INTERCOMM

ASSEMBLER LANGUAGE PROGRAMMERS GUIDE

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

LICENSE: INTERCOMM TELEPROCESSING MONITOR

Copyright (c) 2005, 2022, Tetragon LLC

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Use or redistribution in any form, including derivitave works, must be for non-commercial purposes only.

2. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

3. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Assembler Language Programmers Guide

Publishing History

Publication	Date	Remarks
First Edition	September 1973	This manual corresponds to Intercomm Release 6.0.
Second Edition	June 1989	This manual corresponds to Intercomm Releases 8.0, 9.0, and 10.0.

NOTES:

The following enhancements are for Release 10 only:

- 3-byte MSGHBMN number
- INTSORT (in-core table sort) service routine
- Dynamically loaded programs above the 16meg line
- GETDATE macro
- SCTL command for subsystem activity display
- VSAM data set access under Dynamic File Allocation (DFA)
- Subsystem message flushing.

The following are desupported under Release 10:

- AMIGOS file access method
- DISAM file access method.

The following are not supported under Release 8:

- BINSRCH3 entry for binary search processing
- IJKWHOIT Csect name conversion
- Enhanced VSAM facilities:
  - Sharing files across regions
  - RRDS support
  - ESDS empty file load
  - FAR parameters: DSN, LSR, WRITEOVER.

The material in this document is proprietary and confidential. Any reproduction of this material without the written permission of Isogon Corporation is prohibited.

PREFACE

Intercomm is a state-of-the-art teleprocessing monitor system executing on the IBM System/370 family of computers and operating under the control of IBM Operating Systems (MVS/370, XA, 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.

The Assembler Language Programmers Guide explains the organization of Intercomm from the application programmer's point of view and illustrates the procedures for creating Assembler Language application programs and integrating them into the Intercomm environment.

Syntax used in describing the coding of JCL or application program statements is:

- { } A pair of braces indicates the presence of a choice: code elements contained within the braces represent alternatives, one of which must be chosen. The braces are not to be coded.

- [ ] A pair of brackets indicates an optional parameter which may be omitted depending on access requirements as described in the accompanying text. The brackets are not to be coded.

- A parameter consisting partially or solely of lower case letters represents the generic (Intercomm) name of the value. The programmer must substitute the actual name used for defining the data area within the specific program.

As a prerequisite to this manual, it is assumed that the user is familiar with the Intercomm Concepts and Facilities Manual. The following manuals describe in further detail facilities refererenced in this manual:

- Basic System Macros
- Message Mapping Utilities
- Utilities Users Guide
- Store/Fetch Facility Users Guide
- Dynamic Data Queuing Facility
- Page Facility
- Operating Reference Manual: "Message Management"
  "File Management"

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

		Page
Chapter 1	INTRODUCTORY CONCEPTS OF ON-LINE SYSTEMS	1
1.1	Introduction	1
1.2	The On-Line System Environment	1
1.3	Batch Environment vs. On-Line Environment	3
1.4	Single-Thread vs. Multithread Processing	4
1.5	Program Functions in the On-Line Environment	6
1.5.1	Monitor Control Functions	7
1.5.2	Application Processing Functions	7
Chapter 2	MESSAGE PROCESSING AND CONTROL UNDER INTERCOMM	9
2.1	The Intercomm Environment	9
2.2	System Components	11
2.2.1	Front End	11
2.2.2	Back End	11
2.3	System Programs	12
2.4	Subsystems	16
2.4.1	Reentrant vs Nonreentrant Subsystems	16
2.5	Intercomm Tables	17
2.6	Interfacing with the Intercomm Monitor	18
2.7	Intercomm Message Header	19
2.7.1	MSGHQPR and MSGHVMI Fields	22
2.8	Intercomm Message Flow Using Message Mapping	22
2.9	Intercomm Message Flow Using Edit and Output	24
2.10	The Intercomm System Log	26
2.11	Additional Application Processing Facilities	30
Chapter 3	CODING AN INTERCOMM SUBSYSTEM IN ASSEMBLER LANGUAGE	31
3.1	Program Structure	31
3.2	Message Processing Concepts	35
3.3	Subsystem Coding	38
3.3.1	Subsystem Entry	39
3.3.2	Linkage	40
3.3.3	Message Processing	40
3.3.4	Additional Coding Techniques	42
3.3.5	Subsystem Illustration	44
3.3.6	Message Switching Between Subsystems	46.1
3.4	Restarted Messages	46.1
3.5	MVS/XA Extended Storage Loading Requirements	46.2
Chapter 4	USING THE MESSAGE MAPPING UTILITIES	47
4.1	Concepts	47
4.2	Processing	47
Chapter 5	USING THE EDIT UTILITY	49
5.1	Concepts	49
5.2	Processing Results	50

		Page
Chapter 6	USING THE FILE HANDLER	51
6.1	General Concepts	51
6.1.1	Subsystem Processing	52
6.2	Calling Service Routines	53
6.2.1	Automatic Error Checking	55
6.3	Select, Release Functions	56
6.3.1	Closing a File	57
6.4	Exclusive Control for Non-VSAM Files	57
6.4.1	Release Exclusive Control--RELEX	59
6.5	Sequential Access Method Processing	60
6.5.1	File Handler Service Routines	60
6.5.2	Undefined Record Format and Record Length	61
6.5.3	Variable-Length Record Format and Record Length	61
6.6	Indexed Sequential Access Method Processing	62
6.6.1	File Handler Service Routines	62
6.7	Direct Access Method Processing	64
6.7.1	File Handler Service Routines	64
6.8	Virtual Storage Access Method (VSAM) Processing	67
6.8.1	File Handler Service Routines	67
6.8.2	VSAM Processing Options	69
6.8.3	FHCW Reason Codes for VSAM	70
6.8.4	Exclusive Control for VSAM Files	70
6.8.5	Alternate Index Processing of Keyed VSAM Files	71
6.9	ISAM/VSAM Compatibility Under Intercomm	74
Chapter 7	USING THE OUTPUT UTILITY	75
7.1	Concepts	75
7.2	Processing	75
Chapter 8	CONVERSATIONAL SUBSYSTEMS	79
8.1	General Concepts	79
8.1.1	Conversational Applications	79
8.1.2	Conversational Transactions	79
8.1.3	Retention of Information	80
8.2	Implementing Conversational Subsystems	81
8.3	Saving Information in USERSPA	82
8.4	Saving Information with Store/Fetch	84
8.5	Saving Information on a Dynamic Data Queue	86
8.6	Saving Information via the CONVERSE Service Routine	88
8.6.1	Subsystem Design Using CONVERSE	90
8.7	Design Considerations in Conversational Processing	93
8.7.1	Control of the Input to Conversations	93
8.7.2	Assigning a Verb to a Terminal	93

		Page
Chapter 9	USING INTERCOMM SERVICE ROUTINES AND FACILITIES	95
9.1	Service Routines and Facilities	95
9.2	Message Switching (MSGCOL)	96
9.3	User Log Entries (LOGPUT)	97
9.4	Pass Message to Front End (FESEND, FESENDC)	98
9.5	Front End Control Messages	100
9.5.1	Front End Data Queuing (FECMDDQ)	100
9.5.2	Front End Feedback Messages (FECMFDBK)	101
9.5.3	Front End Queue Release (FECMRLSE)	103
9.6	Performing Binary Table Search (BINSRCH, BINSRCH2, BINSRCH3)	104
9.7	Data Field Search Routines (PMIFINDB,PMIDLTDB)	105
9.7.1	PMIFINDB - Find a Data Field	106
9.7.2	PMIDLTDB - Delete or Add a Data Field	107
9.8	Segmented Message Input (GETSEG)	108
9.8.1	Segmented Message Output Terminal Assignment (DVASN)	109
9.9	Dispatcher Related Routines	110
9.9.1	IJKPRINT - Direct Output Line to SYSPRINT	110
9.9.2	IJKTRACE - Print Dispatcher Queues 110.1	
9.9.3	IJKDELAY - Request Time Delay 110.1	
9.10	In-Core Table Sort Facility (INTSORT) 110.2	
9.11	Other Intercomm Service Facilities 110.3	
9.12	Loading Service Routine Entry Points from the SPA...110.4	
Chapter 10	INTERCOMM MACROS FOR ASSEMBLER LANGUAGE PROGRAMS	111
10.1	Introduction	111
10.2	Macro Descriptions	114
10.2.1	CATCH	114
10.2.2	DISPATCH	114
10.2.3	INTDEQ	115
10.2.4	INTENQ	115
10.2.5	INTPOST	115
10.2.6	INTWAIT	115
10.2.7	MODCNTRL	116
10.2.8	PASS	116
10.2.9	PMISNAP	116
10.2.10	PMIWTO	116
10.2.11	PMIWTOR	117
10.2.12	SUBTASK	117
10.2.12	USRTRACK	117
10.3	Macro Coding Examples	118
10.3.1	DISPATCH Macro Usage	118
10.3.2	PASS/CATCH Macro Usage	120
10.3.3	INTENQ/INTDEQ Macro Usage	121
10.3.4	MODCNTRL Macro Usage	121
Chapter 11	SAMPLE PROCESSING PROGRAMS	123

		Page
Chapter 12	SUBSYSTEM TESTING	129
12.1	Introduction	129
12.2	Debugging Application Program Problems	129
12.3	Testing a Subsystem with the Front End Simulator	130
Chapter 13	SUBSYSTEM TESTING IN TEST MODE	167
13.1	Introduction	167
13.2	Testing a Subsystem in Test Mode	167
Appendix A	ASSEMBLER LANGUAGE JCL PROCEDURES	197
Appendix B	DSECTS FOR ASSEMBLER LANGUAGE PROGRAMS	199
Appendix C	INTERCOMM TABLE SUMMARY	205
Appendix D	SPA AND SPAEXT FIELD NAMES FOR ROUTINE ENTRY POINTS	209
D.1	Fields in the SPA	209
D.2	Fields in the SPAEXT	210
Appendix E	NONREENTRANT SUBSYSTEMS	215
E.1	Introduction	215
INDEX		221

LIST OF ILLUSTRATIONS

Figure		Page
1	On-line Transaction Processing in a Multiprogramming Environment	2
2	Differences Between Batch and On-line Environments	3
3	Multithreading in an On-line Environment	5
4	The Intercomm Environment	9
5	Intercomm Control Sequence	10
6	Intercomm System Components	13
7	Intercomm Message Header Fields	20
8	Intercomm Message Flow Using Message Mapping	23
9	Intercomm Message Flow Using Edit and Output	25
10	Sequence of Log Entries	27
11	INTERLOG Entries	28
12	Reentrant Assembler Subsystem Structure	32
13	Reentrant Application Program Environment	33
14	Intercomm System Return Codes	34
15	Subsystem Logic Using Message Mapping Utilities	36
16	Subsystem Logic Using Edit and Output Utilities	37
17	Echo Message Example; Reentrant Assembler Language	46
18	Message Processing Using MMU	48
19	Edit Utility Processing of Fields Omitted or in Error	50
20	Functions of File Handler Service Routines	51
21	DD Statement Parameters for the File Handler	52
22	Defining File Handler Control Areas	53

Figure		Page
23	File Handler Service Routine Parameters	54
24	Outline of File Handler Return Codes	55
25	File Handler SELECT/RELEASE Return Codes	56
26	Exclusive Control Processing	58
27	File Handler Release Exclusive Control (RELEX) Return Codes	59
28	File Handler Sequential Access Method Return Codes	60
29	File Handler ISAM Return Codes	63
30	File Handler BDAM Option Codes	65
31	File Handler BDAM Return Codes	66
32	File Handler VSAM Call Summary	72
33	File Handler VSAM Return and Feedback Codes	73
34	Message Header Specifications for the Output Utility	77
35	Typical Conversational Transactions	80
36	Input Message Data Retention During a Conversation	80
37	User and Terminal Table Space in the USERSPA	82
38	Sample USERSPA Declaration Within a Subsystem	83
39	Conversational Processing Using Store/Fetch	85
40	Conversational Processing Using Dynamic Data Queuing	87
41	Conversational Subsystem Logic Using Converse	89
42	CONVERSE Return Codes	91
43	Message Collection Return Codes	96
44	FESEND Return Codes	99
45	FECM Return Codes	100
46	GETSEG Return Codes	108
47	INTSORT Return Codes	110.2

Figure		Page
48	Sample Reentrant Subsystem (Assembler)	124
49	Sample Assembler Subroutine	125
50	Table Updates to Implement Simulation Mode Testing	132
51	MMU Maps Used by Sample Subsystem	133
52	Input Test Messages Generated via CREATSIM	134
53	Linkedit and Execution JCL for Simulation Mode	136
54	SIM3270 Printout from Simulation Mode Execution	139
55	Simulation Mode Execution Log Printout	159
56	Sample Inquiry Subsystem; Reentrant Assembler	169
57	Table Updates to Implement Test Mode Testing	180
58	Utilities Table Coding for Test Mode Subsystem	181
59	Test Mode Message Card Formats	183
60	Sample Input Test Messages for Test Mode	184
61	Linkedit and Execution JCL for Test Mode	185
62	Sample Test Mode Execution Snaps	188
63	Test Mode Execution Log Printout	191
A-1	Intercomm-supplied Assembler JCL Procedures	197
B-1	Intercomm Dsects for Assembler Programs	200
C-1	Table Names and Associated Macro Instructions	205
C-2	Components and Associated Table Names	207
E-2	Nonreentrant Assembler Subsystem Structure	216
E-3	Nonreentrant Application Program Environment	217

Chapter 1

INTRODUCTORY CONCEPTS OF ON-LINE SYSTEMS

1.1 INTRODUCTION

The objective of most on-line systems is to reduce the time factor from source of input data to the results of data processing. Typical on-line systems applications in the business environment are:

- Data Collection

  Transactions may be edited partially on receipt, batch totals may be transmitted and verified, but the bulk of processing of the collected data takes place in the batch mode off-line.

- Inquiry/Update Systems

  Transactions are processed immediately to retrieve and/or update information in an on-line data base.

- Message Switching

  Transactions consist of administrative data to be rerouted to other terminals in the system.

On-line systems are characterized by a mode of operation which is nonscheduled and transaction-oriented. An operator at a terminal remote from the data processing center enters a transaction (unit of work) by transmitting a message over communication facilities. Each individual transaction is processed immediately, as opposed to batch systems, where transactions are accumulated for processing on a periodic basis (monthly, daily, etc.).

Online systems are designed to satisfy a response time requirement which is the elapsed time between a request for processing of an input message from a terminal to receipt of an acknowledgement, or response to that input message (completion of a transaction).

1.2 THE ON-LINE SYSTEM ENVIRONMENT

Typical on-line message processing application programs operate on one transaction at a time as they come in from terminals. Application programs are usually designed to process only one type of transaction, and the whole environment can be said to be transaction oriented. Input messages can be processed as received, in any order, and the files to be referenced should not be read from beginning to end for each transaction. Instead, the records in files are accessed directly, either through a specific key or some form of cross-reference look-up.

A few applications might require some sequential or list processing of a file, and while this is possible, message processing times for such applications would tend to be high.

Figure 1. shows a computer system schematic depicting a memory layout with an on-line system such as Intercomm, operating in a region or address space as a job under an operating system such as IBM's MVS. The on-line system has its own Transaction Monitor which schedules the activation of transaction processing according to the varying demands in message traffic.

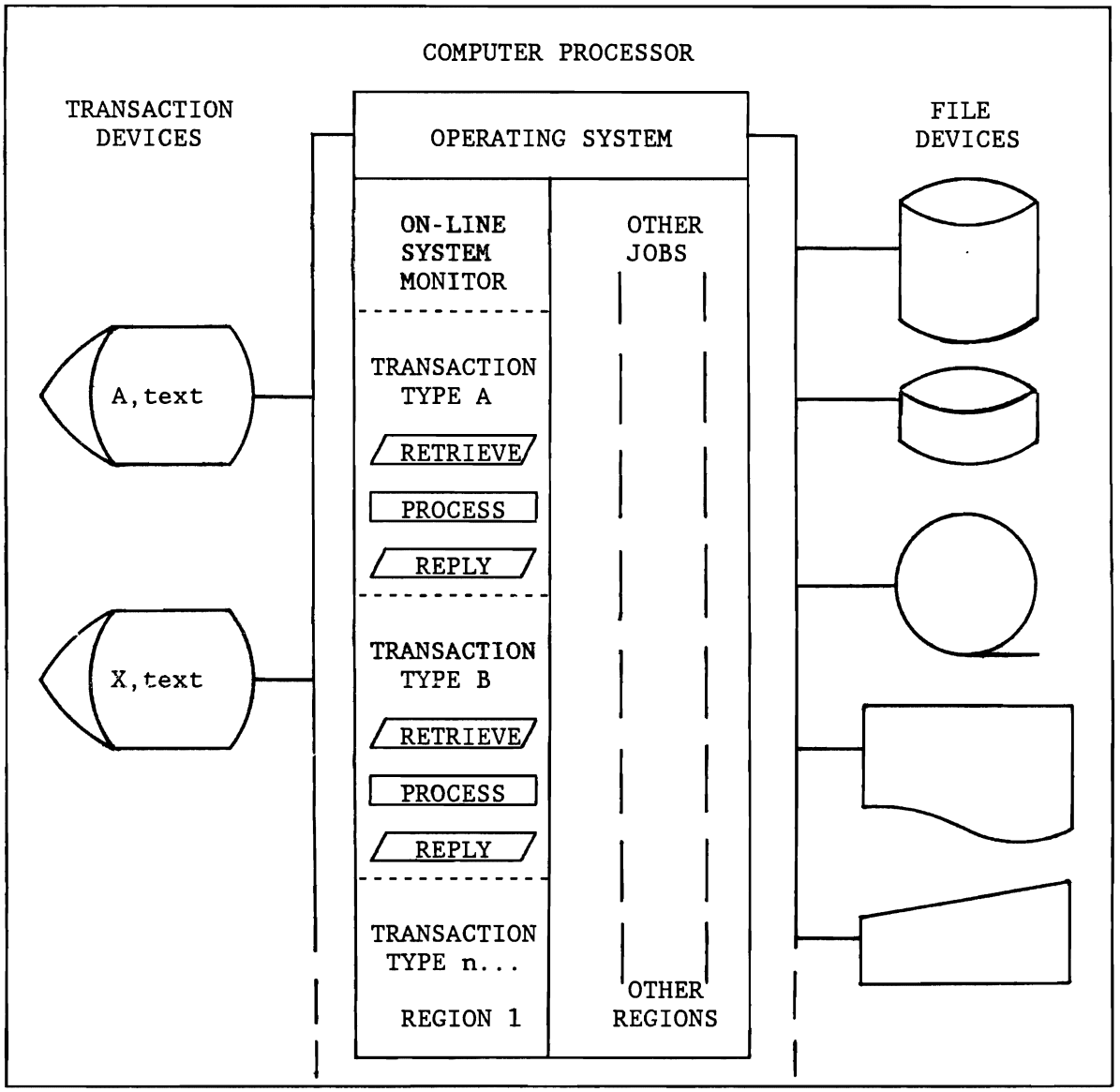


Figure 1. On-line Transaction Processing in a Multiprogramming Environment

The transaction processing programs do not conduct input or output operations with the terminals. This function is provided by the on-line system, which reads input messages from terminals and saves them (queues them) until the appropriate processing program can be activated (scheduled). The message is then retrieved from the queue and passed directly to the processing program by the Monitor. The processing program then requests the Monitor to queue its output response message, and the Monitor handles the terminal output function.

1.3 BATCH ENVIRONMENT VS. ON-LINE ENVIRONMENT

The classical batch processing system flow of input/process/output can be expanded to include message queuing and retrieving in the on-line environment. However, the typical on-line application program need only be concerned with actual transaction processing, because the on-line system does the rest. Figure 2 summarizes some of the differences between batch and on-line environments.

Batch	Online
Scheduled input	Unscheduled input
Single-application job	Multiple-application job
Delayed processing of transactions in batches by type	Immediate processing of individual transactions by type
Transaction input, processing, and output controlled by processing program logic	Terminal input/output events are asynchronous to the processing program

Figure 2. Differences Between Batch and On-line Environments

1.4 SINGLE-THREAD VS. MULTITHREAD PROCESSING

In the on-line environment, the logical path of a program in execution is called a thread. A single-thread system processes one message at a time. However, in a multiple application environment, message volume is such that all message traffic could not be adequately serviced in a single-thread mode. Large queues (waiting lines) tend to develop because messages arrive faster than they can be processed. To alleviate this problem and improve system throughput, the delay time in the processing of one message waiting for an I/O operation may be used for simultaneously processing another message. In this way, several message processing logic paths, or threads, may be active at once. This is referred to as multithreading.

Multithreading is coordinated by the Transaction Monitor, and, depending on message traffic, can occur between two or more programs or within a single program.

To illustrate this, let us assume that we have two transaction processing programs, A and B, and that three messages have arrived for processing; two A-type transactions and one B-type transaction. Programs A and B both require access to records in a file, affording an opportunity for some processing overlap or multithreading. Multithreading would occur between programs A and B if while program A is waiting for file retrieval, program B is activated by the Monitor to carry out its message processing. However, if program A were reentrant, that is, written in such a way that it could handle more than one thread at a time, then multithreading could also occur within program A. This means that while reentrant program A is waiting for a file retrieval for the processing of one message, it may be activated again to carry out the parallel processing of a second, or nth, message. Figure 3 illustrates these concepts.

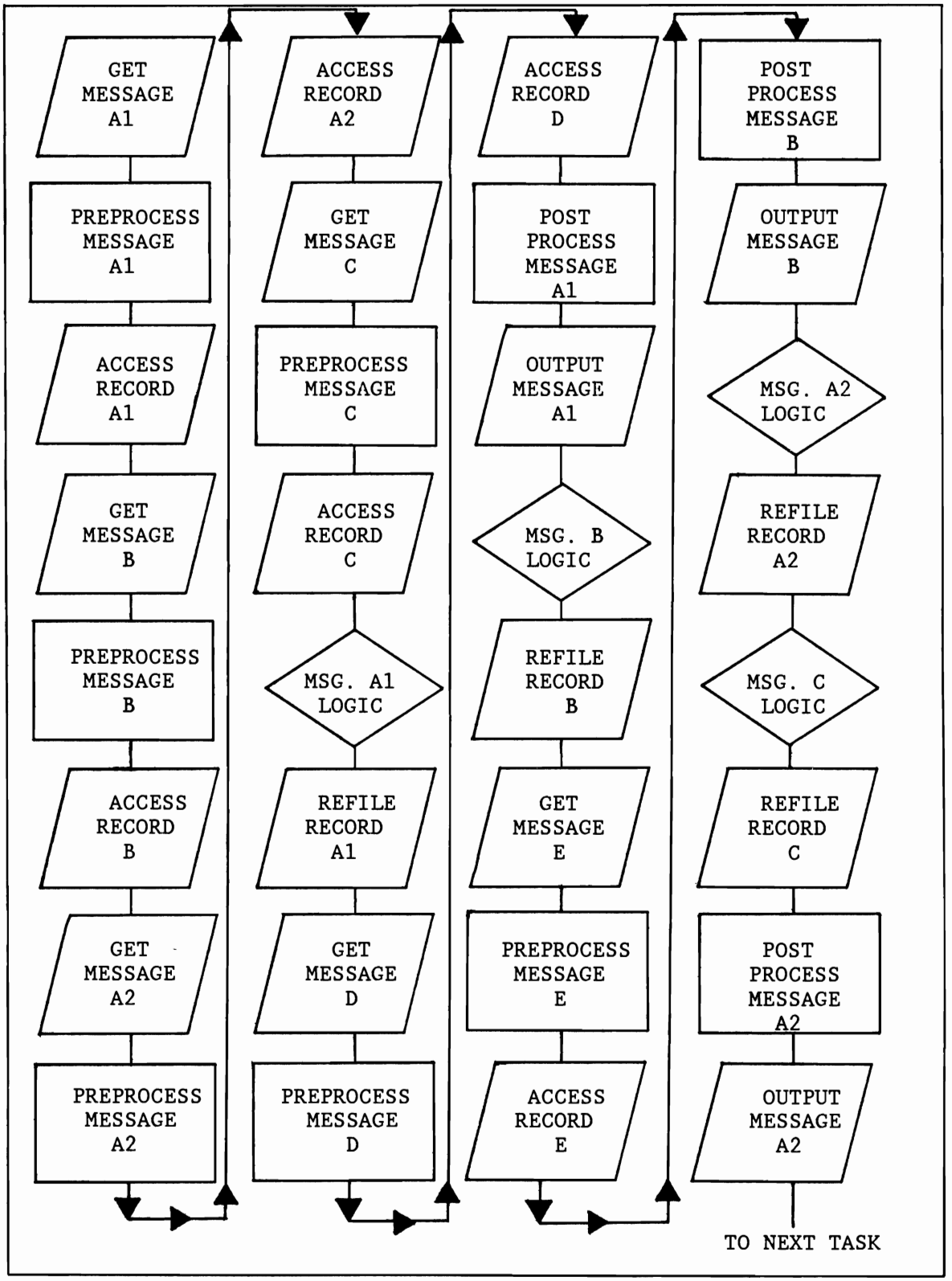


Figure 3. Multithreading in an On-line Environment

1.5 PROGRAM FUNCTIONS IN THE ON-LINE ENVIRONMENT

An on-line system consists of programs to serve four different functions:

- Line Control and Terminal Control
  - -- Servicing input requests from the various terminal types including transmission error recovery
  - -- Directing output to the various terminal types including transmission error recovery
  - -- Intercepting and storing messages to non-operational devices, and retrieval of messages when devices become operational
  - -- Translation of messages to and from terminal transmission code and EBCDIC code for processing
- Message Processing Control
  - -- Queuing new input messages until the associated message processing program is scheduled for execution
  - -- Scheduling message processing programs to obtain best system throughput for message traffic
  - -- Controlling multithread operation for concurrent processing of several messages
  - -- Centralizing data file accesses to eliminate redundant operations and provide exclusive control over records during file updates
- Systems Operation Control
  - -- Security checking functions to restrict certain transactions to specific operators and/or terminals, and to prevent access to unauthorized functions/files.
  - -- Logging (journaling) of all message traffic
  - -- Checkpointing, Message Restart, File Recovery and Backout-On-The-Fly (dynamic file backout) facilities
  - -- Cancellation of message processing programs when a program check or program loop occurs
  - -- Collect and display system statistics
  - -- Display and modify system status

- Message (Transaction) Processing

  -- Editing text data from terminal input, including format conversion and content editing of individual fields

  -- Retrieval and updating of data from on-line files

  -- Preparation of response (output) messages to terminals

  -- Queuing of response messages for output to terminals

1.5.1 Monitor Control Functions

The Intercomm System provides complete facilities for:

- Line control and terminal control
- Message processing control
- Systems operation control

1.5.2 Application Processing Functions

Transaction processing logic lies within the coding domain of the application programmer. Intercomm provides the following message and file handling support:

- Format conversion and editing of input fields
- Centralized control of data files
- Format conversion and placement of constant and variable information in response messages and terminal displays
- Queuing of messages (for the same or another terminal, or another application)

The installation-dependent application logic functions then need include only the following:

- Content editing of individual input message fields
- Retrieval and updating of data from on-line files
- Selection of individual fields for the output message(s)

Chapter 2

MESSAGE PROCESSING AND CONTROL UNDER INTERCOMM

2.1 THE INTERCOMM ENVIRONMENT

Intercomm operates under MVS as a job in a region or address space. The job is loaded at the beginning of on-line operations and continues to operate until the terminal network is closed down. Intercomm contains many system programs and application subsystems. Intercomm system programs include the Monitor and other subprograms to handle such things as terminal and peripheral I/O operations. Subsystems are message processing application programs activated by the monitor. The term "subsystem" includes both application-oriented message processing programs written by users and Intercomm system command processing and utility programs. The Intercomm region contains the execution module itself plus dynamically allocated storage or work space, as illustrated in Figure 4.

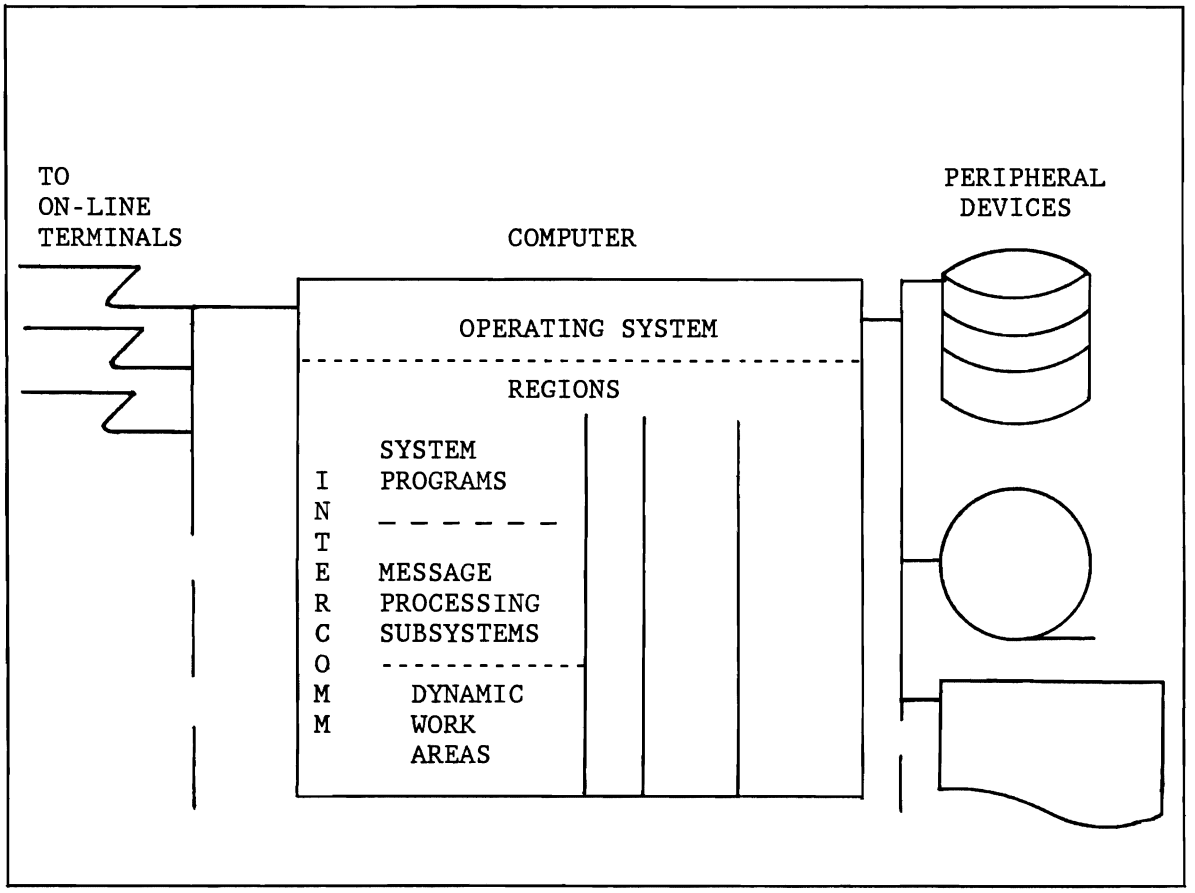


Figure 4. The Intercomm Environment

The system programs are time- or event-driven; the subsystems are message-driven. The Intercomm Monitor calls system programs to handle events and exceptional conditions as they occur, for example, terminal and peripheral I/O interrupts, time-dependent processing, excessive message traffic, and system operator commands.

A subsystem, on the other hand, is called by the system monitor when there are messages queued for it and it has been scheduled for execution. Subsystems, while executing, can use the IBM CALL macro to call user subroutines or to call system programs to perform services, such as accessing data files and queuing messages for output or for additional processing by other subsystems. Figure 5 shows that called system programs and user subroutines will always return to the calling subsystem (or subroutine), just as the subsystem itself, executing as a subroutine of Intercomm, must always return to the system monitor that originally activated it.

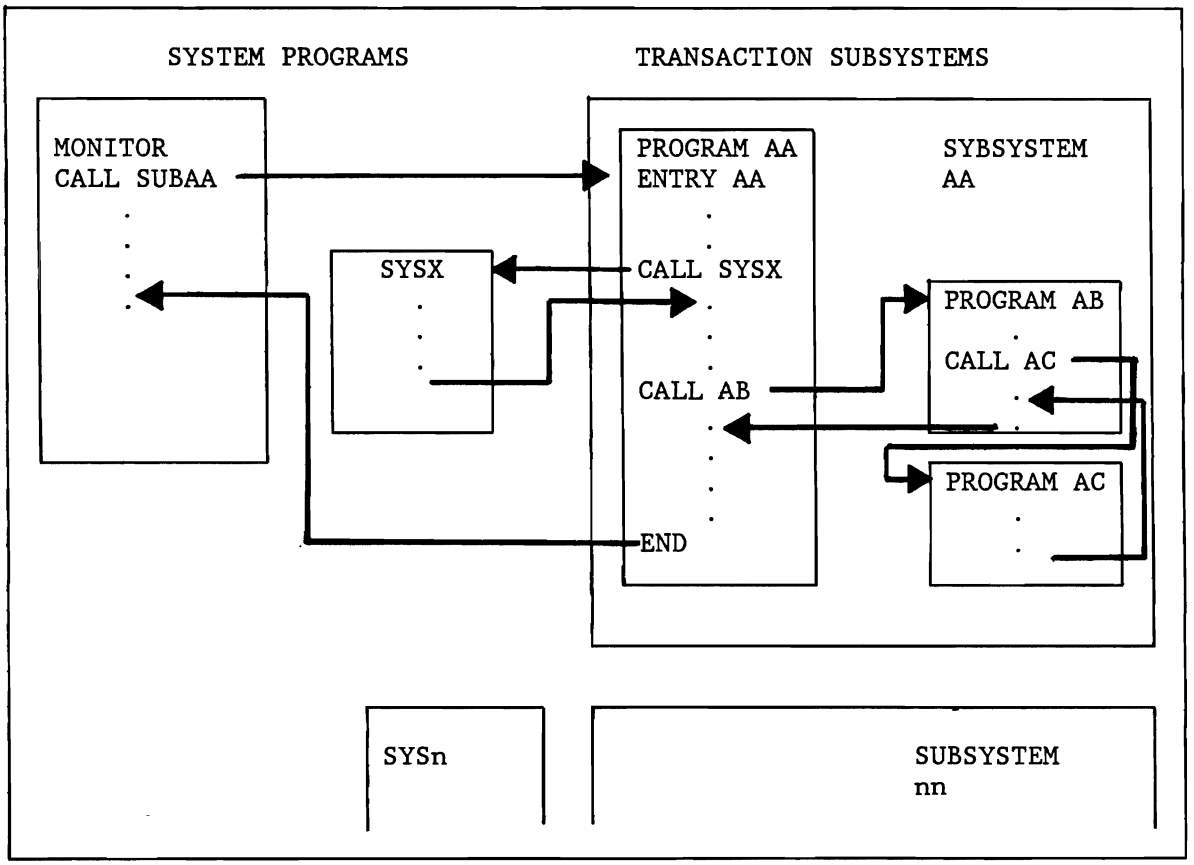


Figure 5. Intercomm Control Sequence

2.2 SYSTEM COMPONENTS

On-line system component programs are often categorized as resident or nonresident, system or user, but typical on-line terminology also distinguishes between Front End and Back End system components.

2.2.1 Front End

The Front End communicates with and monitors all terminals in the network. It receives and sends messages, checks validity, performs security checking if specified, and accomplishes appropriate code translation. The Front End communicates with the Intercomm message processing Back End via input message queuing and output message dequeuing routines. Although Intercomm has its own VTAM Front End, it can also interface with other software Front Ends such as TCAM and BTAM.

2.2.2 Back End

The Back End accomplishes all message processing control, system operation control, and processing of individual messages. It is, essentially, the "director" of the entire on-line system operation.

The Front End and the Monitor portion of the Back End are always resident, whereas message processing subsystems can be any combination of resident and loadable. (See Figure 6.) The decision to make a message processing subsystem permanently resident, or loadable, is based upon the trade-offs between response time, frequency of use, and total system core storage requirements.

2.3 SYSTEM PROGRAMS

Intercomm system programs are written in Assembler language and include the Monitor, File Handler, high-level language interface routines to maintain reentrancy, and message processing service routines.

The Monitor interfaces with the Front End via message queues and controls the processing of messages by subsystems. It is essentially a traffic director, analyzing message traffic and scheduling subsystems based upon traffic volume and priority criteria. The Monitor has four key components:

- The TP queuing interface, which communicates with the Front End to dequeue input messages or to queue output messages created by subsystems.
- The Subsystem Controller, which schedules, loads and activates the application subsystems, and performs clean up processing when the subsystem returns.
- The Dispatcher, which controls the execution of all events in the system to accomplish multithreading.
- The Resource Manager, which allocates/deallocates and controls dynamic resources (such as core storage) used by system and application programs.

The File Handler is the central Intercomm routine where all peripheral I/O service for data files is controlled. The File Handler issues OPENs, CLOSEs, GETs, PUTs, READs, and WRITEs via the operating system data management facility. Subsystems merely call an appropriate File Handler routine. Therefore, all access methods supported by Intercomm are available to any subsystem program, regardless of the programming language used. The File Handler maintains a single set of control blocks for each file defined to it via standard Job Control Language Data Definition statements, and all programs share this one set of control blocks. Intercomm can control overlapping of peripheral I/O processing, as well as provide standardized error analysis. A file is usually opened only once during an on-line session: at the time the first I/O is requested. Since files can be accessed concurrently by different subsystems, an exclusive control feature is provided to eliminate difficulties arising when two or more subsystems (or subsystem threads) attempt to update the same record at the same time.

Language interface routines are described in Chapter 3.

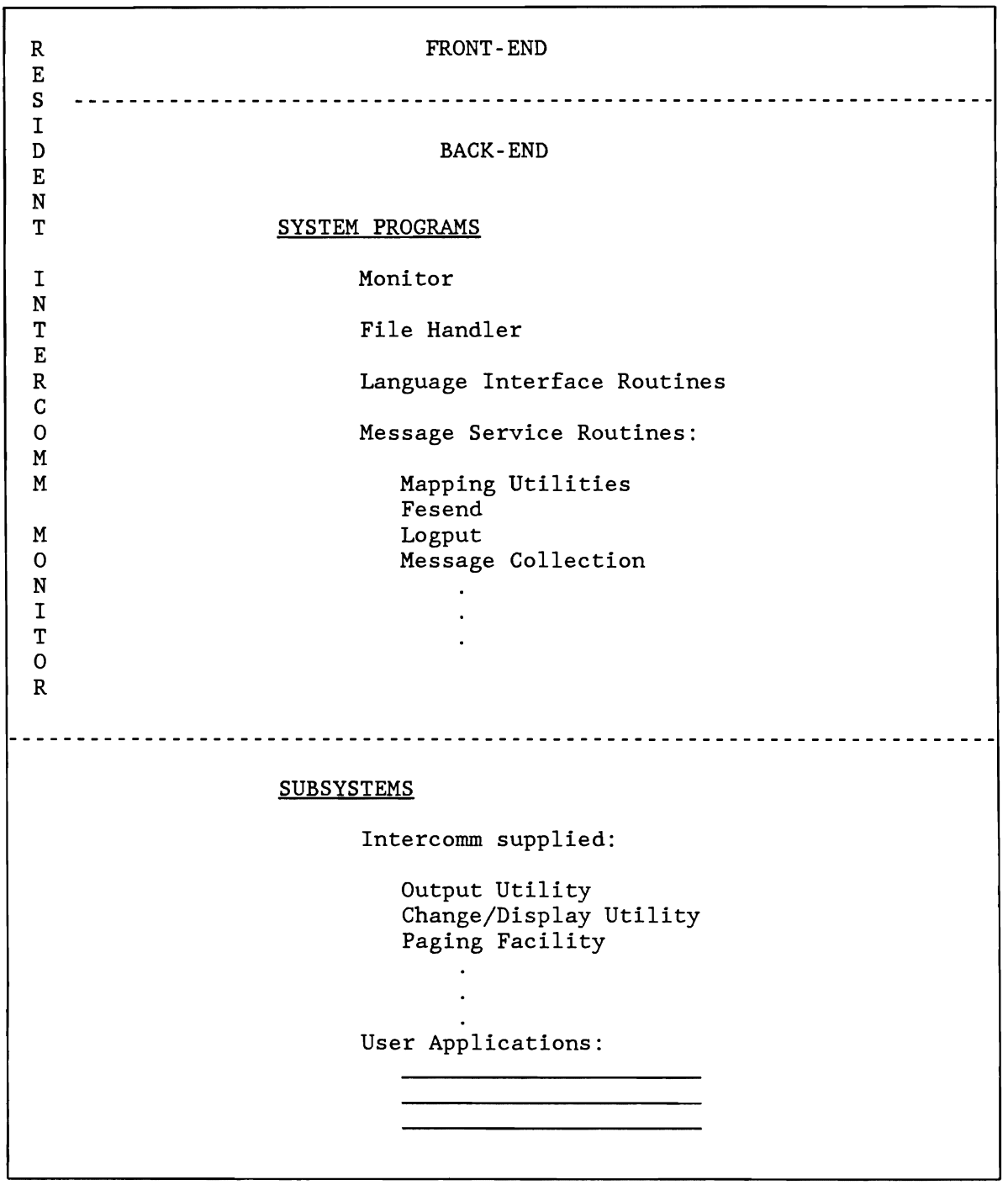


Figure 6. Intercomm System Components

The basic message processing service routines are:

- FESEND--which passes an output message to the Front End for transmission to a terminal.

- LOGPUT--which copies a message onto the system log whenever called by a system program or subsystem.

- MESSAGE COLLECTION--which handles the queuing and dequeuing of all messages destined for subsystems.

Intercomm provides service routines to convert terminal-dependent input messages to a terminal-independent form for application processing. This transformation includes removal of terminal-dependent control characters and conversion of numeric data fields to computational or binary form, if required. Similarly, for output messages, service routines provide transformation from terminal-independent results of application subsystem processing to terminal-dependent messages for transmission. This includes insertion of terminal-dependent control characters, conversion of data fields to character format, if required, and inclusion of title information, if specified. Each of these routines function via user-specified descriptions (tables) of input and output message formats. These service routines are:

- Message Mapping Utilities

  This is a set of service routines called by an application program to perform the device-dependent transformations specified by the user for both input and output messages. Validity checking, conversion, justification and padding/truncation of data fields is also performed. This utility also executes output message disposition (queuing/spooling), if requested.

- Edit Utility

  This is a service routine called by the Monitor to process input messages, performing device-dependent transformations, and field validity checking, conversion and padding according to user-specified editing characteristics.

- Output Utility

  This is a service routine executing as a subsystem to process output messages by performing device-dependent transformations, and then pass the messages to the Front End.

For detailed documentation of these facilities, see Message Mapping Utilities and the Utilities Users Guide.

Other service routines of the Intercomm system for processing requests associated with special subsystem design requirements are:

- Store/Fetch

  This facility allows a subsystem to save and retrieve a temporary or permanent data string identified by a user-defined key. One or more subsystems can access each stored data string. (See Store/Fetch Facility.)

- Dynamic Data Queuing (DDQ)

  This facility allows a subsystem to save and retrieve a set of related data strings (a data queue) identified by a user-defined name. One or more subsystems can access each DDQ which may be transient or permanent. A DDQ may also be used for collecting messages destined for another subsystem, a printer, or even a batch program. (See Dynamic Data Queuing.)

- CRT Page Facility

  This facility allows a subsystem to write a set of output messages to a CRT terminal-oriented Page Data Set. The first message of a set is also sent to the Front End automatically. The terminal operator may then enter commands processed by the Page subsystem to retrieve and browse through the pages of a set of output messages. (See Page Facility.)

- Data Base Management System Support (DBMS)

  This facility consists of separate service routines for each supported DBMS (IDMS, System 2000, Model 204, ADABAS, TOTAL, DL/I, or a user DBMS), which allows access to the DBMS from Intercomm. (See the Data Base Management System Users Guide.)

- Dynamic File Allocation (DFA)

  This facility allows a subsystem to create (allocate) and/or access a sequential data set, or to access a VSAM data set, specifying its DSNAME as part of subsystem logic, rather than with execution JCL. (See Dynamic File Allocation.)

- Signed-on Operator-Id Checking

  When executing under the security control of the Intercomm Extended Security System, a subsystem may call a service routine (SECUSER) to determine the user-ID of the operator at the terminal from which the transaction to be processed was entered. (See Extended Security System.)

2.4 SUBSYSTEMS

Intercomm-supplied subsystems are written in reentrant Assembler Language, and include the Output Utility, the Change/Display Utility, the Page Browsing Subsystem and many command processing subsystems.

The Output Utility allows a programmer to specify predefined report and display formats so that simply constructed output messages from a subsystem can be expanded, columnized, headed and subheaded, and displayed upon different types of devices without concern to the subsystem creating the message. Output Utility display formats can be changed without program modifications.

The Change/Display Utility allows simple inquiry and file maintenance via predefined keyword input messages from terminals causing access to data files defined by tables. The Display Utility is used in conjunction with the Output Utility to produce varied report or display formats.

The Page Facility processes commands from CRT-type terminals to browse through a file of output display screens created by the PAGE system program. Subsystems make use of this feature by calling the page storage program during message processing. The terminal operator interacts with the Page Facility directly.

Command processing subsystems process Intercomm standard messages to accomplish the start/stop of system functions, message switching between terminals, displaying and changing the status of system control parameters, display of statistics, etc. The commands and text syntax are described in System Control Commands.

User-supplied subsystems accomplish application-dependent message processing. Each may call any Intercomm service routine or user-supplied subroutine, and may be written in COBOL, Assembler or PL/1.

2.4.1 Reentrant vs Nonreentrant Subsystems

In an interactive on-line environment, the probability is very high for more than one terminal operator to enter concurrent requests to be processed by the same subsystem. To accomplish the multithreading of concurrent requests, application subsystems should be coded as reentrant, that is, variable data is defined and processed in a dynamic working storage area obtained for the exclusive use of one processing thread. For Assembler Language subsystems, Intercomm provides entry and exit macros for obtaining and freeing the dynamic working storage area (save/work area), in addition to Intercomm equivalents of the IBM GETMAIN and FREEMAIN macros to obtain and free additional storage to hold messages, etc. to be passed to other programs. These macros are described in Chapter 3.

2.5 INTERCOMM TABLES

Intercomm is a generalized on-line system monitor, requiring information about specific operating characteristics of a particular installation. This information is supplied in the form of tables generated with Intercomm macro instructions. Application programmers are usually not involved in defining the Intercomm tables, except for table specifications which pertain to their own applications. The basic tables controlling message processing are as follows:

- Front End Verb Table (BTVRBTB)

  A table listing all valid transaction identifiers (verbs), and relating them to the subsystem required for message processing. There is one entry per verb, defined via a BTVERB macro.

- Front End Network Table

  Tables describing the terminal network (relating individual devices to five-character station identifications), device hardware and operating characteristics, and output message queuing specifications.

- Back End Station Table (PMISTATB) and Device Table (PMIDEVTB)

  Tables describing terminal identifications and device-dependent characteristics to the Message Mapping Utilities and/or the Edit and Output Utilities.

- System Parameter List (SPA)

  A table describing system-wide operating characteristics, and consisting of two Csects: SPA and SPAEXT. The SPA Csect may be extended to include installation-defined table entries, accessible to all user subsystems and subroutines (see Chapter 8). This table is generated via the SPALIST macro.

- Data Set Control Table (DSCT)

  A table generated by the File Handler describing on-line data sets. Information in this table is derived from JCL and file control (FAR) parameters at execution startup time.

- Subsystem Control Table (SCT)

  A table listing the program properties (reentrancy, language, entry point, etc.), message queue specifications (core and/or disk queues), and scheduling (resident or loadable, concurrent message processing limits, priority, etc.) for each subsystem. There is one entry per subsystem, defined via a SYCTTBL macro.

The above listed tables are described in detail in the Operating Reference Manual. Additional tables describe detailed functions for the system programs, service routines and utilities.

2.6 INTERFACING WITH THE INTERCOMM MONITOR

Each message processed by Intercomm consists of a 42-byte header prefix, plus application-oriented message text. The message header is prefixed to each input message by the Front End and is analyzed by the System Monitor for all message processing control. The particular fields of the header which control message routing are Receiving Subsystem Code (MSGHRSC) and Receiving Subsystem Code High-Order (MSGHRSCH). This two-byte code is initialized by the teleprocessing interface when it constructs the header from the verb supplied at the beginning of the message text. The Front End Verb Table relates user verbs to their corresponding subsystem codes via coding of BTVERB macros (see Basic System Macros) in a user member USRBTVRB copied into the system BTVRBTB containing Intercomm system verbs.

All subsystems are defined to Intercomm by an entry in the Subsystem Control Table (SCT). There is one entry for each subsystem which defines the program's general characteristics, scheduling requirements and message queuing specifications. Each subsystem must be assigned a unique two-character subsystem code for message routing. Definition of Intercomm system subsystems for utility and command processing is provided in the released member INTSCT (formerly in PMISPA under Release 8).

The Subsystem Control Table entry for each user subsystem is defined using the SYCTTBL macro which is coded in a user member USRSCTS copied into the system INTSCT at assembly time. A full description of the macro may be found in the Intercomm Basic System Macros manual.

Many installations assign the responsibility of coding the Subsystem Control Table entries for individual user subsystems to the application programmer. At other installations, the Intercomm System Support Manager performs this task. In either case, the SYCTTBL macros must be coded with care, as there is one table controlling all user and system subsystems in operation when Intercomm is executing.

The most significant SYCTTBL macro parameters for Assembler Language subsystems are:

- LANG=RBAL

  For reentrant assembler language subsystems (LANG=NBAL if nonreentrant).

- SBSP=xxxxxxxx or LOADNAM=xxxxxxxx (for dynamic load)

  Label of entry point to which control is transferred when work is forwarded to subsystem (SBSP), or the load module name for dynamically loaded subsystems (LOADNAM).

- TCTV=nnn

  Expected maximum processing time (in seconds) in a high-volume environment before the subsystem is assumed to be looping, or in an extended wait for file or data base access, and should be timed out. Considerations for this value depend on subsystem processing such as data base access, file updates, number and type of file accesses, exclusive control for file updates, number of output messages created, enqueue lock-out possibilities, etc.

- MNCL=nn

  Specifies the maximum number of concurrent threads that can be executed through this specific subsystem during a high activity period (when more than one operator enters transactions routed to this subsystem).

- RESOURC=name

  This parameter is used to control concurrent access to a resource (file, table, data base, etc.) across several subsystems in one Intercomm region. The name is also coded for the ID parameter of a RESOURCE macro (coded before all SYCTTBLs in the SCT) which identifies the shared resource and the maximum concurrent subsystem threads that may be activated for that resource. Note that the maximum share count coded on the RESOURCE macro overrides the combined MNCL value for all the subsystems "naming" that resource. An internal enqueue is issued (no time-out). While using this feature will affect response time during peak activity, it does not affect the TCTV for a subsystem, which goes into effect after shared control of the resource is granted.

2.7 INTERCOMM MESSAGE HEADER

The Intercomm message header is constructed by the Front End for each message when it arrives from a terminal. New messages created within the subsystem must be prefixed with the standard forty-two-byte header format, which is constructed by copying the input message header to an output message area and then altering appropriate fields. Figure 7 lists the names and formats of all the fields in the message header, and describes their contents and changeability.

Field Name	Length	Description	Alter Legend*
MSGHLEN	2	Length of message including header (binary number)	Y
MSGHQPR	1	Teleprocessing segment I/O code: 02/F2=full message; 00/F0=header segment; 01/F1=intermediate segment 03/F3=final (trailer) segment	N
MSGHRSCH	1	High-order receiving subsystem code	Y
MSGHRSC	1	Low-order receiving subsystem code	Y
MSGHSSC	1	Low-order sending subsystem code	M
MSGHMMN	3	Monitor message number assigned by Message Collection (binary)	N
MSGHDAT	6	Julian date (YY.DDD)**	N
MSGHTIM	8	Time stamp (HHMMSSTH)	N
MSGHTID	5	Terminal identification (originating terminal on input messages, destination terminal on output) or Broadcast Group name	Y
MSGHCON	2	Reserved area	N
MSGHCON+1 (MSGHRETN)	(1)	Subsystem return code (for log code X'FA' entries only)	N
MSGHFLGS	2	Message indicator flags	N
MSGHBMN	3	Front End message number (binary)	N
MSGHSSCH	1	High-order sending subsystem code	M
MSGHUSR	1	Reserved***	L
ORG MSGHUSR MSGHADDR	(1) 2	Used for special processing by the Front End (MSGHBMN-Rel. 8/9)	N
MSGHLOG	1	Log code (see Figure 11)	L
MSGHBLK	1	Reserved area	N
MSGHVMI	1	Verb or message identifier interpreted by receiving subsystem as required, and by FESEND	Y

Figure 7. Intercomm Message Header Fields (Page 1 of 2)

* Alter Legend:

Y = Must be filled in for intersubsystem message switching and output messages passed to FESEND (MSGHVMI should be set to X'57' or X'67', as appropriate, for output messages passed directly to FESEND)

M = Should be filled in for user's own information (required by Intercomm for message restart/file recovery and Log Analysis)

N = Do Not Touch (must be copied from input to output message header area)

L = May be modified for user codes based on subsystem logic

** The period represents a one-byte message thread number (for resource management and/or message restart purposes).

***MSGHUSR is used by Intercomm modules as follows:

1. If the BTVERB macro for the input verb has HPRTY=YES coded; contains a C'P' to request priority queuing for the subsystem. The user may move a C'P' to this field to request priority queuing for output messages to a terminal (via FESEND) or to another subsystem (via Message Collection).

2. For messages to be processed by the Edit Utility, contains a C'F' to indicate that the input message was from a 3270 CRT and contains SBA sequences.

3. For output messages to a switched async device (Teletype, Dataspeed 40, and 2740); a C'B' requests disconnect after transmitting the output message.

4. For output messages to a switched Teletype or Dataspeed 40 device; a C'X' requests using the alternate call-list for the next input message (as described in the BTAM Terminal Support Guide).

5. For output messages discarded by the Front End, a C'F' indicates the message was flushed by command, a C'Z' that it was discarded by the VTAM OTQUEUE user exit (Release 10 only).

If none of the above considerations are applicable, the subsystem may use this field for messages queued to other user subsystems, or for special logging information. The LOGPRINT utility always prints the value coded in this field (in hexadecimal).

Figure 7. Intercomm Message Header Fields (Page 2 of 2)

2.7.1 MSGHQPR and MSGHVMI Fields

In general, an Assembler Language application subsystem does not need to be concerned with the MSGHQPR field, unless processing long input from a Teletype or similar device where message input may be segmented. In this case, the DDQ Facility must be used to store and forward the input message segments. Otherwise, input messages from the Front End always contain a QPR of C'2'. Both MMU and the Output Utility set the QPR to X'02' for output messages unless the Output Utility finds it necessary to segment an output message, in which case a segment code is used. The various uses of the MSGHVMI field for input and output message processing may be determined from the index references to this field at the end of this manual.

2.8 INTERCOMM MESSAGE FLOW USING MESSAGE MAPPING

The interaction of Intercomm system components, tables and subsystems with the Message Mapping Utilities (MMU) is summarized in Figure 8; the path of one input message and its corresponding output message is traced, and the numbered arrows in the diagram correspond to the numbered paragraphs below.

1 The Front End reads an input message and prefixes a 42-byte control header containing routing information, time, date, originating terminal and message length. The message is then queued for subsystem processing by Message Collection.

2 The System Monitor schedules the subsystem and retrieves the message based upon the Subsystem Control Table (SCT) scheduling criteria.

3 The message is passed to the subsystem.

4 Input in terminal-dependent format is transformed to a terminal independent form by a call to a Message Mapping Utility (MMU).

5 The subsystem performs message processing logic, requesting I/O service functions from the File Handler or Data Base Manager interface.

6 The subsystem creates one or more terminal-dependent output messages by calling MMU.

7 The subsystem passes the message formatted by MMU to the Front End by a call to FESEND (unless MMU is asked to perform this function).

8 The subsystem returns control to the System Monitor, passing a return code indicating normal completion or an error condition.

In the Intercomm multithread environment, this same sequence of events is carried out concurrently for many messages.

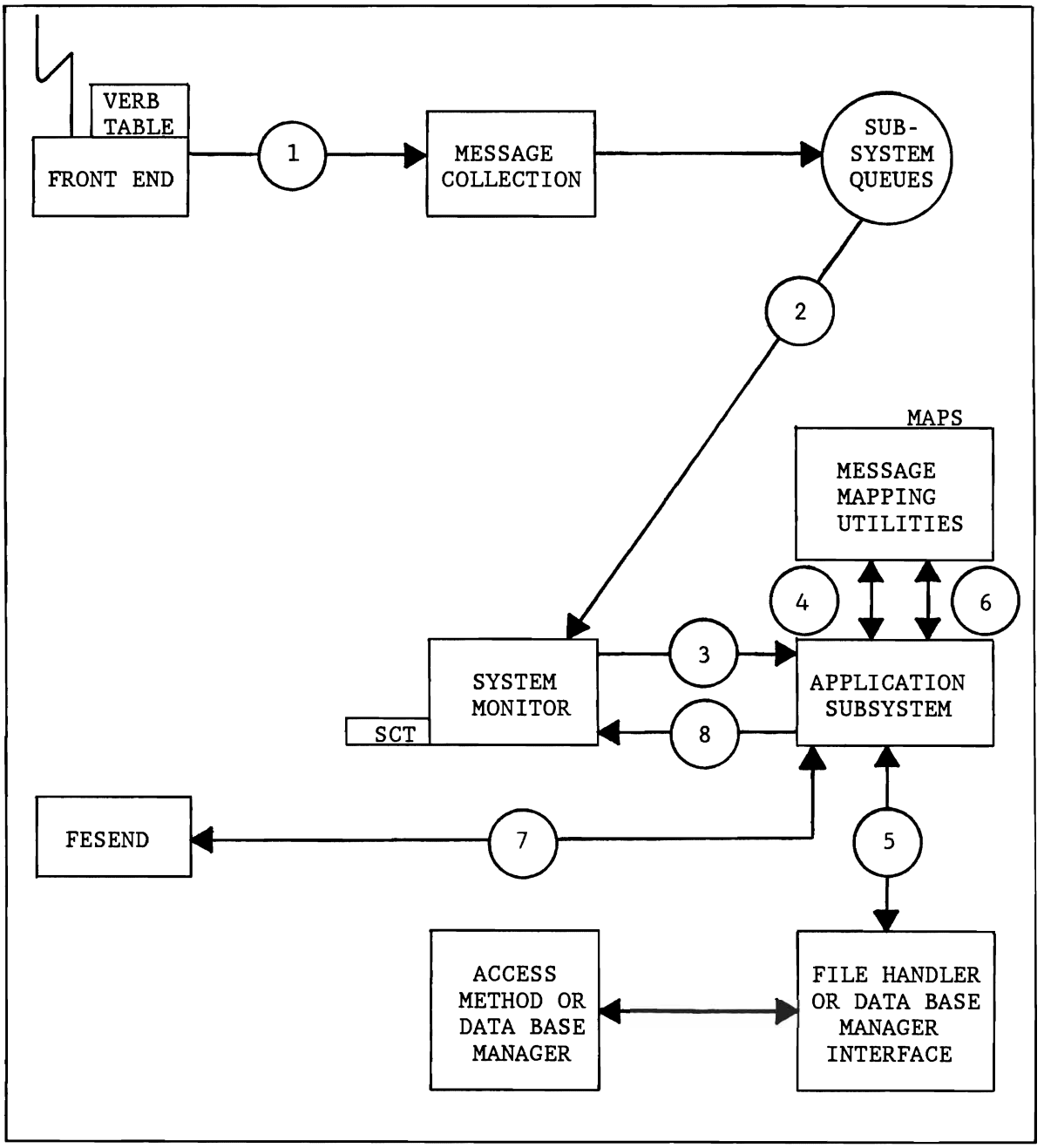


Figure 8. Intercomm Message Flow Using Message Mapping

2.9 INTERCOMM MESSAGE FLOW USING EDIT AND OUTPUT

The path of one input message and its corresponding output message is traced in Figure 9; the numbered arrows in the diagram correspond to the numbered paragraphs below.

1 The Front End reads an input message and prefixes a 42-byte control header containing routing information, time, date, originating terminal, and message length. The message is then queued for subsystem processing by Message Collection.

2 The System Monitor schedules the subsystem and retrieves the message based upon the Subsystem Control Table (SCT) scheduling criteria.

3 The unedited message is passed to the subsystem.

4 The subsystem calls the Edit Utility (if required) and the input message is edited according to the Edit Control Table (ECT).

5 If editing is not successful due to invalid input data, the Edit Utility optionally creates an error message for the originating terminal and queues it for the Output Utility by calling Message Collection, before returning an error code to the subsystem. If editing is successful, the edited message is passed back to the subsystem.

6 The subsystem performs message processing logic, requesting I/O service functions from the File Handler or Data Base Manager interface.

7 The subsystem creates one or more output messages and queues them for the Output Utility by calling Message Collection.

8 The subsystem returns control to the System Monitor, passing a return code indicating normal completion or an error condition.

9 The System Monitor schedules the Output Utility and passes the output message(s) to it for processing.

10 The Output Utility performs formatting, if specified in the header, according to entries in the Output Format Table (OFT), finally passing the message to the Front End via a call to FESEND.

11 The Output Utility returns to the System Monitor.

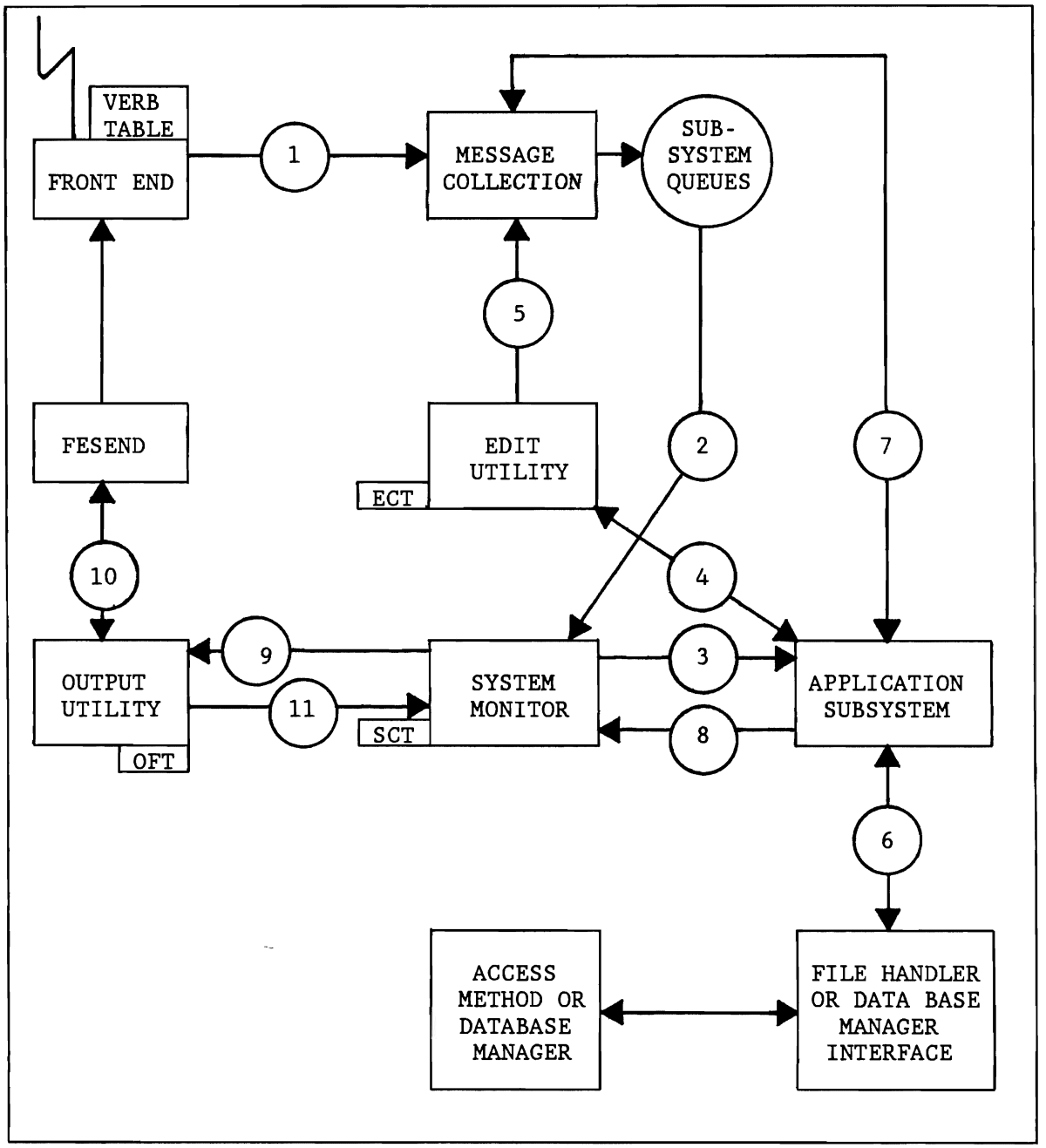


Figure 9. Intercomm Message Flow Using Edit and Output

2.10 THE INTERCOMM SYSTEM LOG

The Intercomm system log (INTERLOG) provides system journaling and maintains a historical record of all traffic within the system. Complete documentation of performance during on-line processing is thus provided, along with system control for restart/recovery.

Message traffic is recorded at the time of entry on a subsystem queue, and at the time message processing begins and ends within each subsystem. Subsystems may make user entries on the system log by calling an Intercomm system program (LOGPUT).

An installation may suppress some or all log entries, depending on its own requirements. The system log is optionally used at Intercomm system restart time to restore message traffic within the system at the time of failure. The logging entries are blocked and written to a variable-length sequential data set which may reside on disk or tape.

Log entries are in one of two formats: HT--42-byte message header and full text, as the message arrives from a terminal and is queued for a subsystem, or queued for a terminal; or HO--header-only entries, to mark progress through the system or error conditions.

Log entries are identified by a code in the MSGHLOG field of the message header. The time and date stamps (MSGHTIM and MSGHDAT) in the message header are updated for each log entry.

Progress of a message through a specific subsystem, or through the Front End, is indicated by the same Monitor Message Number (MSGHMMN) in each log record (01-30-FA or F2-F3). Complete progress of a message, from the first processing subsystem to final transmission, is indicated by the same Front End Message Number (MSGHBMN). The log may be printed completely or selectively via the Intercomm off-line utility LOGPRINT, described in the Operating Reference Manual.

A timing analysis utility (Log Analysis), which is supplied with Intercomm, may be used off-line to produce a report of message queuing and processing time. Statistics for messages by terminal, verb, subsystem, and/or system totals are provided. See the Operating Reference Manual.

The logging entries may be input to user-written batch programs to provide performance analysis in detail, such as traffic vs. network configurations, accounting routines, etc.

Figure 10 illustrates the log entries for one input message and a corresponding output message generated via the Output Utility. Number 6 appears only if executing in Test mode, since there is no Front End.

For live or simulated mode Intercomm, two additional entries are an F2 log code (HT) when the message is queued for the Front End via FESEND (appears in place of the 40 log entry between the 30 and FA entries), and an F3 log code (HO) when the message was transmitted by the Front End. Logging of the message to be transmitted (log code F2) occurs before final Front End processing (idles insertion, New Line to SBA sequence conversion, etc.).

If Message Mapping is used and the message is passed to the Front End via FESEND (Figure 8), only the log entries numbered 1, 2, and 7 appear for each message processing thread. Log codes 3, 4, and 5 represent the additional processing for a message passed to the Output Utility (receiving code U).

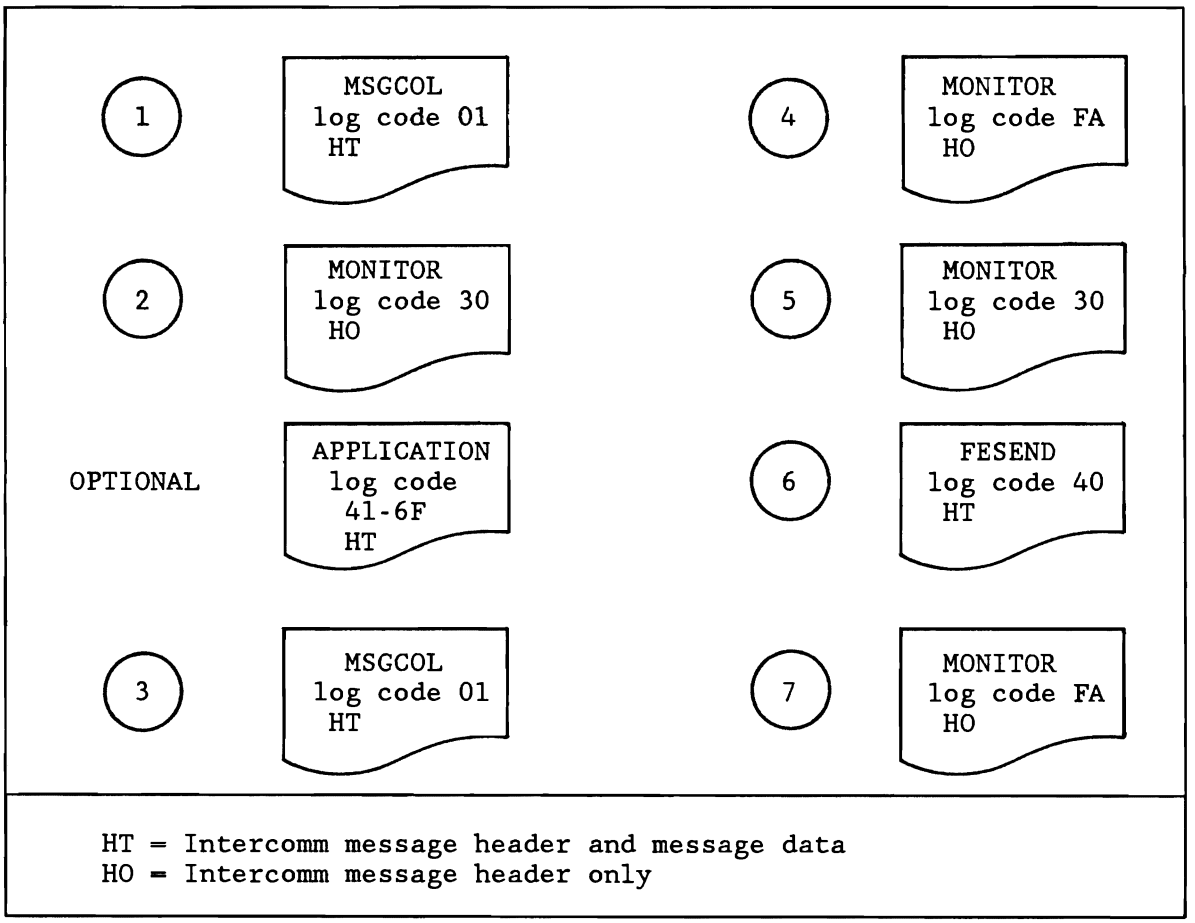


Figure 10. Sequence of Log Entries

Figure 11 describes all the Intercomm log codes. Note that user log entries may only use the codes in the range X'41' to X'6F'.

Internal Code	External Code	Format	Description	Origin	Restart Use
X'00'	00	HT	Checkpoint Record	Checkpoint	Yes
C'2'	01	HT	Message queued for subsystem by Front End or a subsystem	Message Collection	User
C'R'	02	HT	Message restarted through the system	LOGPROC	User
C'P'	03	HT	Message restarted--related to Data Base Recovery	LOGPROC	User
C'T'	30	HO	Message passed to subsystem for processing	Subsystem Controller	User
C'Z'	40	HT	Message passed to Front End (test mode only)	FESEND	No
X'41'-X'6F'	41-6F	HT	User called LOGPUT	Any Subsystem	No
X'80'-X'8E'	80-8E	HT	File Recovery before-images	IXFLOG	User
X'8F	8F	HO	Checkpoint Records indicator	IXFCHKPT	Yes
X'90'-X'9E'	90-9E	HT	File Recovery after-images	IXFLOG	User
X'9F'	9F	HT	Intercomm Startup	LOGPUT	Yes
X'A0'	A0	HO	Message restart begun	LOGPROC	Yes
X'A1'	A1	HO	Message restart finished: all subsequent log entries produced by live Intercomm	LOGPROC	Yes
X'AA'	AA	HT	Intercomm Closedown	LOGPUT	No
X'C0'	C0	HT	Region started (Multiregion only) (Text=Region-id(s))	MRINTER	No
C'A'	C1	HT	Message successfully queued for Satellite Region	MRQMNGR CR only	User

Internal Code: Log code in core during processing (snaps and dumps)
External Code: Log code after translation by LOGPUT (INTERLOG printout)
Format: HT for header and text, HO for header only
Restart Use: Yes, No, User (specified via user-coded system macros)

Figure 11. INTERLOG Entries (Page 1 of 2)

Internal Code	External Code	Format	Description	Origin	Restart Use
C'B'	C2	HO	Message successfully passed to Satellite Region	MRQMNGR CR only	User
C'C'	C3	HO	Message lost (Region/Hold Q full) or flushed (SR/SS down)	MRQMNGR CR only	User
C'I'	C9	HT	Sign on/off processing, security violation messages	ESS	No
C'3'	FA	HO	Normal message complete	Subsystem Controller	User
C'5'	FB	HO	Unprocessed message--invalid subsystem/QPR code	Message Collection	User
C'6'	FC	HO	Unprocessed message--core and disk queue full	Message Collection	User
C'8'	FD	HO	Message cancelled--program error or time-out, I/O error, or flushed by command (Rel 10)	Subsystem Controller	User
C'9'	FE	HO	Message flushed by Retriever, used when application program does not obtain (via GETSEG) all parts of a segmented message; or message failed security check	Retriever SYCT400	No
C'1'	F1	HT	Message after verb verification	USRBTLOG (optional)	No
C'2'	F2	HT	Message queued for transmission	FESEND	User
C'3'	F3	HO	Message transmitted, discarded (MSGHUSR=Z), (Rel 10) or flushed (MSGHUSR=F) (Rel 10)	Front End	User
C'4'	F4	HO	3270 output message content invalid--message dropped.	BLHOT	No
C'5'-C'8'	F5-F6 F7-F8	HO HT	Transmitted DDQ msg status: see SNA Term. Support Gd.	Front End	No
X'FF'	FF	HT	Intercomm Restart Accounting	MSGAC	Yes

Figure 11. INTERLOG Entries (Page 2 of 2)

2.11 ADDITIONAL APPLICATION PROCESSING FACILITIES

In addition to the application programming facilities described in this and related manuals, the application designer should be aware of the following processing options available under Intercomm:

- Off-line batch region execution: the Intercomm File Handler, DFA, DDQ, Store/Fetch and MMU may be executed by an off-line program (coded as non-reentrant) to prepare a file, data strings, or messages for on-line access. See the associated manuals for linkedit considerations.

- Multiregion Facility batch region interface: when executing an on-line Multiregion system, any batch application region may pass a message or a FECMDDQ (see also Chapter 9) to an on-line subsystem or to the Front End via the Output Utility subsystem. See Multiregion Support Facility.

- Time controlled processing: instead of being triggered by an input terminal message, an application may be designed to execute at a particular time of day. See the Operating Reference Manual.

- Segmented input message processing via DDQ: segmented input messages, whether gathered by Intercomm from a remote device (CPU, etc.) or generated by an application program, are placed on a DDQ and may be serially passed to an application subsystem via a DDQ Facility interface. See Dynamic Data Queuing.

- Dynamic linkedit feature: dynamically loaded user subsystems and subroutines are linkedited to called Intercomm resident routines at startup, thus reducing the size of the load modules. The LOAD system control command is used to force a relinkedit of a new version of a dynamically loaded program placed on the load library while Intercomm is executing. See the Operating Reference Manual.

- User exits: various user exits for installation dependent processing are listed in the Operating Reference Manual.

Chapter 3

CODING AN INTERCOMM SUBSYSTEM IN ASSEMBLER LANGUAGE

3.1 PROGRAM STRUCTURE

An application subsystem executing under Intercomm control is activated to process one message. The following examples typify the concerns of message processing logic:

1. Interpretation of message text to reroute administrative data to another terminal.

2. Editing of message text, creation of a record on a sequential data set for later off-line processing and preparation of an acknowledgement message to the originating terminal.

3. Editing and analysis of message text to determine file retrieval and/or update criteria, data file access, preparation of a response message for the operator at the originating terminal.

4. Analysis of an application-oriented control message and appropriate action, such as checking batch totals from example 2, above, or acting on a special request to close a file or perform some other control function.

This chapter presents techniques for coding a BAL application subsystem to execute in the Intercomm region, and to use Intercomm message processing facilities. While some facilities are referenced here, they are fully described in another chapter: check the index for specific routines. To bring all the coding requirements into proper perspective, this chapter includes a sample Intercomm application subsystem. Its objective is to "echo" the text of an incoming message back to the originating terminal.

A BAL application subsystem is coded as a reentrant subroutine, as illustrated in Figure 12. A subsystem's logic is designed to analyze and process one input message, it does not contain logic for terminal I/O operations. Figure 13 depicts the components of a BAL application program environment.

Subsystem Controller activation of the application subsystem is achieved via the equivalent of a CALL macro instruction. On entry to the subsystem, the address of a three-word parameter list, as listed below, is passed via register 1:

- Address (fullword-aligned) of message to be processed, consisting of a 42-byte header and text (edited or unedited)

- Address of the System Parameter Area (SPA), for accessing addresses of Intercomm service routines and user data that may be used for processing the message (see Appendix D)

- Address of the program's Subsystem Control Table (SCT) entry, which allows the subsystem to reference such information as its subsystem code, execution priority, etc.

```
SUBSYSXX CSECT
         REGS
*SUBSYSTEM ENTRY
         LINKAGE   ...
          .
          .
          .
*MESSAGE PROCESSING LOGIC
          .
          .
          .
*GET CORE FOR OUTPUT MESSAGE
         STORAGE   ...
          .
          .          Build Output Message
          .
*PASS THE MESSAGE TO THE FRONT END VIA FESEND, OR
*QUEUE THE OUTPUT MSG VIA MSGCOL
         CALL      ...
          .
          .
          .
*FREE THE INPUT MESSAGE
         STORFREE  ...
          .
          .
          .
*SUBSYSTEM EXIT
         RTNLINK   ...
*
INMSG    DSECT
INHDR    DS        CL42
INTEXT   DS
          .
          .          define input text format
          .
INLEN    EQU       *-INHDR
*
OUTMSG   DSECT
OUTHDR   DS        0CL42
         COPY      MSGHDRC
OUTTEXT  DS
          .
          .          define output text format
          .
OUTLEN   EQU       *-OUTHDR
*
WORKAREA DSECT
REGSAVE  DS        18F
COREADDR DS        F
PARMSAVE DS        5F
SUBSYSWK DS
          .
          .          define subsystem dynamic work area
          .
WORKLEN  EQU       *-REGSAVE
         END
```

Reference description in text

(1) (2) (3) (4) (5) (6) (7) (8) (9)

Figure 12. Reentrant Assembler Subsystem Structure

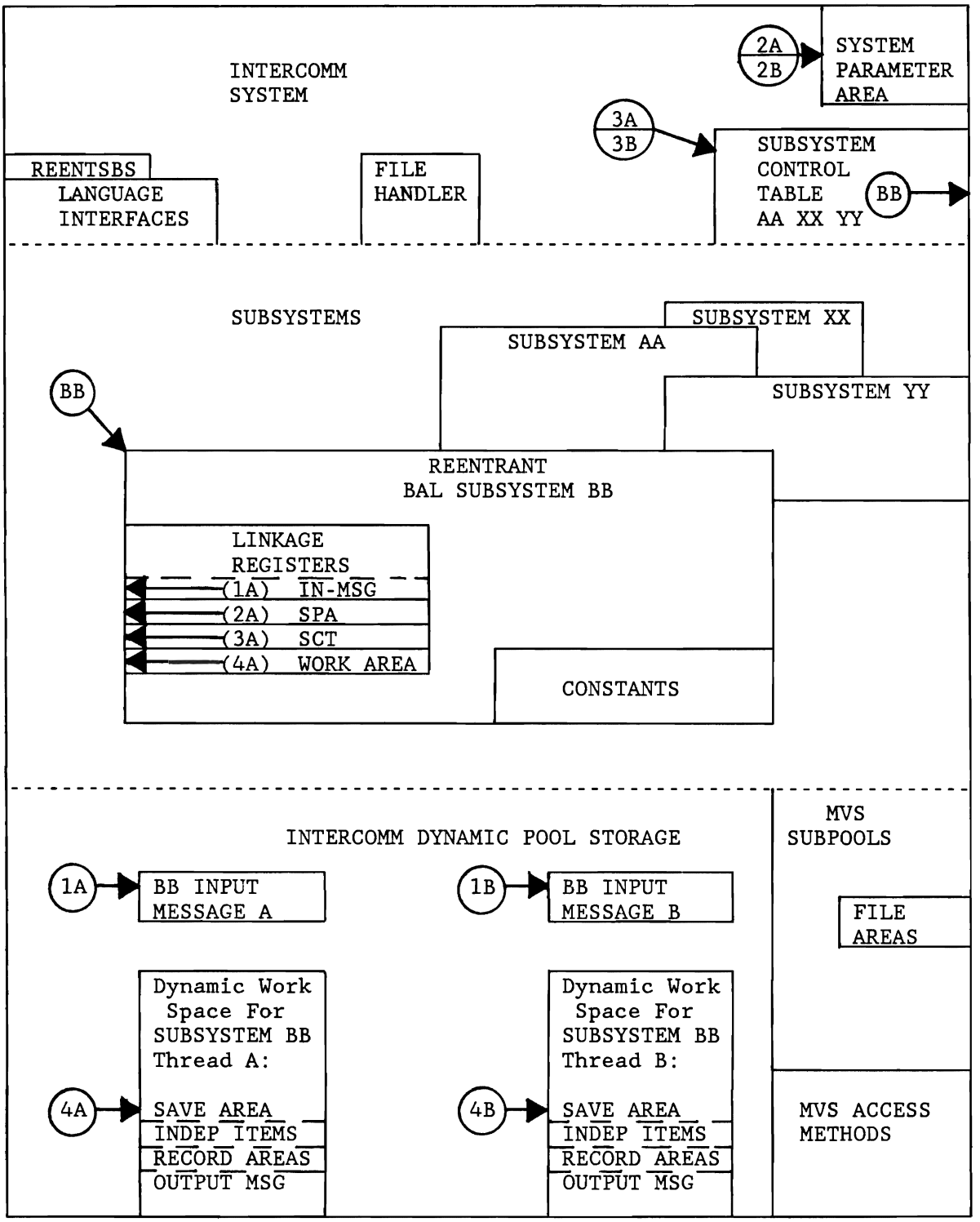


Figure 13. Reentrant Application Program Environment

After a subsystem completes processing and returns control to the Subsystem Controller (see Chapter 2), the Intercomm return code (in Register 15) is checked to determine whether the message should be cancelled due to an error. Then the return code is placed in the externally saved input message header in MSGHRETN (MSGHCON+1), and the header is logged with an appropriate log code (see Chapter 2). Figure 14 describes Intercomm return codes. If the subsystem (or a called subroutine) program checks, or the return code is 8 or 12, USRCANC returns an appropriate error message to the terminal operator. USRCANC is a user exit provided by Intercomm under the name PMICANC, and is described in the Operating Reference Manual.

Return Code	Meaning	Subsystem Controller Error Action
0	Successful completion	None
4	Edit reject (unsuccessful)	None
8	Unrecoverable error condition (no core, MAPEND error, etc.)	Message canceled, CALL to USRCANC
12	I/O error	Message canceled, CALL to USRCANC
16	(Not used, reserved)	---
20-60	User codes to identify unusual condition	None
64	File or DBMS Update Subsystem, no message restart required*	None
68	File or DBMS Inquiry Subsystem, message restart required*	None
72-254	Same as 20-60	None
912	Force Backout-on-the-Fly*	File updates or additions backed out

*See File Recovery Users Guide or Data Base Management System Users Guide

Figure 14. Intercomm System Return Codes

3.2 MESSAGE PROCESSING CONCEPTS

The application program receiving the message may analyze the Verb Message Identifier (MSGHVMI) in the header and/or message text fields to further control message processing logic. The meaning of different VMI values is dependent on the design requirements of the program receiving the message. For example, the Front End sets the VMI to X'00' to indicate to the Assembler subsystem that editing by the Edit Utility is required, based on the specification in the Front End Verb Table for a given verb (BTVERB macro, EDIT parameter). The Assembler subsystem then analyzes the VMI to determine if the Edit Utility should be called. A VMI value of X'FF' (high-values) indicates that no processing is required by, or was performed by, the Edit Utility. Any other value in the VMI indicates that the Edit Utility has already processed the message or that a user subsystem has placed a code in the field before switching (queuing) the message to the currently processing subsystem.

An application subsystem creates an output message by building a 42-byte header and appropriate message text. This new message is either passed to the Front End via FESEND for transmission to the terminal, or is queued for later processing by the Output Utility or some other subsystem by calling the Intercomm system program MSGCOL. The subsystem destined to receive this new message is determined by the receiving subsystem code fields (MSGHRSC, MSGHRSCH) in the message header. The receiving subsystem may then analyze the VMI, as appropriate. The Output Utility, for example, analyzes the VMI to determine whether or not prespecified output message formatting is to be performed. If the output message is passed directly to FESEND, MSGHRSCH and MSGHRSC should be set to binary zeros.

Subsystem logic for input message text analysis and output message text creation varies, depending whether Message Mapping or the Edit and Output Utilities are used. Figures 15 and 16 illustrate subsystem processing logic for these two cases.

It is very important to note that the input message area (Intercomm header and message text) may only be examined (treated as a read-only area) by the application program. It may also be copied to an output message area (header only, or header and text) where it may be added to or changed, depending on program logic. Never add to, or change, the input message text area.

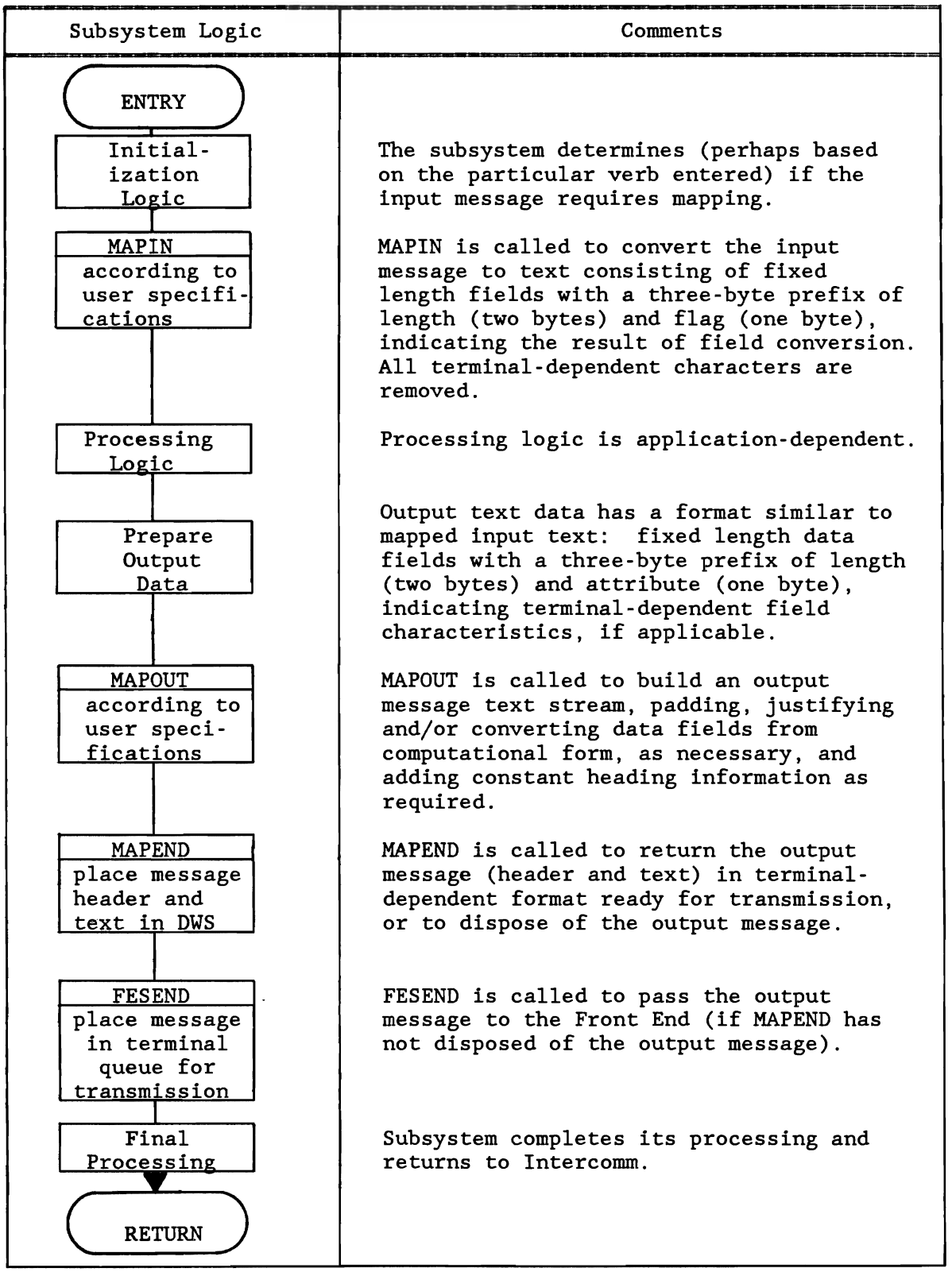


Figure 15. Subsystem Logic Using Message Mapping Utilities

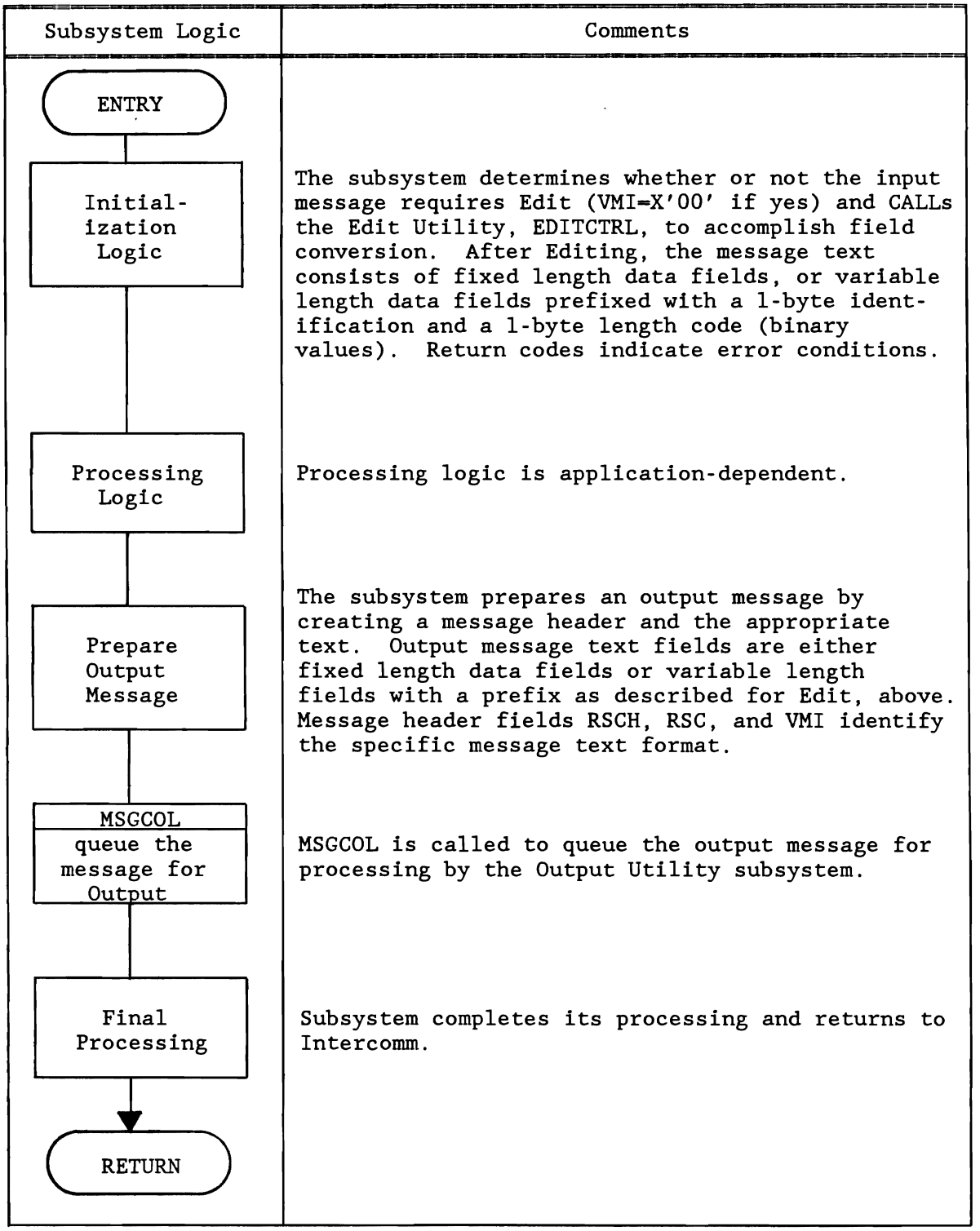


Figure 16. Subsystem Logic Using Edit and Output Utilities (Page 1 of 2)

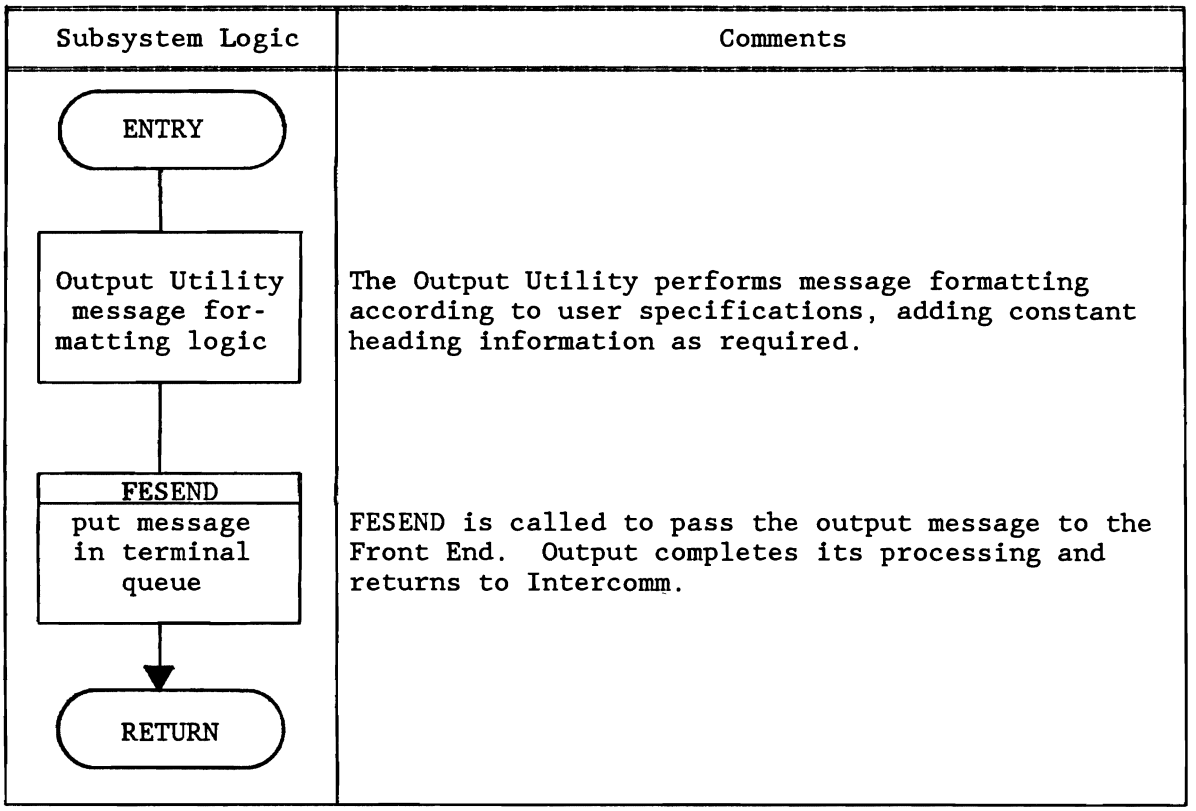


Figure 16. Subsystem Logic Using Edit and Output Utilities (Page 2 of 2)

3.3 SUBSYSTEM CODING

When a message is received in the Intercomm region, it undergoes various preprocessing functions by the system. The Front End prefixes the message with a 42-byte header and the message is queued for processing. The Subsystem Controller, a component of the Intercomm Monitor, will schedule, load (if necessary), and activate the application subsystem for the message. If the subsystem is loaded above the 16meg line under XA, it will receive control in 31 Amode.

At this point, message processing logic begins. At least eight major functions will be included in the subsystem's structure to process a message. These are paired as follows:

- Entry and Exit from the Subsystem

  The subsystem must provide necessary logic to link with Intercomm (that is, initialize registers, acquire and chain a save area, etc.), and to return to the monitor (restore registers, free the save area). The LINKAGE and RTNLINK macros are provided for subsystem linkage processing.

- Edit and Process Message

  The incoming message is often unedited. The subsystem is responsible for all conversion or editing. Message Mapping Utilities or the Edit Utility may be used for this purpose. The message is then analyzed and processed by subsystem logic, using Assembler Language instructions, Intercomm or operating system supplied macros, and/or Intercomm service routines.

- Obtain and Free Core

  The Assembler Language application subsystem is responsible for obtaining storage as needed from the Intercomm dynamic pool area, and then freeing the storage when processing is completed (if necessary). The Intercomm STORAGE and STORFREE macros are provided to obtain and free core.

- Build and Queue the Output Message

  As part of the message processing logic, a subsystem may create one or more messages to transmit to a terminal. Message Mapping Utilities may be used in conjunction with the FESEND routine, or the Page or Dynamic Data Queuing Facilities, to create and queue the output message(s) for transmission. Or, the output message(s) may be created and queued for processing by the Output Utility via a call to MSGCOL. The Output Utility may be used to format the message(s), and subsequently pass the message(s) to FESEND for transmission.

The subsystem consists of two areas: the actual message processing logic; and the definition of areas of core in the Intercomm dynamic pool storage area. Figure 12 illustrates the structural flow of an application subsystem. The circled numbers are included to facilitate reference in the following text discussion.

3.3.1 Subsystem Entry

Entry into the subsystem is identified by the CSECT name or an ENTRY name. The address of the three-word parameter list passed on entry by the Subsystem Controller is in register 1. The input message resides in the dynamic pool area. The message format must be defined within the application program by a DSECT (see (7) in Figure 12). Parameters 2 (SPA) and 3 (SCT-entry) addresses are resident areas in the Intercomm load module, but a DSECT must be generated if these areas are to be referenced. The LINKAGE macro (see (1) in Figure 12) will provide these Dsects (as described in the next section), or the programmer may generate the Dsects as part of the subsystem structure. Intercomm Dsects for use by Assembler Language application programs, and methods for generating them (macro or COPY statement) are described in Appendix B.

3.3.2 Linkage

To allow concurrent processing of messages within a subsystem, the subsystem must be coded in a reentrant form. The LINKAGE and RTNLINK macros generate much of the code necessary to make an Assembler Language subsystem reentrant. The LINKAGE macro generates all of the instructions necessary to establish standard reenterable linkage from the Subsystem Controller into the application subsystem. LINKAGE provides addressability, and can also perform a number of other service functions for the application subsystem:

- Provide a set of register equates
- Issue various USING statements
- Set up one or two base registers
- Set up specified registers with addresses passed in the parameter list
- Obtain a dynamic save/work area from the dynamic pool area and zero the core obtained
- Provide the PARMLIST, SPALIST, SCTLIST, MSGHDR, R13 DSECTs, as desired. (See also (8) and (9) in Figure 12.)

The LINKAGE macro must be the first executable instruction in the application subsystem structure. The coding requirements to ensure reentrancy are:

1) Code the LINKAGE macro at the main entry point of the subsystem (see (1) in Figure 12).
2) Code the RTNLINK macro (see (6) in Figure 12) to return control to the calling program (Subsystem Controller).
3) Do not modify any area within the program. Only modify dynamic storage obtained by using the LINKAGE and STORAGE macros (see (3) in Figure 12).
4) Use the list and execute forms of any IBM operating system macros and Intercomm macros (when applicable). Intercomm processing macros are listed in Chapter 10.

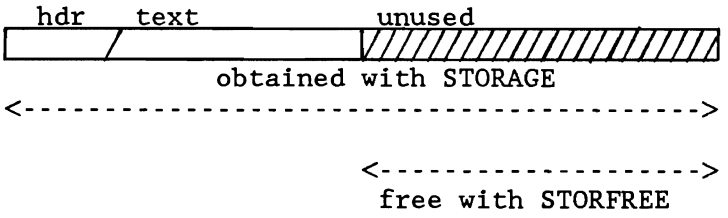
3.3.3 Message Processing

The next section of the subsystem structure contains the message processing logic as indicated by (2) in Figure 12. Further considerations for message editing and text formats are described in Chapter 2. If a file is to be accessed as part of the processing of the message, a File Handler service routine must be called to read from or write to a file (described in detail in Chapter 6). See other chapters in this manual for detailed descriptions of service routine calls for other processing facilities, and Chapter 10 for a list of available macros.

Once the processing has been performed, an area of dynamic core may be needed to construct the output message. The Intercomm STORAGE macro must be used to obtain dynamic core (see (3) in Figure 12). The STORAGE macro instruction issues a request for ownership of a portion of core. If the request is satisfied, register 15 will contain a return code of 0 and the core allocated will commence on a double word boundary. If the request for dynamic core is not satisfied, register 15 will contain a return code of 8. Core obtained via the STORAGE macro must be released through the use of the STORFREE macro instruction, unless that core is utilized for a message queued for another subsystem, or the Front End.

Once dynamic core has been obtained, the subsystem can construct the message for output, consisting of a 42-byte header prefix, plus application-oriented message text. This new message is passed directly to the Front End by calling the system program FESEND, or is queued for later processing by the Output Utility or some other subsystem by calling the Intercomm system program MSGCOL, indicated by (4) in Figure 12. The subsystem destined to receive this new message is determined by the receiving subsystem code fields (RSC, RSCH) in the message header. The Receiving Subsystem may then analyze the Verb/Message Identifier (VMI), as appropriate. The Output Utility, for example, analyzes the VMI to determine whether or not prespecified formatting is to be performed.

If the output message text is shorter than the actual area obtained, the STORFREE macro is used to free the extra area. This is accomplished by calculating the difference between the length of the area obtained and the length (contained in the message header) of the actual output message, as illustrated below:




```
        LA          R14,OUTLEN          Total LEN acquired less
        SH          R14,MSGHLEN         actual LEN used=unused LEN
        SRL         R14,3               Drop down to nearest
        SLL         R14,3                     multiple of 8
        LTR         R0,R14              R0=unused length
        BZ          NOFREE              If less than 8, nothing to free
        LR          R1,R7               Pick up addr of unused core by
        AH          R1,MSGHLEN          bumping to end of used portion
        ROUND       R1                  Up to next multiple of 8
        STORFREE    ADDR=(1),LEN=(0)    Free unused core
NOFREE  DS          0H
```

As indicated by (5) in Figure 12, the subsystem must use the STORFREE macro to free the input message area, unless that area has been freed by a call to MAPIN, or has been referenced in a call to Message Collection (see "Additional Coding Techniques" below). The system routines MSGCOL and FESEND take over "ownership" of the passed message area.

The RTNLINK macro instruction is coded in coordination with the LINKAGE macro instruction. It restores the registers saved at LINKAGE time and optionally frees storage acquired by LINKAGE. In addition, it effects the return to the Subsystem Controller and passes a return code (see Figure 14) in register 15. See (6) on Figure 12.

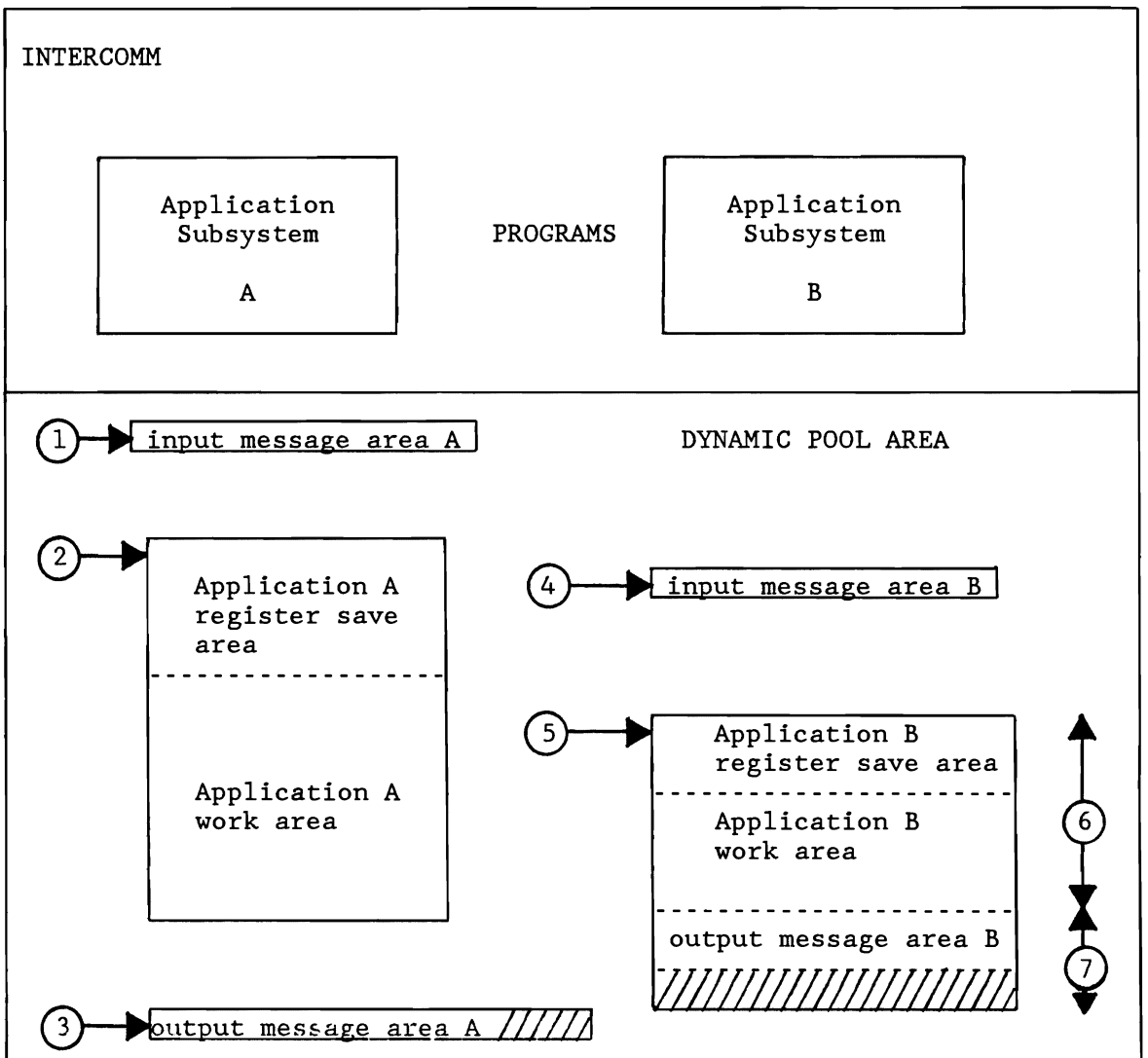
Note that the SPA parameter of the RTNLINK macro must be coded if the register containing the address of the SPA differs at the time of RTNLINK from the register used at the time the LINKAGE macro is issued or if the subsystem is dynamically loaded.

3.3.4 Additional Coding Techniques

To minimize the amount of dynamic subpool space utilized by an application subsystem, the programmer may optionally use the input message area as the output message area as text processing logic allows. However, if using this method, the output message must never be greater in length than the input message. This approach is well suited (although not limited) to fixed-length message text. If the output message is shorter than the input message, additional program logic is required to free the remaining area of the input message not utilized as output message text. Remember that STORFREE operates on doubleword boundary alignment. Also, the new shorter message length must be stored in the message header before calling MSGCOL or FESEND.

A second technique for obtaining core for an output message which minimizes dynamic pool area fragmentation (but adds internal processing overhead) is to allow the LINKAGE macro to obtain space for an output message as well as other work areas. In this instance, the output message area must appear at the "trailing end" of the obtained area. RTNLINK must be coded to free up all of the save/work area except the output message area, which is controlled by Message Collection (or FESEND). However, the subsystem must use STORFREE for any trailing area of core not occupied by the output message.

The following illustrates the areas of dynamic core operated upon by two different subsystems and the associated responsibilities for obtaining and freeing core.



1. area obtained by Monitor, freed by subsystem A
2. area obtained by LINKAGE, freed by RTNLINK
3. area obtained by STORAGE, passed to MSGCOL or FESEND, area not used for message freed by STORFREE (shaded area)
4. area obtained by Monitor, freed by subsystem B
5. area obtained by LINKAGE
6. area freed by RTNLINK
7. passed to MSGCOL, area not used freed by STORFREE (shaded area)

3.3.5 Subsystem Illustration

Figure 17 illustrates the basic coding required to implement an Intercomm subsystem and the definition of an input message and creation of an output message via an application to "echo" the text of an incoming message back to the originating terminal. The Message Mapping Utilities, or the Edit Utility and the formatting capabilities of the Output Utility, are not used.

1. The message header is created by copying the input message header to the output message header area and adjusting the following fields:

   - MSGHSSCH, MSGHSSC--Sending Subsystem Code

     Move the original receiving subsystem code values, MSGHRSCH (to MSGHSSCH) and MSGHRSC (to MSGHSSC), to identify the current subsystem as the sending subsystem.

   - MSGHRSCH, MSGHRSC--Receiving Subsystem Code

     Move in a predefined code to indicate further processing (the next subsystem) for this message (for FESEND, use binary zeros).

   - MSGHVMI--Verb/Message Identifier

     Move in a predefined code to indicate the output message is not fully formatted: X'57'. If an output message is formatted by MMU, do not touch this field.

   - MSGHLEN--Message Length

     Modify to total header and text length of output message (if different from length of input message).

   - MSGHTID--Receiving Terminal Name

     If the originating terminal is to receive the response message, do not change. Otherwise, specify the receiving terminal name for the output message.

   To assist the programmer in defining the message header, there is a source library member, MSGHDRC, that may be copied into the appropriate DSECT area in the source code. See Chapter 2 for a description of individual fields in the message header.

2. The new message text is created by copying the input message text to the output text area.

3. Queuing of the output message for the terminal is accomplished via the service routine FESEND.

4. The return code from the queuing routine must be analyzed to assure that the new message was actually queued, and recovery action taken if not.

5. The last logical activity in the subsystem is to STORFREE the input message and then issue the RTNLINK macro, returning control to the Subsystem Controller and passing back an appropriate return code.

The program Csect or Entry Point name must correspond to the subsystem entry point described in the Subsystem Control Table. If dynamically loadable, the load module name should be the same as the Csect name. The entry parameters for the System Parameter Area (SPA) and Subsystem Control Table (SCT) entry for the subsystem are not detailed as there is no need to reference any of their individual fields.

The input and output message formats are described via Dsects, with the message header detailed for the output message area. Constants and an LTORG statement are usually defined between the last code instructions and the Dsects area. Areas of storage modified during program execution must be defined within the dynamic save/work area Dsect. Such items also include storage areas required for Intercomm service routines and passed to those routines as parameters, whether or not the subsystem references or modifies those areas. For programs eligible for loading above the 16meg line under MVS/XA or ESA, unmodified constant values (map names, file ddnames, etc.) must be copied to the save/work area for passing as parameter values to called Intercomm routines. Additionally, areas passed as parameters to user subroutines must also be defined in the save/work area.

```
RBALECHO CSECT
* REENTRANT BAL SUBSYSTEM TO 'ECHO' A MESSAGE,
* ILLUSTRATING BAL SUBSYSTEM STRUCTURE.
* MESSAGE PROCESSING LOGIC CONSISTS ONLY OF CREATING AND QUEUING
* AN OUTPUT MESSAGE TO RETURN TO THE ORIGINATING TERMINAL
         USING INMSG,R5
         USING OUTMSG,R6
         USING WORKAREA,R13
         REGS                              GENERATE REGISTER EQUATES
* SUBSYSTEM ENTRY
         LINKAGE  BASE=(R12),LEN=WORKLEN,PARM=(R2),SPA=(R3),MSG=(R5)
* GET CORE FOR THE OUTPUT MESSAGE AREA
         LH    R8,INMSG                    INPUT MESSAGE LENGTH
         STORAGE  ADDR=COREADDR,LEN=(R8),LIST=PARMSAVE
         LTR   R15,R15                     TEST RETURN CODE
         BZ    GOTCORE
         LA    R10,8                       RETURN CODE 8 IF NO CORE
         B     FREEIN
GOTCORE  L     R6,COREADDR                 ESTABLISH ADDRESSABILITY
* BUILD OUTPUT MESSAGE
         MVC   OUTHDR,INHDR                INPUT HEADER TO OUTPUT HEADER
         MVC   MSHGSSCH,MSGHRSCH           THIS SUBSYSTEM BECOMES THE
         MVC   MSGHSSC,MSGHRSC                  SENDING SUBSYSTEM
         MVI   MSGHRSCH,X'00'              THERE IS NO
         MVI   MSGHRSC,X'00'                    RECEIVING SUBSYSTEM
         MVI   MSGHVMI,X'57'               VMI FOR PREFORMATTED OUTPUT TEXT
         LA    R9,42
         SR    R8,R9                       TEXT LENGTH=INPUT LENGTH-42
         EXMVE OUTTEXT,INTEXT,R,R8         MOVE TEXT TO OUTPUT MESSAGE AREA
         AR    R8,R9                       RESTORE TOTAL MSG LENGTH
* QUEUE THE MESSAGE FOR THE INPUT TERMINAL
         CALL  FESEND,(OUTMSG),VL,MF=(E,PARMSAVE)
         LTR   R15,R15                     TEST RETURN CODE
         BZ    QUEUED
         LA    R10,12                      RETURN CODE 12 IF NOT QUEUED
         B     FREEIN
QUEUED   LA    R10,0                       RETURN CODE 0 IF ALLS WELL
* FREE THE INPUT MESSAGE
FREEIN   STORFREE  ADDR=(R5),LEN=(R8)
* RETURN TO SUBSYSTEM CONTROLLER
         RTNLINK  ADDR=(R13),LEN=WORKLEN,RC=(R10)
*
WORKAREA DSECT                    LINKAGE WORKAREA,FREED BY RTNLINK
REGSAVE  DS    18F                        REGISTER SAVE AREA
COREADDR DS    F                          STORAGE MACRO, ADDR OF CORE
PARMSAVE DS    5F                         PARAMETER LIST SAVE AREA
WORKLEN  EQU   *-WORKAREA
INMSG    DSECT                            INPUT MESSAGE
INHDR    DS    CL42                            HEADER
INTEXT   DS    CL200                           200 CHARACTER TEXT MAXIMUM
OUTMSG   DSECT                            OUTPUT MESSAGE
OUTHDR   DS    0CL42                           HEADER AREA
         COPY  MSGHDRC                         HEADER FIELDS DEFINITION
OUTTEXT  DS    CL200                           200 CHARACTER TEXT MAXIMUM
         END
```

Figure 17. Echo Message Example; Reentrant Assembler Language

3.3.6 Message Switching Between Subsystems

Any Intercomm subsystem may send a message to any other Intercomm subsystem. If a message is sent to some other subsystem, it is called "message switching." An application subsystem can switch a message to the Output Utility, which is another subsystem. The Change/Display Utility switches messages to the Output Utility. An application subsystem may switch (or requeue) a message to itself in the event that reprocessing or deferred processing of the message is required. An application subsystem may exceed an installation's core limitations and be broken into several subsystems. One subsystem may receive a message input from a terminal, perform partial processing and develop intermediate results in the form of a message sent to a second subsystem. The second subsystem processes the intermediate results as an input message and may complete the message processing or develop additional intermediate results in the form of messages sent or switched to any other subsystem or subsystems. Any one of these subsystems might also switch messages to the Output Utility.

Message switching between subsystems is accomplished by moving the input message to an output message area and then changing the receiving subsystem codes in the header and calling MSGCOL as usual. The Verb/Message Identifier (MSGHVMI) may be initialized for interpretation by the receiving subsystem. A VMI equal to X'00' indicates that the Edit Utility is to be called by the subsystem. To switch messages between terminals, the destination terminal identifier (MSGHTID) would also have to be changed, and the VMI set to X'57'.

3.4 RESTARTED MESSAGES

After an Intercomm system failure (abend or operator cancel) or an operating system failure (requiring a re-IPL of the CPU), Intercomm may be brought up in Restart Mode which permits reprocessing of messages in progress at the time of failure. Additionally, previously cancelled messages (see Figure 14), and unprocessed messages (received and queued, but not started) will be requeued for processing after system startup completes. This is accomplished by retrieving the original input messages from the log created in the previous Intercomm execution as described in the Operating Reference Manual, and may be coordinated with file or database record backout as described in the File Recovery Users Guide and DBMS Users Guide.

Restarting of messages for a particular subsystem is controlled by the RESTART parameter of the SYCTTBL macro defining the subsystem in the SCT. A restarted input message (in progress at failure time) contains a log code of C'R' or C'P' (if data base update may be executed by the subsystem). All other input messages contain a log code of C'2' (see Figure 11). A subsystem may need a different processing path for a restarted message and should be careful about creating an output response message which might confuse a terminal operator.

3.5 MVS/XA EXTENDED STORAGE LOADING REQUIREMENTS

If the user desires that an Assembler Language subsystem or subroutine be dynamically loaded above the 16meg line under XA or ESA, the following is required:

- Programs must be reassembled to ensure that the Release 10 XA support versions of macros are used.

- Programs must be coded and defined to Intercomm as reentrant and use the linkedit AMODE and RMODE override parameters (see Appendix A).

- Subsystems and subroutines must be linked with INTLOAD to provide 24-Amode interface to Intercomm. INTLOAD is serially reusable (not reentrant), therefore do not link with the RENT attribute.

- SYCTTBL macro: code LOADNAM (not SBSP), LANG=RBAL, BLDL=YES (default), and REUSE=YES (default) parameters.

- LINKAGE and RTNLINK macros must be used by subsystems: dynamic save/work area acquired in 24-Amode.

- SUBLINK and RTNLINK macros must be used in dynamically loadable subroutines: dynamic save/work area acquired in 24-Amode.

- If Intercomm service routines (MMU, File Handler, MSGCOL, etc.) are called, they may only be accessed via the IBM CALL macro using the service routine entry point name (which causes a branch to the INTLOAD interface routine which handles mode switching on entry and return). The address of the routine may not be preloaded from the SPA or SPAEXT.

- Intercomm macros may be used (except the SUBTASK and CALLOVLY macros), however the SPA and SPAEXT parameters may not be coded, nor may the LINK parameter be used. INTLOAD also contains entry points for macro processing (for STORAGE, STORFREE, DISPATCH, etc.). The SPA and SPAEXT parameters may only be coded on the LINKAGE and SUBLINK macros. The LIST parameter must be used on the STORAGE macro (RENT=NO may not be used), and the referenced list must be in 24-Amode storage (dynamic save/work area).

- All variable (modifiable) fields and unmodified parameter values (ddnames, map names, etc.) passed to Intercomm service routines or macros (which branch to Intercomm service routines) must be in 24-Amode storage (acquired via LINKAGE, SUBLINK or STORAGE macros). That is, ddnames, enqueue names, etc. defined in the program as constants (including the module name used for MODCNTRL) must be copied to dynamic storage before the service routine call or macro execution.

- Dynamically loadable user subroutines must be defined to Intercomm via the SUBMODS macro in the REENTSBS table: code LNAME (not NAME), TYPE=BAL (default), BLDL=YES (default), and USAGE=REENT (default) parameters.

- Resident (in Intercomm load module) user subroutines, to be called by a program loaded above the 16meg line, must also be defined in the REENTSBS table via a SUBMODS macro: code NAME parameter only (and USAGE if not reentrant).

- Subsystems (and subroutines) must use the MODCNTRL macro to access user subroutines (tables). ACTION=LINK should be used for subroutine access so that the Intercomm MODCNTRL interface program will handle mode-switching when necessary for a loaded subroutine. Parameters passed to a subroutine must be in dynamic 24-Amode storage (in case the subroutine is resident or loaded in 24-Amode).

- If a MODCNTRL macro with ACTION=LOAD is used, the address of the loaded module is returned in register 1 when the issuing program receives control back from MODCNTRL. If the hi-order bit (80) is on in the address (address is negative), the module was loaded above the 16meg line. A program executing above the 16meg line may use (branch to) the loaded module directly (modes are compatible). A program executing in 24-Amode must use the XASWITCH macro to switch modes before processing the loaded module and again at the end of processing before using the MODCNTRL macro with ACTION=DELETE. If the address of the loaded module is not negative (loaded or resident in 24-Amode), a program executing in 31-Amode must immediately execute a MODCNTRL macro with ACTION=DELETE and then use a MODCNTRL macro with ACTION=LINK, to access the 24-Amode subroutine. A 24-Amode table, however, may be processed by a 31-Amode program.

- The 24-Amode interface routine SWMODE must be included in the Intercomm linkedit.

- Check the linkedit map of the program to be loaded above the 16meg line for unresolved external references that may be incorrectly resolved (causing Intercomm to be entered in 31-Amode) by dynamic linkedit processing, if used. References to the SPA (by a hard-coded Vcon) in the SUBLINK, LINKAGE and RTNLINK macro expansions and by INTLOAD may be ignored, as well as references to the BITSECT table by the SSSTART, SSSTOP and SSTEST macros.

- Ensure that the following Intercomm interface modules were assembled under MVS/XA: SYCT400, DYNLLOAD, MANAGER, INTLOAD, SWMODE.

Chapter 4

USING THE MESSAGE MAPPING UTILITIES

4.1 CONCEPTS

The Message Mapping Utilities (MMU) provide an interface between the application subsystem and terminal-dependent message processing logic for both input and output messages. MMU is invoked by calls to Intercomm service routines which perform mapping functions based upon user-specified tables (MAPs). Mapping includes justification, padding, and conversion of character data to computational format and vice versa.

4.2 PROCESSING

MMU input mapping produces fixed length data fields prefixed by a two-byte length and one-byte flag (indicates errors or omissions) unless the data fields are defined in a structured (named) segment (contiguous group of fields). In this case the three-byte prefix occurs for the entire segment, not the individual fields.

MMU output mapping operates upon data in the same format, but the flag byte becomes the field (or segment) attribute character. The mapped input text area and the unmapped output text area are called symbolic maps and are defined by COPY statements in the application program's save/work area. The application program references data fields and the associated prefix by symbolic name. For example, a customer name field (CUSTMER) of twenty-five characters would appear in an MMU symbolic definition as follows:

```
CUSTMERL   DS   XL2            (length)
CUSTMERT   DS   X              (flag/attribute)
CUSTMER    DS   CL25           (data)
```

Output message disposition is determined by options passed to MMU. The formatted message(s) may be returned to the subsystem; passed to FESEND for terminal queuing; passed to the Page Facility for CRT page browsing; or spooled to a DDQ for subsequent transmission as a series of report pages for a printer.

A summary of message processing logic using MMU is shown in Figure 18. For a complete description of Message Mapping and its use by application subsystems, refer to the Intercomm Message Mapping Utilities.

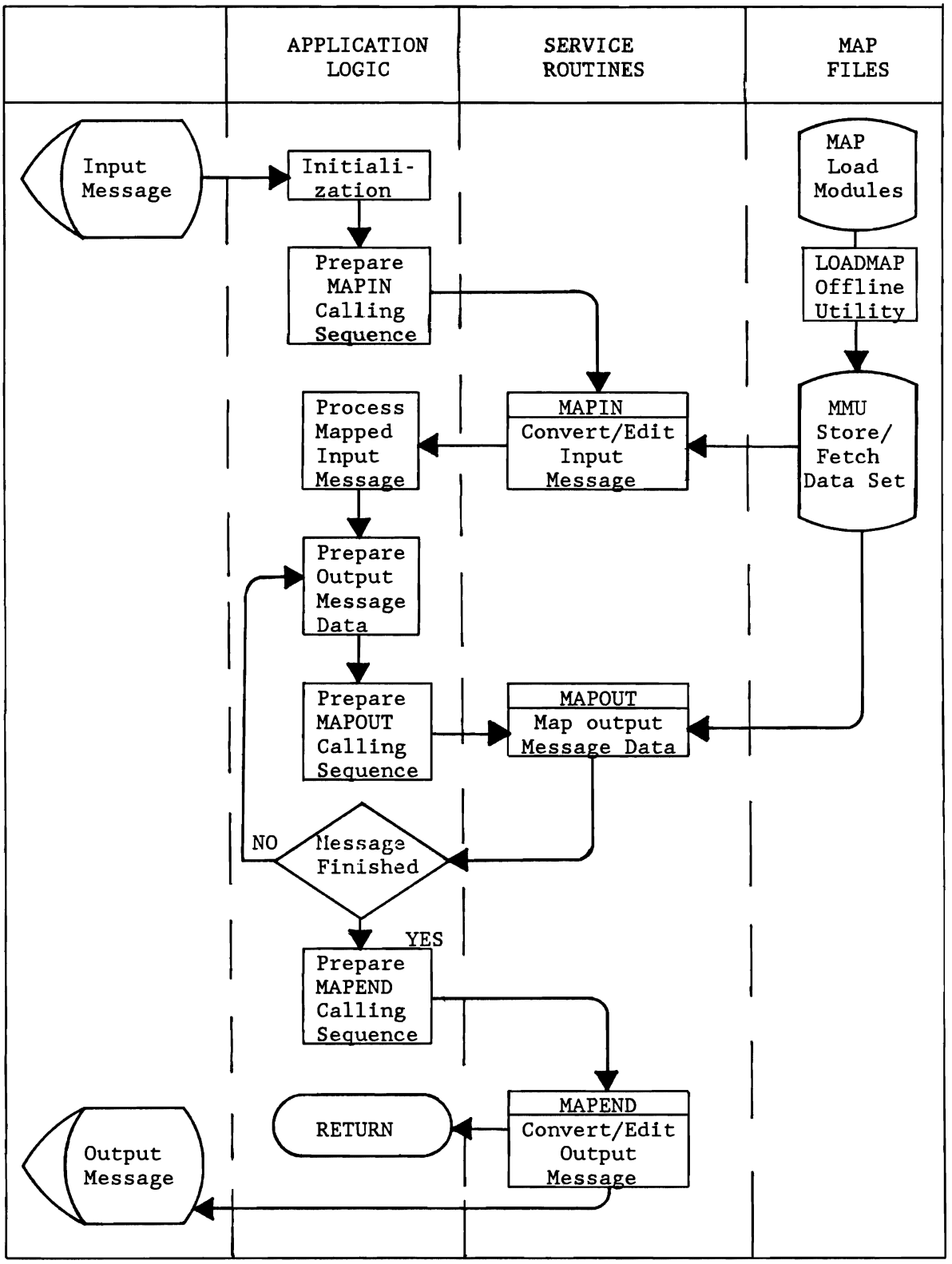


Figure 18. Message Processing Using MMU

Chapter 5

USING THE EDIT UTILITY

5.1 CONCEPTS

The Edit Utility may be used for input messages instead of MMU. It provides an interface to facilitate application program logic for message editing. When pre-editing has been requested for a verb (via Front End Verb Table specification), Intercomm calls the Edit Utility to produce edited message text from data fields entered by the terminal operator, before queuing the message for the subsystem. Otherwise, the BAL subsystem must call the Edit Utility to produce the edited message text. Coding format:

```
[symbol] CALL  EDITCTRL,(input-message,spa,0),VL,MF=(E,list)
```

where:

- input-message is the address of the unedited message.
- spa is the address of the System Parameter Area.
- 0 reserves a third word in the parameter list (used by Edit).

On return from the Edit Utility, register 15 contains a binary return code indicating the results of editing. Zeros indicate the message was edited successfully. The address of the successfully edited message is in the first word of the parameter list passed to Edit. For a nonzero return code, a zero address also indicates the input message was not successfully edited (original message freed). Program logic for editing an input message:

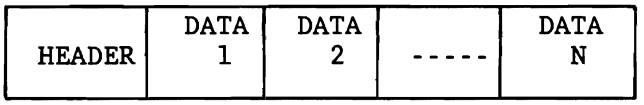
```
          LINKAGE - - -,MSG=(R5),SPA=(R3),- - -
TEST      CLI   MSGHVMI,X'00'       EDIT REQUIRED?
          BNE   OKAY                NO
          CALL  EDITCTRL,((R5),(R3),0),VL,MF=(E,LIST)
          LTR   R15,R15
          BZ    GOOD
          RTNLINK - - -,RC=4        UNSUCCESSFUL
GOOD      L     R5,LIST             EDITED MESSAGE ADDRESS
OKAY      EQU   *
```

The edited message becomes the input message processed by the subsystem. During the course of editing, the Edit Control routine strips field delimiter characters such as the system separator character (defined in the SPA), 3270 CRT SBA sequences, TAB characters, New Line characters, Carriage Return or combined Carriage Return/Line Feed, End of Text, End of Message, etc. All other device control characters not translated or otherwise suppressed by the Front End translation table for a particular device will be treated as text within a field. Editing is controlled by the Edit Control Table (ECT), which contains all information about each message necessary to perform editing. An edit proceeds field by field based upon the user-specified ECT. Data fields may be edited by Intercomm or user-coded Edit subroutines. For a complete description of the Edit Utility, its components and return codes, refer to the Utilities Users Guide.

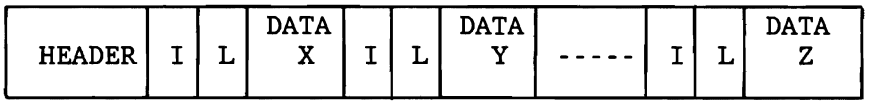
5.2 PROCESSING RESULTS

The result of processing by EDIT is a message with a standard forty-two-byte message header and data fields in one of the following basic formats:

- Fixed Format--each edited field is of fixed length in a predefined sequence as follows:



- Variable Format--each edited field may vary in length and position in the edited result. The PMIFINDB service routine (see Chapter 9) may be used to locate specific fields. Each edited field is prefixed with a one-byte identification code, one-byte length, and possibly a one-byte occurrence number for fields defined as repetitive in the ECT:



The Edit Utility considers a message successfully edited if there are no required fields (as specified by the Edit Control Table) in error or omitted. In the case of unsuccessful editing, Edit sends an error message to the originating terminal for each required field omitted or in error. If none of the required fields are omitted or in error, it remains the responsibility of the application program to analyze the edited result and perform recovery logic for any non-required fields in error. Figure 19 summarizes results of Edit processing for fields in error.

Field Type	Fixed Format	Variable Format
Non-Required Field Omitted	Field appears in edited result, filled with pad character associated with Edit Subroutine, that is, spaces for alphanumeric field, zero for numeric field, or user-assigned.	Field does not appear in edited result.
Non-Required Field in Error	Field appears in edited result filled with high-values (X'FF').	Field does not appear in edited result.
Required Field in Error or Omitted	Message rejected by EDIT.	Message rejected by EDIT.

Figure 19. Edit Utility Processing of Fields Omitted or in Error

Chapter 6

USING THE FILE HANDLER

6.1 GENERAL CONCEPTS

The Intercomm File Handler provides centralized control over all data file access in the on-line system. Requests for data file access are made in message processing subsystems by calling a File Handler service routine.

The correspondence between the normal MVS Data Management Macro Instructions and the Intercomm File Handler service routines is shown in Figure 20.

Function	BAL Macros	Service Routine
Prepare a file for access	OPEN	SELECT
Access logical records sequentially (QSAM,QISAM)	GET,PUT,PUTX	GET,PUT
Access logical records randomly (BISAM,BDAM)	READ,WRITE	READ,WRITE
Access physical blocks (BSAM,BDAM)	READ,WRITE	READ,WRITE
Access VSAM files	GET PUT	GETV PUTV
Conclude file access	CLOSE	RELEASE

Figure 20. Functions of File Handler Service Routines

A data file on-line is identified to the File Handler by the existence of a data definition (DD) statement in the execution JCL. Files must be existing (DISP=OLD or SHR) except for sequential output data sets (DISP=NEW or MOD).

DD statement requirements are illustrated in Figure 21. Additional requirements for VSAM are described in that section. Special processing definitions for particular files are defined to Intercomm at system startup by FAR (File Attribute Record) parameters. These include READONLY (prohibit output), OPEN (at startup), file duplexing, etc., and are described in the Operating Reference Manual. Additional parameters for file recovery (in case of program or system failure) are described in the File Recovery Users Guide.

```
//ddname*   DD  DSNAME=**
//                ,DISP=**
//                ,DCB=(DSORG=**
//                ,OPTCD=**              For BSAM,BDAM,BISAM only.
//                ,RECFM=                Must be specified by existing
//                ,BLKSIZE=              data set label or explicitly
//                ,LRECL=                in DD statement.
//                ,NCP=
//                ,LIMCT=
//                 etc.)
```

*Name used to identify file in calls to SELECT.
**Marks those parameters which must be explicitly specified on the DD statement for each data set.

Figure 21. DD Statement Parameters for the File Handler.

In centralizing data file accesses, the File Handler provides one central set of control blocks for each file, thus reducing core requirements in individual message processing subsystems.

Furthermore, all the facilities of the following Operating System Data Management functions are accessible to any subsystem: BDAM, BSAM, QSAM, BISAM, QISAM and VSAM.

The File Handler also supports the following ISAM replacement access method available from another vendor: IAM.

Data Base interfaces supported under Intercomm (IDMS, ADABAS, TOTAL, DL/I, Model 204, System 2K) are described in the DBMS Users Guide and the respective vendors' manuals.

6.1.1 Subsystem Processing

In the on-line environment, several subsystems in concurrent execution may require access to the same data file. Rather than each subsystem issuing an OPEN and corresponding CLOSE for accessing a particular file, the File Handler will open a file the first time it is accessed (unless already opened at startup) and the file remains open for the duration of the on-line job in execution. A SELECT request simply establishes internal control blocks and the corresponding RELEASE request merely disconnects those internal control blocks. In each subsystem, following a SELECT for a particular file, access functions (READ, WRITE, GET, PUT, GETV, PUTV) may be called as many times as may be necessary for message processing logic. RELEASE must be called for each selected file prior to the RTNLINK to the System Monitor.

Each subsystem must provide space for two File Handler control areas. The information in these areas is unique for each message thread, so they must be defined in the dynamic save/work area of reentrant programs.

For each call to a File Handler service routine, the File Handler is passed the addresses of the two control areas, as illustrated in Figure 22. The first is a full word aligned 12-word (48 bytes) area, called an External DSCT (EXTDSCT), which the File Handler uses to save control information for the subsystem processing thread, from the time that a given file is first SELECTed until it is finally RELEASEd. A unique EXTDSCT must be defined for each file concurrently accessed within the same processing thread and should be cleared to zeros before calling SELECT. The other control field, called the File Handler Control Word (FHCW), is an aligned full word field used for communication between the File Handler and the calling subsystem. Prior to each call to a service routine, the subsystem must clear the FHCW with blanks or initialize it with a predefined request code as described for each routine. A code of space (blank) is indicated in the detailed access descriptions by the lower case letter ⱥ. An example of such a request would be to establish Exclusive Control during a call to READ with intent to update. The File Handler will return a completion code in this word, after servicing a request, to communicate the status of the operation back to the subsystem.

```
*FILE HANDLER CONTROL AREAS
*
FILEAREA    DS     12F              EXTERNAL DSCT
FHCW        DS     0F               FILE HANDLER CONTROL WORD
FHSTATUS    DS     CL1              STATUS-BYTE
FHREQ       DS     CL1              REQUEST-BYTE
FHREST      DS     CL2              UNUSED (except for VSAM)
            .                       I/O AREA DEFINITION might
            .                       follow
```

Figure 22. Defining File Handler Control Areas

6.2 CALLING SERVICE ROUTINES

A reentrant Assembler Language subsystem calls a File Handler service routine using the following format:

```
[symbol]   CALL   function,(parameters),VL,MF=(E,list)
```

where:

- function is the specified File Handler routine being accessed, such as SELECT, READ, etc.

- parameters are the parameters passed to the File Handler for each specific routine
- VL indicates a variable-length parameter list, as illustrated in the description for each File Handler function
- MF=(E,list) indicates the executable form of the macro instruction with the parameter list saved at the location labeled 'list'. 'list' must be defined in the dynamic storage area unique to each processing thread in order to maintain reentrancy.

The parameters for the File Handler service routines are described in Figure 23. The specific parameters passed to a given service routine depend on file requirements and the processing options of the particular service routine called. If the calling subsystem (or subroutine) might be loaded above the 16meg line (under XA or ESA), then all parameters must be in the dynamic save/work area (or other area acquired in 24 Amode).

Parameter	Content
EXTDSCTname	A 48-character fullword-aligned area supplied by the subsystem for the File Handler's use for each file SELECTed
FHCWname	The 4-byte File Handler Control Word, in which the File Handler returns a completion code to the subsystem (see Figure 22)
ddname	An eight-character constant containing the name of the DD statement describing the data set to Intercomm
Record-area	The area for data read from, or to be written to, the file
Key	The key for file access (ISAM, Keyed BDAM, VSAM-KSDS)
VSAM RBA	Four-byte Relative Byte Address number (ESDS)
VSAM RRN	Four-byte Relative Record Number (RRDS)
Block-ID	Applies only to BDAM files: • three-byte relative block number (RBN) • three-byte relative track and record number (TTR) • eight-byte actual address (MBBCCHHR)

Figure 23. File Handler Service Routine Parameters

The File Handler IAM support uses the Intercomm ISAM support routines.

On return from a File Handler service routine, the leftmost position of the FHCW area will contain a character indicating the result of the operation, as shown in Figure 24. Additionally, for VSAM files, the rightmost position of the FHCW will contain a VSAM reason code.

Code	Meaning
0	Normal completion
1	Hardware I/O error
2	Unusual condition (EOF, invalid key, etc.)
3	Exclusive control time-out occurred
4-8	Not used
9	Invalid request (no DD statement, invalid parameter sequence, attempt to output to an input only file, etc.)

Figure 24. Outline of File Handler Return Codes

The application subsystem logic must then analyze this return code and take appropriate error recovery action. An error message might be created and queued for output to the terminal. Otherwise, the subsystem can return to the Subsystem Controller with a return code of 12, indicating that the Subsystem Controller should call the USRCANC routine which in turn will send an error message to the terminal.

6.2.1 Automatic Error Checking

If the application subsystem logic is such that special error recovery processing is not required, the File Handler will perform error checking itself and data will be returned to the subsystem only if the return code is zero. Otherwise, the File Handler will force a program check, which causes cancelling of the input message and return to the Subsystem Controller, which calls the USRCANC routine. To request this function, place a character 'C' in the first byte of the FHCW prior to calling a File Handler service routine.

6.3 SELECT, RELEASE FUNCTIONS

SELECT must be called to initialize the subsystem's EXTDSCT prior to any data access function performed by the File Handler. Prior to the call to SELECT, the subsystem's EXTDSCT must be initialized to binary zeros.

RELEASE must be called to notify the File Handler that its pointers to the subsystem's EXTDSCT should be cleared and that all data access to a particular file within one subsystem thread is complete. There must be a RELEASE corresponding to each SELECT of a file. Multiple SELECTs of the same file using the same EXTDSCT are not permitted without intervening RELEASEs, within the same processing thread. After each RELEASE, the EXTDSCT should be cleared to zeros before being reused.

Coding format:

```
[symbol] CALL SELECT,(EXTDSCTname,FHCWname,ddname),VL,MF=(E,list)

[symbol] CALL RELEASE,(EXTDSCTname,FHCWname),VL,MF=(E,list)
```

Note: the ddname must be in the dynamic save/work area if the calling subsystem (subroutine) can be loaded above the 16meg line under XA or ESA.

Figure 25 describes the return codes for SELECT and RELEASE.

Return Codes (First Byte of FHCW)	SELECT	RELEASE
0	A reusable file (disk input) ready for access; sequential access begins at first record.	Successful release
1	A nonreusable file (SYSOUT, disk output (DISP=NEW/MOD or DISP=SHR/OLD and FAR WRITEOVER parm specified, or a data set on tape) ready for access, begins after last record previously accessed.	Not applicable
9	No ddname found in File Handler internal control table. (No DD statement in JCL or the file has been "locked" by the FILE control command.)	File not selected.

Figure 25. File Handler SELECT/RELEASE Return Codes

6.3.1 Closing a File

Occasionally, it is necessary to close a file, perhaps because it is to be updated by a batch job. A special form of RELEASE requests the File Handler to close a file. However, unless some external control is taken to assure that no other programs have selected the file, a close request could cause other transactions for the file to fail. Also, if new transactions are attempting to access the closed file, the File Handler will open it again and unpredictable results may occur. Intercomm provides the FILE system control command for systemwide file access control.

To close a file from an application subsystem:

- If the file has been previously selected: first release the EXTDSCT by calling RELEASE referencing the EXTDSCTname used when the file was selected (as described above), then

- Move a character C to the second byte of the FHCW ('ØCØØ') and call RELEASE supplying the ddname of the file to be closed; use the following coding format:

```
[symbol] CALL  RELEASE,(ddname,FHCWname),VL,MF=(E,list)
```

6.4 EXCLUSIVE CONTROL FOR NON-VSAM FILES

In a multithread environment with only inquiry applications, the fact that several message processing programs may concurrently retrieve data from the same file or files presents no operational problems. However, when more than one message processing program attempts to update or add records to a file, data integrity problems can occur. Figure 26 illustrates the problems of concurrent updates; program B's update nullifies that of program A. Exclusive control implies that while one program is operating on a record, that is, the time between a READ and a WRITE, all other requests to read or write that particular record will be delayed. A program requesting a record held during exclusive control by another program is not notified of this delay, but rather stops execution in the File Handler until exclusive control is either removed or expires so that the File Handler can then proceed with the requested function. Exclusive control, when required, must be requested separately with each call to File Handler READ or GET functions. Exclusive control for basic access methods operates at the block or record level. Exclusive control for queued access methods operates at the data set level; thus applications should be designed to avoid GET for update whenever feasible.

To obtain exclusive control over the entire data set in a QISAM file or over a physical block in a BDAM or BISAM file, move 'ØXØØ' to the File Handler Control Word prior to calling GET or READ. Exclusive control does not apply to physical sequential (QSAM/BSAM) files.

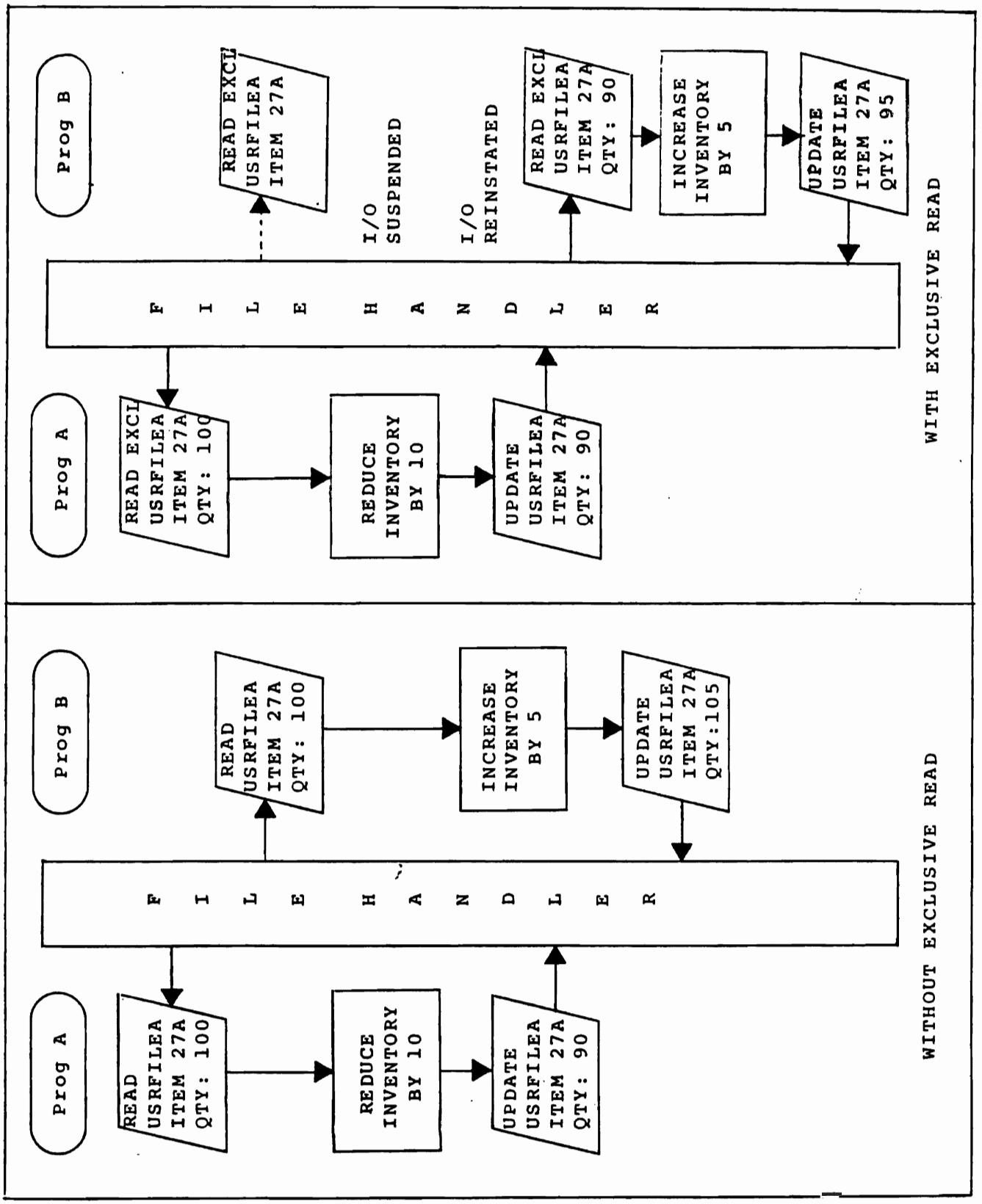


Figure 26. Exclusive Control Processing

Exclusive control will be released by:

- A call to WRITE or PUT referencing the same EXTDSCTname, that is, the update of the previously acquired record, and no key or block-id specified.
- A call to WRITE referencing the same EXTDSCTname and a key and/or block-id is specified.
- A call to READ or GET referencing the same EXTDSCTname (retrieving a new record from the file).
- A call to RELEASE referencing the same EXTDSCTname.
- An elapsed time after the call to READ with Exclusive Control greater than the exclusive control time-out value of the File Handler. This is set at two minutes for any given record and a maximum of ten minutes for consecutive exclusive accesses to a QISAM file.

  NOTE: A return code of 3 after a call to WRITE or PUT to update a record held in exclusive control indicates that exclusive control timed out: the WRITE or PUT did not take place. The program should re-READ or re-GET the same record with exclusive control and WRITE or PUT again.

- A call to RELEX, if the program logic is such that the record does not need to be updated, or additional and time-consuming activity (accessing other files) is required before resuming access to the file. Such a program could call RELEX to release exclusive control without actually RELEASEing the file until later in the program logic.

6.4.1 Release Exclusive Control--RELEX

RELEX is called to release Intercomm or VSAM exclusive control without having to read, update, time-out, or RELEASE the file.

Coding format:

```
[symbol] CALL  RELEX,(EXTDSCTname,FHCWname),VL,MF=(E,list)
```

Return Code	Meaning
0	Exclusive control released
9	File not selected or invalid function

Figure 27. File Handler Release Exclusive Control (RELEX) Return Codes

6.5 SEQUENTIAL ACCESS METHOD PROCESSING

6.5.1 File Handler Service Routines--GET, PUT (QSAM); READ, WRITE (BSAM)

GET is called to access the next sequential logical record from a file. PUT is called to write the next sequential logical record to a file. READ is called to access the next sequential physical block. WRITE is called to write the next sequential physical block. If PUT or WRITE is called referencing a disk data set, the record last accessed by a GET or READ will be updated, however, the length may not be changed. GET processing is subtasked by the File Handler in order to provide multithreading facilities; for further details, see the Operating Reference Manual.

Coding format:

```
[symbol] CALL  GET,(EXTDSCTname,FHCWname,record-area
               [,record-length]),VL,MF=(E,list)

[symbol] CALL  PUT,(EXTDSCTname,FHCWname,record-area
               [,record-length]),VL,MF=(E,list)

[symbol] CALL  READ,(EXTDSCTname,FHCWname,record-area
               [,record-length]),VL,MF=(E,list)

[symbol] CALL  WRITE,(EXTDSCTname,FHCWname,record-area
               [,record-length]),VL,MF=(E,list)
```

Return Codes	GET, READ	PUT, WRITE
0	Successful	Successful
1	I/O Error	I/O Error
2	End-of-file	(Not applicable)*
9	Not selected or invalid function; that is, using an output-only file	Not selected or invalid function; that is, using a tape input file or readonly file, or file not sequential.
* For WRITE to a disk file: indicates End-of-file (write not done)		

Figure 28. File Handler Sequential Access Method Return Codes

6.5.2 Undefined Record Format and Record Length

The record-length parameter is valid and required only when a file with an undefined record format (DCB=RECFM=U) is accessed. The record-length parameter points to a fullword containing the length of the output record before a PUT or WRITE operation, or to contain the length of the input record after a GET or READ operation. The second character of the File Handler Control Word must be set to U to utilize this feature. Do not code the DCB subparameter LRECL on the DD statement for the file in the Intercomm execution JCL. The BLKSIZE, RECFM and DSORG subparameters are required.

6.5.3 Variable-Length Record Format and Record Length

Variable-length records start with a Record Descriptor Word (RDW) which must be fullword aligned. The first two bytes of the word contain the record length in binary (+4 for the RDW); the second two bytes contain binary zeros (low values). The RDW is followed immediately by the record data, and must be recognized by the subsystem on input, and provided and initialized on output.

For blocked files, if GET or PUT are used, the access method will perform the blocking and deblocking. If READ or WRITE are used, the application program must perform the deblocking (READ) and blocking (WRITE). In this case, the block must start with a Block Descriptor Word (BDW) of four bytes (aligned); the first two bytes contain, in binary, the total block length (including 4 for the BDW), and the second two bytes contain binary zeros (low values). For JCL details, and FAR options for defining and accessing the file, see the Operating Reference Manual.

6.6 INDEXED SEQUENTIAL ACCESS METHOD PROCESSING

To use an ISAM file on-line under Intercomm, do not define three DD statements (INDEX/PRIME/OVERFLOW) for either the off-line creation of the ISAM data set, or the on-line execution DD statement. For creation, let the access method set up the index and overflow areas (use CYLOFL parameter on DD statement). For on-line execution, define only DISP=OLD and the data set name, volser and unit parameters if not catalogued, and the DCB parameter DSORG=IS. Optionally, the DCB parameter OPTCD may also be specified. See also the descriptions of FAR parameters applicable to ISAM data sets described in the Operating Reference Manual.

6.6.1 File Handler Service Routines--GET, PUT (QISAM); READ, WRITE (BISAM)

GET is called to access the next sequential record, or to reposition (if a key is specified) and access the next sequential record. READ is called to retrieve a specific record at random. PUT is called to update the last record retrieved by a call to GET. WRITE is called to update the last record retrieved by a call to READ, or to add a record to the file (if a key is specified). For update, exclusive control may be requested; otherwise use blanks in the FHCW.

Coding format:

to retrieve next sequential record:

```
[symbol]  CALL  GET,(EXTDSCTname,FHCWname,record-area),
                VL,MF=(E,list)
```

to reposition and retrieve record with key equal or high:

```
[symbol]  CALL  GET,(EXTDSCTname,FHCWname,record-area,key),
                VL,MF=(E,list)
```

to update last GET:

```
[symbol]  CALL  PUT,(EXTDSCTname,FHCWname,record-area),
                VL,MF=(E,list)
```

to retrieve a specific record:

```
[symbol]  CALL  READ,(EXTDSCTname,FHCWname,record-area,key),
                VL,MF=(E,list)
```

to update last READ:

```
[symbol]  CALL  WRITE,(EXTDSCTname,FHCWname,record-area),
                VL,MF=(E,list)
```

to add a specific record:

```
[symbol]  CALL  WRITE,(EXTDSCTname,FHCWname,record-area,key),
                VL,MF=(E,list)
```

Figure 29 describes return codes for ISAM access.

QISAM Return Codes	GET w/o Key	GET w/Key	PUT
0	Next sequential record retrieved	Record with equal or next higher key retrieved	Record from previous GET updated
1	I/O error	I/O error	I/O error
2	End of File	Key out of range	N/A
3	N/A	N/A	Exclusive Control Time-out
9	File not selected or invalid function	File not selected or invalid function	File not selected or invalid function

BISAM Return Codes	WRITE w/o Key	WRITE w/Key	READ
0	Record from previous READ updated	Record with specified key added	Record with equal key retrieved
1	I/O error	I/O error	I/O error
2	N/A	Key already exists or no room to add new record	Key does not exist
3	Exclusive Control Time-out	N/A	N/A
9	File not selected or invalid function	File not selected or invalid function	File not selected or invalid function

Figure 29. File Handler ISAM Return Codes

6.7 DIRECT ACCESS METHOD PROCESSING

BDAM files are accessed by block-id. The form of the block-id is defined in the OPTCD subparameter of the DCB parameter of the DD statement and the same form must be used by all programs accessing the file:

- OPTCD=RF--block-id is three-byte binary RBN (relative block number) for fixed-length files only
- OPTCD=AF--block-id is eight-byte actual MBBCCHHR
- OPTCD=F--block-id is three-byte binary TTR (relative track and record number) for fixed- or variable-length files.

The F permits feedback (of block-id) requests: the form of the block-id is that requested by the OPTCD parameter. For Keyed BDAM with extended search, insert an E immediately after the = sign (that is, code OPTCD=ERF, etc.), and specify the LIMCT subparameter on the DCB parameter of the DD statement.

6.7.1 File Handler Service Routines--READ, WRITE (BDAM)

READ is called to retrieve a physical block. WRITE is called to update a block previously read, to replace an existing block in a preformatted file, or to add a new block.

Coding format:

```
[symbol]  CALL  READ,(EXTDSCTname,FHCWname,record-area[,key],
                block-id),VL,MF=(E,list)

[symbol]  CALL  WRITE,(EXTDSCTname,FHCWname,record-area[,key]
                [,block-id]),VL,MF=(E,list)
```

Figure 30 shows FHCW options (byte 2) for standard and keyed BDAM files, and when to use key and/or block-id fields. Figure 31 describes the corresponding return codes. When reading a keyed BDAM file, the key will be read into the key field if a key parameter is passed and the key is not used as the search argument (w/o extended search). For a keyed BDAM file, replace requires a previous read; update and replace are synonymous.

Intercomm provides two utilities for off-line preformatting of fixed-length BDAM files:

- CREATEGF for BDAM files without keys
- KEYCREAT for BDAM files with keys.

These utilities are described in the Operating Reference Manual.

1. BDAM Files Without Keys

Code	Request	Macro
∅	READ w/o exclusive control, w/block-id	READ DIF
X	READ w/exclusive control, w/block-id	READ DIX
∅	WRITE to update last READ, w/o block-id	WRITE DI/DIX
∅	WRITE to update/replace w/o previous READ, w/block-id	WRITE DI
A	WRITE to add a record--variable-length only (record address returned automatically in caller's block-id field)	WRITE DAF

2. BDAM Files With Keys

Code	Request	Macro
*∅	READ data block only w/o exclusive control (w/extended search) w/key, w/block-id	READ DKF
*X	READ data only w/exclusive control (w/extended search) w/key, w/block-id	READ DKX
J	READ key and data block w/o exclusive control w/o extended search, w/block-id (w/key)	READ DIF
I	READ key and data w/exclusive control w/o extended search, w/block-id (w/key)	READ DIX
*∅	WRITE to update data only w/o extended search w/key	WRITE DKF/DKX
I	WRITE to update key and data w/o extended search, w/key	WRITE DI/DIX
*A	WRITE to add a record--next available space w/key, w/block-id (w/extended search)	WRITE DAF

*Feedback of record addresses may be requested for these options only by placing an F in byte 3 of the FHCW.

Figure 30. File Handler BDAM Option Codes.

NOTE: The DI form of the macros (issued in the File Handler) requires that the block-id field contains the exact address of the data record in the form specified by the OPTCD subparameter on the DD statement. With the DK form, if

extended search is not specified (via E on the OPTCD subparameter), only one track is searched for a record with key matching that passed in the key field, and starting at the address specified in the block-id field. A WRITE for update of last READ does not need a block-id, as positioning is remembered internally.

1. BDAM Files Without Keys

Return Codes	READ	WRITE w/o block-id	WRITE w/block-id
0	Block retrieved	Block from previous READ updated	Specified block added/replaced
1	I/O error	I/O error	I/O error
2	Block out of range	N/A	RECFM=F... Block out of range RECFM=V... No space available/ block out of range
3	N/A	Exclusive Control Time-Out	N/A
9	File not selected or invalid function	File not selected or invalid function	File not selected or invalid function

2. BDAM Files With Keys

Return Codes	READ	WRITE w/o block-id	WRITE w/block-id
0	Logical record retrieved	Record from previous READ updated	Specified record added
1	I/O error	I/O error	I/O error
2	Key not found (READ w/key)	Key not found at block-id saved from previous READ (WRITE DK only)	RECFM=F... No dummy record found RECFM=V... No space available
3	N/A	Exclusive Control Time-Out	N/A
9	File not selected or invalid function	File not selected or invalid function	File not selected or invalid function

Figure 31. File Handler BDAM Return Codes

6.8 VIRTUAL STORAGE ACCESS METHOD (VSAM) PROCESSING

VSAM support is provided for all three file types: KSDS, ESDS, and RRDS. Subsystems designed to access VSAM files use two File Handler service routines; GETV and PUTV. SELECT and RELEASE function for VSAM as they do for OS data sets. Calls are similar to the standard File Handler format, with the File Handler Control Word (FHCW) used to specify VSAM options. DD statements for VSAM must specify AMP=(AMORG) and for fixed-length data records, 'RECFM=F' must also be specified on the AMP parameter: AMP=(AMORG,'RECFM=F'). FAR options and execution options for VSAM files such as LSR buffer pool support, empty ESDS file load or overwrite, and data set name sharing, are described in the Operating Reference Manual. Most users converting ISAM to VSAM can continue to use their current File Handler calls. Refer to "ISAM/VSAM Compatibility under Intercomm" later in this chapter for further details.

6.8.1 File Handler Service Routines--GETV, PUTV (VSAM)

A VSAM call may request either sequential or direct access and may specify access for KSDS via keys (keyed access) or for ESDS via Relative Byte Addresses (addressed access). A keyed access call for direct retrieval may provide either a generic key or a full key, and may specify a search for either an equal (generic) key or for the first greater-or-equal (generic) key.

A VSAM Relative Record Number Data Set (RRDS) may be accessed sequentially, or directly by Relative Record Number. A direct access request to a RRDS is made by suppling the Relative Record Number of the desired record instead of a key or RBA. All direct accesses to an RRDS must specify "full key, search equal." RBA access is not allowed and RRNs should not be converted to RBAs for access to an RRDS. Records may be inserted into emply slots in an RRDS but a record may not be added with a higher relative record number than the maximum allowed. This maximum is specified when the data set is defined to VSAM.

GETV calls are processed assuming that no update will be performed unless the caller so specifies. The caller may switch back and forth from direct to sequential access, provided VSAM rules are not violated, for example, keyed request against an entry-sequenced data set. The File Handler service routine GETV is called for retrieval. The File Handler service routine PUTV is called for storage or deletion.

Coding formats:

For sequential access

```
[symbol] CALL  GETV,(EXTDSCTname,FHCWname,record-area),
               VL,MF=(E,list)
```

Coding formats (continued):

For direct access

```
[symbol] CALL  GETV,(EXTDSCTname,FHCWname,record-area,{rba}),
               VL,MF=(E,list)                          {key}
                                                       {rrn}
```

For update of record retrieved by preceding GETV or for sequential addition

```
[symbol] CALL  PUTV,(EXTDSCTname,FHCWname,record-area),
               VL,MF=(E,list)
```

For direct addition of a new record

```
[symbol] CALL  PUTV,(EXTDSCTname,FHCWname,record-area,{rba}),
               VL,MF=(E,list)                          {key}
                                                       {rrn}
```

where:

EXTDSCTname is the standard File Handler parameter.

FHCWname is the standard File Handler parameter. Its VSAM use is to define processing options and to return completion codes to the caller (see Figures 32 and 33).

record-area is the label of the user's I/O area. For fixed length records, no length is specified and data will start in the beginning of the area. For variable length records, the first four bytes of the area are used as an OS-type, fullword-aligned, variable record descriptor word (RDW), the first two bytes of which specify the appropriate length in binary (data length +4); data begins in the fifth byte. For GETV, the File Handler will return this length to the caller and for PUTV, the caller must provide the length to the File Handler.

rba is the label of an aligned fullword containing the Relative Byte Address when required for addressed access.

key is the label of a field providing a key, when required for keyed access. If a generic key is provided, then the first two bytes of this field must be the length, in binary, of the generic key which must begin in byte 3, and the field must be fullword-aligned.

rrn is the address of a fullword-aligned field providing a four-byte binary Relative Record Number whose value is 1 to n, where n is the maximum record number defined for the data set.

6.8.2 VSAM Processing Options

The following determine the mode of VSAM access to be performed:

- The preceding call

  A VSAM call is dependent upon the preceding call only in two cases: PUTV for update, or sequential GETV or PUTV calls requiring initial positioning.

  In the first case, the PUTV call must be immediately preceded by a GETV for update, which identifies the record to be updated. The PUTV for update has no fourth parameter because the key, RRN or RBA was defined by the prior GETV. In the second case, a direct call providing a key, RRN or RBA and requesting positioning must be issued in order to process sequentially starting from that point in the file. To request positioning in this manner, specify S in the second byte of the FHCW for the direct call to GETV; the first record in the sequence will be returned. For an ESDS file, a GETV call without a fourth parameter results in sequential reads from the beginning of the file; the S in the FHCW is unnecessary.

- The presence or absence of the fourth parameter

  With the exception of a PUTV for update, all calls for direct access specify a fourth parameter and all subsequent calls for sequential access specify only three parameters.

- The contents of the File Handler Control Word

  The second and third bytes of the FHCW are used to complete the definition of the options desired. Alphabetic codes are used and positive tests are made for each defined code. When no defined code is present, the default option (blank) is used.

Bytes 1 and 2 of the FHCW are utilized the same as for OS Access Methods for Return Codes (Byte 1) and Special Requests (Byte 2). The first byte of the FHCW will contain a zoned decimal digit upon return from GETV or PUTV. A nonzero value indicates an error or an exceptional condition.

Byte 2 is used in conjunction with direct access. When an S is provided in byte 2, the direct access is treated as the first of a series of sequential requests which begins at a point specified by the fourth parameter. Therefore, a VSAM POINT will be issued and sequential access will subsequently be performed for the next call.

Byte 3 is used for all VSAM calls as illustrated in Figure 32. There are five default (blank) cases:

- GETV with three parameters (subsequent sequential access)
- GETV with four parameters (search key/RRN equal, no update)
- PUTV with three parameters with no prior GETV for update (sequential add/insert)
- PUTV with three parameters and with a prior GETV for update
- PUTV with four parameters (direct key/RRN add/insert)

6.8.3 FHCW Reason Codes for VSAM

Byte 4 is used to provide VSAM reason codes (from the RPL feedback field) upon completion of a VSAM file access request. In VSAM, a distinction is made between logical and physical errors. In either case VSAM returns a supplementary reason code in hexadecimal defining the condition more precisely. Accordingly, the File Handler will return this reason code in FHCW byte 4, for the caller's use. If the File Handler was called at an ISAM entry point (GET/PUT, READ/WRITE), the code returned in FHCW byte 1 may differ from GETV/PUTV calls (in order to maintain compatibility with existing ISAM subsystems). Figure 33 summarizes VSAM and ISAM/VSAM return codes. VSAM reason codes are fully documented in IBM's VSAM Administration Guide.

6.8.4 Exclusive Control for VSAM Files

VSAM automatically provides exclusive control of a control interval (physical block) whenever a GETV for update is processed if the file was defined with SHAREOPTION 1 or 2. The subsystem must release this exclusive control via a call to RELEX before another GETV is issued for the same file, unless an intervening PUTV for update or erase is issued. If no subsequent GETV will be issued, the call to RELEASE will also release exclusive control. There is no VSAM exclusive control time-out. If the VSAM file is accessed by more than one region (Intercomm and/or batch), see IBM documentation on VSAM SHAREOPTIONs, and the Intercomm Operating Reference Manual.

6.8.5 Alternate Path Processing of Keyed VSAM Files

Base Cluster and Alternate Path processing of keyed VSAM files is supported with the following (VSAM-imposed) restrictions:

- If defined in the JCL, the DD statement for the base cluster must be before those for any related paths, and open at startup must be requested via a FAR. Also, both the base cluster and the paths must be connected to an LSR buffer pool.

- Each path to be accessed on-line must be defined in the JCL and be SELECTed with the corresponding ddname. When created, the path must be defined with the UPDATE option.

- The FAR READONLY option must be specified for all paths and the base cluster (if defined) except for the path used for updating, when Shareoption 2 is in effect for the base cluster. If updating is only via the base cluster, then READONLY must be specified for all associated paths. VSAM will not allow any accesses to a base cluster under Shareoption 1 when one path has opened it for update. A base cluster under Shareoption 3 may be accessed for reads or updates by more than one path at any time, however no exclusive control (read/write file integrity) is provided by either VSAM or Intercomm. For Intercomm-provided exclusive control for Shareoption 4, see the Operating Reference Manual.

- If multiple paths are accessed, and/or retrieval/update is done via the path(s) and the base cluster, retrieval of updated versions of the records can be ensured via the FAR DSN and LSR parameters.

- Since duplicate keys may occur in an Alternate Index, the application program is responsible for checking for duplicate keys. Sequential processing (GETV type 1) can be used after the first GETV with key (and an S in byte 2 of the FHCW) in order to retrieve subsequent records. The program can test to see if the last record under a duplicate key was retrieved by checking the VSAM reason code which will be placed in byte 4 of the FHCW. See the IBM VSAM Administration Guide for reason code values.

- The alternate index data set must be defined with the UPGRADE attribute and be built prior to Intercomm startup. An attempt to retrieve a record from an empty file will cause a program check.

- Alternate index data sets should not be defined in the JCL unless access to a data record containing the prime keys is desired, or path processing is not used. Only readonly processing should be done for an AIX and for any related paths and for the base cluster, otherwise, retrieval of the current version of a record is unpredictable.

Type	Service Routine	Access or Action	FHCW Byte 3		KEY/RRN or RBA	Comments
			Update	No Update		
1	GETV	Sequential	U	default	---	In KEY or RRN sequence
2	GETV	Sequential	A	R	---	In RBA sequence (default for ESDS)
3	GETV	Direct	U	default	Full Key or RRN	Search =
4	GETV	Direct	L	F	Full Key	Search greater or = (not valid for RRDS)
5	GETV	Direct	=	E	Generic Key	Search = (not valid for RRDS)
6	GETV	Direct	>	G	Generic Key	Search greater or = (not valid for RRDS)
7	GETV	Direct	A	R	RBA	Addressed Access
8	PUTV	Sequential Add or Insert	default		---	No prior GETV for update (insert not allowed for Addressed Access)
9	PUTV	Update	default		---	Prior GETV for update (addressed update may not change length)
10	PUTV	Erase	E		---	Prior GETV for update (not permitted for addressed access)
11	PUTV	Direct Add or Insert	default		Key or RRN	(no prior GETV)
12	PUTV	Add	A		RBA	Insert not valid

Figure 32. File Handler VSAM Call Summary

Condition at Completion of Operation*	FHCW Byte 1 (char) VSAM	FHCW Byte 1 (char) ISAM	FHCW Byte 4 (hexadecimal)
Successful completion (A)	0	0	04,08,0C,10,1C
Physical I/O error (A)	1	1	04,08,0C,10,14,18
End of data (1, 2)	2	2	04
No record found (3, 4, 5, 6, 7)	2	2	10
Key not within defined key ranges (3, 4, 5, 6, 7)	2	1	24
Duplicate key (8, 11)	9	2	08
Key out of ascending sequence (8)	9	2	0C
Update attempt with new key (9)	9	9	60
Key exceeds maximum (5, 6)	9	**	70
Addressed update changes length (9)	9	**	64
Invalid RBA provided (7, 12)	9	**	20
Required positioning not performed (1, 2, 8)	9	**	58
Direct or update call while loading (8) GETV for ESDS while loading (2,7)	9	9	74
Insufficient disk space (8, 9, 11, 12)	9	9	1C
Record on unmountable volume (1-7, 11, 12)	9	9	18
Invalid Relative Record Number (3,11)	9	**	C0
Invalid RBA access to a RRDS file (7,12)	9	**	C4

*Characters in parentheses reference the type(s) of VSAM Call (Figure 32) which apply. A = all cases.

**Should not occur. The File Handler will force a program check condition to terminate the message in progress.

Figure 33. File Handler VSAM Return and Feedback Codes

6.9 ISAM/VSAM COMPATIBILITY UNDER INTERCOMM

Subsystems accessing ISAM files can function with little or no modification when their files are converted to VSAM. Intercomm's ISAM/VSAM interface does not use IBM's VSAM/ISAM interface modules. See the Operating Reference Manual for steps necessary to activate the interface. When processing a VSAM data set, the File Handler uses QISAM compatible access for a GET or PUT call and BISAM compatible access for a READ or WRITE call.

An ISAM retrieval is converted to a VSAM GET for update. If a key is provided, it is, of course, treated as a full key. For GET with a key, positioning and a search for a greater or equal key is performed. For READ, a search is made for an equal key. File Handler logic will initialize the user FHCW prior to performing the VSAM function as follows:

- Byte 2 is set to 'S' to force sequential positioning.
- Byte 3 is set to 'U' or 'L' to force update mode.

ISAM delete code processing continues to function as usual via the OPTCD subparameter of AMP on the DD statement. The new OPTCD parameters (I, IL) which specify supplementary delete code processing are supported also.

The following considerations apply to ISAM users converting to VSAM and should be carefully observed:

- ISAM subsystems must be operational when accessing ISAM files. Erroneous ISAM parameter lists will cause unpredictable results.
- Between a SELECT and a RELEASE, neither READ and GET nor WRITE and PUT may be intermixed.
- The caller may not provide his own DCB.
- The FHCW will be modified in order to convert the call to its VSAM equivalent.
- There is no equivalent to a QISAM physical block once the file has been converted to VSAM. All VSAM data records are equivalent to ISAM logical records. This means that users processing the file via READ in one subsubsystem and GET in another will both retrieve what would have been an ISAM logical record.

Figure 33 describes return codes when ISAM/VSAM compatibility is used.

Chapter 7

USING THE OUTPUT UTILITY

7.1 CONCEPTS

The Output Utility is a subsystem that processes messages destined for terminals operating under control of Intercomm. It is responsible for completing any device-dependent formatting requirements in a message before passing it to the teleprocessing interface (FESEND) for eventual transmission to the terminal device. It also checks the operational status of destination terminals. Should it find a destination terminal not operational, it will redirect messages to an alternate terminal, if one has been named for that particular destination terminal. Otherwise, the Front End will intercept a message to a nonoperational terminal and queue it in the output queue assigned to that terminal to await its availability.

7.2 PROCESSING

An application subsystem may create four different types of output message text, identified by a value in the message header VMI field (MSGHVMI):

- Preformatted (VMI=X'57')

  Text consists of both data and device control characters. All spacing and other formatting (titles, column headings, etc.) is included in the message text. Output processing consists merely of passing the message to the Front End via FESEND. If the destination terminal (MSGHTID) is the name of a broadcast group, rather than an individual terminal, a separate message is created for each terminal of the group. Except for broadcast terminal-ids, subsystems should use the service routine FESEND, which is more efficient than queuing via Output.

- Formatting Required, Variable Text (VMI=X'50)

  Text consists of a string of character data items to be inserted into a final message format defined by an Output Format Table (OFT) entry. Each data field is prefixed with an item code and length prefix, and an occurence factor (if a repetitive field), to identify the field. The OFT defines the position and content of titles, headings, etc., and defines the position where data fields from the message text are to be inserted. Output formats the final message, adding device-dependent control characters, and performs broadcast group processing, as described above.

- Formatting Required, Multiple Segments (VMI=n)

  This form is used when multiple messages are to be created for the same hardcopy terminal (such as a printer) and interleaving of other messages for the same device is not desired. The text is variable format as described above. The VMI code for the first (or header) segment is X'51'; for intermediate segments is X'52' or X'5C' depending on line types desired; and for the final seqment is X'53'. The final segment must be queued, even if no intermediate segments are created, in order that Output may release the terminal for other messages. See also the description of the DVASN service routine in Chapter 9.

- Formatting Required, Fixed Text (VMI=X'72')

  Text consists of fixed length text fields in character or computational format. This type of message is routed to the Change/Display Utility, where it is converted to a Variable Text message and routed to the Output Utility. The fixed text is described to Change/Display by a Format Description Record (FDR). The first twelve bytes of the fixed format text identify the particular FDR which details the fixed fields of the message. Byte 9 within this header provides the segment type (see Figure 34).

The application subsystem creates its output message (header and text) and directs the message to either the Output Utility or the Change/Display Utility by calling the service routine MSGCOL. The receiving subsystem codes and VMI in the message header specify the destination subsystem and message text formatting requirements. Figure 34 summarizes message header specifications. In addition, the MSGHQPR field in the message header must be set to C'2' if the originating subsystem might process segmented input.

For complete details regarding the Output Utility and Change/Display Utility, refer to the Utilities Users Guide.

OUTPUT Message Type	Message Header Fields			Change/ Display Prefix
	MSGHRSCH	MSGHRSC	MSGHVMI	
Preformatted (device-dependent)	X'00'	C'U'	X'57'	N/A
Variable Text Formatting:				
Single Segment Messages:				
binary format for item code, length, (and occurrence number)	X'00'	C'U'	X'50'	N/A
Multi-Segment Messages:				
binary format first segment	X'00'	C'V'	X'51'	N/A
detail segment - repetitive items			X'52'	
detail segment - non-repetitive items			X'5C'	
final segment			X'53'	
Fixed Field Formatting:	X'00'	C'H'	X'72'	
Single Segment Messages:				C'0'
Multi-Segment Messages:				
first segment				C'1'
detail segment - repetitive items				C'2'
detail segment - non-repetitive items				C'4'
final segment				C'3'

Figure 34. Message Header Specifications for the Output Utility

Chapter 8

CONVERSATIONAL SUBSYSTEMS

8.1 GENERAL CONCEPTS

Conversational subsystems are defined as one or more subsystems designed to process more than one input message to complete a transaction. They effectively carry on a dialogue with the terminal operator, receiving an input message, retaining it and/or associated results of processing, issuing a response (perhaps a prompt for additional information), receiving another input message, retaining it, etc., until the transaction is complete. At the end of the conversation, appropriate files may be updated.

8.1.1 Conversational Applications

Typical applications which lend themselves to conversational processing are:

- Operator prompting (multiscreen input)
- Batch Data collection

Prompting, or multiscreen input, applications typically consist of dialogues in which the terminal operator enters an input message, the information is analyzed by the application subsystem and the results of processing are saved; the application subsystem then sends an output message to the terminal, prompting the operator for the next piece of information required. This dialogue continues until the application subsystem has obtained all the necessary information to complete processing for the given transaction.

Batch data collection may be conversational in that even though the input data is saved for later retrieval, the collecting application may need to return an error message requesting correction of invalid input data before saving the input record, or the application may need to request the input of a different type of record (for more detailed subsidiary information, intermediate totals, etc.).

8.1.2 Conversational Transactions

Conversational transactions involve the sending and receiving of more than one message in a terminal session. Each input message may be processed by related subsystems or by the same subsystem. A two-part conversational transaction is illustrated in Figure 35.

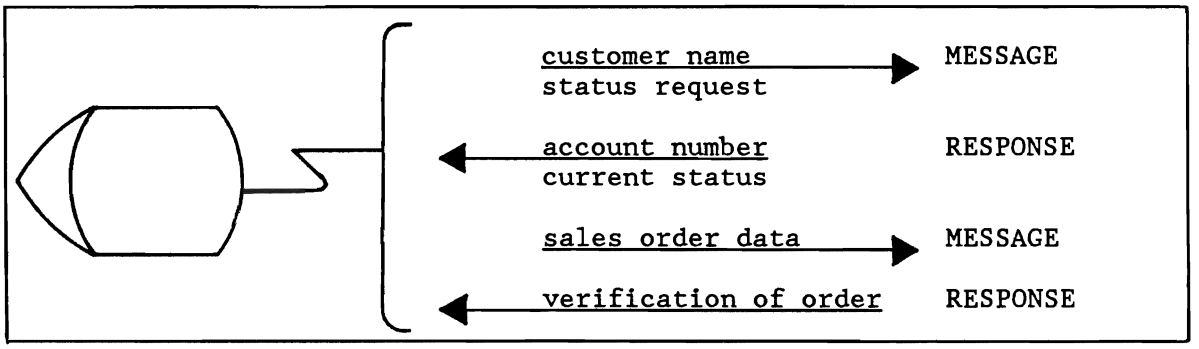


Figure 35. Typical Conversational Transactions

8.1.3 Retention of Information

Assume a conversation in which three input messages and three responses are necessary to complete the transaction. A terminal, a subsystem and a storage medium on which to save the input messages and/or corresponding intermediate results of the processing are necessary components in the conversational environment. In the example illustrated in Figure 36, the subsystem receives information and prompts the terminal operator for additional information until it obtains all the required data. This intermediate information is also stored either in core or on a disk data set. After the final input message is received and processed, appropriate files are updated, intermediate data is deleted, and a final response is issued.

Terminal XYZ	Subsystem ABC	Storage
Input Message 1--->	Receive, process and store---->	Input Message 1 + results
Output Message 1<---	Prompt for additional information	
Input Message 2--->	Receive, access Input Message 1<--	Input Message 1 + results
	Process	
	Also store Input Message 2----->	Input Message 2 + results
Output Message 2<---	Prompt for additional information	
Input Message 3--->	Receive, analyze with prior messages and results <----	Input Message 1 & 2 + results
	Update files, delete prior data	
Output Message 3<---	Final response	

Figure 36. Input Message Data Retention During a Conversation

8.2 IMPLEMENTING CONVERSATIONAL SUBSYSTEMS

Conversational subsystems may be implemented in several ways, each characterized by the retention of initial and subsequent input and processing results. The method of retention differs, depending upon the method of implementation chosen.

Control of the conversation, or the retention of the input messages and/or corresponding results of processing may be accomplished by using any one of the following methods of implementation:

- The User SPA (User Extension to System Parameter List)
- The Store/Fetch Facility
- The Dynamic Data Queuing Facility
- The CONVERSE Service Routine

In addition to the retention of the input environment, conversational subsystems have design considerations with respect to file updates and control of input verbs. These design considerations are discussed following a review of the four methods of retention of input messages and corresponding results of processing.

Intercomm provides Front End conversational support to ensure that duplicate input is not processed. This is accomplished by defining applicable verbs and interactive terminals as conversational in the Front End tables. See the Operating Reference Manual.

8.3 SAVING INFORMATION IN USERSPA

The user extension to the SPA is called USERSPA and is accessible to all Intercomm subsystems since the SPA is the second entry parameter to all subsystems. The SPA (Csect) is a 500-byte core-resident table. The user extention to the SPA begins at the 501st byte and may include application-oriented areas, such as tables, counters, and switches for application subsystem use. Thus, the size of USERSPA is installation-dependent. The user portion of the SPA is optionally checkpointable and can be restored at system restart time.

A portion of USERSPA may be divided into sections associating table space for each terminal, as illustrated by Figure 37. Each terminal-oriented area might be used for control data during conversational processing, until the conversation with that terminal completes.

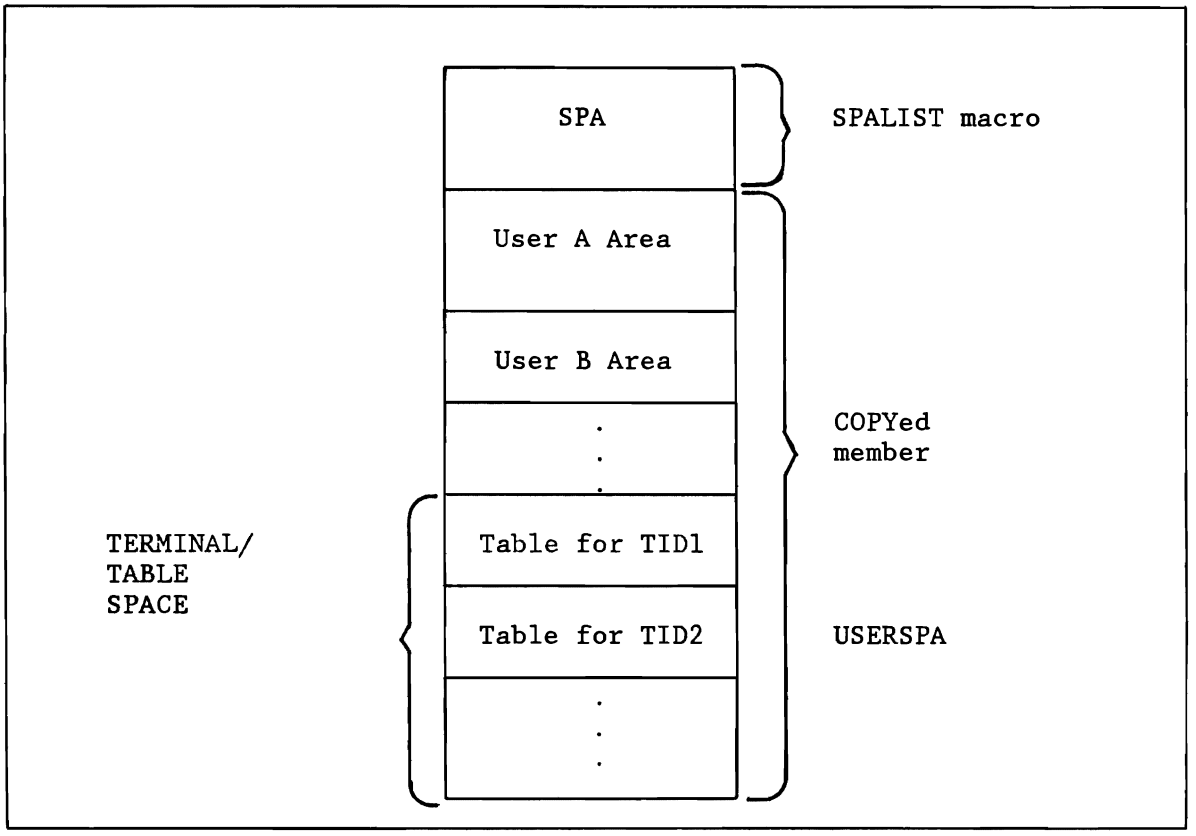


Figure 37. User and Terminal Table Space in the USERSPA

The SPA is expanded by updating the Assembler Language member USERSPA on the system release library SYMREL. The updated version should be stored on SYMUSR. When assembling INTSPA, USERSPA is copied as the last entry in the SPA Csect. Therefore, any user additions would be referenced beginning with the 501st byte. Any such additions should ordinarily be coordinated through the System Manager, as most application subsystems could be affected.

In the Dsect definition of SPA, as shown in Figure 38, three different applications have their own 50-byte areas defined: (USERA, USERB, USERC) plus a table for their common use (COMTAB). The Assembler Language member USERSPA for this example would contain a definition of an area corresponding to SPAUSER. USERSPA could be defined as a systemwide COPY member for all Assembler Language routines. The Dsect is generated via the LINKAGE macro, or by coding the SPALIST macro, with no parameters. In the latter case, the macro must be preceded by a labeled DSECT statement which is the subject of a USING statement to establish addressability.

```
SPALIST  DSECT
SPA...   DC    ...
         .
         .
SPAUSER  DS    0X
COMTAB   DS    XL200
USERA    DS    XL50
USERB    DS    0XL50
USERB1   DS    F
USERB2   DS    X
USERB3   DS    XL45
USERC    DS    XL50
```

Figure 38. Sample USERSPA Declaration Within a Subsystem

The following chart summarizes the advantages and disadvantages of the USERSPA method of implementation of conversational processing.

Advantages	Information saved in Core; no I/O overhead. Accessed easily. Checkpointable and restorable at restart.
Disadvantages	The entire USERSPA is accessible to all Intercomm subsystems. Therefore a problem of control develops with respect to the possiblity of destruction of data by another subsystem, or security problems. Updating and maintenance of USERSPA may require reassembly of all subsytems which reference it. A potentially large area of storage must be allocated. Addressability, if area larger that 3595 bytes.

8.4 SAVING INFORMATION WITH STORE/FETCH

Conversational information may be stored and later retrieved (either in storage or on a disk data set) by the Store/Fetch Facility. Information is retained via the STORE function, and retrieved via the FETCH function. The storage space may be released via the UNSTORE function. Saved information may also be updated.

An operator prompting type of conversation involving one terminal and one or more application subsystem(s) could use Store/Fetch very efficiently for retaining information. Store/Fetch performs its function upon data strings. Data strings are logical entities of information (input messages to be retained or whatever other data the user intends to save), which are identified by unique user-defined keys. The information is accessible only to those subsystems which call a Store/Fetch service routine naming the data string by its unique key, which could include the current terminal-ID from the input message header. Therefore, there is more control over the information than there would be if it were to be saved in the USERSPA. The data strings are classified as either transient, semipermanent or permanent. The differences between these classifications are as follows:

Disposition	Availability	Storage Medium
Transient	Not available across restart	Core or disk
Semipermanent	Available across restart	Disk
Permanent	Available across every system start until explicitly unstored	Disk

In conversational processing, permanent data strings should not be used. As to whether to use transient or semipermanent strings, the user must decide whether the information is critical enough to be preserved across system restart. If so, the data strings would be classified as semipermanent and would reside on disk. At restart time, the operator could then resume a conversation at the point of failure if subsystem logic can determine when the conversation was interrupted. If stored data is specified as transient, data is eligible to reside in core. Processing would thus be speeded up, as I/O overhead would be eliminated. At restart time, the operator would then start the conversation from the beginning.

Detailed information on Store/Fetch, including the interface between application subsystems and the Store/Fetch service routines, may be found in Store/Fetch Facility. Application subsystem logic must determine whether the input message in progress is initial, intermediate or final. This determination is necessary to assure that the proper calls to Store/Fetch are issued when data is to be saved or retrieved. Once the determination is made, Store/Fetch may be used to manage the conversational information as shown in Figure 39.

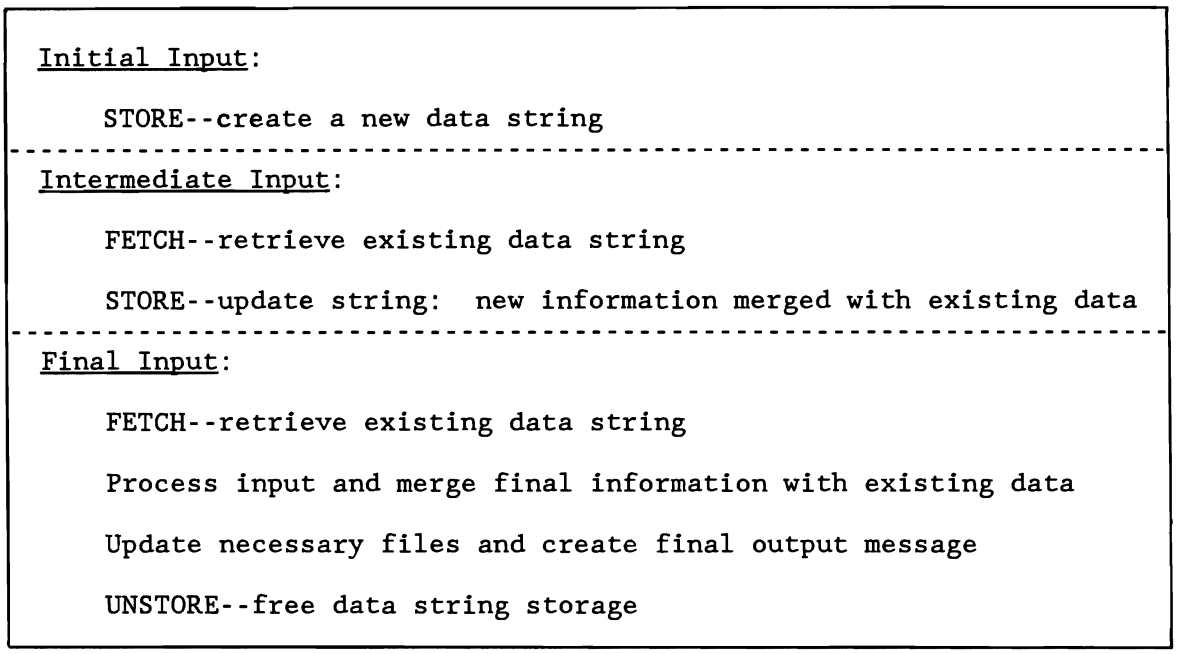


Figure 39. Conversational Processing Using Store/Fetch

Subsystem processing logic can be simplified by using one or more of the following techniques:

- A 'string-not-found' return code from a FETCH request indicates intial input (no intermediate data stored).
- A FETCH with the Delete option forces restart of the conversation from the beginning if the system fails, or the subsystem times out or program checks before the STORE of the intermediate data can be done. This technique also saves Store/Fetch and core storage resource overhead.
- The STORE of the intermediate data should be done after the output message is processed.
- File record(s) should not be updated until all intermediate data is collected. At this time the record(s) should be retrieved for update (exclusive control) and checked for external updates by unrelated processing since the conversation began.
- Do not send the final confirmation output message until successfully updating the file(s).

8.5 SAVING INFORMATION ON A DYNAMIC DATA QUEUE

The Dynamic Data Queuing Facility (DDQ) is a Special Feature available to Intercomm users. Detailed specifications on using DDQ may be found in Dynamic Data Queuing Facility. A DDQ provides the application subsystem with the ability to dynamically create, retrieve and delete logical data sets (or queues) of records on a BDAM data set. As illustrated in Figure 40, more calls are required to interface with the DDQ routines than are required to interface with Store/Fetch to obtain the same functions. However, a DDQ provides the ability to save several related data strings as a type of sequential file. The entire DDQ can then be processed by another subsystem or postponed for batch processing. A DDQ is most effectively used, not as a means for temporary storage of data during a conversation, but as a means for accumulating conversational results for subsequent processing, that is, for data collection. This facility can also be used for collecting data from related conversations with more than one terminal.

The data queues may be either transient, single-retrieval transient, semipermanent or permanent. Single-retrieval transient queues cannot be read more than once. This type of DDQ, therefore, would not be suitable for conversational processing. The other queue types are distinguished by the following characteristics:

Queue Type	Characteristics
Transient	Must be passed to another subsystem or freed. Cannot be retrieved later. Not preserved across restart or normal startup.
Semipermanent	Retrieved at a later point in time via a user-provided Queue Identifier (QID). Extra I/O overhead is involved in saving the queue. Can be freed by user requests. Queue must be completed (closed) in order to be preserved across restart. Existing semipermanent queues freed at normal startup.
Permanent	Same characteristics as semipermanent except that permanent queues are always preserved across any Intercomm start, warm or cold, if closed at least once.

Figure 40 illustrates typical use of DDQ facilities in conversational processing. The application subsystem logic must determine whether input is initial, intermediate, or final. Final input, in this example, causes the queue to be closed and passed to another subsystem for asynchronous or postponed file updating. Thus, the terminal operator, upon receipt of the final output message, can begin another conversation without waiting for file updates to occur. This technique is particularly useful for files which do not require up-to-date inquiry response such as order entry, personnel, etc.

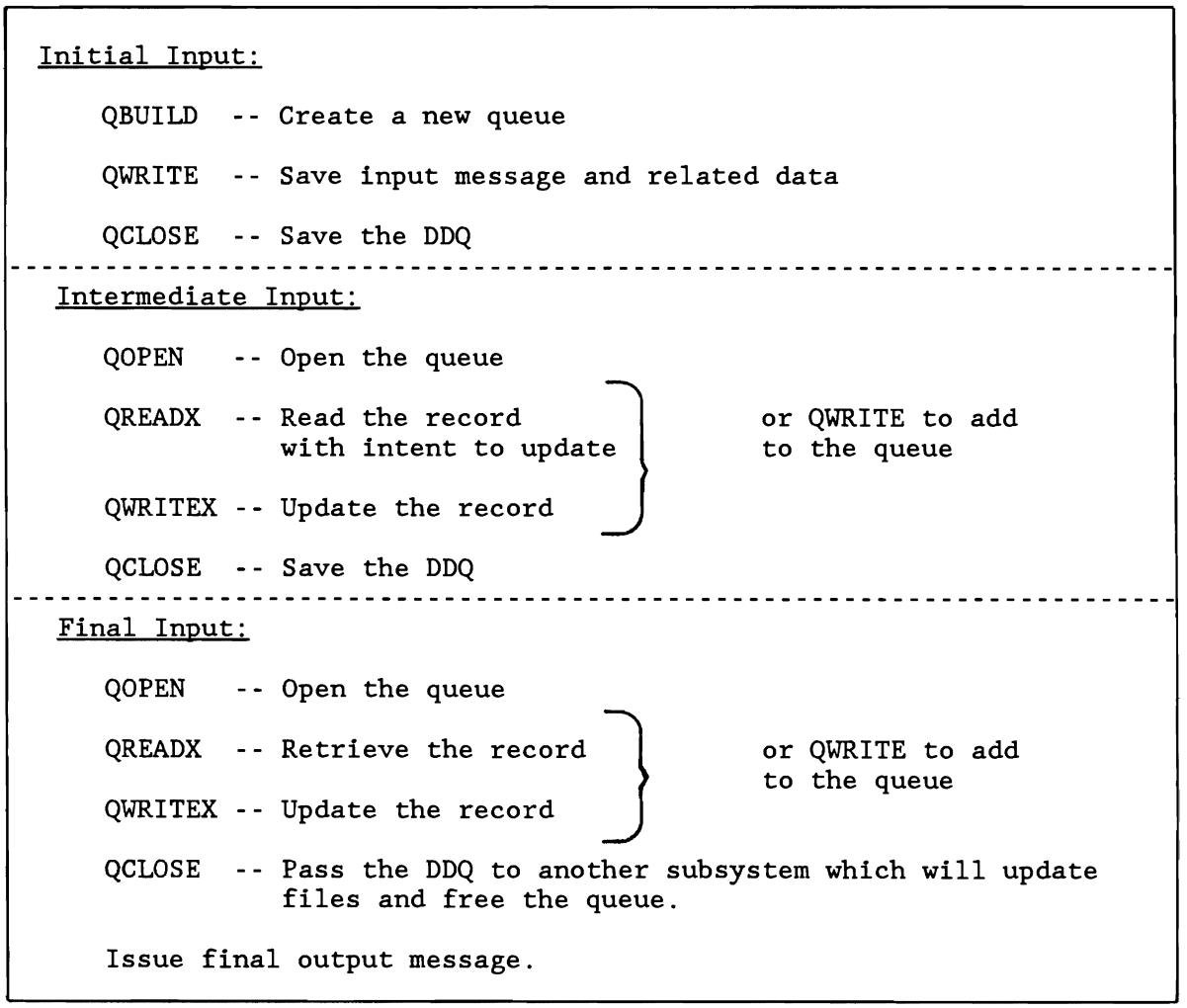


Figure 40. Conversational Processing Using Dynamic Data Queuing

8.6 SAVING INFORMATION VIA THE CONVERSE SERVICE ROUTINE

The final method of retaining information for a conversation is to use the Intercomm system service routine CONVERSE. The CONVERSE routine is called by an application subsystem when input from a terminal is required to continue processing a transaction. The application subsystem stops processing until the next input message is received. Control is returned to the next sequential instruction following the call to CONVERSE.

Application subsystems are designed more easily with CONVERSE, as it is simpler to control the sequential order of the messages. However, the use of CONVERSE is not encouraged, as it ties up Intercomm resources. Dynamic storage associated with the initial and subsequent input messages is retained during the call to CONVERSE. Storage requirements for subsystems would be greater than when other conversational techniques are used, because one subsystem contains logic for all message types of a conversational transaction. It is far more efficient to design conversational subsystems which retain control only for the amount of time necessary to process one message than to tie up system resources while each input message in the conversation is in turn received, kept, analyzed and responded to in one execution of one application subsystem. When CONVERSE is used, dynamically loaded subsystems remain in storage until all "conversations in progress" have terminated. Intercomm restart processing of such subsystems restarts the conversation from the beginning. All intermediate messages are discarded.

The saving of information in the USERSPA or in a Store/Fetch data set or in a DDQ does not require an application subsystem to contain logic for time-outs. The use of CONVERSE does. If the next input message is not received in the time limit specified by the user, a time-out occurs, which must be handled by subsystem logic.

The CONVERSE program keeps track of conversational requests by terminal and subsystem, and separates messages accordingly. Hence, any subsystem may be in conversation with any number of terminals simultaneously.

An example of the use of CONVERSE in a two-part conversation is illustrated in Figure 41.

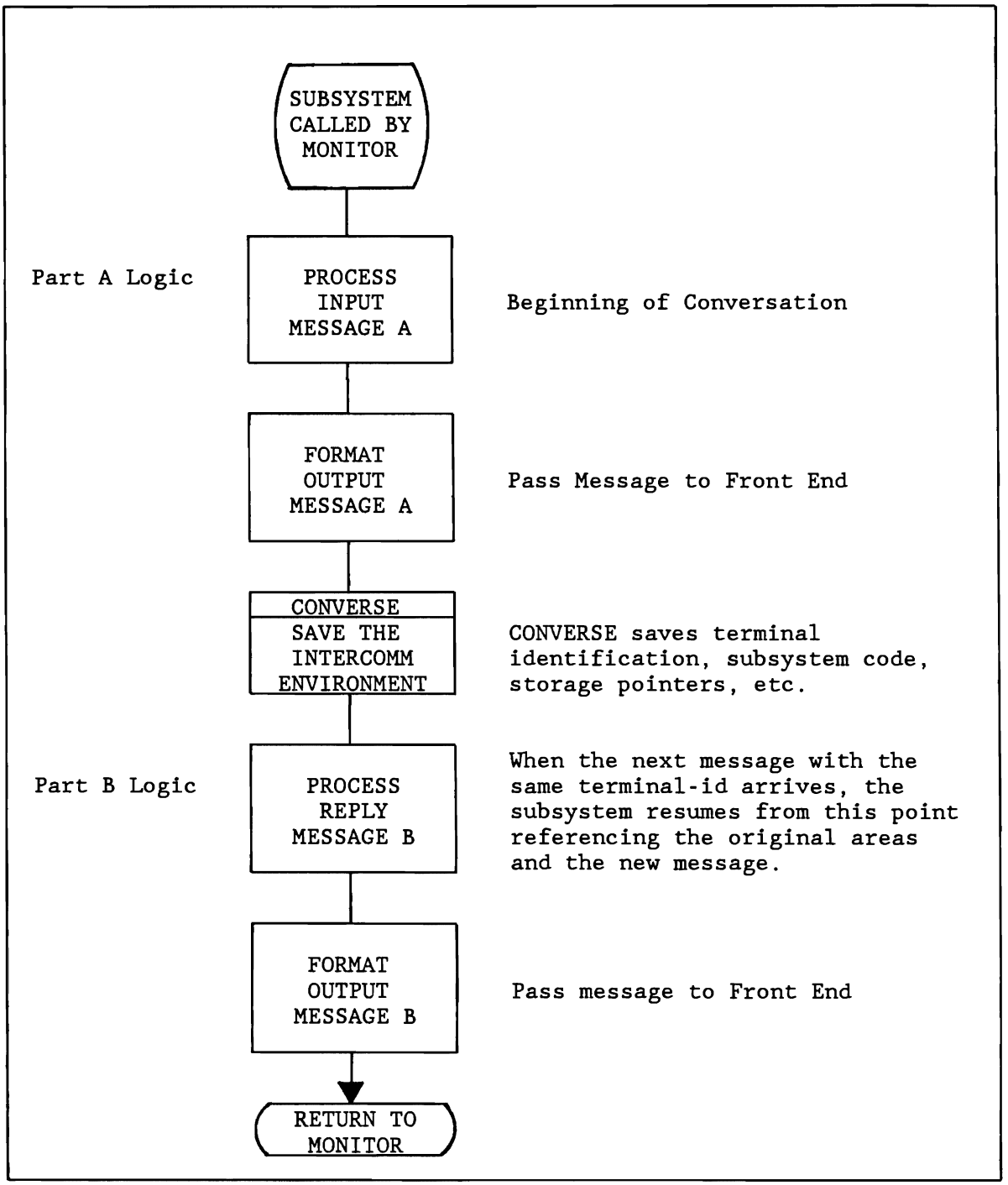


Figure 41. Conversational Subsystem Logic Using Converse

8.6.1 Subsystem Design Using CONVERSE

The Intercomm system service routine CONVERSE is called when awaiting additional input in response to some prompting message. Since any interval may elapse before the next message is received, CONVERSE will save information in its own control table for each conversation and return to the Subsystem Controller while waiting for the response.

The call to CONVERSE specifies a time limit within which a reply message should be received. If it is not received during the specified interval, then the subsystem is entered at the next instruction following the call to CONVERSE and its message parameter is adjusted to point to a time-out message supplied by CONVERSE. That message (header plus text) could then be switched to the Output Utility. The terminal identification in the header is that of the non-responding terminal.

Coding format:

There are two coding formats available to reentrant Assembler Language subsystems for calling the CONVERSE subroutine: as core resident or in an overlay area.

The coding format, if CONVERSE is always resident, is as follows:

```
[symbol] CALL  CONVERSE,(parm,time),VL,MF=(E,list)
```

If CONVERSE may be in a transient overlay area:

```
[symbol] CALLOVLY  CONVERSE,(parm,time),VL,MF=(E,list)
```

where:

- parm is the address of the parameter list passed to the subsystem at its entry point. The contents of the parm register is specified on the LINKAGE macro. If the input message is not freed by the subsystem (or a call to MAPIN) CONVERSE will free it. If freed by the subsystem, zero the first word of the input parameter list. If the input message is edited by a call to EDITCTRL, set the address of the edited message into the first word of the input parameter list.

- time is the label of a fullword binary value indicating an interval limit (in seconds) within which a subsequent message is expected. A zero value for the time limit will bypass the automatic time-out feature.

When processing resumes following the call to CONVERSE, the environment appears as it was before the call--except the input message parameter (unless there was a time-out) now points to the most recent message from the terminal. It is the subsystem's responsibility to verify that the message received following the call to CONVERSE is actually the appropriate message expected in the logical sequence of the conversation.

In calling CONVERSE from an Assembler Language subsystem, the address of the parameter list for the previous message being processed by the subsystem must be passed. Upon return, the address of a new parameter list (with the address of the new message) must be loaded to the appropriate register. This coding sequence is illustrated below:

```
LINKAGE MSG=(R8),PARM=(R7),SPA=(R9)...
   .
   .
   .
CALL  MSGCOL,(. . . . . )                Q output message
CALL  CONVERSE,((R7),TIME),VL,MF=(E,list)
L     R7,list
L     R8,PARMMSG
```

If the new message from the terminal requires editing, CONVERSE will call the Edit Utility before passing the new input message address to the subsystem. If editing is unsuccessful, error messages are sent back to the terminal (see Figure 42).

Figure 42 shows the CONVERSE return codes and the contents of message text for a time-out condition. These return codes are fullword binary values in register 15 indicating the condition for return.

Return Codes	Meaning
0 (X'00')	Normal return: the entry parameter input-message reflects the address of the new input message. The message will have been edited successfully if the Front End Verb Table shows editing required. (If editing is unsuccessful, error messages will be sent to the terminal, and the subsystem is not reactivated until either a subsequent input message is edited successfully or an automatic time-out occurs.) CAUTION: The CONVERSE automatic time-out is not extended if a message is found in error by the Edit Utility.
17 (X'11')	No core available for CONVERSE control blocks; conversational mode not initiated.
18 (X'12')	Time-out expired. The entry parameter input-message reflects the address of an error message generated by CONVERSE. The message header contains the appropriate terminal identification. The message text is: *PMI*CONVERSE*ANTICIPATED MESSAGE NOT RECEIVED WITHIN USER SPECIFIED TIME INTERVAL

Figure 42. CONVERSE Return Codes

Control of the conversational program environment is accomplished by Intercomm in different ways, depending on the subsystem's residency:

- Resident

  The dynamic work space for one message is retained pending arrival of the next message from the terminal; the subsystem will continue to process messages from other terminals.

- Overlay Loaded

  Same as above, except the loaded overlay region may contain other subsystems to process other messages during (and after) "CONVERSE time."

- Dynamically Loaded

  Same as above, except the subsystem remains in core until all "conversations in progress" have terminated.

Conversational subsystem logic must be designed with care regarding file access. Selected files should be released prior to the call to CONVERSE. If not, other subsystems accessing the same files or other messages in process in the same subsystem may "time out." This may occur because an operating system control block is associated with the access to the file and is not "freed" until the file is released. If a file is accessed prior to the call to CONVERSE and released after the call to CONVERSE a "lock out" situation may occur.

Assembler Language subroutines may not call CONVERSE.

8.7 DESIGN CONSIDERATIONS IN CONVERSATIONAL PROCESSING

In order to ensure file integrity, conversational subsystems performing file and/or data base updates should be designed to perform the updates for the last message in the conversation. Alternatively, control may be passed (via message queuing) to a non-conversational subsystem to perform the updates.

8.7.1 Control of the Input to Conversations

Conversational subsystems expect ordered input. They must be designed to analyze input messages and to determine which message in the sequence has been received. Control of the input may be exercised by the terminal operator or by the application subsystem(s).

The terminal operator may be given a specific sequential list of messages to input at the terminal for a given verb or verbs. This method would probably be used for data collection applications, in which more messages are sent to the application subsystem than are received at the terminal. It could also be used for any conversational application in which the order of input is fixed.

The application subsystem may control the input sequence by analyzing an input message, processing it, and issuing a response informing the operator about the content or format of the next input message. The response may direct the operator to input another verb (that of a related subsystem). Subsystem-controlled input is good for conversations in which the "next" desired piece of information may vary depending upon the contents of a file record, or a table, or the setting of a switch in an area saved between subsystem activations.

8.7.2 Assigning a Verb to a Terminal

To eliminate the requirement for an operator to key in a verb with each input message, the operator may enter a system control command message to LOCK a specific terminal to a particular verb. The Front End then prefixes that verb to each input message from that terminal. The operator may enter another control message, UNLK, to unlock the terminal from the verb. See System Control Commands.

The LOCK/UNLK commands processed by the Front End can also be issued by a subsystem. When a LOCK is in effect, all subsequent messages from the specified terminal will be automatically prefixed by the verb specified in the LOCK command. This LOCK remains in effect until UNLK is issued. With LOCK in effect, some advantages are:

- The terminal operator does not have to keep reentering the same verb.
- A new verb cannot be entered during the conversation.

Either the subsystem or the operator may control the input sequence by locking and unlocking the terminal to different verbs at different points in, or at the end of, the conversation.

Optionally, the Intercomm AUTOLOK feature may be defined for the verb in the Front End Verb Table, which dictates that when that verb is input from the terminal, the terminal is to be automatically locked to that verb. Subsequently, the terminal is to remain locked until specifically UNLKed by the operator or processing subsystem.

The format for the LOCK/UNLK commands (message text) is as follows:

```
LOCK$TPUxxxxx$vvvv@
UNLK$TPUxxxxx@
```

where:

xxxxx
: is the five-character terminal identification

vvvv
: is the four-character verb

@
: is the end-of-transmission character (X'26')

$
: is the system separator character as defined for the installation.

The preformatted message constructed by a subsystem must be prefixed with the standard message header for FESEND (MSGHRSCH=X'00',MSGHRSC=X'00',VMI=X'57'). This message is passed to the Front End via FESEND (see Chapter 9) and the LOCK or UNLK takes place. No response message is sent to the terminal when such processing is requested by a subsystem.

Chapter 9

USING INTERCOMM SERVICE ROUTINES AND FACILITIES

9.1 SERVICE ROUTINES AND FACILITIES

This chapter further describes use of Intercomm service routines and facilities available to Assembler Language subsystems. These are as follows:

- Pass message to another subsystem (MSGCOL)
- Message Logging (LOGPUT)
- Pass Message to Front End (FESEND)
- Front End Control Messages (FECMs)
- Perform Binary Table Search (BINSRCH, BINSRCH2, BINSRCH3)
- Data Field Search Routines (PMIFINDB, PMIDLTDB)
- Segmented Message Routines (GETSEG, DVASN)
- Dispatcher Related Routines (IJKPRINT, IJKTRACE, IJKDELAY)
- In-core Table Sort (INTSORT)
- Other Intercomm Service Facilities
- Loading Service Routine Entry Points from the SPA

9.2 MESSAGE SWITCHING (MSGCOL)

Message Collection is a system service routine. It is responsible for queuing messages destined for processing by another subsystem, that is, message switching. MSGCOL controls the queuing of messages via the message header fields MSGHRSCH and MSGHRSC, the receiving subsystem code.

The logic of an application subsystem might be such that the input message is modified within its dynamic area to become an output message to switch to another subsystem. To do this, the length of the input message must not be altered (data may not be added). Queuing the message for the next subsystem is then done by calling Message Collection (MSGCOL); Message Collection then owns and is responsible for the management of the message area. In this case, the subsystem is not responsible for freeing the input message area.

Coding format:

```
[symbol] CALL  MSGCOL,(message,SPA),VL,MF=(E,list)
```

where:

message is the address of the (input) message to be queued

SPA is the address of the System Parameter Area.

MSGCOL return codes indicate the result of the queuing. The return code is a fullword binary value in Register 15. (See Figure 43.) Regardless of the result, the calling program no longer has any control over the area of dynamic storage occupied by the message.

Return Code	Meaning
0	Message queued successfully
4	No room on queue (entry made on system log) or message rejected for delayed subsystem
8	No core for disk queue I/O area
12	I/O error on disk queue
16	Invalid subsystem code (entry made on system log)

Figure 43. Message Collection Return Codes

Recovery action for unsuccessful queuing might be to return to the System Monitor with a return code of 8 or 12. A message would then be sent to the terminal that originated the input message being processed, if USRCANC (PMICANC) is included in the Intercomm linkedit.

9.3 USER LOG ENTRIES (LOGPUT)

An application subsystem may require entries on the system log for many different situations:

- Application-dependent security violation or other application-dependent error recording.
- Log entries rather than snaps used to trace the progress of a message while testing.
- Any application-oriented requirement for a record on the system log.
- Before- and/or after-image records of file updates (if not using the Intercomm File Recovery special feature).

User log entries are identified by unique codes in the message header log code field (MSGHLOG) and hence can be recognized by any batch program processing the log off-line. Messages to be logged consist of a standard 42-byte header and message text. The log code field in the message header may have any value from X'41' to X'6F'. Logging is performed by calling the Intercomm system service routine LOGPUT. The date and time stamp in the message header (MSGHDAT and MSGHTIM) will be updated by LOGPUT prior to writing to the log. Log entries may subsequently be suppressed for later Intercomm executions by modifying the LOGTROUT translate table in the LOGPUT routine. Any message having a log code in the header which translates to X'FF' will not be logged.

The length of the record on the log is controlled by the value of MSGHLEN in the message header and must be at least 42. LOGPUT will not write out messages longer than the logical record size of the log (see INTERLOG JCL description in the Operating Reference Manual).

Coding format:

```
[symbol] CALL  LOGPUT,(message),VL,MF=(E,list)
```

where:

message is the label (address) of the message (header plus text) to be logged.

There is no return code from LOGPUT.

9.4 PASS MESSAGE TO FRONT END (FESEND, FESENDC)

FESEND (or FESENDC) is called to pass a message to the Intercomm Front End for transmission to a terminal. The entry point FESENDC of FESEND copies the message to a new area of storage, and then proceeds with FESEND logic. The message header field MSGHTID specifies the destination terminal or broadcast group name. FESEND then requests queuing of the message on the associated terminal queue. If a broadcast group is specified, FESEND creates an individual message for each terminal of the group and requests queuing for each of those messages. All terminals in the broadcast group must be of the same type, as defined in the Back End Station and Device tables (see Chapter 2).

FESEND accepts two types of messages: preformatted (VMI=X'57') message text, which contains the control characters and data for transmission to the terminal except for start-of-text sequence(s) to be added by the Front End; and fully-formatted (VMI=X'67') message text, which contains all control characters and data ready for transmission to the terminal. (MMU produces fully-formatted messages.) If segmented input messages may be processed, set MSGHQPR to C'2' before calling FESEND. If passing the message to the Front End is for any reason unsuccessful, the subsystem is notified by a return code in Register 15, and recovery action may be taken.

FESEND tests whether messages sent to the Front End might be system commands or for control purposes. Such messages control Front End operation and generally cause no output to a terminal. Front End Control Messages (FECMs) are described later in this chapter. All system control commands and message text contents are documented in System Control Commands.

FESEND becomes the owner of the area of storage occupied by the message. Do not attempt to free this area or reference it once FESEND is called. FESENDC copies the message to a new area, the original area still belongs to the caller (and may be in the dynamic save/work area and ultimately freed via RTNLINK, rather than an acquired area which requires freeing by STORFREE before the RTNLINK).

Coding format:

```
[symbol]  CALL  {FESEND },(msg-addr[,ret-code[,option-codes]]),
                {FESENDC}
                        VL,MF=(E,list)
```

where:

msg-addr points to the first byte of the message (header and text) to be passed (copied) to the terminal queue.

return-code optionally points to a two-byte character field where FESEND will place a return code indicating whether or not processing was successfully completed (see Figure 44).

option-codes optionally points to a four-byte character field containing Front End processing codes as follows:

Byte 1: CRT Release option code:
blank or X'00'--do not release (prevent screen overlay) next message (default)
C'R'--release (allow overlay) next message to CRT
C'C'--release next message, but do not cancel Front End conversational time-out

Byte 2: VTAM Response option code (overrides Front End Network Table definition for terminal):
blank or X'00'--no override (default)
C'O'--D1 response
C'E'--E1 response
C'F'--D2 response
C'G'--E2 response

Bytes 3 and 4: Not used (set to blanks or binary zeros)

FESEND also returns codes in hex in register 15; the codes and possible recovery actions are listed in Figure 44. A nonzero return code means the message was not queued for the Front End. Return codes 16-24 should only occur during subsystem testing. Regardless of the result, the calling program no longer has any control over the area of dynamic storage occupied by the message if FESEND was called.

Return Code		Meaning
00	X'00'	Message queued successfully.
04	X'04'	Queue-full condition encountered; attempt a retry by invoking FESEND again, after a timed delay.
08	X'08'	Low-core condition encountered; attempt a retry by invoking FESEND again or return to Intercomm. (See Figure 14.)
12	X'0C'	I/O error (see Figure 14) encountered on disk queue; return to Intercomm.
16	X'10'	Invalid terminal-ID; no recovery action required. Check with System Manager to verify terminal/ broadcast group named in MSGHTID field.
20	X'14'	Invalid VMI or syntax error in Front End control or command message text.
24	X'18'	Invalid message header; return to Intercomm. See also error message MG602I and Snap 51.

Figure 44. FESEND Return Codes

9.5 FRONT END CONTROL MESSAGES

The Front End Control Message (FECM) facility provides three types of Front End control messages which may be used by application subsystems for:

- Front End data queuing (FECMDDQ)
- Front End feedback messages (FECMFDBK)
- Front End queue release (FECMRLSE)

A FECM is generated by an application program call to a service routine. The generated feedback message text is complete. The header field MSGHLEN has been set; bytes 3-42 are not modified. If the user has copied a valid header to the FECM message area prior to the call, only the sending subsystem codes (SSCH,SSC) and the VMI must be set (X'57'). The generated FECM must then be passed to the Front End by a call to FESEND in the application program.

After a call to any Front End Control Message facility, a return code is placed in the first byte of the status word and the binary value of the return code is returned in register 15:

Return Code Value	Meaning
C'0'	FECM successfully created
C'8'	No storage available to build FECM

Figure 45. FECM Return Codes

9.5.1 Front End Data Queuing

Front End data queuing (FECMDDQ) works in conjunction with the Dynamic Data Queuing Facility. It provides the user with a more efficient way of handling groups of related output messages. An application may pass a Dynamic Data Queue (DDQ) to the Front End via a FECM. The DDQ contains messages to be sent to a terminal. This is a more efficient design approach than sending one message at a time to the Front End via FESEND, and prevents interleaving of unsolicited messages with those on the DDQ. This feature is particularly useful for printed reports. The messages on the DDQ must be preformatted (VMI=X'57') or fully formatted (VMI=X'67'). The Dynamic Data Queuing Facility manual contains detailed information on DDQ concepts, facilities and implementation, and specific design considerations for Front End Data Queuing. MMU uses this facility (FECMDDQ), when requested for multipage printer output.

Coding format:

```
[symbol] CALL  FECMDDQ,(status-word,fecm-area,ddq-id[,ddq-disp]),
               VL,MF=(E,list)
```

where:

status-word is a fullword (aligned) required by the facility.

fecm-area is a 112-byte area to contain the FECM (header and text). The user should initialize the header prior to the call, probably by copying the input message header to this area. If this parameter is zero, the facility will acquire the area of storage for the caller. (Register 15 has a value of 8 on the return from the call if storage was not acquired.) The caller must then complete the message header area of the FECM. Only MSGHLEN is set by the facility.

ddq-id is the sixteen (16) byte DDQ identifier.

ddq-disp is a one-byte code indicating DDQ disposition after all messages are transmitted:

C'S' means SAVE the DDQ (required if MSGHTID is a broadcast group name)

C'F' means FREE the DDQ (default)

NOTE: The ddq-disp parameter may be omitted if the DDQ is to be freed after all the messages are transmitted (default). All the above parameters must be in dynamic storage if the calling program is loaded above the 16meg line under MVS/XA or ESA.

9.5.2 Front End Feedback Messages

This type of FECM (FECMFDBK) is used by an application to determine that all prior messages queued for a terminal (before the FECM) have been transmitted. In this way, an application subsystem can be notified that certain critical messages have indeed been successfully transmitted.

Subsystem logic creates all normal output messages and passes them to the Front End (via FESEND, MMU, or by queuing messages for Output). Generation of a feedback message is then requested by a call to a FECM service routine. The feedback message is then processed in the same way as the other messages for the terminal (queued via FESEND or the Output Utility). When the Front End retrieves the feedback message, it is routed to the subsystem specified when the feedback message was generated rather than to the destination terminal.

Feedback messages may also be used in conjunction with Front End Data Queuing. A feedback message could be an intermediate, or the last, message on a DDQ passed to the Front End. If the DDQ was created via MMU (a MAPEND call option), then the feedback FECM must be created and queued by the subsystem on return from the MAPEND call.

Coding format:

```
[symbol] CALL  FECMFDBK,(status-word,fecm-area,fecm-rsc,
               fecm-text),VL,MF=(E,list)
```

where:

status-word is a fullword (aligned) required by the facility.

fecm-area is a 78-byte area to contain the FECM (header and text). The user should initialize the header area prior to the CALL, probably by copying the input message header to this area. If this parameter is zero, the facility will acquire the area of storage for the caller. (Register 15 has a value of 8 on the return from the call if storage was not acquired.) The caller must then complete the message header area of the FECM, only MSGHLEN is set by the facility.

fecm-rsc is a two-byte receiving subsystem code (high/low) to specify the feedback message destination subsystem.

fecm-text is a 16-byte area containing the desired feedback message text.

The generated feedback message text is complete. The header field MSGHLEN has been set; bytes 3-42 are not modified by FECMFDBK. If the user has copied and set a valid header prior to the call, no further modification to the header is required.

9.5.3 Front End Queue Release

This type of FECM (FECMRLSE) allows the subsystem to override the normal Front End Logic for CRTs, which requires a one-for-one correspondence between input and output messages. When the release FECM is processed by the Front End, it causes a subsequent response message queued for the terminal identified by MSGHTID in the FECMRLSE message header to be transmitted immediately, rather than waiting for input (RLSE command) from the terminal operator. Because of protocol restrictions (HDFF) on VTAM Front End IBM SDLC 3270 CRT processing, the CRT release option for the first call to FESEND should be used (see Section 9.4) as a release; because if the terminal is already in send mode, it is necessary to turn the line around before sending the released message, which may confuse the terminal operator.

A release FECM might be used if a subsystem queues more than one output message to the CRT terminal due to a considerable amount of processing (file/data base I/O) being necessary between messages. The first message might be an immediate response to the terminal operator indicating the input request is being processed, while the second message is the ultimate result of the requested processing. A release FECM could also be used to force immediate transmission of a critical message to another CRT (other than the input terminal). Such processing should be used with caution because unsolicited messages can cause confusion for the terminal operator and may clear an existing screen format or displayed message.

Coding format:

```
[symbol] CALL  FECMRLSE,(status-word,fecm-area),VL,MF=(E,list)
```

where:

status-word is a fullword (aligned) area required by the facility.

fecm-area is the label of a 60-byte area to contain the generated FECM (header and text). The user should initialize the header area prior to the call, probably by copying the input message header to this area. If this parameter is zero, the facility will acquire the area of storage for the caller. (Register 15 has a value of 8 on the return from the call if storage was not acquired.) The caller must then complete the message header area of the FECM, only MSGHLEN is set by the facility.

9.6 PERFORM BINARY TABLE SEARCH (BINSRCH,BINSRCH2,BINSRCH3)

The module BINSRCH is automatically included as a resident Monitor module. This module performs two functions: it will search a sorted table for an entry equal to a passed argument, using a binary search technique--the entry point to perform this function is BINSRCH2; and it will search an index table whose first halfword or fullword of each entry is an offset/2 from the base of the actual table to be searched (points to an individual table entry). The sequence of index table entries reflects the ascending sequence of values (keys) in the actual table (which is not sorted). The entry points to perform this function are BINSRCH (if the first halfword of the index entry contains an offset to the corresponding entry in the actual table) or BINSRCH3 (if the first fullword of the index contains the offset).

Coding format:

```
[symbol] CALL {BINSRCH },(arg,#entries,entry-len,table,offset,arg-len,
              {BINSRCH2}
              {BINSRCH3}
                        index[,shift]),VL,MF=(E,list)
```

where:

arg is the address of the search argument.

#entries is a fullword containing the total number of entries in the table (BINSRCH2) or index (BINSRCH and BINSRCH3).

entry-len is a fullword containing the length of a table entry (BINSRCH2) or of an index entry (BINSRCH and BINSRCH3).

table is the base address of the table to be searched.

offset is a fullword containing the offset within an actual table entry at which a comparison is to be done.

arg-len is a fullword containing the length of the search argument.

index is the address of the first index entry (BINSRCH or BINSRCH3) or zeroes (BINSRCH2).

shift is a fullword containing the shift amount to shift the offset in an index entry to displace to the beginning of the associated table entry. This parameter is optional, and if absent, a shift value of 1 is assumed. The displacement calculated will be equal to Index-Offset*2*Shift. (BINSRCH and BINSRCH3 only.)

BINSRCH, BINSRCH2 or BINSRCH3 will return the following to the calling program:

- Register 15: the address of the matching table entry (BINSRCH2), or the address of the matching index entry (BINSRCH or BINSRCH3), or, if not found, the first table (or index) entry whose key exceeds the search key.

- Register 1: the address of the actual matching table entry, or zeros if none found.

9.7 DATA FIELD SEARCH ROUTINES (PMIFINDB,PMIDLTDB)

When using the Edit, Output or Change/Display Utilities, the search routines allow the user to add, delete, or locate a variable format data field in an area (message text). These routines are entry points in the system module PMISERC3. The variable format data field must always be formatted as follows:

Byte 1: Item Code (in binary), identifying the data.

Byte 2: Length (in binary), containing the number of bytes which follow. This length does not include the item code or the length byte itself.

Byte 3: Beginning of the data text, or if an occurrence number (line number) for the item code exists, it is contained in the 3rd byte (in binary) and the data will follow starting in the 4th byte.

The area containing the variable format data fields will be referred to as the text area in the following discussion.

Specifically, the search routines are used to find the address of an item code, (or an item code and occurrence number), or to update a text area by adding or deleting a data field. The search routines are:

PMIFINDB - Finds the address of a data field with a given item code (by line number or by occurence number) in a text area.

PMIDLTDB - Adds or deletes a data field with a given item code (by line number or occurence number) within a text area.

9.7.1 PMIFINDB - Find a Data Field

The purpose of this routine is to find a data field with a given item code (optionally by occurrence or line number) within a text area. Search by occurrence number will locate a data field with a specific occurrence number, or with a specific line number if the data fields in a message area occur vertically rather than horizontally.

Coding Format:

```
[symbol]  CALL  PMIFINDB,(start-address,SPA,end-address,codes,0),
                VL,MF=(E,list)
```

where:

start-address is the beginning location address for the search

SPA is the address of the System Parameter Area

end-address is the ending location address for the search (end of text area)

codes is the address of a 3-byte field containing

byte 1: item code (binary)

byte 2: occurrence/line number (binary) - X'00'if no occurrence/line search is required

byte 3: character action code--
1: search by occurrence/line number
2: search by sequence (obsolete)
3: next occurrence following specified occurrence number (obsolete)

0 saves space in the parameter list for the address of the found variable length data field (starting with the requested item code).

If the search is successful, the address of the found data field will be returned in the 5th parameter. If the search is unsuccessful, one of the following conditions occurs:

- If the item code could not be found, binary zeros will be returned in the 5th parameter field, return code in register 15 is 0.
- If the location addresses are invalid (end lower than begin) or the action code is invalid (other than 1), a return code of 1 (in register 15) will be returned to the calling module, and the 5th parameter field will be zeroed.

9.7.2 PMIDLTDB - Delete or Add a Data Field

The purpose of this routine is to delete or add a variable length data field with a given Item Code (optionally by occurrence number) within a particular text area. An added field is moved to the end of the used (non-zero) text area and the next byte is set to X'00'. For a deleted field, the remaining text area is shifted left over the deleted field, and the trailing unused portion is set to binary zeroes.

Coding format:

```
[symbol]  CALL  PMIDLTDB,(start-address,SPA,end-address,field,action),
                VL,MF=(E,list)
```

where:

start-address is the beginning location address for the search

SPA is the address of the System Parameter Area

end-address is the ending location address for the search (end of text area)

field is the address of a field as follows:

For Delete Action - Two-byte field containing:
- a) Item Code (in binary)
- b) Occurrence Number (in binary)
  (if no occurrence number is necessary, this byte must contain X'00')

For Add Action - Variable length data field containing:

a)	Item Code	1 byte
b)	Length of c below	1 byte
c)	Occurrence Number (optional)	1 byte
	and Data	x bytes

action is the address of 1-byte action code field containing:
C'1' for add action, or
C'2' for delete action.

If an add or delete action took place, a return code of 0 will be passed to the subsystem in register 15. If the operation was not successful, one of the following return codes will be passed in register 15:

1--invalid action code or location addresses

2--no match on Item Code for delete action

3--no room to add variable length data field.

9.8 SEGMENTED MESSAGE INPUT (GETSEG)

In designing the message processing logic for an application subsystem, the possibility of receiving a multisegmented message for processing must be considered. This type of input message requires a special Intercomm service routine module GETSEG. When an application subsystem receives the first segment of a multisegmented message, identified by a value of 0 (X'F0') in the message header MSGHQPR field, it must call the GETSEG subroutine in order to receive the remaining segments of the message. GETSEG must be called for each message segment (intermediate segment MSGHQPR=1), until the final segment is obtained (MSGHQPR=3).

Coding format:

```
[symbol] CALL  GETSEG,(msgarea,ret-code,sctaddr),VL,MF=(E,list)
```

where:

msgarea is the name (address) of an area, defined within an area acquired by a STORAGE macro or in the dynamic save/work area, in which to place the next message segment

ret-code is the address of a one-byte area into which GETSEG passes a return code

sctaddr is the address of the area passed upon entry to the subsystem which defines the Subsystem Control Table (SCT) entry for the application program.

As a result of this call, GETSEG will obtain the next message segment and will call the Edit Utility (if the message requires editing) before passing it to the subsystem for processing. If the incoming message goes through the Edit Utility, care should be used in selecting parameters used by the Edit Utility; they must appear in identical form in every segment of the message. The return code passed to the application subsystem by GETSEG in the specified area, is in character format. The possible return codes are listed below:

Return Code	Meaning
C'0'	Message is present.
C'4'	There is no message present.
C'8'	There is a message present, but core is not available. The subsystem should return to the Subsystem Controller with an identical return code.
C'9'	Message is present, but Edit Utility could not process it.

Figure 46. GETSEG Return Codes

9.8.1 Segmented Message Output Terminal Assignment (DVASN)

The DVASN message processing service routine is used in conjunction with the Output Utility. DVASN is called by a subsystem to obtain exclusive use of a terminal for the purpose of transmitting a multisegment message without interruption, that is, without interleaving of messages (such as printer report pages queued by other subsystems). The DVASN subroutine (module name PMIDVASN) must be called before queuing (via MSGCOL) the first segment of a multisegment message for formatting by Output.

Coding format:

```
[symbol] CALL  DVASN,(cmp,SPA,term,oft,ret),VL,MF=(E,list)
```

where:

cmp is the address of a halfword field containing the number of the company or division being serviced (obsolete - use binary zeros for the number).

SPA is the address of the System Parameter Area.

term is the address of a field containing the destination terminal (broadcast group) name, or of the field MSGHTID in the message header.

oft is the address of a halfword field containing the OFT number of the format about to be started.

ret is the address of a five-byte field in which will be returned the assigned terminal-ID (alternate tid-name if original down in Back End Station Table), or binary zeros if not found.

As a result of this call, DVASN will assign the terminal to the subsystem and designate it in a "multi-segmented-message-transmission-in-progress" condition in its respective entry in the Back End Station Table (see Chapter 2). This action thus prevents other messages from being transmitted to the designated terminal until its busy status is subsequently freed by the Output Utility (only in Control Region in a Multiregion environment).

9.9 DISPATCHER RELATED ROUTINES

Three Intercomm service routines are available to an application subsystem during execution. They are Dispatcher-related in that they were developed for internal use, but are also applicable within message processing logic.

- IJKPRINT formats a print line for SYSPRINT and includes page overflow logic.
- IJKTRACE provides a list of Dispatcher task queues for purposes of debugging. IJKTRACE utilizes IJKPRINT.
- IJKDELAY provides a simple means of coding a time delay (for multi-threading) of approximately 100 milliseconds within an application subsystem.

9.9.1 IJKPRINT - Direct Output Line to SYSPRINT

This subroutine calls the PUT entry point in the File Handler to output a print line image of IBM standard format V (variable-length) records, with an ASA printer spacing control character as the first text byte. (Maximum logical record length is 137.) A count is maintained of the number of lines printed on the text page; when the count exceeds a pre-assembled value, the next line output will specify a skip to head of form (ASA control character '1'), and the line count will be reset.

Output is directed to the file with ddname SYSPRINT. If the file is undefined or incorrectly defined, no output is produced, but no diagnostic indication is given. The DD statement for SYSPRINT is described in the Operating Reference Manual. The call made to the File Handler refers to the output file by name without the use of a File Handler work area, thus causing the File Handler to bypass the use of the Dispatcher to accomplish multi-tasking; control is retained in the calling program path.

Coding format:

```
[symbol]  CALL  IJKPRINT,(print-line),VL,MF=(E,list)
```

where:

print-line is the address of the format V record (line-image) to be directed to SYSPRINT. (Variable-length record formats are described in Chapter 6.)

9.9.2 IJKTRACE - Print Dispatcher Queues

This subroutine constructs print line images producing a formatted display of all Dispatcher task queues. It is called automatically whenever the program check handler (SPIESNAP) is entered, or a subsystem time-out occurs, and may also be called for diagnostic purposes by any other program.

Control is retained in the current program path for the duration of processing by this module; the dispatcher is not entered and no other system work is performed.

Coding format:

```
[symbol]  CALL  IJKTRACE
```

Each print line image is passed to the module IJKPRINT for output to the SYSOUT data set called SYSPRINT. The Operating Reference Manual details the exact format of IJKTRACE output.

9.9.3 IJKDELAY - Request Time Delay

This module may be called to introduce a timed delay averaging 100 milliseconds into a program path. The Dispatcher is given control to perform other processing and returns at the expiration of the delay interval.

This facility may be utilized to give a time-slicing effect within a routine which would otherwise monopolize the use of CPU time. It can also force the buildup of parallel program paths for reentrancy testing purposes in an environment which otherwise might not result in actual parallel execution, or it may be invoked to await the passing of a temporary condition which will be resolved by another scheduled program.

Coding format:

```
[symbol]  CALL  IJKDELAY
```

9.10 IN-CORE TABLE SORT FACILITY (INTSORT)

To sort an in-core table, the INTSORT Facility is provided. Such a table might be data stored in a Store/Fetch string or file data record via online transactions or offline processing. The table can have any number of fixed-length entries up to 32767, and each entry can have a total size of 1 to 255 bytes. The key to be sorted on can be anywhere within the entry, but must be in the same place, and of the same length, in each entry. Coding format:

```
[symbol] CALL  INTSORT,(entries,entry-length,table,key-offset,
               key-length),VL,MF=(E,list)
```

where:

entries is a fullword (aligned) containing the number of table entries (up to 32767).

entry-length is a fullword (aligned) containing the size of each entry (up to 255).

table is the address of the area containing the table to be sorted.

key-offset is a fullword (aligned) containing the offset (-1) of the key within each entry (value must be zero if at the beginning of the table entry).

key-length is a fullword containing the length of the key (to be sorted on) of each entry (can be the same as entry-length).

The return code is a fullword binary value returned in register 15, as listed in Figure 47. For all non-zero return codes, the sort is not executed.

Return Code	Meaning
X'00'	INTSORT completed successfully
X'04'	Number of entries less than 1 or more than 32767.
X'08'	Length of an entry is less than 1 or greater than 255.
X'12'	No table address supplied.
X'16'	Key-offset greater than 254.
X'20'	The key-length plus key-offset exceeds maximum (255) entry-length.

Figure 47. INTSORT Return Codes

9.11 OTHER INTERCOMM SERVICE FACILITIES

The following service routines for application programs are accessed via the following subroutine entry names:

- MMU (MAPIN, MAPOUT, MAPEND, MAPCLR, MAPURGE, MAPFREE)
- Store/Fetch (INTSTORE, INTFETCH, INTUNSTO)
- DDQ (QBUILD, QOPEN, QREAD, QREADX, QWRITE, QWRITEX, QCLOSE)
- Page Facility (PAGE)
- DBMS (DBINT) - data base interfacing
- Dynamic File Allocation (ALLOCATE, ACCESS)
- ESS operator-id checking (SECUSER)

Detailed documentation for use of the above facilities is provided in separate manuals (see Chapter 2). Special coding and call conventions for specific data base support are described in Data Base Management System Users Guide and vendor manuals.

Macros to access other system routines and services are described in Chapter 10. In addition, the following facilities may be of interest:

- Assigning a verb to a terminal (see Chapter 8)
- Locating a Csect (and Entry) name from a hexadecimal address (IJKWHOIT - see Operating Reference Manual).
- Generating a system command message (see System Control Commands)
- Generating a display or printout of dynamic storage (save/work area) during program execution via the SCTL system control command (Release 10 only). Acquire and initialize storage for the command message and queue it for the SYSCNTL subsystem (as released, SUBH=000, SUBC=C). Use the form:

  SCTL$DSPCH$address(len)[$address(len)...]$TID=name@

  See the Release 10 System Control Commands for syntax details and length restrictions. Use the HEXCON or LAYOUT macro to convert a binary address to a printable hex format in the message text. Omit the TID parameter if the storage is to be displayed at the subsystem's input message terminal.

9.12 LOADING SERVICE ROUTINE ENTRY POINTS FROM THE SPA

Several Intercomm service routines may optionally be assigned to the Link Pack Area. (Refer to the Operating Reference Manual.) In such cases, the service routine entry point should not be coded in the CALL macro; rather, the address of the service routine is loaded from the SPA or SPAEXT into register 15, and the CALL macro is coded with register notation, as follows:

```
L     R15,SPA...              load service routine address from SPA
                                        or
L     R15,SPAEXTAD            point to SPA Extension
USING SPAEXT,R15              set addressability
L     R15,SEX...              load service routine address from SPAEXT
                                       then
CALL  (15),(parameters),VL,MF=(E,list)
DROP  R15                     cancel SPAEXT addressability
```

Appendix D lists the SPA and SPAEXT field names for service routine entry points referenced in this manual. For dynamically loaded subsystems, this sequence of instructions saves dynamic linkedit time at Intercomm startup, unless the subsystem has been linked with the INTLOAD system program which performs a similar function.

The above coding sequence may not be used if the subsystem is eligible for loading above the 16meg line under MVS/XA or ESA; service routine entry names must be used. Mode switching is accomplished by linking the subsystem with INTLOAD (also contains XA interface code). Also, do not code the SPA or SPAEXT parameters on any Intercomm macros except the LINKAGE and SUBLINK macros.

Chapter 10

INTERCOMM MACROS FOR ASSEMBLER LANGUAGE PROGRAMS

10.1 INTRODUCTION

Intercomm provides many macro instructions to facilitate coding of user Assembler Language programs (subsystems, subroutines, user exits). Coding specifications for each macro are detailed in Basic System Macros, which should be referenced in conjunction with this chapter. Several different categories of macros are provided:

- Basic macros for subsystem structure (LINKAGE, RTNLINK, STORAGE, STORFREE), discussed in Chapter 3.

- Macros to simplify program coding, virtually self-explanatory in their use:

  -- CALLIF — Conditional call, transfers control if subroutine is linkedited in the Intercomm load module

  -- CALLOVLY — Call a subroutine which may be linkedited within the transient subroutine overlay region

  -- DDNFIND — Test presence of a ddname in execution JCL

  -- EXMVE — Extended MVC to move n characters, where n may be greater than 256

  -- EXSS — Execute a storage-to-storage instruction

  -- EXTRT — Extended translate and test

  -- GETDATE — Get current date from CPU (yyddd)

  -- GETSPA — Find Intercomm SPA address (Link Pack Area module)

  -- HEXCON — Convert data from binary to printable hexadecimal

  -- INTTIME — Optimized TIME macro to request CPU time, date

  -- LAYOUT — Transfer data fields to a contiguous area of core, effecting conversion of format

  -- MSGHDR — Establish symbolic names for the fields in Intercomm's message header

--	REGA, REGS	Generate register name equates
--	ROUND	Round a register value to a power of 2
--	SECTEST	Test user function authority under ESS (Extended Security System)
--	SSCONV	Convert subsystem codes to printable form
--	SSSTART	Start a STRT/STOP function under program control
--	SSSTOP	Stop a STRT/STOP function under program control
--	SSTEST	Test for bit settings controlled by the General Purpose Subsystem
--	SUBLINK	Provide subroutine linkage (similar to LINKAGE macro). Note: can be paired with the RTNLINK macro
--	XASWITCH	Switch MVS/XA address modes (24/31)

- Macros for debugging functions:

--	PMISNAP	Issue a snap
--	PMIWTO, PMIWTOR	Issue a WTO (WTOR)
--	USRTRACK	Track user data for SAM (System Accounting Facility)

- Macros which request system control functions to be performed:

  -- DISPATCH — Create or cancel a Dispatcher task

  -- INTWAIT — Wait on event completion, or request a timed processing delay

  -- INTPOST — Post internal event

  -- PASS, CATCH — Transfer control of a system resource for a message processing thread to or from the Intercomm system (thread 0)

  -- INTENQ, INTDEQ — Enqueue/dequeue request for exclusive or shared control of a resource

  -- SUBTASK — Create an application program subtask

  -- MODCNTRL — Request LOAD/LINK/DELETE of a user subroutine defined in REENTSBS table

The macros used for debugging and to request system control functions are described in alphabetical order. Illustrations of the use of several of these macros for system control functions conclude this section.

10.2 MACRO DESCRIPTIONS

10.2.1 CATCH -- Transfer Ownership of a Storage Area from Intercomm to an Application

The CATCH macro is issued by an application, in a system with the Resource Management Auditing and Purging facility in use, to take control of an area of storage belonging to Intercomm. CATCHing an area ensures that it will be freed by the Resource Management purge routine if the subsystem program checks or times out.

10.2.2 DISPATCH -- Request Multithread Dispatcher Queuing Service

The DISPATCH macro provides the facility for requesting one of several queuing services from Intercomm's Multithreading Dispatcher. The following types of service requests are available:

- A request for a unit of work to be placed on a specific priority execution queue and executed as soon as priority permits
- A request for a unit of work to be placed on a timer queue and executed upon the elapse of a specified duration of time
- A request for a unit of work to be placed on an event queue and executed upon the completion of a specified event
- A request to delete (cancel) a previously queued request
- A request to terminate control and initiate processing by the highest priority unit of work awaiting execution

Three kinds of queues exist: event, timer and execution queues. There are two event queues, one timer queue, and four execution queues, corresponding to the highest-lowest Intercomm priority codes of 0, 1, 2, 3. All units of work placed on an event or timer queue remain queued until the event transpires or the duration expires. They are then, depending upon assigned priority, transferred to one of the execution queues.

The INTWAIT macro may be used in lieu of the DISPATCH macro for timer or event waiting.

NOTE: The IBM STIMER and TTIMER macros are not allowed to be issued by any user program.

10.2.3 INTDEQ -- Dequeue

The INTDEQ macro is used in conjunction with the INTENQ macro in order to signal to the enqueuing-dequeuing module that a requestor, having already been INTENQed and subsequently granted access to a resource, has no further need of that resource. For every INTENQ macro issued for a resource, there must be an INTDEQ macro subsequently issued for the same resource. If, after the issuance of an INTDEQ macro, the enqueuing-dequeuing module identifies a requestor as having been previously INTENQed, and not having timed out, register 15 will contain a return code of 0. If, however, a previous INTENQ was not issued, or if the previous INTENQ request timed out, register 15 will contain a return code of 4.

10.2.4 INTENQ -- Enqueue

The INTENQ macro is used in conjunction with the INTDEQ macro to serialize the use of a particular resource and, if necessary, delimit the number of concurrent users of that resource. The INTENQ macro is essentially a request to be placed upon a resource queue. Control is not returned to the issuer of INTENQ until all previous requestors on that queue have been given resource access. However, if the SHARE parameter is coded, all previous requestors may or may not have dequeued themselves by the time control is received. When a requestor is placed upon a queue, all registers are saved, therefore register 13 must point to a save area. The INTENQ macro expansion uses registers 0, 1, 14 and 15. No return code is employed, except if the TEST option is used.

10.2.5 INTPOST -- Post Internal ECB

The INTPOST macro is used to post an ECB awaited via the INTRNL=IPOST option of the DISPATCH or INTWAIT macros. This provides the most efficient synchronization technique for two threads within the same Intercomm task. INTPOST may also be used when INTRNL=YES was specified on the DISPATCH or INTWAIT macros. If the object ECB is already posted, then no over-posting will take place.

10.2.6 INTWAIT -- Temporarily Relinquish Control

The INTWAIT macro causes the issuing module to temporarily relinquish control until either an ECB is posted or a time interval expires. It assumes only that the caller's register 13 is pointing to a save area.

This macro is a convenient way to replace the following frequently occuring coding sequence:

```
        STM   14,12,12(13)
        DISPATCH 'S',label,(13),EXIT,{ECB=  }
                                     {INTVL=}
label   LR    13,1
        LM    14,12,12(13)
```

10.2.7 MODCNTRL--Control Dynamically Loaded Subroutines

The MODCNTRL macro requests loading or linking, and then deleting, of separately linkedited user-written load modules. The referenced subroutines or tables must be defined using the SUBMODS macro within the REENTSBS table. Register 15 is set to X'FFFFFFFF' when the SUBMODS entry cannot be found, or if the requested module is not available. No other return code is set in register 15. If the module may be loaded above the 16meg line under XA, a requesting program executing in 24-Amode is responsible for address mode switching using the XASWITCH macro when the LOAD option is used.

10.2.8 PASS--Transfer Ownership of a Storage Area from an Application to Intercomm

The PASS macro is used, in a system with Resource Auditing and Purging, to protect an area of core acquired by an application thread from being freed by the purge routine when the thread completes. That is, the ownership is passed to the Intercomm system thread (thread 0). (See also STORAGE macro, SYS parameter).

10.2.9 PMISNAP--Issue a Snap

The PMISNAP macro should be used by subsystems in place of the IBM SNAP macro. It deducts the time taken by the snap operation from total elapsed time, thereby avoiding a subsystem time-out which could occur when taking a snap.

10.2.10 PMIWTO--Write to Operator

The PMIWTO macro generates a parameter list and call to the Intercomm module WTOMOD, which centralizes all WTOs to the CPU console, and/or the control terminal, and/or SYSPRINT, based on macro coding options. Do not use the PMIWTO macro if issuing a multiline WTO; use the IBM WTO macro.

10.2.11 PMIWTOR--Write to Operator with Reply

The PMIWTOR macro generates a parameter list and calls the Intercomm module WTOMOD. The program issuing the macro may regain control after the reply takes place by issuing a DISPATCH or INTWAIT macro to wait on the ECB specified in the macro parameters. If issuing a multiline WTOR, do not use this macro; use the IBM WTOR macro.

10.2.12 SUBTASK--Dynamic Subtasking

The SUBTASK macro allows part of a thread's logic to execute as a MVS subtask. The program linkage between the main Intercomm task and the subtasked logic may be viewed as being equivalent to a call to a subroutine. Registers 1 and 13 can be used as if a call was issued. SUBTASK generates the instruction BALR 14,15 with Register 15 containing the ENTRY parameter value.

All registers are passed from the main task portion of the application subsystem to the subtask. Intercomm suspends execution of the SUBTASKing thread (via DISPATCH WAIT) until the subtasked code completes, but other threads, perhaps of the same subsystem, may be processed by the main task during this time. When the subtasked code completes, register contents when the SUBTASKing thread is redispatched depends upon what action the subtasked code took to save and restore them.

The subtasked code can be a piece of code in the application subsystem itself or an external subroutine. The main use of the SUBTASK macro is to allow code which may impede Intercomm performance to be executed in a subtask, instead of slowing the main task. Usually, the code would include SVCs with implied WAITS in them. For example, if an application subsystem wanted to issue a LOAD macro, SUBTASK could be used to allow the main task to continue while the subtask would be held up by MVS until the LOAD completed. The Intercomm File Handler subtasks the issuance of QSAM GETS in this manner.

A SUBTASK macro may not be issued by any program eligible for loading above the 16meg line under XA.

10.2.13 USRTRACK-- Track User Data Using SAM

If the System Accounting and Measurement Facility (see Operating Reference Manual) is installed and activated for the subsystem (SYCTTBL macro, SAM parameter), this macro can be used to increment a user-defined accumulator (for Data Base calls, for example) or to invoke a user SAM exit routine.

10.3 MACRO CODING EXAMPLES

See Chapter 3 regarding usage of macros by programs loaded above the 16meg line.

10.3.1 DISPATCH Macro Usage

The following coding examples show several uses of the DISPATCH macro.

In the first example, the programmer wishes to allow other tasks to continue execution while this routine waits for the completion of an event (an input/output operation, for example). Assume that the ECB address has been previously loaded into general register 8:

```
*USING DISPATCH TO WAIT FOR AN EVENT

WAIT     STM          2,12,28(13)              SAVE REGISTERS
         DISPATCH     'S',DONE,(13),EXIT,ECB=(8)
DONE     LR           13,1                     RESTORE REGISTER 13
         LM           2,12,28(13)              RESTORE REGISTERS
```

The routine in progress uses its own save area to store all necessary registers before exiting, and restores the registers after regaining control. The address of the program's save area is the parameter passed through the Dispatcher. This example is reentrant. Note that this code could be replaced by an INTWAIT macro.

In the second example, the programmer wishes to allow other tasks awaiting CPU time to be dispatched, returning to this routine after the execution of higher or equal priority tasks which were awaiting events that may by now have been completed, or after the execution of equal or higher priority tasks which this routine may have just previously placed on the Dispatcher execution queue. The programmer wishes to give this task the same priority it had received when it gained control ('S' parameter):

```
*USING DISPATCH FOR TASK ROTATION

ROTATE   STM          2,12,28(13)              SAVE REGISTERS
         DISPATCH     'S',RESUME,(13),EXIT
RESUME   LR           13,1                     RESTORE REGISTER 13
         LM           2,12,28(13)              RESTORE REGISTERS
```

In the third example, the programmer wishes to schedule the execution of a subprogram which will be executed concurrently with the program or after the program terminates, depending upon Dispatcher scheduling. Assume that the address of a calling program parameter list has been preloaded into general register 1:

```
*USING DISPATCH TO SCHEDULE SUBPROGRAM

         L            0,=V(SUBPROG)
SCHED    DISPATCH     'S',(0),(1),SYS=YES
```

The execution of this program continues. The subprogram gains control after this program returns to the Monitor, or if this program gives up control in any way (I/O operation through the File Handler, task rotation, etc.). The SYS parameter ensures that the dispatched routine will not be purged from its execute queue if the issuing program completes before the subprogram is given control. Note also that the passed parameter list, and the parameter values, may not be in dynamic storage owned by the issuing program (see also STORAGE and STORFREE macros, SYS parameter). The dispatched subprogram will receive control in thread zero (0), and execute as a system program.

In the fourth example, the programmer does not wish to continue processing until three events (ECB1, ECB2, and ECB3) have all completed. Control returns to the issuing program when all three ECBs have been posted:

```
*USING DISPATCH FOR A MULTIPLE WAIT

MULTWAIT DISPATCH     'S',COUNT,(13),ECB=ECB1
         DISPATCH     'S',COUNT,(13),ECB=ECB2
         DISPATCH     'S',COUNT,(13),ECB=ECB3
         LA           12,3                    SET COUNTER TO 3
WAIT     STM          2,12,28(13)             SAVE REGISTERS
         DISPATCH     EXIT                    EXIT
COUNT    LR           13,1                    RESTORE REGISTER 13
         LM           2,12,28(13)             RESTORE REGISTERS
         BCT          12,WAIT                 DECREMENT COUNTER &
                                              BRANCH NOT ZERO
```

10.3.2 PASS/CATCH Macro Usage

One subsystem acquires (and initializes) an area of core (by issuing a STORAGE macro) and passes the address of that core via message switching to another subsystem.

Subsystem A

```
     .
     .
     .
     STORAGE  LEN=256,ADDR=(R1),
              LIST=PARMSAVE
     LR       R11,R1
     .
     .
* INITIALIZE STORAGE AREA
     .
     .
     .
* CREATE MESSAGE FOR SUBSYSTEM
* B WITH ADDRESS OF ACQUIRED
* CORE IN THE MESSAGE TEXT
     .
     .
     .
     PASS     LEN=256,ADDR=(R11)
     .
* QUEUE THE MESSAGE
     .
     CALL     MSGCOL,.......
     .
```

Subsystem B

```
     .
* GET STORAGE ADDRESS FROM
* INPUT MESSAGE
     L          R7,area-address
     CATCH      LEN=256,ADDR=(R7)
     .
     .
     .
* PROCESS AS IF THE AREA HAD
* BEEN ACQUIRED BY THIS SUB-
* SYSTEM
     .
     .
     .
     .
     .
     .
     .
     .
     .
     .
     .
     .
     STORFREE   LEN=256,ADDR=(R7)
     .
```

NOTE: the PASS macro would be redundant if the storage was acquired using the SYS=YES parameter. However, if the subsystem program checked or timed out, the acquired storage would not be freed and would be permanently allocated. Use the PASS and the call to MSGCOL at the end of the program (just before the RTNLINK).

10.3.3 INTENQ/INTDEQ Macro Usage

One subsystem updates a table entry in USERSPA. During the time of this update, other subsystems should not be allowed access to the table. All subsystems must use the same resource identification. The update subsystem has exclusive control during the time of the update; the access subsystem merely tests to ensure that exclusive control is not in effect before accessing the updated table area, as follows:

Update Subsystem

```
         .
* ENQUEUE RESOURCE
         LA R6,TABLENQ
         INTENQ (R6)
         .
         .
* PERFORM TABLE UPDATE
         .
* DEQUEUE RESOURCE
         INTDEQ (R6)
         .
         .
         .
TABLENQ DC CL16'USERTABLE'
```

Access Subsystem

```
         .
         .
* TEST IF RESOURCE ENQUEUED
         LA R4,TABLENQ
         INTENQ (R4)
         INTDEQ (R4)
* ACCESS TABLE DATA
         .
         .
         .
         .
         .
         .
TABLENQ DC CL16'USERTABLE'
```

10.3.4 MODCNTRL Macro Usage

A user routine for error processing (ERRORRTN) is infrequently used. Therefore, it might be defined in REENTSBS as a routine eligible for dynamic load. A subsystem calling this routine might be coded as follows:

```
 *LOAD or LINK must be paired with DELETE
       MODCNTRL     MODNAME=ERRRTN,ACTION=LOAD
       LTR          R15,R15
       BNZ          NOLOAD
       LR           R15,R1                          address of routine
       CALL         (15),(parameters-for-ERRORRTN),VL,MF=(E,list)
       MODCNTRL     MODNAME=ERRRTN,ACTION=DELETE
NOLOAD DS           0H
       .
       .
ERRRTN DC           CL8'ERRORRTN'
```

Chapter 11

SAMPLE PROCESSING PROGRAMS

The sample program SQASMA, shown in Figure 48, demonstrates coding of a BAL subsystem which is either resident or dynamically loadable below the 16meg line (if MVS/XA or ESA). The program processes an inquiry transaction (MURA) containing a part number and a warehouse number for a stock status display. MMU is used to transform the incoming message into a fixed field format. The part number is transformed into a RBN for accessing a BDAM part description file (PARTFILE). The RBN and a part description record area are passed as parameters to a called BAL subroutine SQASMB, illustrated in Figure 49, which also resides below the 16meg line. The subroutine retrieves the requested record from PARTFILE and passes back the File Handler return code to the calling subsystem via register 15.

Together, the part number and warehouse number provide a VSAM key for accessing a stock status file (STOKFILE). The File Handler is used for accessing both files. MMU is used for formatting an output display. Error messages, for conditions such as non-existent or erroneous warehouse or part numbers, or file I/O errors, are built within the program and formatted by MMU using an error map area.

The MSGHDRC source text member defining the Intercomm message header fields is COPY'd from the Intercomm source library (SYMREL) by the Assembler. The ASMLOGCH source text member used for terminal attribute and command override for MMU processing, and the symbolic map areas, are also copied into the program.

All required table entries, JCL, sample input messages and testing procedures, plus sample execution output, are illustrated in Chapter 12, "Subsystem Testing." The subsystem code used in the SYCTTBL macro to identify the sample subsystem is RA. Intercomm's BTAM simulator is used for testing. Test messages are included to test as many error combinations as possible. Chapter 13 illustrates a similar subsystem (without the subroutine) coded for the same purpose but using the Edit and Output Utilities, a MSGCOL call, and Test Mode for testing.

```
 2 **  SAMPLE REENTRANT ASSEMBLER SUBSYSTEM USING THE FILE HANDLER    **
 3 **  TO ACCESS A VSAM FILE AND SUBROUTINE SQASMB TO ACCESS A BDAM   **
 4 **  FILE. MMU IS USED FOR INPUT AND OUTPUT MAPPING AND MAPPING     **
 5 **  OF ERROR MESSAGES.                                            **
 6 **                  - REGISTER USAGE -                            **
 7 **  R2  I/O MAP                                                   **
 8 **  R3  WORK                                                      **
 9 **  R4  RETURN CODE                                               **
10 **  R5  ERROR MAP                                                 **
11 **  R6  BAL INSTRUCTIONS                                          **
12 **  R7  BAL INSTRUCTIONS                                          **
13 **  R8  PARMS.                                                    **
14 **  R9  INPUT MESSAGE                                             **
15 **  RA  SPA                                                       **
16 **  RB  SPAEXT                                                    **
17 **  RC  BASE REGISTER                                             **
18 **  RD  SAVE AREA (WORKAREA DSECT)                                **
19 **                                                                **


21          PRINT NOGEN
22 SQASMA   CSECT
23          LINKAGE BASE=(RC),SPA=(RA),LEN=DYNLEN,GPREQ=REGA,
                  MSG=(R9),PARM=(R8)
25+         PRINT NOGEN
62+         PRINT NOGEN
67+         PRINT GEN
68+         PUSH  PRINT               TURN OFF PRINT GENERATION
69+         PRINT NOGEN
785+        POP   PRINT               RESUME PRINT GENERATION
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 1 of 15)

```
 812+          PRINT NOGEN
 913           USING INMSG,R9
 914           USING WORKAREA,RD
 915           USING STKSTAT,R2
 916           USING SPAEXT,RB
 917           XR    R4,R4                  INITIALIZE RETURN CODE
 918           LA    R5,WORKLEN+MAP1L(,RD) ADDRESS ERROR MAP
 919           L     RB,SPAEXTAD
 920           BAL   R6,MOVEHDR             MOVE INP HDR TO OUTP HDR
 921           BAL   R6,MAPIN               MAP AND THEN FREE THE INPUT MSG
 922           LA    R6,PRERTN              WHERE TO RETURN
 923           CLI   MCW,C'3'               IF A FIELD IS IN ERROR
 924           BE    INVINPUT                  SEND ERROR MSG
 925           CLI   MCW,C'0'               IF MAPPING NOT OK
 926           BNE   MAPER                     SEND ERROR MSG
 927           BAL   R6,CLEARMAP            CLEAR THE MAP
 928           BAL   R6,RDPARTFL            PREPARE TO READ PARTFILE
 929           BAL   R6,BDAMREAD            READ A RECORD
 930           TM    FHFLAG,BDRDOK          READ OK?
 931           BZ    PRERTN                 NO
 932           BAL   R6,RDSTKFIL            PREPARE TO READ STOCK FILE
 933           BAL   R6,VSAMREAD            READ A RECORD
 934           TM    FHFLAG,VSRDOK          READ OK?
 935           BZ    PRERTN                 NO
 936           BAL   R6,MAPOUT              PREPARE OUTPUT MAP
 937           CLI   MCW,C'0'               OK?
 938           BE    PRERTN                 YUP
 939           BAL   R6,MAPER               NO..SEND ERROR MSG
 940 PRERTN    DS    0H
 941           TM    MAPFLG,NOMAP           UNABLE TO SEND MAP?
 942           BC    RETURN                 YES - JUST FREE MAP AND GO
 943           LA    R6,RETURN              WHERE TO RETURN
 944           TM    MAPFLG,MAPERR          ARE WE SENDING AN ERROR MAP?
 945           BO    ERMAPEND               YES - GO DO IT
 946           BAL   R6,GDMAPEND            SEND GOOD MAP
 947 RETURN    DS    0H
 948 ***    FREE INPUT MAP AREA AND RETURN
 949           XC    MCW,MCW                CLEAR MAP CONTROL WORD
 950           CALL  MAPFREE,(MCW,IOGRP,IOMAP,ADDRMSG,CMSGHTID),          -
                     VL,MF=(E,CALLIST)
 965           RTNLINK ADDR=(RD),LEN=DYNLEN,RC=(R4)
 975+          PRINT NOGEN
1011**,GETSPA - V7.0 - 11/76 - SM
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 2 of 15)

```
1035 MOVEHDR  DS    OH
1036          MVC   OMSGHTID,MSGHTID     SAVE TID
1037          MVC   IOGRP,IOGRPNM        MOVE MAP GROUP NAME TO WORK AREA
1038          MVC   IOMAP,IOMAPNM        MOVE MAP NAME            "    "
1039          MVC   ERMAP,ERMAPNM        MOVE ERR MAP NAME        "    "
1040          BR    R6

1042 MAPIN    DS    OH
1043          ST    R9,ADDRMSG           STORE INPUT MSG ADDRESS
1044          XC    MCW(4),MCW           CLEAR MAP CONTROL WORD
1045          CALL  MAPIN,(MCB,IOGRP,IOMAP,ADDRMSG,MCW),VL,                  R
                    MF=(E,CALLIST)      MAP INPUT MESSAGE
1060          L     R2,ADDRMSG           MAPPED MSG DATA ADDRESS
1061          BR    R6

1063 CLEARMAP DS    OH
1064          XC    MCW(4),MCW           CLEAR MAP CONTROL WORD
1065          MVI   MCWOPT4,C'A'         ONLY CLEAR ATTRIBUTE BYTES
1066          CALL  MAPCLR,(MCW,IOGRP,IOMAP,MAP1,OMSGHTID),VL,               -
                    MF=(E,CALLIST)
1081          BR    R6

1083 RDPARTFL DS    OH
1084          PACK  DWORD,RBNBYTE        PACK RBN #
1085          CVB   R3,DWORD              AND CONVERT TO BINARY
1086          ST    R3,RBNWORD           STORE RBN
1087          MVC   CURRFILE,DDPART      DDNAME OF BDAM FILE
1088          BR    R6

1090 BDAMREAD DS    OH
1091 ** USE MODCNTRL MACRO TO LINK TO SQASMB SUBRTN WHICH READS A BDAM FILE
1092          MVC   SUBNAME,SQASMB       MOVE SUBRTN NAME INTO WORK AREA
1093          MODCNTRL (PARTREC,RBNWORD),VL,MF=(E,CALLIST),                  -
                    ACTION=LINK,MODNAME=SUBNAME
1108          LTR   RF,RF                DID SQASMB GET CONTROL?
1109          BM    NOSUBRTN             NO
1110          B     *+4(RF)              BRANCH ON RETURN CODE
1111          B     SUBOK                0 RC
1112          B     IOERROR              4 RC
1113          B     NOTFOUND             8 RC
1114          B     NODD                 12 RC
1115 SUBOK    DS    OH
1116          CLC   RECPIN,PARTNO        CORRECT RECORD?
1117          BNE   NOTFOUND             NO
1118          MVC   PRTDATA,RECDES       MOVE DATA TO OUTP MSG
1119          MVC   ORDUNT,RECUNT         "    "    "  "    "
1120          MVC   PRTPRC,RECPRC         "    "    "  "    "
1121          OI    FHFLAG,BDRDOK        SHOW EVERYTHINGS OK
1122          BR    R6
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 3 of 15)

```
1124 RDSTKFIL DS    OH
1125          MVC   CURRFILE,DDSTOCK     GET DDNAME OF FILE
1126          MVC   RECWHS,WHSNO         FIND KEY OF RECORD WE WANT
1127          MVC   RECPNO,PARTNO
1128          MVC   KEYFIELD,KEYFLD      AND MOVE IT INTO KEY AREA
1129          BR    R6


1131 VSAMREAD DS    OH
1132          XC    EXTDSCT(48),EXTDSCT CLEAR IT
1133          BAL   R7,SELECT           GO SELECT THE FILE
1134          CLI   FHSTAT1,C'9'        NO DD?
1135          BE    NODD
1136          BAL   R7,VSREAD2          READ A RECORD
1137          CLI   FHSTAT1,C'1'        I/O ERROR?
1138          BE    IOERROR             YES
1139          CLI   FHSTAT1,C'2'        RECORD NOT FOUND?
1140          BE    VSRECNF             NOT FOUND - GO BUILD ERROR MSG
1141          OI    FHFLAG,VSRDOK       INDICATE READ WORKED
1142          MVC   WHSLOC,RECWLC       BUILD OUTPUT MESSAGE
1143          MVC   STKLEV,RECLEV
1144          MVC   DATEDIT,RECLDT
1145          BAL   R7,EDITDATE         MAKE DATE PRINTABLE
1146          MVC   LEVDATE,DATEMOVE
1147          MVC   STKORD,RECORD
1148          MVC   DATEDIT,RECODT
1149          BAL   R7,EDITDATE         MAKE DATE PRINTABLE
1150          MVC   ORDDATE,DATEMOVE
1151 DORLSE   DS    OH
1152          BAL   R7,RELEASE          GO RELEASE THE FILE
1153          BR    R6


1155 VSREAD2  DS    OH
1156          XC    FHSTAT,FHSTAT       CLEAR FH CONTROL WORD
1157          CALL  GETV,(EXTDSCT,FHSTAT,STOCKREC,KEYFIELD),                -
                    VL,MF=(E,CALLIST)
1171          BR    R7                  RETURN TO SUBRTN


1173 SELECT   DS    OH
1174          XC    FHSTAT,FHSTAT       CLEAR FILE HANDLER CONTROL WORD
1175          CALL  SELECT,(EXTDSCT,FHSTAT,CURRFILE),VL,MF=(E,CALLIST)
1187          BR    R7                  RETURN TO SUBRTN


1189 RELEASE  DS    OH
1190          XC    FHSTAT,FHSTAT       CLEAR FILE HANDLER CONTROL WORD
1191          CALL  RELEASE,(EXTDSCT,FHSTAT),VL,MF=(E,CALLIST)
1202          BR    R7                  RETURN TO SUBRTN
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 4 of 15)

```
1204 EDITDATE DS    OH          EDIT DATE TO MM/DD/YY FORM
1205          MVC   DMYEAR,DEYEAR
1206          MVC   SLASH1,SLASH
1207          MVC   DMDAY,DEDAY
1208          MVC   SLASH2,SLASH
1209          MVC   DMMO,DEMO
1210          BR    R7                    RETURN TO SUBRTN


1212 GDMAPEND DS    OH
1213          XC    MCW,MCW               CLEAR MAP CONTROL WORD
1214          MVI   MCWOPT2,C'C'          TRANSMIT ENTIRE MSG
1215          BAL   R7,MAPEND             GO CALL MAPEND
1216          CLI   MCW,C'8'              MAPEND SUCCESSFUL?
1217          BER   R6                    YES
1218          BAL   R7,MAPURGE            NO..PURGE THE MAP
1219          ST    R6,SAVE               SAVE THE LINK REGISTER
1220          BAL   R6,MAPER              PREPARE AN ERROR MSG
1221          BAL   R6,ERMAPEND           SEND THE ERROR MSG
1222          L     R6,SAVE               RESTORE LINK REGISTER
1223          BR    R6                    RETURN TO MAINLINE


1225 ERMAPEND DS    OH
1226          XC    MCW,MCW               CLEAR MAP CONTROL WORD
1227          MVI   MCWOPT2,C'C'          TRANSMIT MSG
1228          MVI   MCWOPT3,WRITE1        OVERWRITE EXISTING SCREEN
1229          BAL   R7,MAPEND             GO CALL MAPEND
1230          CLI   MCW,C'8'              SUCCESSFUL?
1231          BER   R6                    YES
1232          BAL   R7,MAPURGE            NO..PURGE THE MAP
1233          LA    R4,8                  RETURN CODE = 8
1234          BR    R6                    RETURN TO MAINLINE


1236 MAPEND   DS    OH
1237          CALL  MAPEND,(MCB,0,MCW),VL,MF=(E,CALLIST)
1249          BR    R7                    RETURN TO SUBRTN


1251 MAPURGE  DS    OH
1252          CALL  MAPURGE,(MCB),VL,MF=(E,CALLIST)
1262          BR    R7                    RETURN TO SUBRTN


1264 MAPOUT   DS    OH
1265          XC    MCW,MCW               CLEAR MAP CONTROL WORD
1266          CALL  MAPOUT,(MCB,IOGRP,ICMAP,MAP1,MCW,DMSGHTIO),
                    VL,MF=(E,CALLIST)
1282          BR    R6                    RETURN TO MAINLINE
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 5 of 15)

```
1284 ****  ERROR ROUTINES  ****
1285 *
1286 *
1287 *
1288           DROP  R2
1289           USING ERRMAP,R5
1290 INVINPUT DS    OH
1291           CLC   PARTNOT-MAP1(1,R2),WHSNOT-MAP1(R2) BOTH NON-NUMERIC?
1292           BNE   ONLY1                 NO - ONLY 1 IS
1293           MVC   MSG7,MSGI             MOVE IN APPROPRIATE MSG
1294           B     GOSNDMSG              MAP THE ERROR MSG
1295 ONLY1    DS    OH
1296           MVC   MSG7,MSGG             MOVE IN APPROPRIATE MSG
1297           BH    GOSNDMSG              PARTNO NOT NUMERIC
1298           MVC   MSG7,MSGH             WHSNO NOT NUMERIC
1299 GOSNDMSG DS    OH
1300           MVC   ERRMSG,INVINMSG       MOVE MSG INTO MAP
1301           BAL   R7,SENDERR            GO CALL MAPOUT
1302           BR    R6                    RETURN TO MAINLINE


1304 NOSUBRTN DS    OH
1305           MVC   MSG8,MSGJ             BUILD ERROR MSG
1306           MVC   NOFILE,CURRFILE
1307           MVC   MSG9,MSGK
1308           MVC   ERRMSG(L'NOSUBMSG),NOSUBMSG MOVE INTO MAP
1309           BAL   R7,SENDERR            DO MAPOUT
1310           BR    R6                    RETURN TO MAINLINE


1312 VSRECNF  DS    OH
1313           MVC   MSG3,MSGC             NOT FOUND - BUILD ERROR MSG
1314           MVC   MSG4,MSGD
1315           MVC   NOWARENO,PARTNO-MAP1(R2)
1316           MVC   NOWARWHS,WHSNO-MAP1(R2)
1317           MVC   ERRMSG(37),NOWARMSG MOVE MSG INTO ERROR MAP
1318           BAL   R7,SENDERR            GO MAP ERROR MESSAGE
1319           B     DORLSE


1321 NOTFOUND DS    OH
1322           MVC   MSG1,MSGA             BUILD ERROR MSG
1323           MVC   MSG2,MSGB
1324           MVC   NOPART,PARTNO-MAP1(R2)  MOVE IN MISSING PART #
1325           MVC   ERRMSG(L'NOPRTMSG),NOPRTMSG MOVE INTO MAP
1326           BAL   R7,SENDERR            DO MAPOUT
1327           BR    R6                    RETURN TO MAINLINE
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 6 of 15)

```
1329 NODD      DS    OH
1330           MVC   CANCODE,=CL15'NO DD FOR FILE'
1331           B     CONTIN
1332 IOERROR   DS    OH
1333           OI    FHFLAG,IOERR          INDICATE AN IO-ERROR
1334           MVC   CANCODE,=CL15'IO ERROR ON'
1335 CONTIN    DS    OH
1336           MVC   MSG5,MSGE             MOVE IN APPROPRIATE MSG
1337           MVC   CANFLNM,CURRFILE      MOVE IN FILE NAME
1338           MVC   ERRMSG(L'CANMSG),CANMSG MOVE INTO MAP
1339           BAL   R7,SENDERR            DO MAPOUT
1340           TM    FHFLAG,BDRDOK+IOERR   IF IOERROR ON VSAM FILE
1341           BO    DORLSE                  THEN GO RELEASE IT
1342           BR    R6                    RETURN TO MAINLINE


1344 MAPER     DS    OH
1345           MVC   MSG6,MSGF             MOVE IN APPROPRIATE MSG
1346           MVC   ERRTAG,MCW            MOVE BYTES 1 & 2 INTO MSG
1347           MVC   ERRMSG(L'MAPERMSG),MAPERMSG MOVE INTO MAP
1348           BAL   R7,SENDERR            DO MAPOUT
1349           BR    R6                    RETURN TO MAINLINE


1351 SENDERR   DS    OH
1352           XC    MCW,MCW               CLEAR MAP CONTROL WORD
1353           OI    MAPFLG,MAPERR         INDICATE MAPOUT FOR ERROR MAP
1354           CALL  MAPOUT,(MCB,IOGRP,ERRMAP,ERRMAP,MCW,OMSGHTID),          -
                     VL,MF=(E,CALLIST)
1370           CLI   MCW,C'0'              MAPOUT OK?
1371           BER   R7                    YES
1372           LA    R4,8                  NO - RETURN CODE = 8
1373           OI    MAPFLG,NOMAP          SHOW NO MAP SENT
1374           BR    R7                    RETURN TO SUBRTN
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 7 of 15)

```
1376            PRINT GEN
1377 **         CONSTANTS
1378 SLASH      DC    C'/'
1379 DDSTOCK    DC    C'STOKFILE'
1380 DDPART     DC    C'PARTFILE'
1381 IOGRPNM    DC    CL8'STKSTAT'
1382 IOMAPNM    DC    CL8'MAP1'
1383 ERMAPNM    DC    CL8'ERRMAP'
1384 SQASMB     DC    CL8'SQASMB'
1385 MSGTBL     DS    0CL206
1386 MSGA       DC    C'PART NUMBER '
1387 MSGB       DC    C' NOT FOUND.'
1388 MSGC       DC    C'PART '
1389 MSGD       DC    C' NOT FOUND IN WAREHOUSE '
1390 MSGE       DC    C'. MESSAGE CANCELLED.'
1391 MSGF       DC    C'MAP ERROR MCW IS '
1392 MSGG       DC    CL50'INVALID DATA: PARTNO MUST BE NUMERIC'
1393 MSGH       DC    CL50'INVALID DATA: WHSNO MUST BE NUMERIC'
1394 MSGI       DC    CL50'INVALID DATA: PARTNO AND WHSNO MUST BE NUMERIC'
1395 MSGJ       DC    C'SUBROUTINE TO READ '
1396 MSGK       DC    C' NOT AVAILABLE'
1397            COPY  ASMLOGCH              SYMBOLIC CONTROL CHARS AND ATTRIBS.

1399 *          LOGICAL ATTRIBUTE BYTE DEFINITIONS FOR IBM3270
1400 *
1401 UAN        EQU   1  UNPROT/ALPHA/NORMAL
1402 UANMDT     EQU   2  UNPROT/ALPHA/MDTON
1403 UANSEL     EQU   3  UNPROT/ALPHA/SELPEN
1404 UANMDSEL   EQU   4
1405 UAHSEL     EQU   5
1406 UAHMDSEL   EQU   6
1407 UAX        EQU   7
1408 UAXMDT     EQU   8
1409 UNN        EQU   9
1410 UNNMDT     EQU   10
1411 UNNSEL     EQU   11
1412 UNNMDSEL   EQU   12
1413 UNHSEL     EQU   13
1414 UNHMDSEL   EQU   14
1415 UNX        EQU   15
1416 UNXMDT     EQU   16
1417 PAN        EQU   17
1418 PANMDT     EQU   18
1419 PANSEL     EQU   19
1420 PANMDSEL   EQU   20
1421 PAHSEL     EQU   21
1422 PAHMDSEL   EQU   22
1423 PAX        EQU   23
1424 PAXMDT     EQU   24
1425 PSN        EQU   25
1426 PSNMDT     EQU   26
1427 PSNSEL     EQU   27
1428 PSNMDSEL   EQU   28
1429 PSHSEL     EQU   29
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 8 of 15)

```
1430 PSFMDSEL EQU   30
1431 PSX      EQU   31
1432 PSXMDT   EQU   32
1433 SUPR     EQU   33


1435 *   LOGICAL COMMAND CHARACTER DEFINITIONS FOR IBM3270
1436 *
1437 WRITE1   EQU   1
1438 ERASWRIT EQU   2
1439 ERASWRAL EQU   3


1441 *   LOGICAL CONTROL CHARACTER DEFINITIONS FOR IBM3270
1442 *
1443 RMDT     EQU   1
1444 RKEYBD   EQU   2
1445 RMDTKEYB EQU   3
1446 ALARM    EQU   4
1447 ALRMRMDT EQU   5
1448 ALRMRKEY EQU   6
1449 ALRMRMKY EQU   7
1450 PRNTNL   EQU   8
1451 PRNT40   EQU   9
1452 PRNT64   EQU   10
1453 PRNT80   EQU   11
1454 PRNLRMDT EQU   12
1455 PR40RMDT EQU   13
1456 PR64RMDT EQU   14
1457 PR80RMDT EQU   15
1458 PRNLRKEY EQU   16
1459 PR40RKEY EQU   17
1460 PR64RKEY EQU   18
1461 PR80RKEY EQU   19
1462 PRNLRMKY EQU   20
1463 PR4CRMKY EQU   21
1464 PR64RMKY EQU   22
1465 PR80RMKY EQU   23
1466 PRNLALRM EQU   24
1467 PR4CALRM EQU   25
1468 PR64ALRM EQU   26
1469 PR80ALRM EQU   27
1470 PRNLARMD EQU   28
1471 PR4CARMD EQU   29
1472 PR64ARMD EQU   30
1473 PR8CARMD EQU   31
1474 PRNLARKY EQU   32
1475 PR40ARKY EQU   33
1476 PR64ARKY EQU   34
1477 PR8CARKY EQU   35
1478 PRNLAMKY EQU   36
1479 PR4CAMKY EQU   37
1480 PR64AMKY EQU   38
1481 PR80AMKY EQU   39
1482 NULL     EQU   40
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 9 of 15)

```
1484 *         LOGICAL ATTRIBUTE BYTE DEFINITIONS FOR DS40
1485 *
1486 *  UAN FOR DS40=UNPROT/ALPHA/NORMAL
1487 *  UANMDT FOR DS40=UNPROT/ALPHA/MDTON
1488 *  UANSEL FOR DS40=UNPROT/ALPHA/SELPEN


1490 *   LOGICAL COMMAND CHARACTER DEFINITIONS FOR DS40
1491 *
1492 *  WRITE1 FOR DS40=HOME CURSOR ONLY (ESC,H)
1493 *  ERASWRIT FOR DS40=ESC,R=HOME CURSOR,CLEAR SCREEN


1495 *   LOGICAL CONTROL CHARACTER DEFINITIONS FOR DS40
1496 *


1498 *         LOGICAL ATTRIBUTE BYTE DEFINITIONS FOR IBM3270P
1499 *


1501 *   LOGICAL COMMAND CHARACTER DEFINITIONS FOR IBM3270P
1502 *


1504 *   LOGICAL CONTROL CHARACTER DEFINITIONS FOR IBM3270P
1505 *
1506 NL        EQU   51
1507 FF        EQU   52
1508 CR        EQU   53
1509 SI        EQU   54
1510           DC    C'END OF WORKING STORAGE'
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 10 of 15)

```
1512           LTORG
1513                 =V(PMIRTLR)
1514                 =V(DYNLLOAD)
1515                 =CL15'NO DD FOR FILE'
1516                 =CL15'IO ERROR ON'
1517 INMSG    DSECT
1518           COPY  MSGHDRC             MESSAGE HEADER DSECT
1519 *
1520 *       MESSAGE HEADER LAYOUT
1521 *       *********************
1522 *                                 LAST REVISION 10/20/82-RELEASE 9.C
1523 *                                 LAST REVISION 07/30/85-LU 6.2 SUPPORT
1524 *
1525 MSGHLEN  DS    BL2         LENGTH OF MESSAGE
1526 MSGHQPR  DS    BL1         CTAM/BTAM I/O PREFIX BLANK IF SS MSG
1527 MSGHRSCH DS    XL1         HI-ORDER BYTE OF RECEIVING SUBSYSTEM CODE
1528 MSGHRSC  DS    CL1         RECEIVING SUBSYSTEM CODE
1529 MSGHSSC  DS    CL1         SENDING SUBSYSTEM CODE
1530 MSGHMMN  DS    OBL3        MONITOR SEQUENCE NUMBER                    X1078
1531 MSGHTXTL DS    BL2               RECORD LENGTH (FILE RECOVERY)        X1078
1532 MSGHKEYL DS    CL1               KEY LENGTH (FILE RECOVERY)           X1078
1533 MSGHDAT  DS    OCL6        DATE (YY.DDD)                              X1078
1534 MSGHYR   DS    CL2               YEAR                                 X1078
1535 MSGHTHRD DS    BL1               THREAD NUMBER                        X1078
1536 MSGHDAY  DS    CL3               DAY                                  X1078
1537 MSGHTIM  DS    CL8         TIME    (HH.MM.SS)
1538           ORG   MSGHTIM            FIELDS USED IN SCANVERB DURING     JA
1539 *                           CONSTRUCTION OF MESSAGE IN LINE HANDLERS   JA
1540 MSGHVFLG DS    B                 FLAGS                                JA
1541 MSGHVFND EQU   X'80'          VERB WAS ANALYZED BEFORE CALLING BTSEARCH JA
1542 MSGHVBA  DS    AL3         A(BTVERB ENTRY) IF MSGHVFND FLAG ON        JA
1543           ORG   MSGHTIM+L'MSGHTIM                                     JA
1544 MSGHTID  DS    CL5         TERMINAL ID   (AAANN)  AAA=CITY,NN=DEVICE ID
1545 MSGHMRDX DS    OX          INDEX TO MULTIREGION MCT ENTRY
1546 MSGHCON  DS    BL2         COMPANY NUMBER
1547 *         SPECIAL VALUES OF MSGHCON                                   JA
1548 MSGHCFLA EQU   X'BB01'         FLUSH-ALL CHASER MSG                   JA
1549 MSGHCP12 EQU   X'BB03'    3270 COPY FORM 1 (REM.-SAME CU),2 (3275-WR) JA
1550 *  MSGHCP12:  COPY TYPE 1 OR 2, ISSUING TERM REQUEST RESPONSE     SM1124
1551 MSGHCN12 EQU   X'BB13'    COPY TYPE 1 OR 2, NO RESPONSE TO ISSUER SM1124
1552 MSGHCP3  EQU   X'BB02'    3270 COPY FORM3 (READ FULL BUF REQUEST)     JA
1553 MSGHR129 EQU   X'BB04'    IBM129 CARD READER RESET I/P INHIBITED MSG  JA
1554 MSGHFEVR EQU   X'BB'      SET IN MSGHCON+1 OF RESPONSES TO F.E.VERBS  JA
1555           ORG   MSGHCON+1
1556 MSGHRETN DS    BL1         RETURN CODE.
1557 MSGHCONV EQU   C'C'              30 LOGGED FROM CONVERSE.
1558 MSGHFLGS DS    OFL2        MESSAGE INDICATOR FLAGS                   SM1166
1559 MSGHFLG1 DS    FL1         MESSAGE INDICATOR FLAG-BYTE-1             SM1166
1560 MSGHFSDR EQU   X'80'       ASK FOR DEFINITE RESPONSE                VTAM
1561 MSGHFSER EQU   X'40'       ASK FOR EXCEPTION RESPONSE               VTAM
1562 *              IF MSGHFSDR+MSGHFSER=0 THEN NO RESPONSE               VTAM
1563 *              SPECIFICATION, USE OTHER SOURCES TO DETERMINE.         VTAM
1564 MSGHFRSP EQU   MSGHFSDR+MSGHFSER   MASK TO CHECK 'SRESP'             VTAM
1565 MSGHFSR1 EQU   X'20'       1 -> RESPONSE TYPE 1 (FME)               VTAM
1566 MSGHFSR2 EQU   X'10'       1 -> RESPONSE TYPE 2 (RRN)               VTAM
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 11 of 15)

```
1567 MSGHFSEB EQU   X'08'     SEND EB WITH THIS MESSAGE                VTAM
1568 MSGHNCON EQU   X'04'     DO NOT CANCEL CONVERSATION TIMEOUT       XM0215
1569 MSGHFNF3 EQU   X'02'     1 -> DONT WRITE X'F3' LOG RECORD FOR MSG
1570 MSGHFRLS EQU   X'01'     RELEASE NEXT OUTPUT MESSAGE              SM1166
1571 *                                                                 SM1166
1572 MSGHFLG2 DS    FL1       MESSAGE INDICATOR FLAG-BYTE-2            SM1166
1573 MSGHFTRM EQU   X'80'     MSGHADDR POINTS TO SOURCE BTERM/LUC      SM1166
1574 MSGHSRST EQU   X'40'     SERIALLY RESTARTED MESSAGE INDICATOR   (9.0) CH
1575 MSGHSYSC EQU   X'20'     QUEUE THIS MSG TO A 6.2 SESSION EVEN       51MD
1576 *                        IF NO CONVERSATION CURRENTLY ACTIVE        51MD
1577 MSGHFMHI EQU   X'10'     THIS MESSAGE CONTAINS 6.2 FMHDR            51MD
1578 *                                                                    JS
1579 MSGHBMN  DS    BL3       BTAM SEQUENCE NUMBER                       JS
1580 *                                                                    JS
1581 MSGHPMN  EQU   *
1582 MSGHSSCH DS    XL1       HI/ORDER BYTE OF SENDING SUBSYSTEM
1583 MSGHUSR  DS    XL1       AVAILABLE TO USER
1584          ORG   MSGHUSR                                              JS
1585 MSGHADDR DS    AL3       ADDRESS OF AN AUXILIARY AREA (FE ONLY)     JS
1586          ORG   MSGHTID             FOR FILE RECOVERY               X1078
1587 MSGHBKID DS    CL8       BDAM BLOCK ID (FILE RECOVERY)            X1078
1588 MSGHDD   DS    CL8       FILE DDNAME (FILE RECOVERY)              X1078
1589 MSGHLOG  DC    C'0'      LOG TYPE CODE -SEE MONITOR WRITEUP
1590 RVZONE   EQU   X'80'          FILE REVERSAL ENTRY.
1591 RCZONE   EQU   X'90'          FILE RECREATION ENTRY
1592 MSGHXFIL EQU   RVZONE+15      CHECKPOINT RECORD.
1593 RCSTUP   EQU   RCZONE+15      STARTUP RECORD.
1594 MSGHRQST EQU   X'A0'          LOGPROC REQUEUEING STARTED.
1595 MSGHRQND EQU   X'A1'          LOGPROC REQUEUEING ENDED.
1596 MSGHRBUF DS    0H        BUFFER LENGTH (BDAM FILE RECOVERY)       X1078
1597 MSGHMACR DS    0BL1      FILE HANDLER MACRO #                        JT
1598 MSGHBLK  DS    CL1                BLANK  (BINARY ZERO)
1599 MSGHVMI  DS    BL1       VERB/MSG ID
1600 MSGHFFVM EQU   X'67'          SPECIAL VMI FOR FULLY FORMATTED MSGS    JA
1601 DDQVMI   EQU   X'EE'          SPECIAL VMI FOR DYN. DATA QUEING        MM
1602 *
1603 MSGHEND  EQU   *
1604 MSGHLNTH EQU   MSGHEND-MSGHLEN          LENGTH OF MESSAGE HEADER
1605 *
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 12 of 15)

```
1607 WORKAREA DSECT
1608 SAVE     DS    18F
1609 CALLIST  DS    6F
1610 DWCRD    DS    D
1611 *
1612 ECB      DS    F
1613 *
1614 ADDRMSG  DS    A
1615 OMSGHTID DS    CL5
1616 RECAREA  DS    OCL100
1617 PARTREC  DS    OCL100
1618 PARTDATA DS    OCL64
1619 RECPIN   DS    CL5
1620 RECDES   DS    CL54
1621 RECLNT   DS    CL5
1622 RECPRC   DS    PL4
1623 RECMFR#  DS    CL15
1624          DS    CL17
1625 STCCKREC DS    OCL80
1626 DELECHR  DS    X
1627 KEYFLD   DS    OCL8
1628 RECWHS   DS    XL3
1629 RECPNO   DS    XL5
1630          DS    XL28
1631 RECSTKDT DS    OCL43
1632 RECWLC   DS    XL23
1633 RECLEV   DS    PL4
1634 RECLDT   DS    XL6
1635 RECCRD   DS    PL4
1636 RECCDT   DS    XL6
1637 STATWD   DS    OF
1638 FHSTAT   DS    OF
1639 FHSTAT1  DS    X
1640 FHSTAT2  DS    X
1641          DS    H
1642 EXTDSCT  DS    12F
1643 RBNWORD  DS    OF
1644          DS    XL3
1645 RBNWRC   DS    X
1646 CURRFILE DS    CL8
1647 MCW      DS    OF
1648 MCWRETCD DS    X
1649 MCWOPT2  DS    X
1650 MCWOPT3  DS    X
1651 MCWOPT4  DS    X
1652          ORG   MCW
1653 MCWCD12  DS    XL2
1654          ORG
1655 MCB      DS    12F
1656 KEYFIELD DS    CL8
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 13 of 15)

```
1658 DATEDIT  DS    OCL6
1659 DEMC     DS    CL2
1660 DEDAY    DS    CL2
1661 DEYEAR   DS    CL2
1662 DATEMOVE DS    OCL8
1663 DMMO     DS    CL2
1664 SLASH2   DS    X
1665 DMDAY    DS    CL2
1666 SLASH1   DS    X
1667 DMYEAR   DS    CL2
1668 INVINMSG DS    OCL50
1669 MSG7     DS    CL50
1670          ORG   INVINMSG
1671 NOPRTMSG DS    OCL28
1672 MSG1     DS    CL12
1673 NOPART   DS    CL5
1674 MSG2     DS    CL11
1675          ORG   INVINMSG
1676 NOWARMSG DS    OCL37
1677 MSG3     DS    CL5
1678 NOWARENO DS    CL5
1679 MSG4     DS    CL24
1680 NOWARWHS DS    CL3
1681          ORG   INVINMSG
1682 CANMSG   DS    OCL43
1683 CANCODE  DS    CL15
1684 CANFLNM  DS    CL8
1685 MSG5     DS    CL20
1686          ORG   INVINMSG
1687 MAPERMSG DS    OCL19
1688 MSG6     DS    CL17
1689 ERRTAG   DS    CL2
1690          ORG   INVINMSG
1691 NOSUBMSG DS    OCL(L'MSG8+L'NOFILE+L'MSG9)
1692 MSG8     DS    CL(L'MSGJ)
1693 NOFILE   DS    CL8
1694 MSG9     DS    CL(L'MSGK)
1695          ORG
1696 IOGRP    DS    CL8                 MAP GROUP NAME
1697 ERMAP    DS    CL8                 ERROR MAP NAME
1698 IOMAP    DS    CL8                 MAP NAME
1699 SUBNAME  DS    CL8                 SUBROUTINE NAME
1700 MAPFLG   DS    X
1701 MAPERR   EQU   X'40'                ERROR MAP BEING SENT
1702 NOMAP    EQU   X'01'                UNABLE TO SEND MAP
1703 FMFLAG   DS    X
1704 BDRDOK   EQU   X'80'                BDAM READ SUCCESSFUL
1705 VSRDOK   EQU   X'08'                VSAM READ SUCCESSFUL
1706 IOERR    EQU   X'02'                I/O ERROR OCCURRED
1707 OUTMAP   DS    0D
1708 WORKLEN  EQU   *-SAVE
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 14 of 15)

```
1710            COPY  STKSTATA              SYMBOLIC I/O MAP DSECT
1711 STKSTAT   DSECT
1712 MAP1      EQU   *               START OF MAP
1713 VERBL     DS    XL2                     FIELD LENGTH
1714 VERBT     DS    X                       FIELD TAG
1715 VERB      DS    CL4
1716 PARTNOF   DS    CXL3                STRUCTURED SEGMENT START
1717 PARTNOL   DS    XL2                     STRUCTURED SEGMENT LENGTH
1718 PARTNOT   DS    X                       STRUCTURED SEGMENT TAG
1719 PARTNO    EQU   *
1720 FILLER    DS    ZL4
1721 RBNBYTE   DS    Z
1722 USEG1     EQU   *                   SEGMENT DELIMITER
1723 WHSNOL    DS    XL2                     FIELD LENGTH
1724 WHSNOT    DS    X                       FIELD TAG
1725 WHSNO     DS    ZL3
1726 PRTDATAL  DS    XL2                     FIELD LENGTH
1727 PRTDATAT  DS    X                       FIELD TAG
1728 PRTDATA   DS    CL54
1729 ORDUNTL   DS    XL2                     FIELD LENGTH
1730 ORDUNTT   DS    X                       FIELD TAG
1731 ORDUNT    DS    CL5
1732 PRTPRCL   DS    XL2                     FIELD LENGTH
1733 PRTPRCT   DS    X                       FIELD TAG
1734 PRTPRC    DS    PL4
1735 WHSLOCL   DS    XL2                     FIELD LENGTH
1736 WHSLOCT   DS    X                       FIELD TAG
1737 WHSLOC    DS    CL23
1738 STKLEVL   DS    XL2                     FIELD LENGTH
1739 STKLEVT   DS    X                       FIELD TAG
1740 STKLEV    DS    PL4
1741 LEVDATEL  DS    XL2                     FIELD LENGTH
1742 LEVDATET  DS    X                       FIELD TAG
1743 LEVDATE   DS    CL8
1744 STKORDL   DS    XL2                     FIELD LENGTH
1745 STKORDT   DS    X                       FIELD TAG
1746 STKORD    DS    PL4
1747 ORDDATEL  DS    XL2                     FIELD LENGTH
1748 ORDDATET  DS    X                       FIELD TAG
1749 ORDDATE   DS    CL8
1750 MAP1L     EQU   *-MAP1            SINGLE MAP LENGTH
1751           ORG
1752 ERRMAP    EQU   *               START OF MAP
1753 ERRMSGL   DS    XL2                     FIELD LENGTH
1754 ERRMSGT   DS    X                       FIELD TAG
1755 ERRMSG    DS    CL50
1756 ERRMAPL   EQU   *-ERRMAP            SINGLE MAP LENGTH
1757           ORG
1758 STKSTATL  EQU   *-STKSTAT      MAP GROUP LENGTH
1759 DYNLEN    EQU   WORKLEN+STKSTATL
1760           END
```

Figure 48. Sample Reentrant Subsystem (Assembler) (Page 15 of 15)

```
  2 ** REGISTER USAGE  **
  3 ** R3  WORK REGISTER **
  4 ** R4  WORK REGISTER **
  5 ** R6  BAL REGISTER  **
  6 ** R11 PARM POINTER  **
  7 ** R12 BASE REGISTER **
  8 ** R13 SAVE AREA     **
  9 *
 10           PRINT NOGEN
 11 SQASMB    CSECT
 12           REGA  REGISTER EQUATES
 35           SUBLINK LEN=WORKLEN,BASE=(RC),PARM=(RB)
 36+*,SUBLINK - V9.0 - 08/82
 52+          PRINT NOGEN
 56+*,GETSPA - V7.0 - 11/76 - SM
174           USING WORKAREA,RD
175 **        SELECT BDAM FILE
176           MVC   DDNAME,DDPART       MOVE DD NAME INTO WORK AREA
177           CALL  SELECT,(EXTDSCT,FHSTAT,DDNAME),VL,MF=(E,CALLIST)
189           CLI   FHSTAT1,C'9'        SELECT OK?
190           BNE   SELECTOK            YES
191           MVI   RETCD+1,12          NO..RETURN CODE = 12
192           B     RETURN
193 SELECTOK DS     0H
194           XC    FHSTAT,FHSTAT       CLEAR FH CONTROL WORD
195           BAL   R6,READ             GO READ A RECORD
196           CLI   FHSTAT1,C'1'        IOERROR?
197           BNE   NOT1                NO
198           MVI   RETCD+1,4           YES..RETURN CODE = 4
199           B     DORLSE              GO RELEASE THE FILE
200 NOT1      DS    0H
201           CLI   FHSTAT1,C'2'        RECORD NOT FOUND?
202           BNE   DORLSE              FOUND, RETURN CODE = 0
203           MVI   RETCD+1,8           NOT FOUND, RETURN CODE = 8
204 DORLSE    DS    0H
205           XC    FHSTAT,FHSTAT       CLEAR FH CONTROL WORD
206           CALL  RELEASE,(EXTDSCT,FHSTAT),VL,MF=(E,CALLIST)
217 RETURN    DS    0H
218           LH    RF,RETCD            LOAD RETURN CODE
219           RTNLINK ADDR=(RD),LEN=WORKLEN,RC=(RF)
229+          PRINT NOGEN
248+*,GETSPA - V7.0 - 11/76 - SM
```

Figure 49. Sample Assembler Subroutine (Page 1 of 2)

```
270 READ      DS    0H
271 * REGISTER 11 CONTAINS PARM LIST FROM SQASMA
272           L     R3,0(RB)              FIRST PARM = ADDRESS OF RECORD AREA
273           L     R4,4(RB)              SECOND PARM = ADDRESS OF RBN
274           CALL  READ,(EXTDSCT,FHSTAT,0(R3),1(R4)),VL,MF=(E,CALLIST)
288           BR    R6                    RETURN


290 DDPART    DC    CL8'PARTFILE'


292           LTORG
293                 =V(PMISUBL2)
294                 =V(PMIRTLR)


296 WORKAREA  DSECT
297 SA        DS    18F
298 FHSTAT    DS    0F
299 FHSTAT1   DS    X
300           DS    3X
301 EXTDSCT   DS    12F
302 DDNAME    DS    CL8
303 CALLIST   DS    4F
304 RETCD     DS    H
305 WORKLEN   EQU   *-SA
306           END
```

Figure 49. Sample Assembler Subroutine (Page 2 of 2)

PAGES 127-128 INTENTIONALLY MISSING

Chapter 12

SUBSYSTEM TESTING

12.1 INTRODUCTION

After a new subsystem has been thoroughly desk-checked and assembles cleanly, it becomes necessary to test the subsystem's execution under the control of Intercomm. Three methods of testing are available:

- Simulated--batch execution of Intercomm with a simulated BTAM Front End. Message input streams are created via the CREATSIM utility program. Additionally, 3270 terminal input and output screen, or output printer, images are formatted if the SIM3270 utility is implemented for the simulation mode execution. Illustration of this mode of testing is provided in this Chapter, and is particularly useful for testing messages processed via the Message Mapping Utilities.

- Test Mode--batch execution of a Back End Intercomm with message input from a card-image data set, as described in Chapter 13.

- On-line Testing--an on-line system is necessary for final testing of all error conditions, multithread processing, etc. and can be either a single region system, or a satellite region used primarily for testing within a Multiregion production system.

12.2 DEBUGGING APPLICATION PROGRAM PROBLEMS

Text and descriptions of error messages issued by Intercomm as a result of invalid program logic paths, along with descriptions of general debugging techniques for accompanying snaps and abends are available in Message and Codes. Additional debugging facilities such as dispatcher trace reports, thread dumps and indicative dumps are described in the Operating Reference Manual.

12.3 TESTING A SUBSYSTEM WITH THE FRONT END SIMULATOR

As described in the Operating Reference Manual, a test execution with a simulated Front End is very useful to determine Front End message interface problems that may be harder to debug when using an on-line test system. Although the simulation is of certain BTAM devices, including a local 3270, the access method interfaces required for a remote 3270 or a TCAM or VTAM Front End are essentially transparent to the application programmer as the interface dependent code is handled by Intercomm.

This chapter illustrates testing of the subsystem and subroutine described in Chapter 11 using the BTAM simulator for 3270 CRT messages processed via maps defined for the Message Mapping Utilities.

To test an application system in a simulated Intercomm environment, do the following:

NOTE: Steps preceded by an asterisk (*) may often be performed for the application programmer by an installation's Intercomm System Manager. Appendix C summarizes the Intercomm Table entries.

1. Compile and linkedit the user subsystem(s) and subroutine(s), if any. Appendix A describes Intercomm-supplied Assembler JCL procedures.

*2. Create or add to a USRSCTS member on a user test library to contain a Subsystem Control Table Entry (SYCTTBL macro) which describes the subsystem. Reassemble and link INTSCT which copies the USRSCTS member from the test library (see Figure 50).

*3. Define input message verbs in the copy member USRBTVRB via BTVERB macros and reassemble and link the Front End Verb Table BTVRBTB (see Figure 50).

*4. Code a SUBMODS macro addition to the COPY member USRSUBS to define the Assembler subroutine and reassemble and linkedit REENTSBS which copies USRSUBS (see Figure 50).

5. Assemble and linkedit MMU maps (Map Group STKSTAT--see Figure 51) to the MMU load module library. Load maps to the appropriate Store/Fetch data set. See Message Mapping Utilities.

6. Prepare input test message data set(s) using the CREATSIM utility as illustrated in Figure 52. Note that the first message generates, via the MMU command MMUC, the screen template to be used for entering an inquiry transaction. All subsequent input messages are for testing the Assembler subsystem and subroutine, including input error conditions handled by the application program.

*7. Add control cards to the linkedit deck for the user programs, unless the routines are dynamically loadable (see Figure 53).

*8. Add INCLUDE statements for the simulator (BTAMSIM) and 3270 display formatter (SIM3270) to an Intercomm linkedit deck which was created for the BTAM Front End (see Figure 53).

*9. Linkedit to create a new Intercomm load module (see Figure 53).

10. Add DD statements to the Intercomm execution JCL for the printed SIM3270 output and the input message data set(s) (see Figure 53).

11. Create test data sets and add DD statements for them to the execution JCL (see Figure 53). Note that if a VSAM data set is used with a user catalogue, place the STEPCAT DD statement after the //PMISTOP DD statement (see Figure 53); do not use a JOBCAT DD statement. Omit the STEPCAT statement if an ICF catalogue is used.

*12. Execute in simulation mode:

a. Single-thread test all subsystems; to test a reentrant subsystem, specify MNCL=1 in the subsystem's SYCTTBL macro.

b. Multithread test reentrant subsystems (change MNCL) using several test message input data sets or use a single data set as input from more than one terminal.

The parameter 'STARTUP' must be coded on the Intercomm EXEC statement. Figure 53 illustrates a sample execution deck with test message input (DD statement TEST1) for the sample inquiry program and JCL to print the system log.

The resulting SIM3270 printouts for the simulated execution of the sample inquiry subsystem are illustrated in Figure 54. Note that the underlined positions on each screen display indicate attribute byte positions; codes are described under the display. On an actual terminal, the attribute byte position appears as a blank to the terminal operator. See Message Mapping Utilities and IBM documentation on programming for the 3270 CRT for further information on attribute codes.

The Intercomm Log printed after the simulated execution of the sample inquiry subsystem is shown in Figure 55.

13. Test the subsystem concurrently with other application subsystems.

```
//TABLES      JOB
//*
//*                 DEFINE SYCTTBL FOR SUBSYTEM
//*
//STEP1       EXEC  LIBELINK,Q=TEST,NAME=INTSCT,LMOD=INTSCT
//LIB.SYSIN   DD    *
./ ADD NAME=USRSCTS
./ NUMBER     NEW1=100,INCR=100
USRSCTS       DS    0H
RA            SYCTTBL SUBH=R,SUBC=A,SBSP=SQASMA,LANG=RBAL,OVLY=0,       X
                    NUMCL=10,MNCL=2,TCTV=60
/*
//ASM.SYSIN   DD    DSN=INT.SYMREL(INTSCT),DISP=SHR
//*
//*                 DEFINE BTVERB FOR SUBSYSTEM
//*
//STEP2       EXEC  LIBELINK,Q=TEST,NAME=BTVRBTB,LMOD=BTVRBTB
//LIB.SYSIN   DD    *
./ ADD  NAME=USRBTVRB
./ NUMBER     NEW1=100,INCR=100
USRBTVRB      DS    0H
              BTVERB VERB=MURA,SSCH=R,SSC=A,CONV=18000
/*
//ASM.SYSIN   DD    DSN=INT.SYMREL(BTVRBTB),DISP=SHR
//*
//*                 DEFINE SUBMODS FOR SUBROUTINE
//*
//STEP3       EXEC  LIBELINK,Q=TEST,NAME=REENTSBS,LMOD=REENTSBS
//LIB.SYSIN   DD    *
./ ADD  NAME=USRSUBS
./ NUMBER     NEW1=100,INCR=100
USRSUBS       DS    0H
              SUBMODS LNAME=SQASMB,TYPE=BAL,DELTIME=30
/*
//ASM.SYSIN   DD    DSN=INT.SYMREL(REENTSBS),DISP=SHR
//
```

Figure 50. Table Updates to Implement Simulation Mode Testing

```
STKSTAT  MAPGROUP MODE=I/O,DEVICE=IBM3270                                    00000010
MAP1     MAP    SIZE=(20,80),START=(1,1)                                     00000020
VERB     FIELD RELPOS=VERB                                                   00000030
         FIELD RELPOS=(1,7),INITIAL='ENTER TRANSACTION CODE',ATTRIB=PSN      00000040
         FIELD RELPOS=(3,23),INITIAL='ENTER DATA:',ATTRIB=PSN                00000050
         FIELD RELPOS=(5,7),INITIAL='PART NO:',ATTRIB=PAHSEL                 00000060
PARTNO   SEGMENT                                                             00000065
FILLER   FIELD RELPOS=(5,16),FORMAT=(4,,ZD),ATTRIB=UNN                       00000070
RBNBYTE  FIELD RELPOS=(5,20),FORMAT=(1,,ZD)                                  00000075
         SEGMENT                                                             00000077
         FIELD RELPOS=(5,22),FORMAT=1,ATTRIB=PSN                             00000080
         FIELD RELPOS=(6,7),INITIAL='WHS NO:',ATTRIB=PAHSEL                  00000090
WHSNO    FIELD RELPOS=(6,15),FORMAT=(3,,ZD),ATTRIB=UNN                       00000100
         FIELD RELPOS=(6,19),FORMAT=1,ATTRIB=PSN                             00000110
         FIELD RELPOS=(8,23),INITIAL='STOCK STATUS:',ATTRIB=PSN              00000120
         FIELD RELPOS=(10,7),INITIAL='DESCRIPTION:',ATTRIB=PSN               00000130
PRTDATA  FIELD RELPOS=(10,20),FORMAT=54,ATTRIB=UAN                           00000140
         FIELD RELPOS=(10,76),FORMAT=1,ATTRIB=PSN                            00000150
         FIELD RELPOS=(11,7),INITIAL='ORDER UNITS:',ATTRIB=PSN               00000160
ORDUNT   FIELD RELPOS=(11,20),FORMAT=5,ATTRIB=UAN                            00000170
         FIELD RELPOS=(11,26),FORMAT=1,ATTRIB=PSN                            00000180
         FIELD RELPOS=(11,40),INITIAL='PRICE:',ATTRIB=PSN                    00000190
PRTPRC   FIELD RELPOS=(11,47),FORMAT=(9,4,$PDS4),ATTRIB=UAN                  00000200
         FIELD RELPOS=(11,57),FORMAT=1,ATTRIB=PSN                            00000210
         FIELD RELPOS=(13,23),INITIAL='STOCK STATUS AT WAREHOUSE:',     X00000220
               ATTRIB=PSN                                                    00000230
         FIELD RELPOS=(15,7),INITIAL='LOCATION:',ATTRIB=PSN                  00000240
WHSLOC   FIELD RELPOS=(15,17),FORMAT=23,ATTRIB=UAN                           00000250
         FIELD RELPOS=(15,41),FORMAT=1,ATTRIB=PSN                            00000260
         FIELD RELPOS=(16,7),INITIAL='ON HAND:',ATTRIB=PSN                   00000270
STKLEV   FIELD RELPOS=(16,16),FORMAT=(7,4,PD),ATTRIB=UAN                     00000280
         FIELD RELPOS=(16,24),FORMAT=1,ATTRIB=PSN                            00000290
         FIELD RELPOS=(16,40),INITIAL='AS OF:',ATTRIB=PSN                    00000300
LEVDATE  FIELD RELPOS=(16,47),FORMAT=8,ATTRIB=UAN                            00000310
         FIELD RELPOS=(16,56),FORMAT=1,ATTRIB=PSN                            00000320
         FIELD RELPOS=(17,7),INITIAL='ON ORDER:',ATTRIB=PSN                  00000330
STKORD   FIELD RELPOS=(17,17),FORMAT=(7,4,PD),ATTRIB=UAN                     00000340
         FIELD RELPOS=(17,25),FORMAT=1,ATTRIB=PSN                            00000350
         FIELD RELPOS=(17,40),INITIAL='AS OF:',ATTRIB=PSN                    00000360
ORDDATE  FIELD RELPOS=(17,47),FORMAT=8,ATTRIB=UAN                            00000370
         FIELD RELPOS=(17,56),FORMAT=1,ATTRIB=PSN                            00000380
ERRMAP   MAP    SIZE=(15,80),START=(10,1)                                    00000390
         FIELD RELPOS=(1,1),ATTRIB=SUPR,INITIAL=X'125B5F'                    00000400
***  ABOVE CLEARS STOCK STATUS INFO. WHEN ERROR MESSAGE APPEARS  ***         00000410
         FIELD RELPOS=(14,33),INITIAL='ERROR MESSAGE:',ATTRIB=PAHSEL         00000420
ERRMSG   FIELD RELPOS=(15,10),FORMAT=50,ATTRIB=UAHSEL                        00000430
         FIELD RELPOS=(15,61),FORMAT=1,ATTRIB=PSN                            00000440
         ENDGROUP                                                            00000450
         END                                                                 00000460
```

Figure 51. MMU Maps Used by Sample Subsystem

```
//CREATSIM JOB                                                          00000100
//CRS     PROC  T=                                                      00000200
//*    SCRATCH OLD TEST INPUT DATA SET (IF ANY)                         00000300
//S       EXEC  PGM=IEFBR14                                             00000400
//SCR       DD  DSN=INT.T&T,DISP=(OLD,DELETE)                           00000500
//CRS     EXEC  PGM=CREATSIM                                            00000600
//*    CREATE NEW TEST INPUT DATA STREAM FOR 3270 DEVICE                00000700
//STEPLIB   DD  DSN=INT.MODLIB,DISP=SHR                                 00000800
//           DD  DSN=INT.MODREL,DISP=SHR                                00000900
//SYSPRINT DD SYSOUT=A                                                  00001000
//SYSUT2    DD  DSN=INT.T&T,DISP=(,CATLG,CATLG),UNIT=ONLINE,            00001100
//              VOL=SER=INT001,SPACE=(TRK,(1,1))                        00001200
//DUMP    EXEC  PGM=IEBPTPCH                                            00001300
//*    PRINT MESSAGES GENERATED ON TEST INPUT DAT SET                   00001400
//SYSPRINT DD  SYSOUT=A                                                 00001500
//SYSUT1    DD  DSN=*.CRS.SYSUT2,DISP=OLD                               00001600
//SYSUT2    DD  SYSOUT=A                                                00001700
//         PEND                                                         00001800
//*  FOR THIS EXECUTION OF CREATSIM, THE END-OF-CARD CHARACTER IS A     00001900
//*    SEMI-COLON, (USE ALSO AFTER THE VERB-FRONT END SEES THE SBA),    00002000
//*    THE MESSAGE END CHARACTER IS AN EXCLAMATION POINT (EOB).         00002100
//EXECCRS EXEC CRS,T=TEST1                                              00002200
//CRS.SYSIN  DD  *                                                      00002300
GRAPHIC,ADD,;FF                 CONTINUATION CODE                     00002400
GRAPHIC,ADD,<7D                 ENTER KEY                             00002500
SBA,M2                          USING MODEL 2 SCREEN SIZE             00002600
<  MMUC,SHOW,(STKSTAT,MAP1)                                           00002700
<  ;                                                                  00002800
SBA,C102;                                                             00002900
MURA;                                                                 00003000
SBA,C516;                                                             00003100
12345;                                                                00003200
SBA,C615;                                                             00003300
200                                                                   00003400
<  ;                                                                  00003500
SBA,0102;                                                             00003600
MURA;                                                                 00003700
SBA,C516;                                                             00003800
55555;                                                                00003900
SBA,C615;                                                             00004000
200                                                                   00004100
<  ;                                                                  00004200
SBA,C102;                                                             00004300
MURA;                                                                 00004400
SBA,C516;                                                             00004500
12348;                                                                00004600
SBA,C615;                                                             00004700
3C0                                                                   00004800
<  ;                                                                  00004900
SBA,C102;                                                             00005000
MURA;                                                                 00005100
SBA,0516;                                                             00005200
12341;                                                                00005300
SBA,C615;                                                             00005400
600                                                                   00005500
<  ;                                                                  00005600
SBA,C102;                                                             00005700
```

Figure 52. Input Test Messages Generated via CREATSIM (Page 1 of 2)

```
MURA;                                                          00005800
SBA,C516;                                                      00005900
A2345;                                                         00006000
SBA,C615;                                                      00006100
200                                                            00006200
<  ;                                                           00006300
SBA,C102;                                                      00006400
MURA;                                                          00006500
SBA,C516;                                                      00006600
12345;                                                         00006700
SBA,C615;                                                      00006800
BCO                                                            00006900
<  ;                                                           00007000
SBA,C102;                                                      00007100
MURA;                                                          00007200
SBA,C516;                                                      00007300
1234X;                                                         00007400
SBA,C615;                                                      00007500
20Y                                                            00007600
<  ;                                                           00007700
SBA,C102;                                                      00007800
MURA;                                                          00007900
SBA,C516;                                                      00008000
12349;                                                         00008100
SBA,C615;                                                      00008200
100                                                            00008300
<  ;                                                           00008400
SBA,C102;                                                      00008500
MURA;                                                          00008600
SBA,C516;                                                      00008700
12342;                                                         00008800
SBA,C615;                                                      00008900
100                                                            00009000
//DUMP.SYSIN DD *                                              00009100
  PRINT TYPORG=PS,TOTCONV=XE,CNTRL=2                           00009200
//                                                             00009300
```

Figure 52. Input Test Messages Generated via CREATSIM (Page 2 of 2)

```
//EXECTEST JOB (ICOMTEST,,,20),'SQASMA TEST',CLASS=A,
//   RESTART=(GENLINK.ASM)
//PROCLIB  DD  DSN=INT.PROCLIB,DISP=SHR    (AS NEEDED)
//*********************************************************************
//*  THE RESTART PARM IN THE JOB STATEMENT RESTARTS THE TEST AT THE   *
//*  BEGINNING.  IF YOU WISH TO RESTART AT A DIFFERENT STEP, CODE     *
//*  RESTART=STEPNAME  OR  RESTART=STEPNAME.PROCSTEPNAME              *
//*                                                                   *
//*  NOTE: WHEN USING A VSAM FILE, IT IS NECESSARY TO EXECUTE IDCAMS  *
//*        TO VERIFY THE FILE IF A PREVIOUS EXECUTION ABENDED.        *
//*********************************************************************
//*
//*********************************************************************
//*  STEP GENLINK GENERATES A STANDARD BTAM FRONT END LINKEDIT DECK    *
//*  VIA ASSEMBLY OF THE ICOMLINK MACRO. IF ONLY A VTAM FRONT END IS  *
//*  USED, A SETGLOBE WITH THE BTAM GLOBAL SET ON MUST BE IN THE      *
//*  LIBRARY SPECIFIED BY THE Q= PARM. ADD OR CHANGE PARMS FOR THE    *
//*  ICOMLINK MACRO BASED ON INTERCOMM FACILITIES USED.               *
//*  THE GENERATED DECK (SIMLINK) IS PLACED ON INT.SYMTEST.           *
//*  NOTE: THE SPECIFIED FRONT END NETWORK TABLE (FENETWRK) CONTAINS A
//*        DEFINITION FOR THE TEST TERMINAL TEST1 AS A LOCAL BTAM 3270.
//*********************************************************************
//GENLINK  EXEC ASMPC,Q=TEST,DECK=DECK
//ASM.SYSIN DD *
         ICOMLINK MMU=YES,FETABLE=FENETWRK
         END
/*
//SYSPUNCH DD DSN=INT.SYMTEST(SIMLINK),DISP=SHR
//*
//*********************************************************************
//*  STEPS SCRSCR AND ALLOCSCR DELETE AND RE-ALLOCATE THE LOAD        *
//*   MODULE LIBRARY USED IN THE TEST (ALSO USED FOR DYNLLIB)         *
//*********************************************************************
//SCRSCR   EXEC PGM=IEFBR14
//FILE1    DD   DSN=INT.MODSCR,DISP=(OLD,DELETE)
//ALLOCSCR EXEC PGM=IEFBR14
//A        DD   DSN=INT.MODSCR,DISP=(,CATLG),UNIT=SYSDA,
//    DCB=INT.MODLIB,VOL=SER=INT001,SPACE=(CYL,(3,,7)
```

Figure 53. Linkedit and Execution JCL for Simulation Mode (Page 1 of 3)

```
//*********************************************************************
//*  STEP GENINCL CREATES INCLUDE DECK USED BY THE LINK EDIT STEP:    *
//*  THE ADDED INCLUDE STATEMENTS ARE FOR THE SAMPLE SUBSYSTEM AND    *
//*      THE REQUIRED SIMULATION MODE MODULES.                        *
//*  IF THE TEST1 TERMINAL IS NOT IN THE SYSTEM PMISTATB TABLE, ADD:  *
//*         INCLUDE MODREL(PMISTATB)                                  *
//*         INCLUDE MODREL(PMIDEVTB)                                  *
//*         INCLUDE MODREL(PMIBRCAD)                                  *
//*      THE ABOVE ASSUMES THE CONTROL TERMINAL IS NAMED CNT01.       *
//*********************************************************************
//GENINCL  EXEC PGM=IEBUPDTE
//SYSPRINT DD   SYSOUT=A
//SYSUT1   DD   DSN=INT.SYMTEST,DISP=SHR
//SYSUT2 DD DSN=&&INCL,DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1,1)),
//     DCB=(BLKSIZE=80,LRECL=80)
//SYSIN    DD   *
./  CHANGE NAME=SIMLINK,LIST=ALL
         INCLUDE SYSLIB(SQASMA)                                      00000010
         INCLUDE SYSLIB(BTAMSIM)                                     00000020
         INCLUDE SYSLIB(SIM3270)                                     00000030
/*
//*********************************************************************
//*  LINK EDIT THE TEST INTERCOMM SYSTEM                              *
//*  NOTE THAT THE INTERCOMM LKEDT PROC PLACES THE OUTPUT ON THE      *
//*  MODSCR LOAD LIBRARY CREATED ABOVE.                               *
//*  IT IS NOT NECESSARY TO RE-DO THE WHOLE LINK TO REPLACE 1 MODULE  *
//*  IN THIS CASE, ALL YOU SHOULD DO IS:                              *
//*   1)  REASSEMBLE OR RECOMPILE THE CHANGED/NEW MODULE INTO A       *
//*       SEPARATE LOAD LIBRARY                                       *
//*   2)  OVERRIDE THE SYSLIN DD STMT TO //LKED.SYSLIN DD *           *
//*       FOLLOW IT WITH INCLUDE CARDS                                *
//*       FOR THE MODULES YOU WISH TO REPLACE                         *
//*   3)  FOLLOW THOSE INCLUDES WITH THE FOLLOWING 3 CARDS:           *
//*            INCLUDE SYSLMOD(SIMICOM)                               *
//*            ENTRY   PMISTUP                                        *
//*            NAME    SIMICOM(R)                                     *
//*   4)  INSERT A DD STMT FOR THE LOAD LIBRARY ON WHICH THE          *
//*       REPLACEMENT MODULES RESIDE                                  *
//*   5)  CHANGE THE RESTART PARM ON THE JOB STATEMENT                *
//*       TO POINT TO THE LKED STEP.                                  *
//*********************************************************************
//LKED     EXEC LKEDT,Q=TEST,LMOD=SIMICOM,
//       PARM.LKED='LIST,LET,XREF,NCAL,SIZE=(250K,100K)'
//LKED.SYSLIN DD DSN=&&INCL(SIMLINK),DISP=(OLD,PASS)
//MODREL   DD   DSN=INT.MODREL,DISP=SHR
//*********************************************************************
//*  LINKEDIT THE DYNAMICALLY LOADED SUBROUTINE                       *
//*********************************************************************
//LINKSQB  EXEC LKEDT,Q=TEST,LMOD=SQASMB
//SYSLIN   DD   *
         INCLUDE SYSLIB(SQASMB)
/*
```

Figure 53. Linkedit and Execution JCL for Simulation Mode (Page 2 of 3)

```
//*********************************************************************
//*  EXECUTE INTERCOMM IN SIMULATION MODE                             *
//*********************************************************************
//GO       EXEC PGM=SIMICOM,PARM='STARTUP',TIME=(,30)
//STEPLIB  DD   DSN=INT.MODSCR,DISP=SHR
//              DSN=INT.MODUSR,DISP=SHR
//              DSN=INT.MODLIB,DISP=SHR
//              DSN=INT.MODREL,DISP=SHR
//INTERLOG DD   DSN=&&INTLOG,DISP=(NEW,PASS),
//  DCB=(DSORG=PS,RECFM=VB,BLKSIZE=4096,LRECL=4092,NCP=8,OPTCD=C),
//  SPACE=(TRK,(10,5)),VOL=SER=INT001,UNIT=SYSDA
//SMLOG    DD   SYSOUT=A,DCB=(DSORG=PS,BLKSIZE=120,RECFM=FA)
//STSLOG   DD   SYSOUT=A,DCB=(DSORG=PS,BLKSIZE=120,RECFM=FA)
//SYSPRINT DD   SYSOUT=A,DCB=(DSORG=PS,BLKSIZE=141,LRECL=137,RECFM=VA)
//RCT000   DD   DSN=INT.RCT000,DCB=(DSORG=DA,OPTCD=RF),DISP=SHR
//PMIQUE   DD   DSN=INT.PMIQUE,DCB=(DSORG=DA,OPTCD=R),DISP=OLD
//BTAMQ    DD   DSN=INT.BTAMQ,DCB=(DSORG=DA,OPTCD=R),DISP=SHR
//INTSTOR0 DD   DSN=INTSTOR0,DCB=(DSORG=DA,OPTCD=EF,LIMCT=3),DISP=OLD
//INTSTOR2 DD   DSN=INTSTOR2,DCB=(DSORG=DA,OPTCD=EF,LIMCT=3),DISP=SHR
//INTSTOR3 DD   DSN=INTSTOR3,DCB=(DSORG=DA,OPTCD=EF,LIMCT=3),DISP=SHR
//*         TEST DATA SETS FOR SAMPLE SUBSYSTEM
//STCKFILE DD   DSN=VSAMSD1.STCKFILE.CLUSTER,DISP=OLD,
//     AMP=(AMORG,'RECFM=F')
//PARTFILE DD   DSN=INT.TEST.PARTFILE,DISP=OLD,
//     DCB=(DSORG=DA,OPTCD=R)
//*         DATA SETS FOR SIMULATED TERMINAL -- TEST1
//TEST1    DD   DSN=INT.TEST1,DCB=DSORG=PS,DISP=OLD
//SCRTEST1 DD   SYSOUT=A,DCB=(DSORG=PS,RECFM=FA,BLKSIZE=121)
//SIMCARDS DD   *
TEST1,001
//PMISTOP  DD   DUMMY
//*         FAR PARAMETERS
//ICCMIN   DD   *
INTSTOR0,ICCMBDAMXCTRL
/*
//SNAPDD   DD   SYSOUT=A
//SYSUDUMP DD   SYSOUT=A
//STEPCAT  DD   DSN=VSAMSD1,DISP=SHR
//*         DYNAMIC LINKEDIT DATA SETS
//DYNLLIB  DD   DSN=INT.MODSCR,DISP=SHR
//DYNLPRNT DD   SYSOUT=A
//DYNLWORK DD   UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(,PASS)
//*********************************************************************
//*  PRINT THE INTERCOMM LOG GENERATED BY THE TEST                    *
//*********************************************************************
//INTERLOG EXEC PGM=LOGPRINT,COND=EVEN
//STEPLIB  DD   DSN=INT.MODREL,DISP=SHR
//SYSPRINT DD   SYSOUT=A,DCB=(DSORG=PS,BLKSIZE=121)
//INTERLOG DD   DSN=&&INTLOG,DISP=OLD,DCB=BLKSIZE=5000
//SYSIN    DD   DUMMY
//
```

Figure 53. Linkedit and Execution JCL for Simulation Mode (Page 3 of 3)

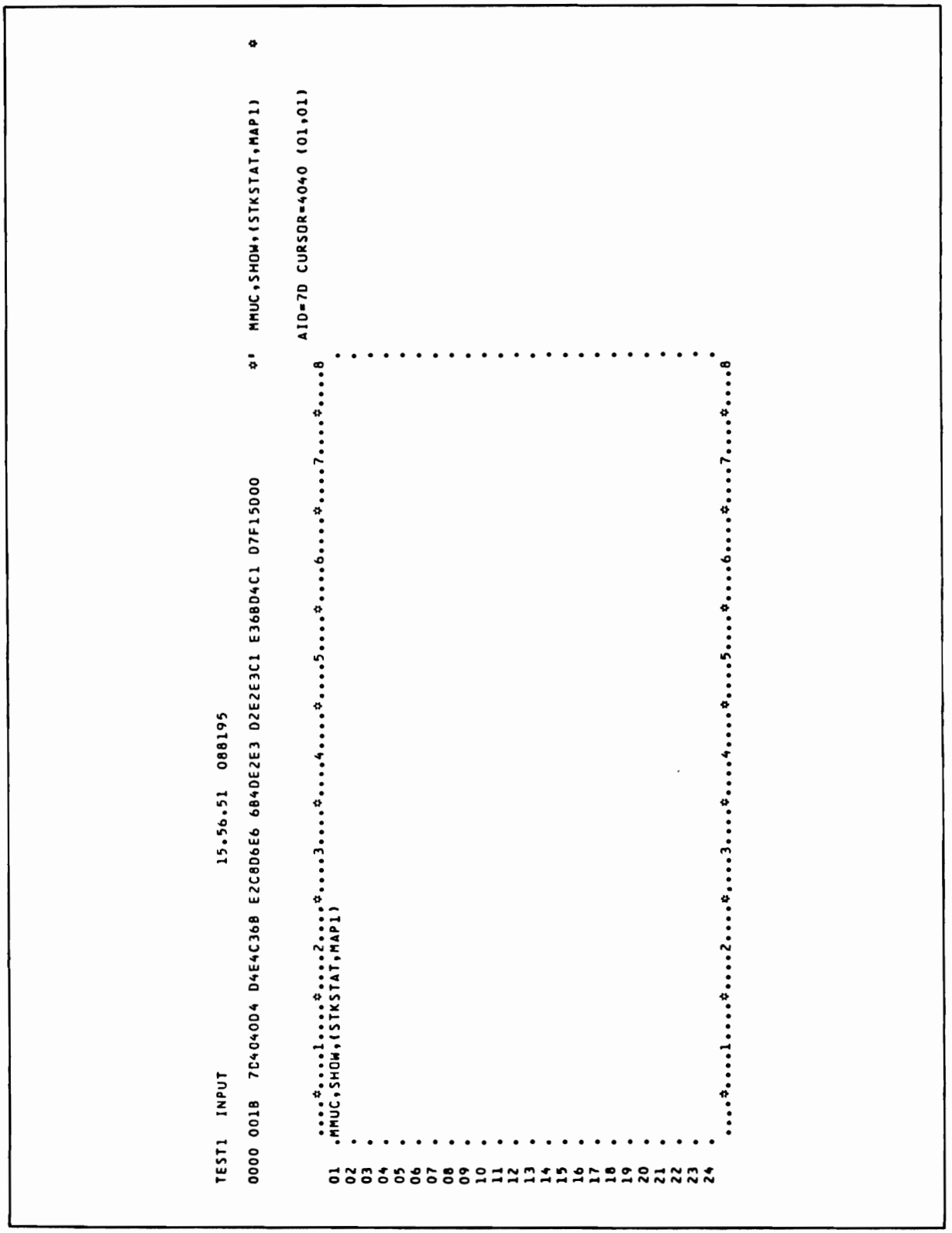
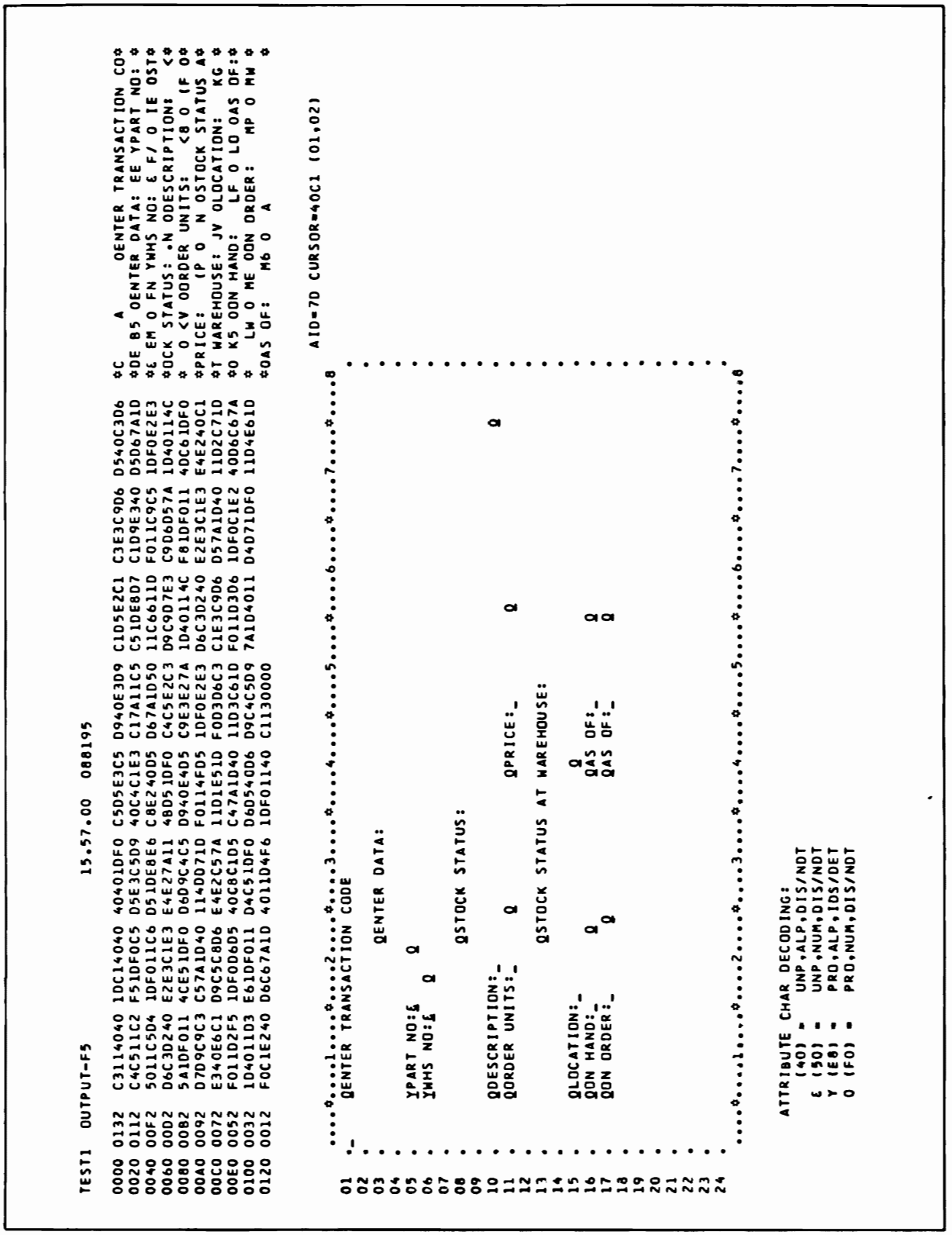
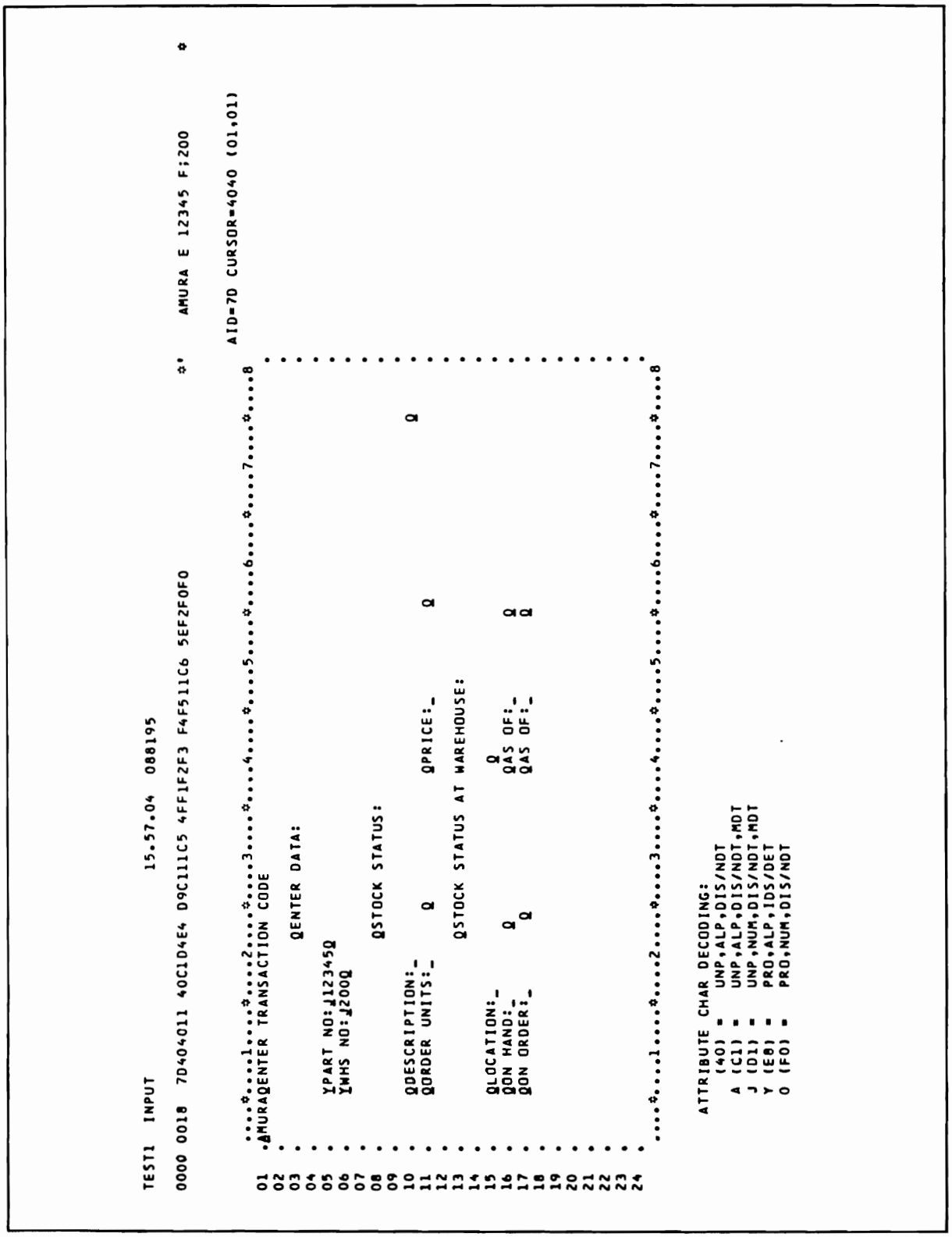


Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 1 of 20)



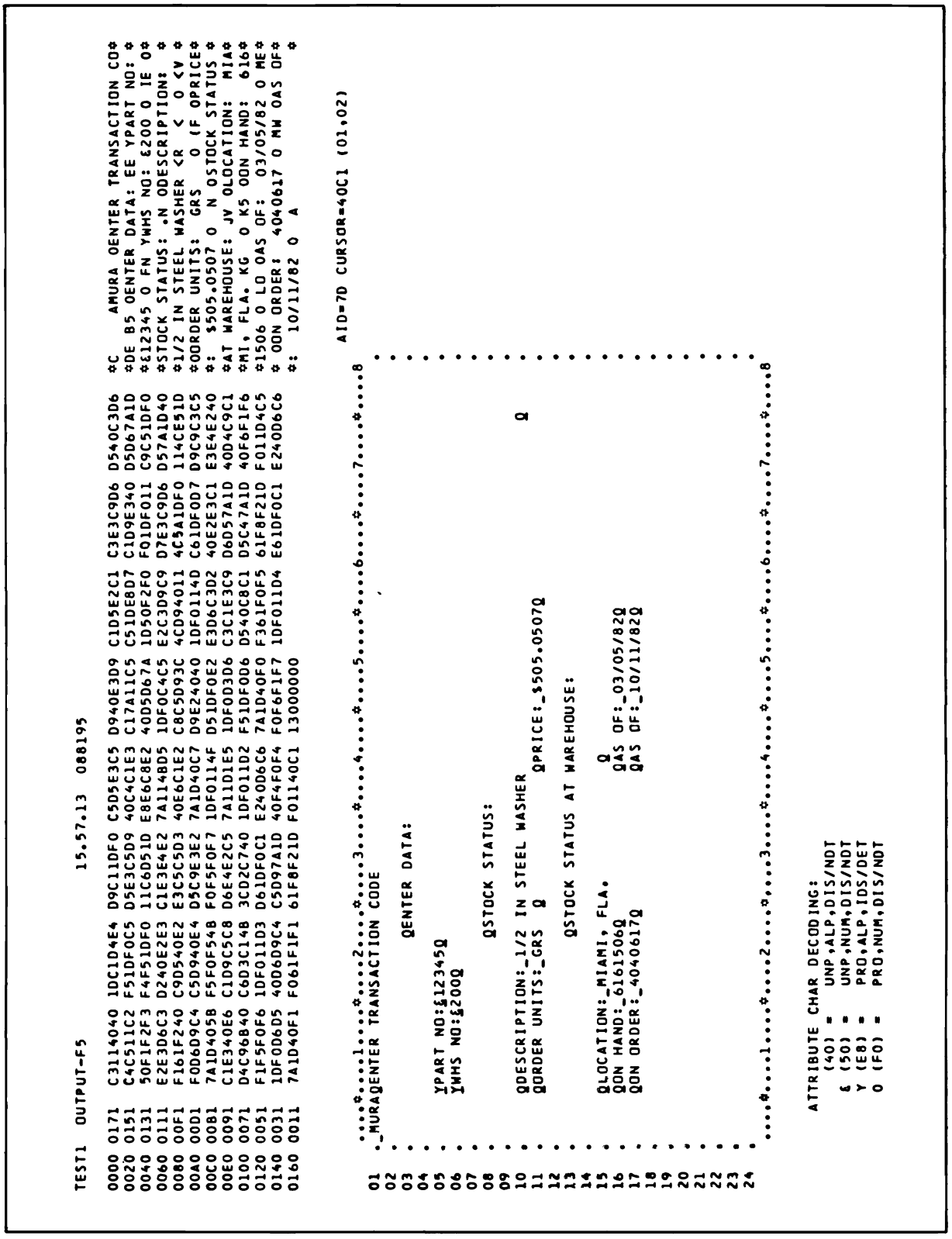
```
TEST1  OUTPUT-F5                    15.57.00  088195

0000 0132  C3114040 1DC14040 40401DF0 C5D5E3C5 D940E3D9 C1D5E2C1 C3E3C9D6 D540C3D6  *C     A      OENTER TRANSACTION CO*
0020 0112  C4C511C2 F51DF0C5 D5E3C5D9 40C4C1E3 C17A11C5 C51DE8D7 C1D9E340 D5D67A1D  *DE B5 OENTER DATA: EE YPART NO: *
0040 00F2  5011C5D4 1DF011C6 D51DE8E6 C8E240D5 D67A1D50 11C6611D F011C9C5 1DF0E2E3  *& EM O FN YWHS NO: & F/ O IE OST*
0060 00D2  D6C3D240 E2E3C1E3 E4E27A11 4BD51DF0 C4C5E2C3 D9C9D7E3 C9D6D57A 1D40114C  *OCK STATUS: .N ODESCRIPTION:   <*
0080 00B2  5A1DF011 4CE51DF0 D6D9C4C5 D940E4D5 C9E3E27A 1D40114C F81DF011 4DC61DF0  *  O <V OORDER UNITS:   <8 O (F O*
00A0 0092  D7D9C9C3 C57A1D40 114DD71D F0114FD5 1DF0E2E3 D6C3D240 E2E3C1E3 E4E240C1  *PRICE:   (P O  N OSTOCK STATUS A*
00C0 0072  E340E6C1 D9C5C8D6 E4E2C57A 11D1E51D F0D3D6C3 C1E3C9D6 D57A1D40 11D2C71D  *T WAREHOUSE: JV OLOCATION:   KG *
00E0 0052  F011D2F5 1DF0D6D5 40C8C1D5 C47A1D40 11D3C61D F011D3D6 1DF0C1E2 40D6C67A  *O K5 OON HAND:   LF O LO OAS OF:*
0100 0032  1D4011D3 E61DF011 D4C51DF0 D6D540D6 D9C4C5D9 7A1D4011 D4D71DF0 11D4E61D  *   LW O ME OON ORDER:   MP O MW *
0120 0012  F0C1E240 D6C67A1D 4011D4F6 1DF01140 C1130000                             *OAS OF:   M6 O  A                *

                                                                                    AID=7D CURSOR=40C1 (01,02)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  ._    QENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .    YPART NO:&     Q                                                            .
06  .    YWHS NO:&   Q                                                               .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .    QDESCRIPTION:_                                                    Q         .
11  .    QORDER UNITS:_     Q              QPRICE:_         Q                       .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .    QLOCATION:_                        Q                                        .
16  .    QON HAND:_       Q                QAS OF:_        Q                        .
17  .    QON ORDER:_       Q               QAS OF:_        Q                        .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                                                                                .
24  .                                                                                .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

       ATTRIBUTE CHAR DECODING:
          (40) =   UNP,ALP,DIS/NDT
        & (50) =   UNP,NUM,DIS/NDT
        Y (E8) =   PRO,ALP,IDS/DET
        O (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 2 of 20)



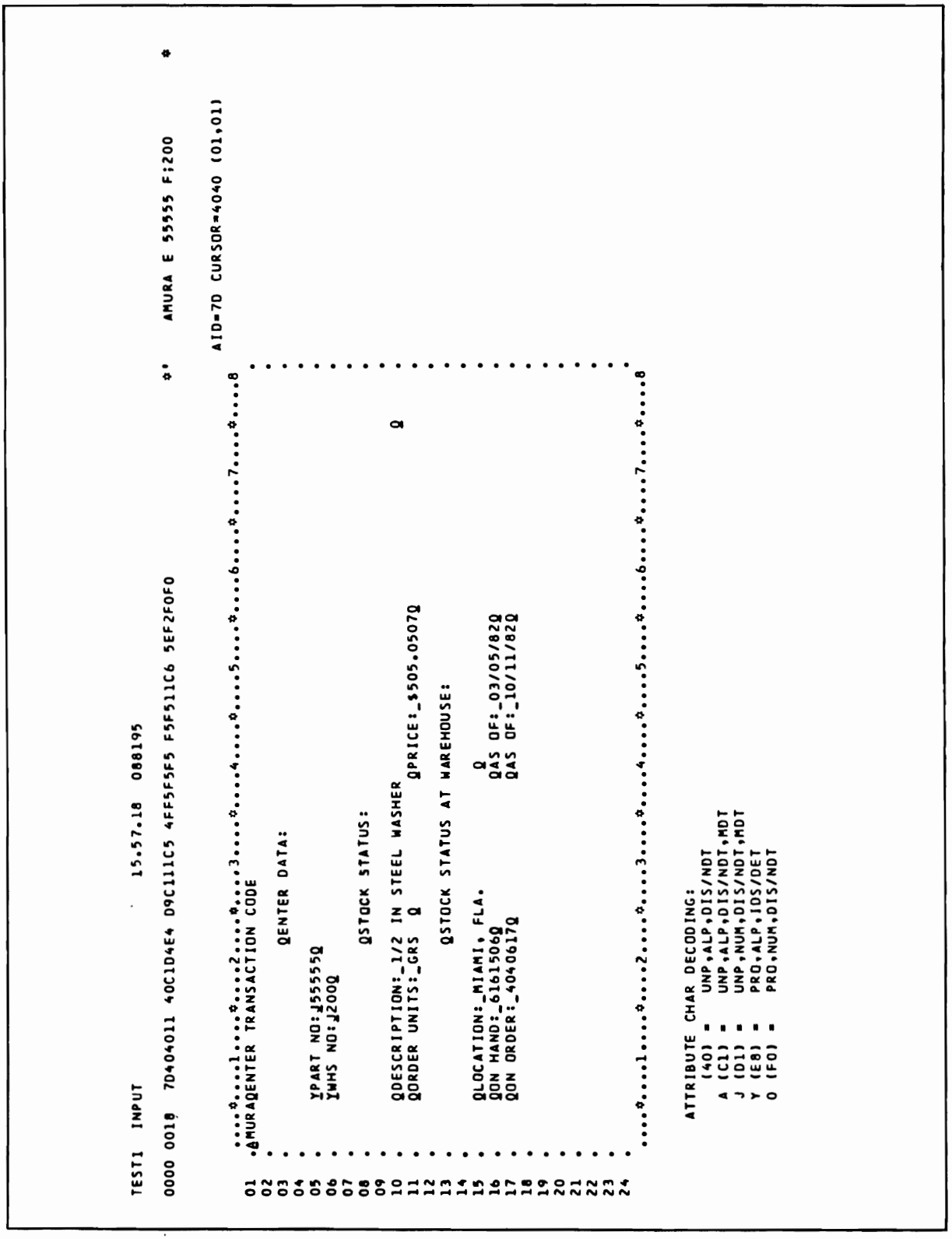
```
TEST1  INPUT                    15.57.04  088195

0000 0018  7D404011 40C1D4E4 D9C111C5 4FF1F2F3 F4F511C6 5EF2F0F0                    *'    AMURA E 12345 F;200        *

                                                                                      AID=7D CURSOR=4040 (01,01)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  .AMURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:J12345Q                                                           .
06  .     YWHS NO:J200Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_                                                    Q        .
11  .     QORDER UNITS:_     Q              QPRICE:_         Q                      .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_                        Q                                       .
16  .     QON HAND:_       Q                QAS OF:_        Q                       .
17  .     QON ORDER:_       Q               QAS OF:_        Q                       .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                                                                                .
24  .                                                                                .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
            (40) =   UNP,ALP,DIS/NDT
          A (C1) =   UNP,ALP,DIS/NDT,MDT
          J (D1) =   UNP,NUM,DIS/NDT,MDT
          Y (E8) =   PRO,ALP,IDS/DET
          0 (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 3 of 20)



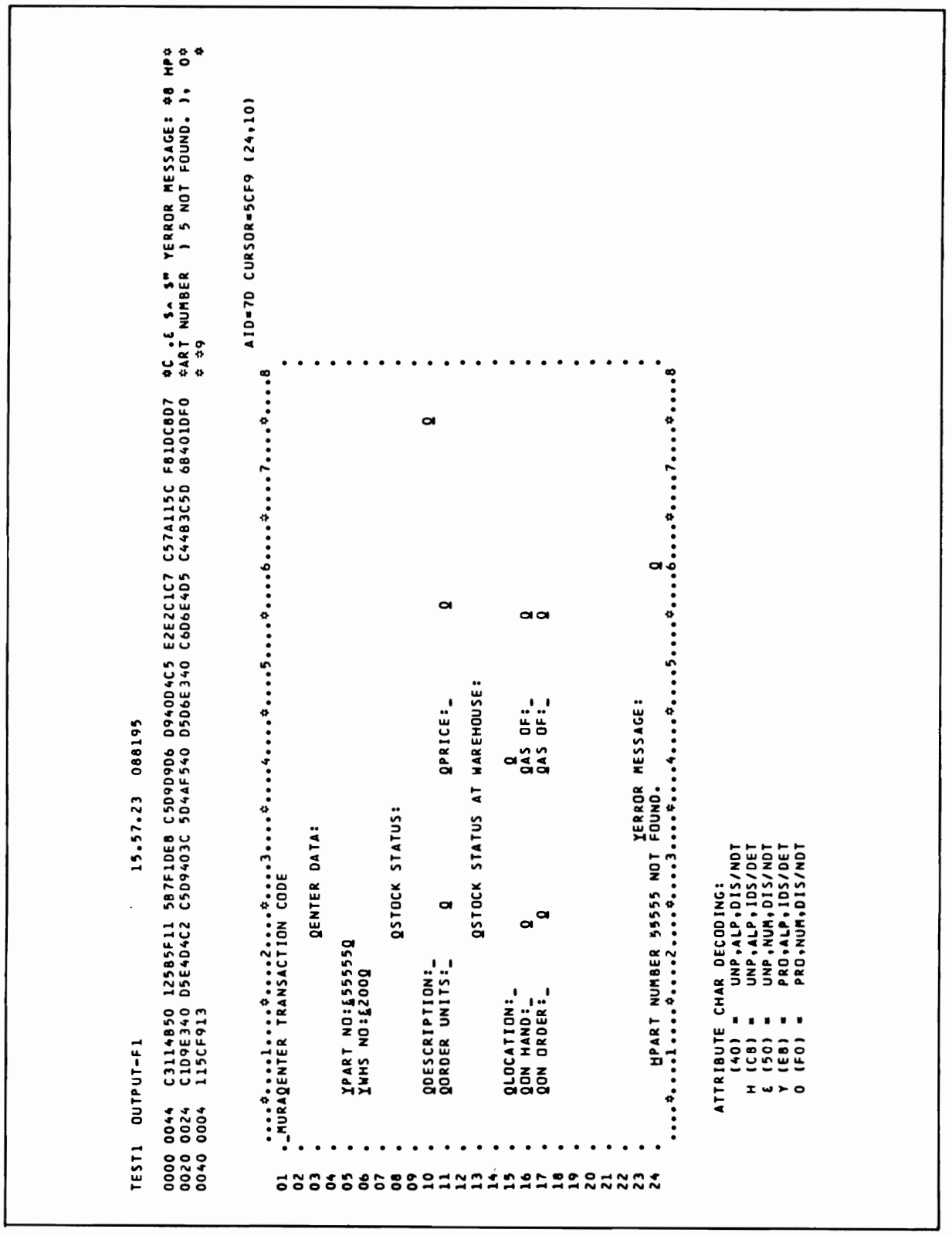
```
TEST1  OUTPUT-F5                    15.57.13  088195

0000 0171  C3114040 1DC1D4E4 D9C11DF0 C5D5E3C5 D940E3D9 C1D5E2C1 C3E3C9D6 D540C3D6  *C    AMURA OENTER TRANSACTION CO*
0020 0151  C4C511C2 F51DF0C5 D5E3C5D9 40C4C1E3 C17A11C5 C51DE8D7 C1D9E340 D5D67A1D  *DE B5 OENTER DATA: EE YPART NO: *
0040 0131  50F1F2F3 F4F51DF0 11C6D51D E8E6C8E2 40D5D67A 1D50F2F0 F01DF011 C9C51DF0  *&12345 0 FN YWHS NO: &200 0 IE 0*
0060 0111  E2E3D6C3 D240E2E3 C1E3E4E2 7A114BD5 1DF0C4C5 E2C3D9C9 D7E3C9D6 D57A1D40  *STOCK STATUS: .N ODESCRIPTION:  *
0080 00F1  F161F240 C9D540E2 E3C5C5D3 40E6C1E2 C8C5D93C 4CD94011 4C5A1DF0 114CE51D  *1/2 IN STEEL WASHER <R  <  0 <V *
00A0 00D1  F0D6D9C4 C5D940E4 D5C9E3E2 7A1D40C7 D9E24040 1DF0114D C61DF0D7 D9C9C3C5  *OORDER UNITS:  GRS   0 (F OPRICE*
00C0 00B1  7A1D405B F5F0F54B F0F5F0F7 1DF0114F D51DF0E2 E3D6C3D2 40E2E3C1 E3E4E240  *:  $505.0507 0  N OSTOCK STATUS *
00E0 0091  C1E340E6 C1D9C5C8 D6E4E2C5 7A11D1E5 1DF0D3D6 C3C1E3C9 D6D57A1D 40D4C9C1  *AT WAREHOUSE: JV OLOCATION:  MIA*
0100 0071  D4C96B40 C6D3C14B 3CD2C740 1DF011D2 F51DF0D6 D540C8C1 D5C47A1D 40F6F1F6  *MI, FLA. KG  0 K5 OON HAND:  616*
0120 0051  F1F5F0F6 1DF011D3 D61DF0C1 E240D6C6 7A1D40F0 F361F0F5 61F8F21D F011D4C5  *1506 0 LO OAS OF:  03/05/82 0 ME*
0140 0031  1DF0D6D5 40D6D9C4 C5D97A1D 40F4F0F4 F0F6F1F7 1DF011D4 E61DF0C1 E240D6C6  * OON ORDER:  4040617 0 MW OAS OF*
0160 0011  7A1D40F1 F061F1F1 61F8F21D F01140C1 13000000                             *:  10/11/82 0  A                *

                                                                                    AID=7D CURSOR=40C1 (01,02)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  ._MURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:&12345Q                                                           .
06  .     YWHS NO:&200Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_1/2 IN STEEL WASHER                                  Q       .
11  .     QORDER UNITS:_GRS  Q              QPRICE:_$505.0507Q                       .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_MIAMI, FLA.             Q                                      .
16  .     QON HAND:_6161506Q                QAS OF:_03/05/82Q                        .
17  .     QON ORDER:_4040617Q               QAS OF:_10/11/82Q                        .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                                                                                .
24  .                                                                                .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
          & (50) =   UNP,NUM,DIS/NDT
          Y (E8) =   PRO,ALP,IDS/DET
          0 (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 4 of 20)



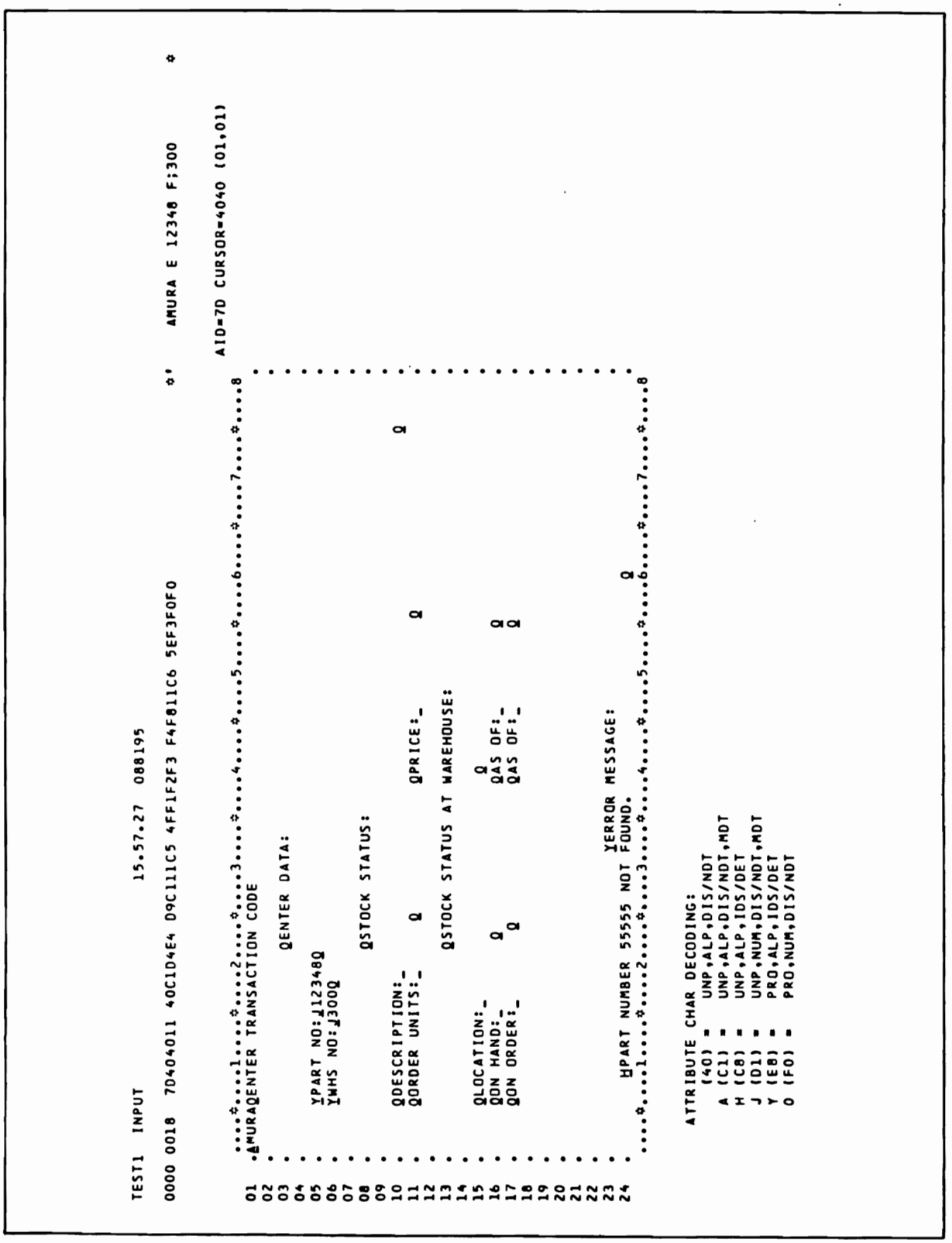
```
TEST1  INPUT                    15.57.18  088195

0000 0018  7D404011 40C1D4E4 D9C111C5 4FF5F5F5 F5F511C6 5EF2F0F0                    *'    AMURA E 55555 F;200        *

                                                                                      AID=7D CURSOR=4040 (01,01)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  .AMURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:J55555Q                                                           .
06  .     YWHS NO:J200Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_1/2 IN STEEL WASHER                                    Q     .
11  .     QORDER UNITS:_GRS  Q              QPRICE:_$505.0507Q                       .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_MIAMI, FLA.             Q                                      .
16  .     QON HAND:_6161506Q                QAS OF:_03/05/82Q                        .
17  .     QON ORDER:_4040617Q               QAS OF:_10/11/82Q                        .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                                                                                .
24  .                                                                                .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
         A (C1) =   UNP,ALP,DIS/NDT,MDT
         J (D1) =   UNP,NUM,DIS/NDT,MDT
         Y (E8) =   PRO,ALP,IDS/DET
         O (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 5 of 20)



```
TEST1  OUTPUT-F1              15.57.23  088195

0000 0044  C3114B50 12585F11 587F1DEB C5D9D9D6 D940D4C5 E2E2C1C7 C57A115C FB10C8D7  *C .£ $* $" YERROR MESSAGE: ** MP*
0020 0024  C1D9E340 D5E4D4C2 C5D9403C 5D4AF540 D5D6E340 C6D6E4D5 C448B3C5D 6B401DF0  *ART NUMBER  ) 5 NOT FOUND. ),  0*
0040 0004  115CF913                                                                  *  *9
                                                                                    *

                                                                    AID=7D CURSOR=5CF9 (24,10)

    ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
 01 _MURAQENTER TRANSACTION CODE
 02 .                                                                              .
 03 .                 QENTER DATA:                                                 .
 04 .                                                                              .
 05 .      YPART NO:&555555Q                                                       .
 06 .      YWHS NO:&200Q                                                           .
 07 .                                                                              .
 08 .                 QSTOCK STATUS:                                               .
 09 .                                                                              .
 10 .      QDESCRIPTION:_                                 Q                        .
 11 .      QORDER UNITS:_      Q          QPRICE:_      Q                          .
 12 .                                                                              .
 13 .                 QSTOCK STATUS AT WAREHOUSE:                                  .
 14 .                                                                              .
 15 .      QLOCATION:_              Q                                              .
 16 .      QON HAND:_       Q       QAS OF:_      Q                                .
 17 .      QON ORDER:_      Q       QAS OF:_      Q                                .
 18 .                                                                              .
 19 .                                                                              .
 20 .                                                                              .
 21 .                                                                              .
 22 .                                                                              .
 23 .                 YERROR MESSAGE:                                              .
 24 .      HPART NUMBER 55555 NOT FOUND.                Q                       .
    ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

    ATTRIBUTE CHAR DECODING:
        (40) =  UNP,ALP,DIS/NDT
      H (C8) =  UNP,ALP,IDS/DET
      & (50) =  UNP,NUM,DIS/NDT
      Y (E8) =  PRO,ALP,IDS/DET
      O (F0) =  PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 6 of 20)

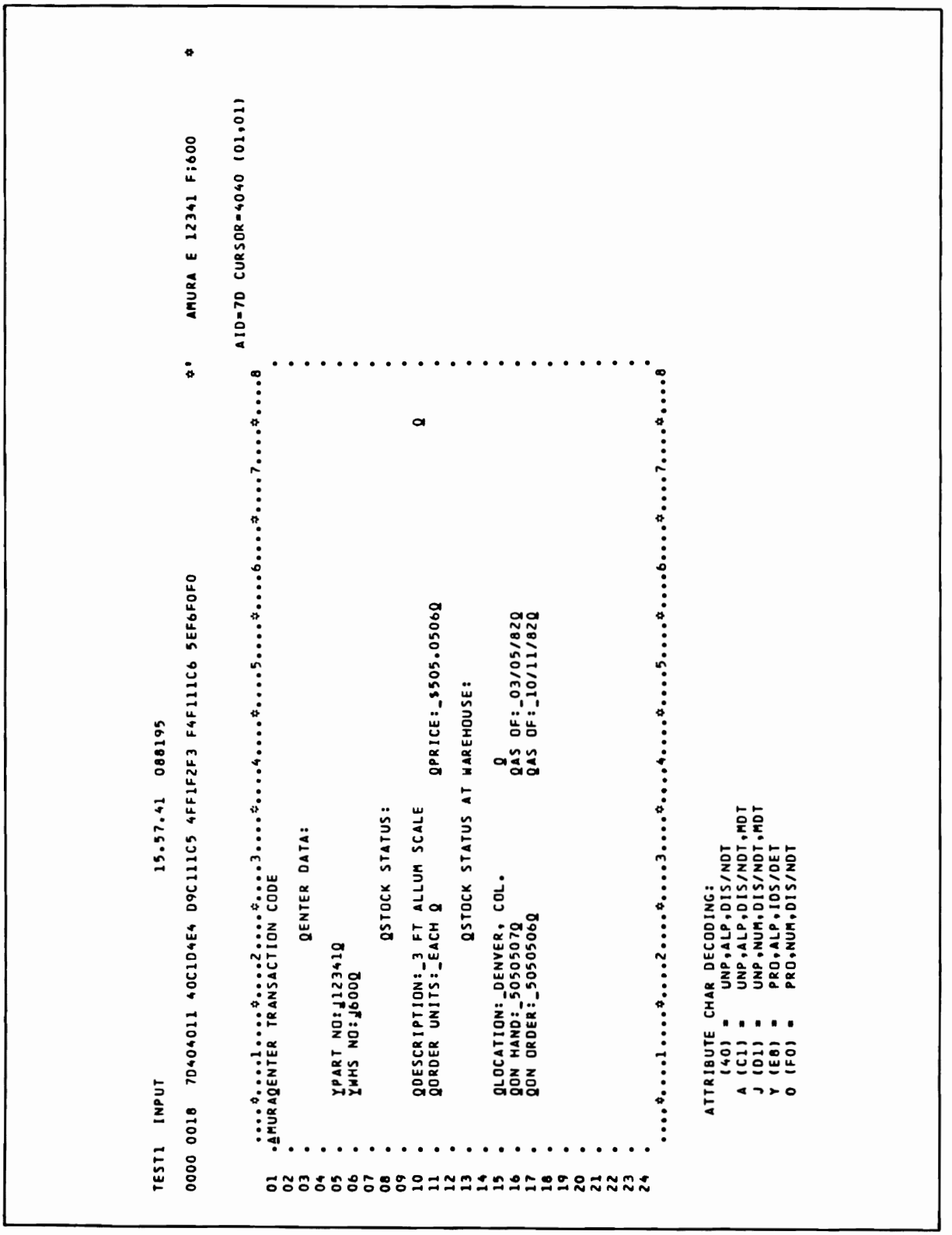


```
TEST1  INPUT                    15.57.27  088195

0000 0018  7D404011 40C1D4E4 D9C111C5 4FF1F2F3 F4F811C6 5EF3F0F0                    *'    AMURA E 12348 F;300        *

                                                                                    AID=7D CURSOR=4040 (01,01)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01   .AMURAQENTER TRANSACTION CODE                                                  .
02   .                                                                              .
03   .                    QENTER DATA:                                              .
04   .                                                                              .
05   .     YPART NO:J12348Q                                                         .
06   .     YWHS NO:J300Q                                                            .
07   .                                                                              .
08   .                    QSTOCK STATUS:                                            .
09   .                                                                              .
10   .     QDESCRIPTION:_                                                    Q      .
11   .     QORDER UNITS:_     Q              QPRICE:_         Q                    .
12   .                                                                              .
13   .                    QSTOCK STATUS AT WAREHOUSE:                               .
14   .                                                                              .
15   .     QLOCATION:_                        Q                                     .
16   .     QON HAND:_       Q                QAS OF:_        Q                      .
17   .     QON ORDER:_       Q               QAS OF:_        Q                      .
18   .                                                                              .
19   .                                                                              .
20   .                                                                              .
21   .                                                                              .
22   .                                                                              .
23   .                              YERROR MESSAGE:                                 .
24   .        HPART NUMBER 55555 NOT FOUND.                    Q                   .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
         A (C1) =   UNP,ALP,DIS/NDT,MDT
         H (C8) =   UNP,ALP,IDS/DET
         J (D1) =   UNP,NUM,DIS/NDT,MDT
         Y (E8) =   PRO,ALP,IDS/DET
         O (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 7 of 20)

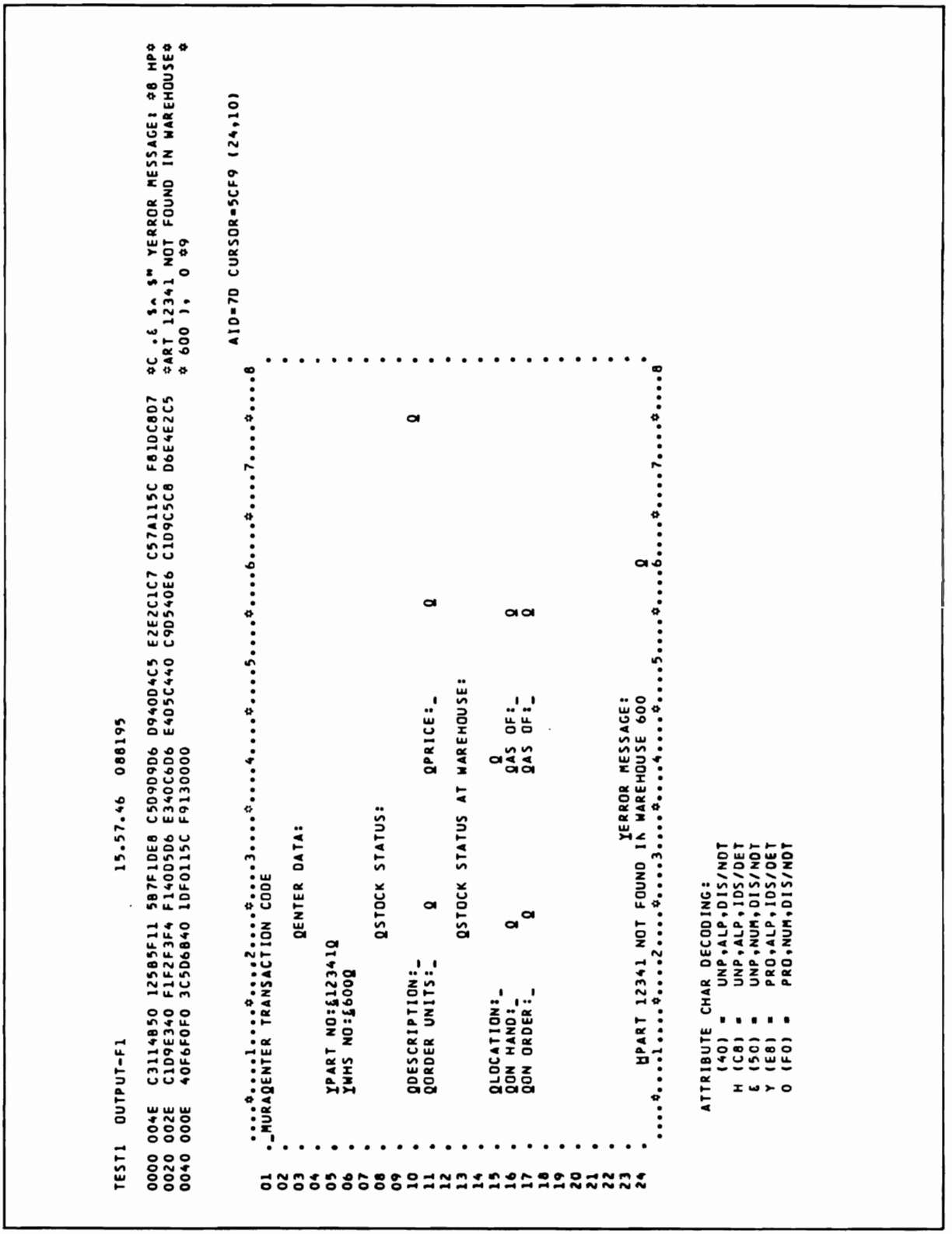
```
TEST1  OUTPUT-F5              15.57.36  088195

0000 016F  C3114040 1DC1D4E4 D9C11DF0 C5D5E3C5 D940E3D9 C1D5E2C1 C3E3C9D6 D540C3D6  *C    AMURA OENTER TRANSACTION CO*
0020 014F  C4C511C2 F51DF0C5 D5E3C5D9 40C4C1E3 C17A11C5 C51DE8D7 C1D9E340 D5D67A1D  *DE 85 OENTER DATA: EE YPART NO: *
0040 012F  50F1F2F3 F4F81DF0 11C6D51D E8E6C8E2 40D5D67A 1D50F3F0 F01DF011 C9C51DF0  *612348 O FN YWHS NO: 6300 O IE O*
0060 010F  E2E3D6C3 D240E2E3 C1E3E4E2 7A1148D5 1DF0C4C5 E2C3D9C9 D7E3C9D6 D57A1D40  *STOCK STATUS: .N ODESCRIPTION: *
0080 0CEF  F340C6E3 40C1D3D3 E4D440E2 C3C1D3C5 3C4CD940 114C5A1D F0114CE5 1DF0D6D9  *3 FT ALLUM SCALE <R  <  O <V OOR*
00A0 00CF  C4C5D940 E4D5C9E3 E27A1D40 C5C1C3C8 401DF011 40DC61DF0 D7D9C9C3 C57A1D40  *DER UNITS:  EACH  O (F OPRICE: *
00C0 00AF  5BF5F0F5 48F0F5F0 F61DF011 4FD51DF0 E2E3D6C3 D240E2E3 C1E3E4E2 40C1E340  *$505.0506 O  N OSTOCK STATUS AT *
00E0 008F  E6C1D9C5 C8D6E4E2 C57A11D1 E51DF0D3 D6C3C1E3 C9D6D57A 1D40C4C5 D5E5C5D9  *WAREHOUSE: JV OLOCATION:  DENVER*
0100 006F  6B4CC3D6 D34B3CD2 C7401DF0 11D2F51D F0D6D540 C8C1D5C4 7A1D40F5 F0F5F0F5  *, COL. KG  O K5 OON HAND:  50505*
0120 004F  F0F71DF0 11D3D61D F0C1E240 D6C67A1D 40F0F361 F0F561F8 F21DF011 D4C51DF0  *07 O LO OAS OF:  03/05/82 O ME O*
0140 002F  D6D540D6 D9C4C5D9 7A1D40F5 F0F5F0F5 F0F61DF0 11D4E61D F0C1E240 D6C67A1D  *ON ORDER:  5050506 O MW OAS OF: *
0160 000F  40F1F061 F1F161F8 F21DF011 40C11300                                      *  10/11/82 O  A                *

                                                                                                AID=7D CURSOR=40C1 (01,02)
 ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01 ._MURAQENTER TRANSACTION CODE                                                    .
02 .                                                                               .
03 .            QENTER DATA:                                                       .
04 .                                                                               .
05 .    YPART NO:&12348Q                                                           .
06 .    YWHS NO:&300Q                                                              .
07 .                                                                               .
08 .            QSTOCK STATUS:                                                     .
09 .                                                                               .
10 .    QDESCRIPTION:_3 FT ALLUM SCALE                  Q                          .
11 .    QORDER UNITS:_EACH Q            QPRICE:_$505.0506Q                         .
12 .                                                                               .
13 .            QSTOCK STATUS AT WAREHOUSE:                                        .
14 .                                                                               .
15 .    QLOCATION:_DENVER, COL.          Q                                         .
16 .    QON HAND:_50505070              QAS OF:_03/05/82Q                          .
17 .    QON ORDER:_5050506Q             QAS OF:_10/11/82Q                          .
18 .                                                                               .
19 .                                                                               .
20 .                                                                               .
21 .                                                                               .
22 .                                                                               .
23 .                                                                               .
24 ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

     ATTRIBUTE CHAR DECODING:
        (40) =  UNP,ALP,DIS/NDT
        (50) =  UNP,NUM,DIS/NDT
        (E8) =  PRO,ALP,IDS/DET
        (F0) =  PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 8 of 20)



```
TEST1  INPUT                    15.57.41  088195

0000 0018  7D404011 40C1D4E4 D9C111C5 4FF1F2F3 F4F111C6 5EF6F0F0                    *'    AMURA E 12341 F;600        *

                                                                                      AID=7D CURSOR=4040 (01,01)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  .AMURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:J12341Q                                                           .
06  .     YWHS NO:J600Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_3 FT ALLUM SCALE                                    Q        .
11  .     QORDER UNITS:_EACH Q              QPRICE:_$505.0506Q                       .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_DENVER, COL.            Q                                       .
16  .     QON HAND:_5050507Q                QAS OF:_03/05/82Q                        .
17  .     QON ORDER:_5050506Q               QAS OF:_10/11/82Q                        .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                                                                                .
24  .                                                                                .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
            (40) =   UNP,ALP,DIS/NDT
          A (C1) =   UNP,ALP,DIS/NDT,MDT
          J (D1) =   UNP,NUM,DIS/NDT,MDT
          Y (E8) =   PRO,ALP,IDS/DET
          O (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 9 of 20)



```
TEST1  OUTPUT-F1                15.57.46  088195

0000 004E  C3114850 12585F11 587F1DE8 C5D9D9D6 D940D4C5 E2E2C1C7 C57A115C F81DC8D7  *C .é §a $" YERROR MESSAGE: *8 HP*
0020 002E  C1D9E340 F1F2F3F4 F140D5D6 E340C6D6 E4D5C440 C9D540E6 C1D9C5C8 D6E4E2C5  *ART 12341 NOT FOUND IN WAREHOUSE*
0040 000E  40F6F0F0 3C5D6B40 1DF0115C F9130000                                     * 600 ), 0 *9
                                                                                                  *

                                                                                AID=7D CURSOR=5CF9 (24,10)

    ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
 01 ._MURAQENTER TRANSACTION CODE
 02 .                                                                              .
 03 .                                                                              .
 04 .                 QENTER DATA:                                                 .
 05 .                                                                              .
 06 .    YPART NO:§12341Q                                                          .
 07 .    YWHS NO:§600Q                                                             .
 08 .                                                                              .
 09 .                 QSTOCK STATUS:                                               .
 10 .                                                                              .
 11 .    QDESCRIPTION:_                                  Q                         .
 12 .    QORDER UNITS:_    Q          QPRICE:_      Q                              .
 13 .                                                                              .
 14 .                 QSTOCK STATUS AT WAREHOUSE:                                  .
 15 .                                                                              .
 16 .    QLOCATION:_               Q                                               .
 17 .    QON HAND:_        Q       QAS OF:_      Q                                 .
 18 .    QON ORDER:_       Q       QAS OF:_      Q                                 .
 19 .                                                                              .
 20 .                                                                              .
 21 .                                                                              .
 22 .                                                                              .
 23 .                 YERROR MESSAGE:                                              .
 24 .    HPART 12341 NOT FOUND IN WAREHOUSE 600                                    .
    ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

    ATTRIBUTE CHAR DECODING:
        (40) =  UNP,ALP,DIS/NOT
      H (C8) =  UNP,ALP,IDS/DET
      E (50) =  UNP,NUM,DIS/NOT
      Y (E8) =  PRO,ALP,IDS/DET
      O (F0) =  PRO,NUM,DIS/NOT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 10 of 20)

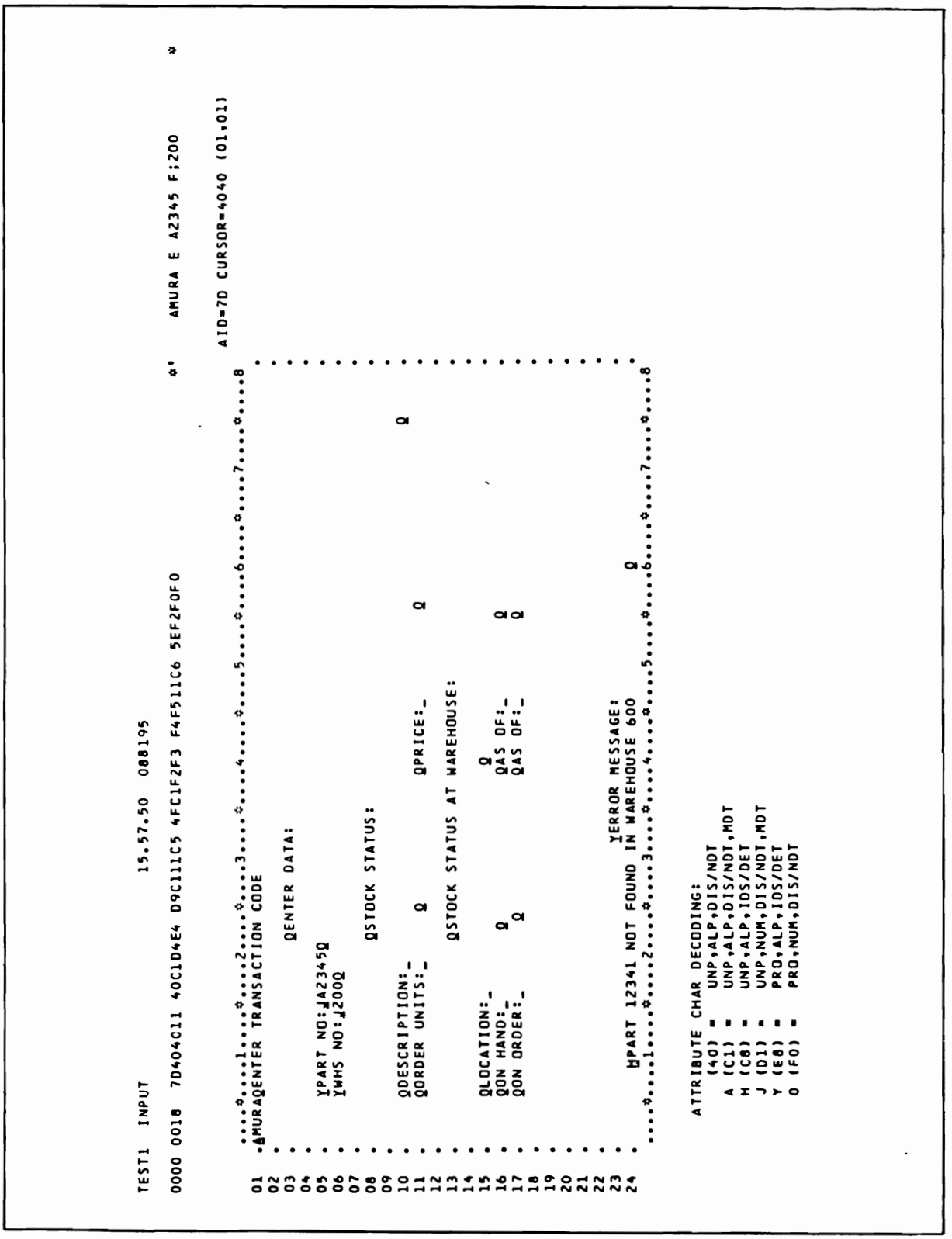


Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 11 of 20)

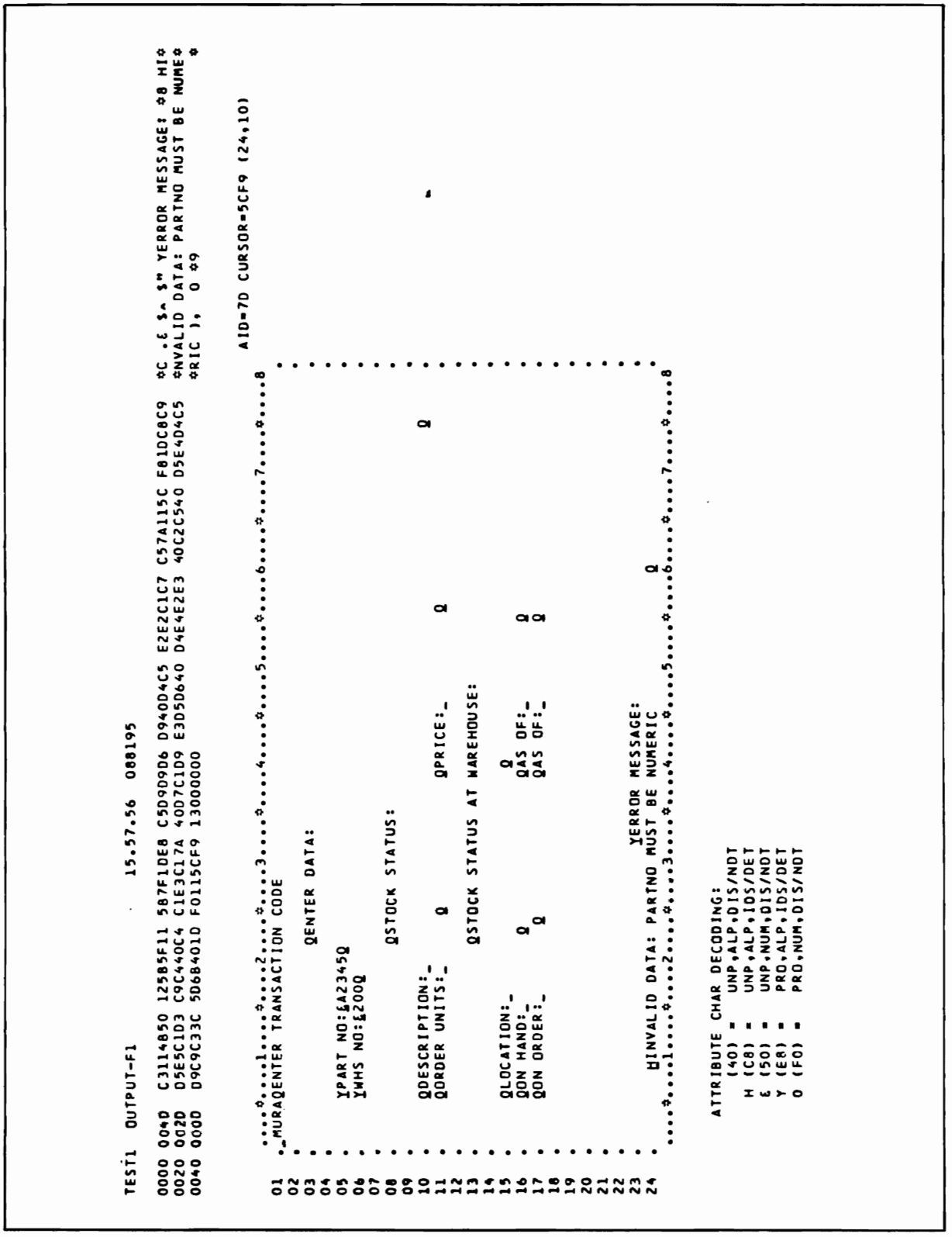


Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 12 of 20)

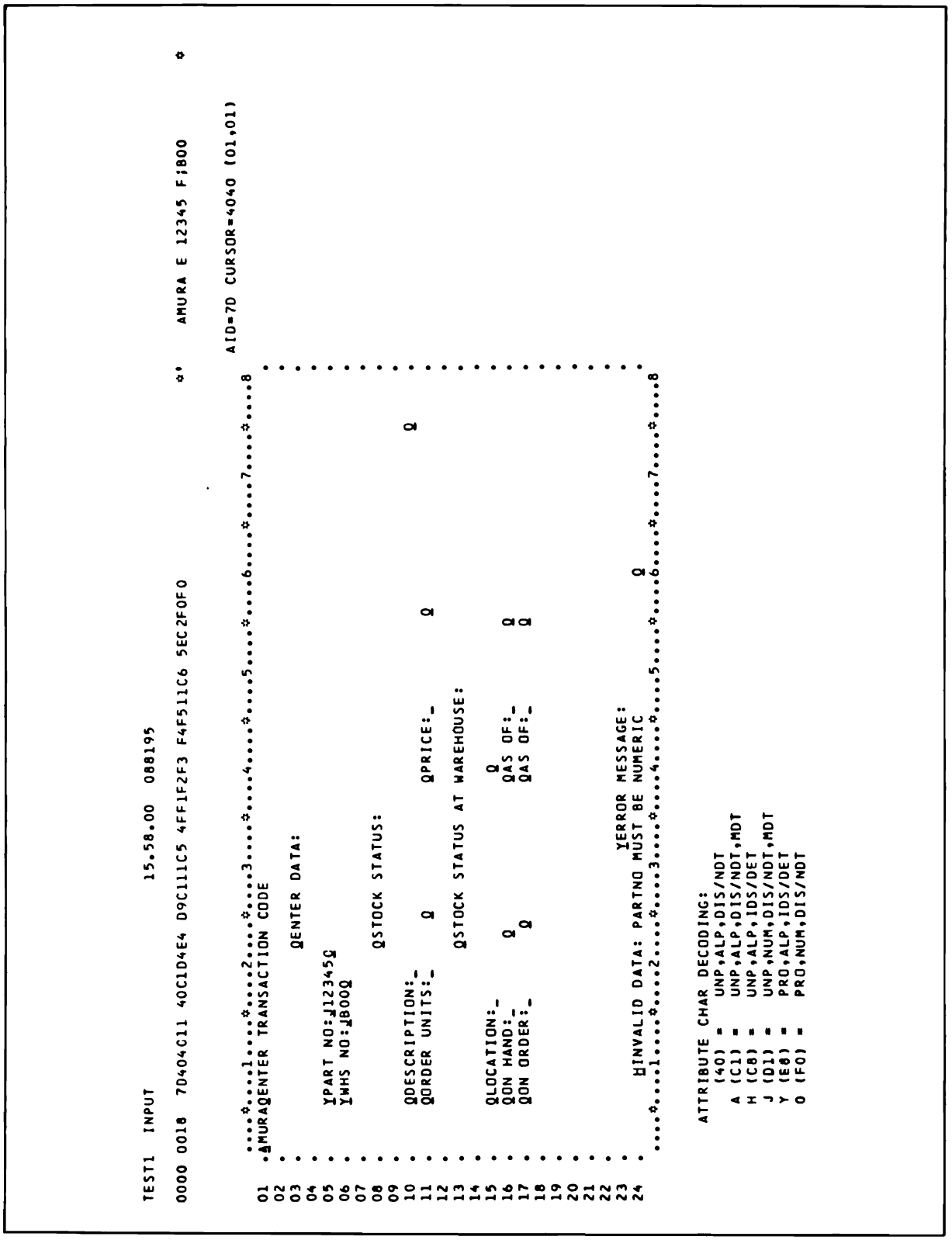
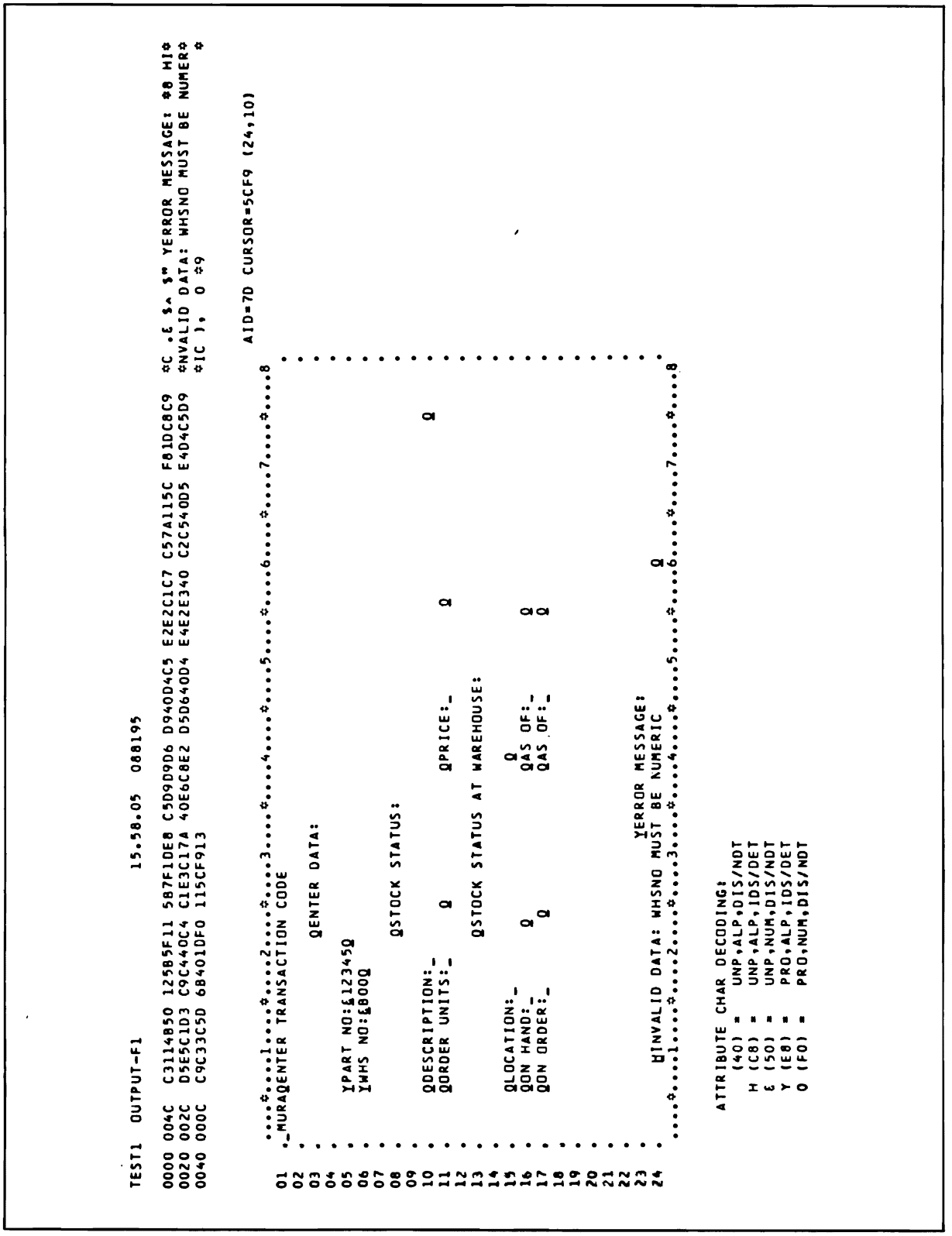
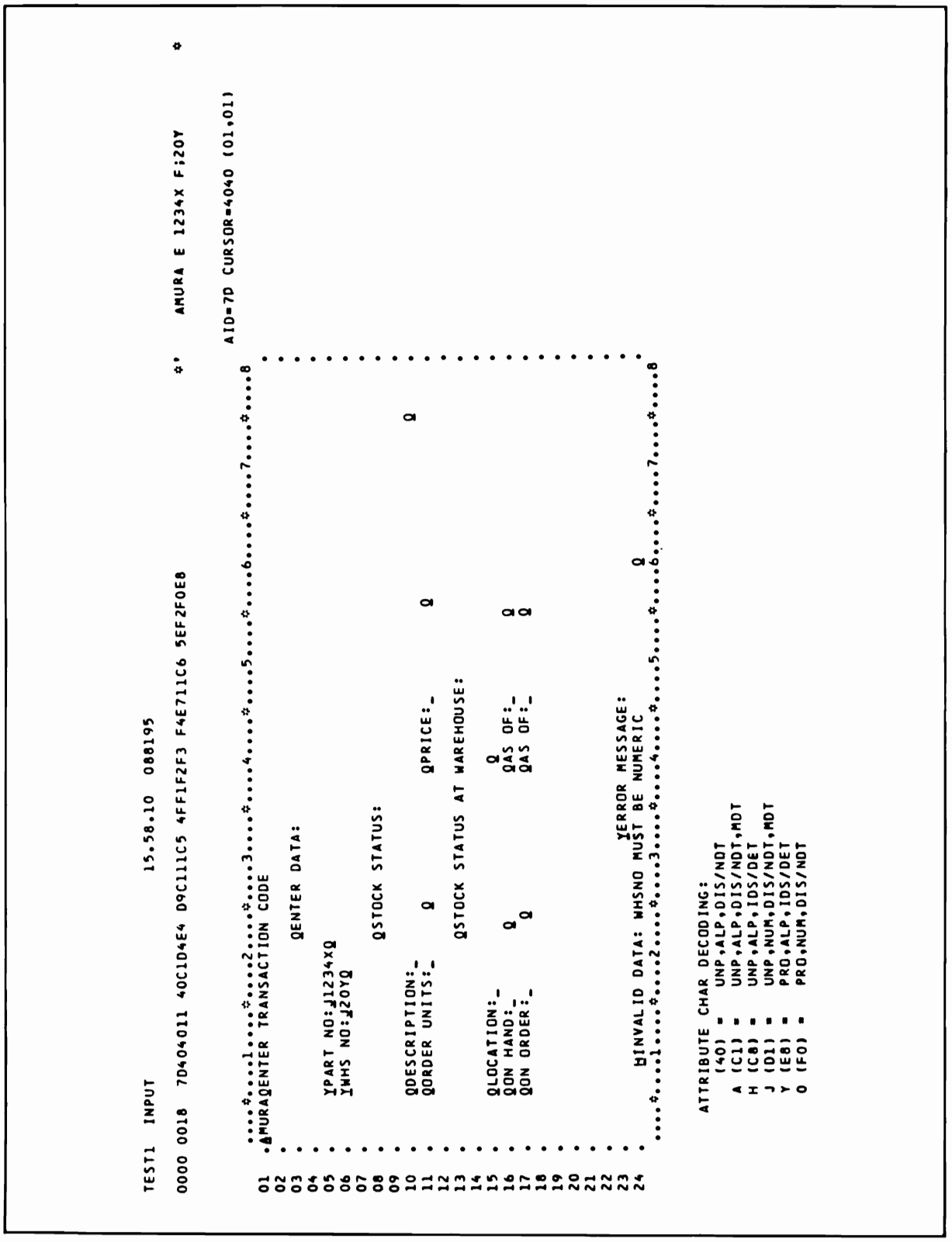


Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 13 of 20)



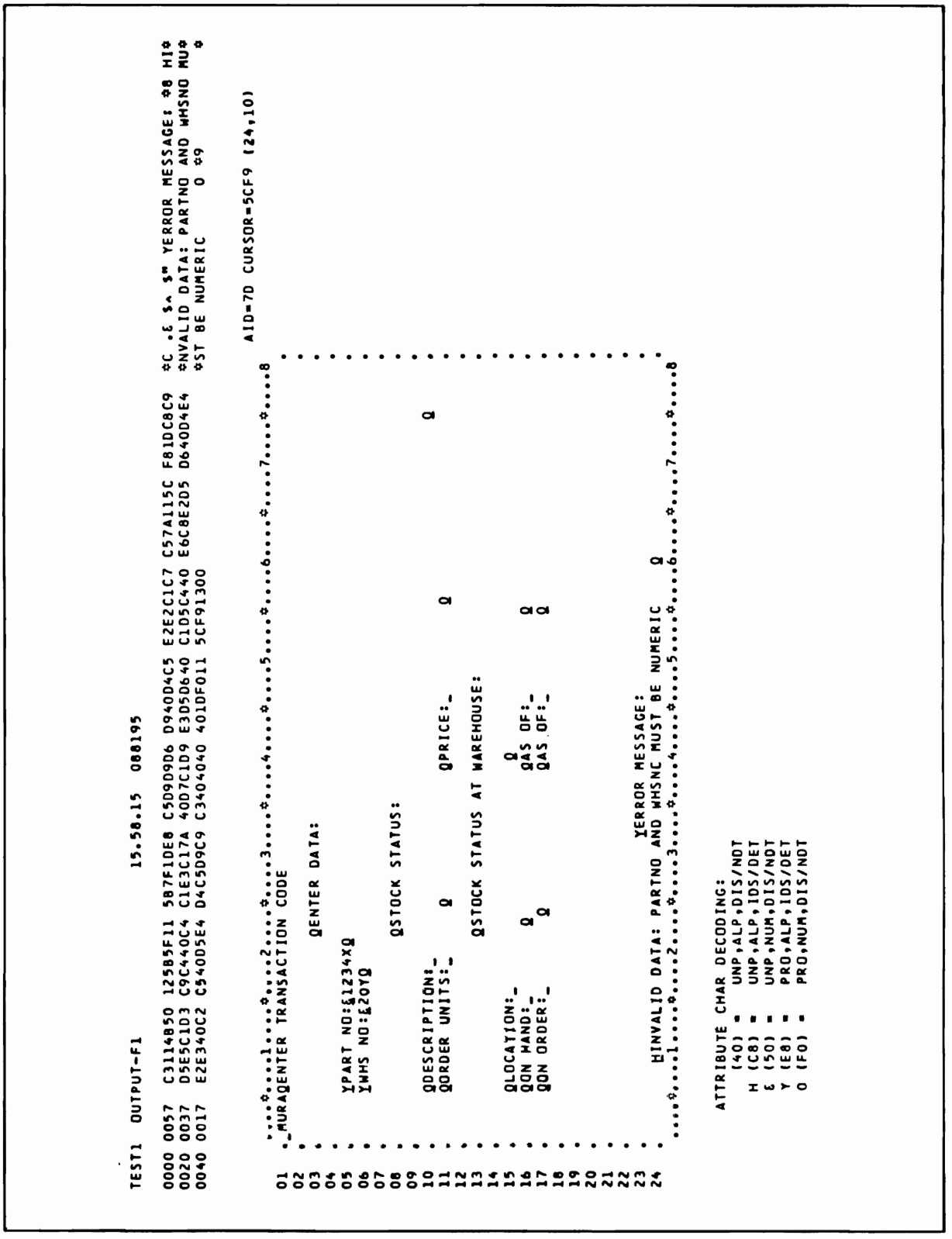
```
TEST1  OUTPUT-F1                    15.58.05  088195

0000 004C  C3114B50 125B5F11 5B7F1DE8 C5D9D9D6 D940D4C5 E2E2C1C7 C57A115C F81DC8C9  *C .£ $^ $" YERROR MESSAGE: *8 HI*
0020 002C  D5E5C1D3 C9C440C4 C1E3C17A 40E6C8E2 D5D640D4 E4E2E340 C2C540D5 E4D4C5D9  *NVALID DATA: WHSNO MUST BE NUMER*
0040 000C  C9C33C5D 6B401DF0 115CF913                                                *IC ),  0 *9                     *

                                                                                      AID=7D CURSOR=5CF9 (24,10)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  ._MURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:£12345Q                                                           .
06  .     YWHS NO:£800Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_                                                      Q      .
11  .     QORDER UNITS:_     Q              QPRICE:_         Q                      .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_                        Q                                       .
16  .     QON HAND:_       Q                QAS OF:_        Q                       .
17  .     QON ORDER:_       Q               QAS OF:_        Q                       .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                              YERROR MESSAGE:                                   .
24  .        HINVALID DATA: WHSNO MUST BE NUMERIC                Q                  .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
         H (C8) =   UNP,ALP,IDS/DET
         £ (50) =   UNP,NUM,DIS/NDT
         Y (E8) =   PRO,ALP,IDS/DET
         Q (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 14 of 20)



```
TEST1  INPUT                    15.58.10  088195

0000 0018  7D404011 40C1D4E4 D9C111C5 4FF1F2F3 F4E711C6 5EF2F0E8                    *'    AMURA E 1234X F;20Y        *

                                                                                      AID=7D CURSOR=4040 (01,01)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  .AMURAQENTER TRANSACTION CODE                                                     .
02  .                                                                                 .
03  .                    QENTER DATA:                                                 .
04  .                                                                                 .
05  .     YPART NO:J1234XQ                                                            .
06  .     YWHS NO:J20YQ                                                               .
07  .                                                                                 .
08  .                    QSTOCK STATUS:                                               .
09  .                                                                                 .
10  .     QDESCRIPTION:_                                                     Q        .
11  .     QORDER UNITS:_     Q             QPRICE:_         Q                        .
12  .                                                                                 .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                  .
14  .                                                                                 .
15  .     QLOCATION:_                       Q                                         .
16  .     QON HAND:_       Q               QAS OF:_        Q                         .
17  .     QON ORDER:_       Q              QAS OF:_        Q                         .
18  .                                                                                 .
19  .                                                                                 .
20  .                                                                                 .
21  .                                                                                 .
22  .                                                                                 .
23  .                              YERROR MESSAGE:                                    .
24  .        HINVALID DATA: WHSNO MUST BE NUMERIC              Q                     .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
         A (C1) =   UNP,ALP,DIS/NDT,MDT
         H (C8) =   UNP,ALP,IDS/DET
         J (D1) =   UNP,NUM,DIS/NDT,MDT
         Y (E8) =   PRO,ALP,IDS/DET
         0 (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 15 of 20)



```
TEST1  OUTPUT-F1                    15.58.15  088195

0000 0057  C3114B50 125B5F11 5B7F1DE8 C5D9D9D6 D940D4C5 E2E2C1C7 C57A115C F81DC8C9  *C .& $^ $" YERROR MESSAGE: *8 HI*
0020 0037  D5E5C1D3 C9C440C4 C1E3C17A 40D7C1D9 E3D5D640 C1D5C440 E6C8E2D5 D640D4E4  *NVALID DATA: PARTNO AND WHSNO MU*
0040 0017  E2E340C2 C540D5E4 D4C5D9C9 C3404040 401DF011 5CF91300                    *ST BE NUMERIC     0 *9          *

                                                                                      AID=7D CURSOR=5CF9 (24,10)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  ._MURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:&1234XQ                                                           .
06  .     YWHS NO:&20YQ                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_                                                    Q        .
11  .     QORDER UNITS:_     Q             QPRICE:_         Q                       .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_                       Q                                        .
16  .     QON HAND:_       Q               QAS OF:_        Q                        .
17  .     QON ORDER:_       Q              QAS OF:_        Q                        .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                              YERROR MESSAGE:                                   .
24  .        HINVALID DATA: PARTNO AND WHSNC MUST BE NUMERIC    Q                   .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
         H (C8) =   UNP,ALP,IDS/DET
         & (50) =   UNP,NUM,DIS/NDT
         Y (E8) =   PRO,ALP,IDS/DET
         0 (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 16 of 20)

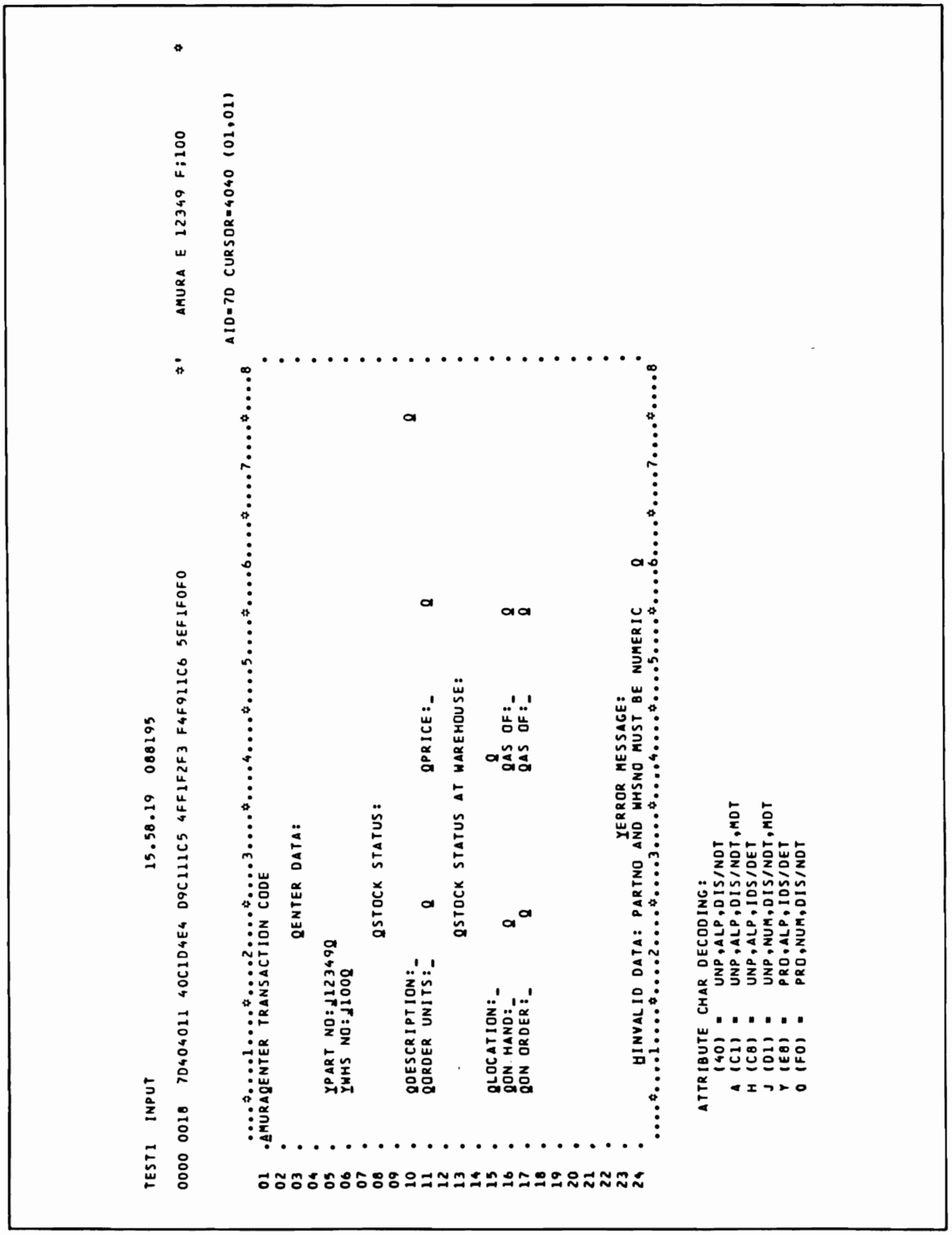
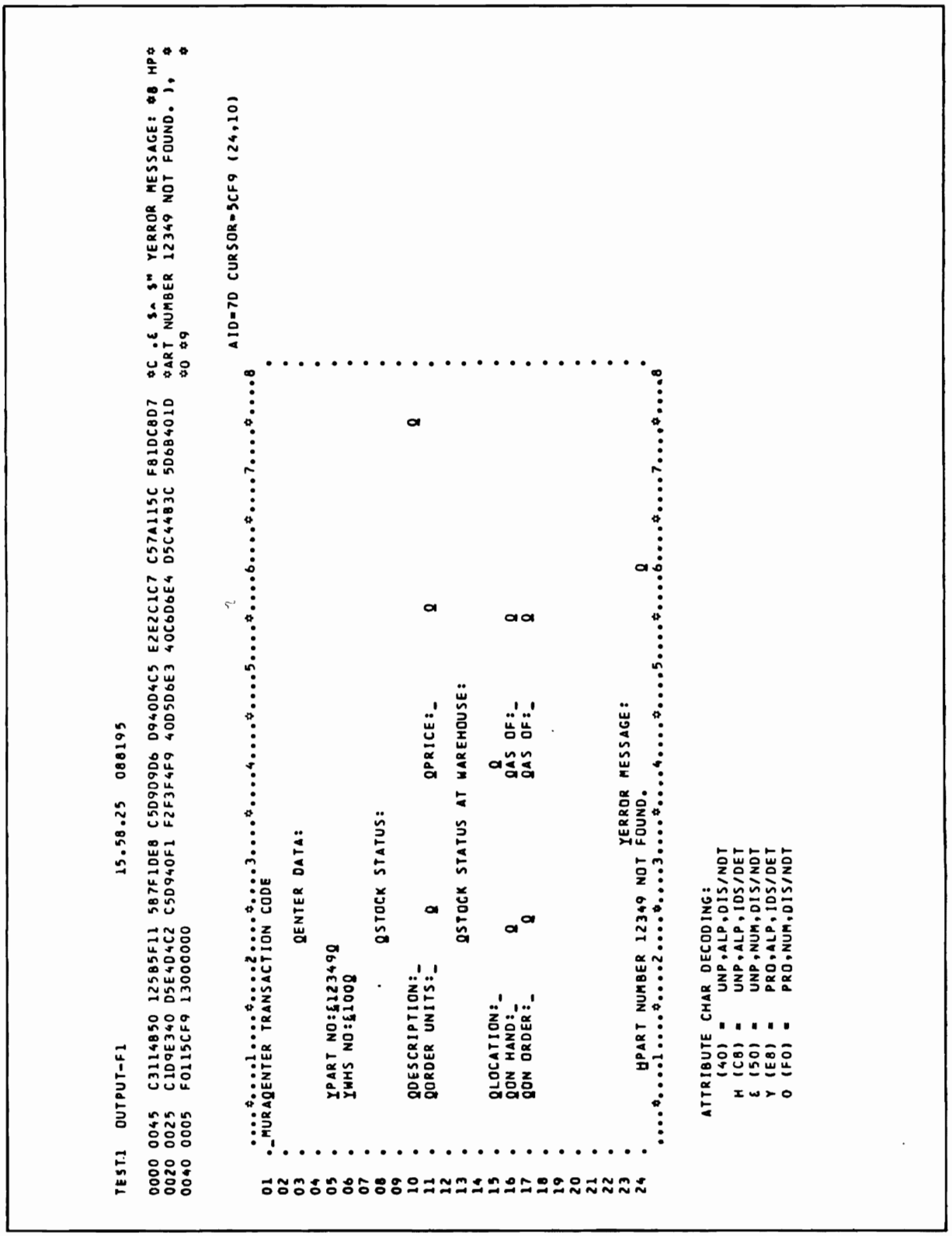
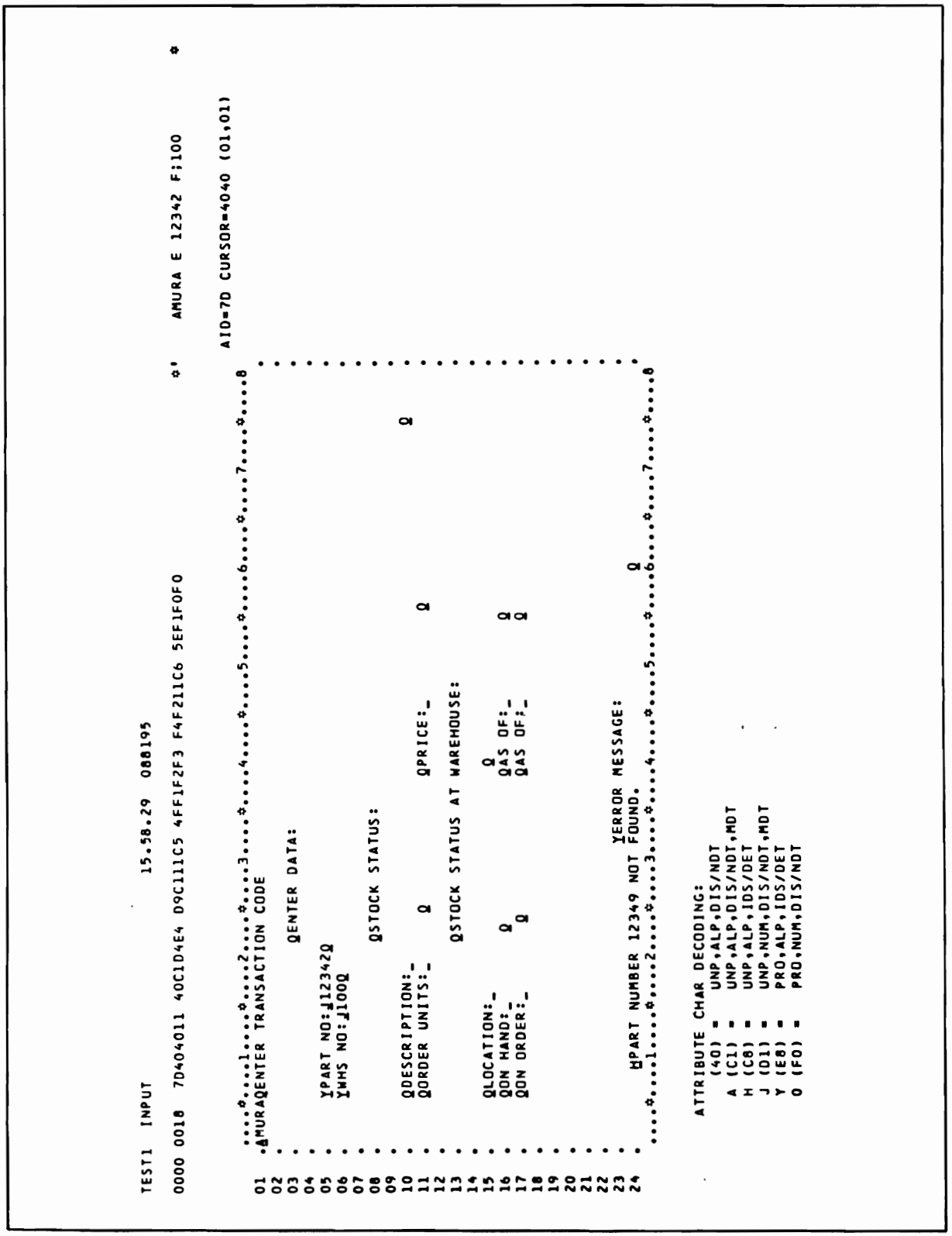


Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 17 of 20)



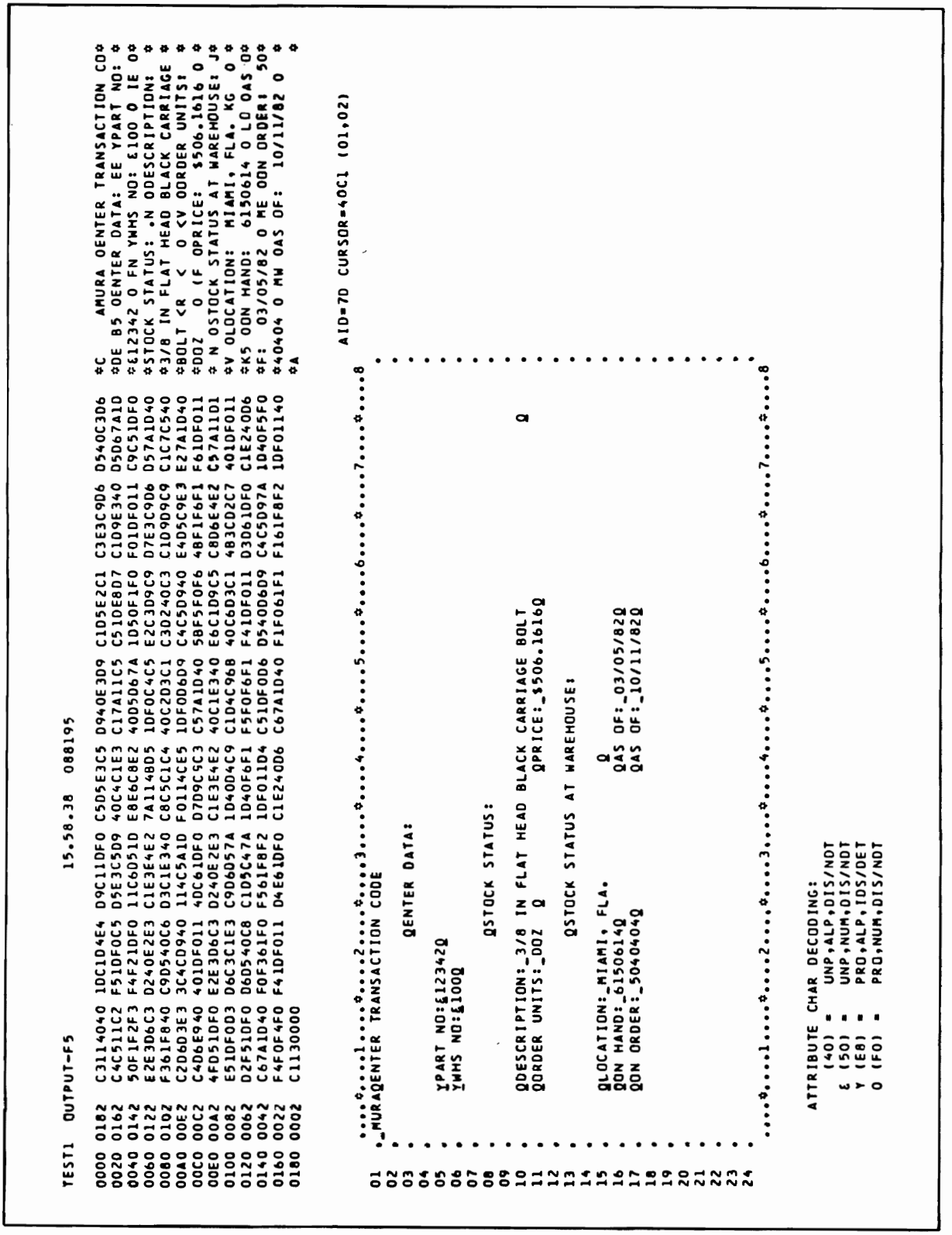
```
TEST1  OUTPUT-F1                    15.58.25  088195

0000 0045  C3114850 12585F11 587F1DE8 C5D9D9D6 D940D4C5 E2E2C1C7 C57A115C F81DC8D7  *C .£ $^ $" YERROR MESSAGE: *8 HP*
0020 0025  C1D9E340 D5E4D4C2 C5D940F1 F2F3F4F9 40D5D6E3 40C6D6E4 D5C4483C 5D6B401D  *ART NUMBER 12349 NOT FOUND. ),  *
0040 0005  F0115CF9 13000000                                                        *0 *9                            *

                                                                                      AID=7D CURSOR=5CF9 (24,10)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  ._MURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:£12349Q                                                           .
06  .     YWHS NO:£100Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_                                                    Q        .
11  .     QORDER UNITS:_     Q            QPRICE:_         Q                        .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_                      Q                                        .
16  .     QON HAND:_       Q              QAS OF:_        Q                         .
17  .     QON ORDER:_       Q             QAS OF:_        Q                         .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                              YERROR MESSAGE:                                   .
24  .        HPART NUMBER 12349 NOT FOUND.                    Q                     .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
          H (C8) =   UNP,ALP,IDS/DET
          £ (50) =   UNP,NUM,DIS/NDT
          Y (E8) =   PRO,ALP,IDS/DET
          0 (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 18 of 20)



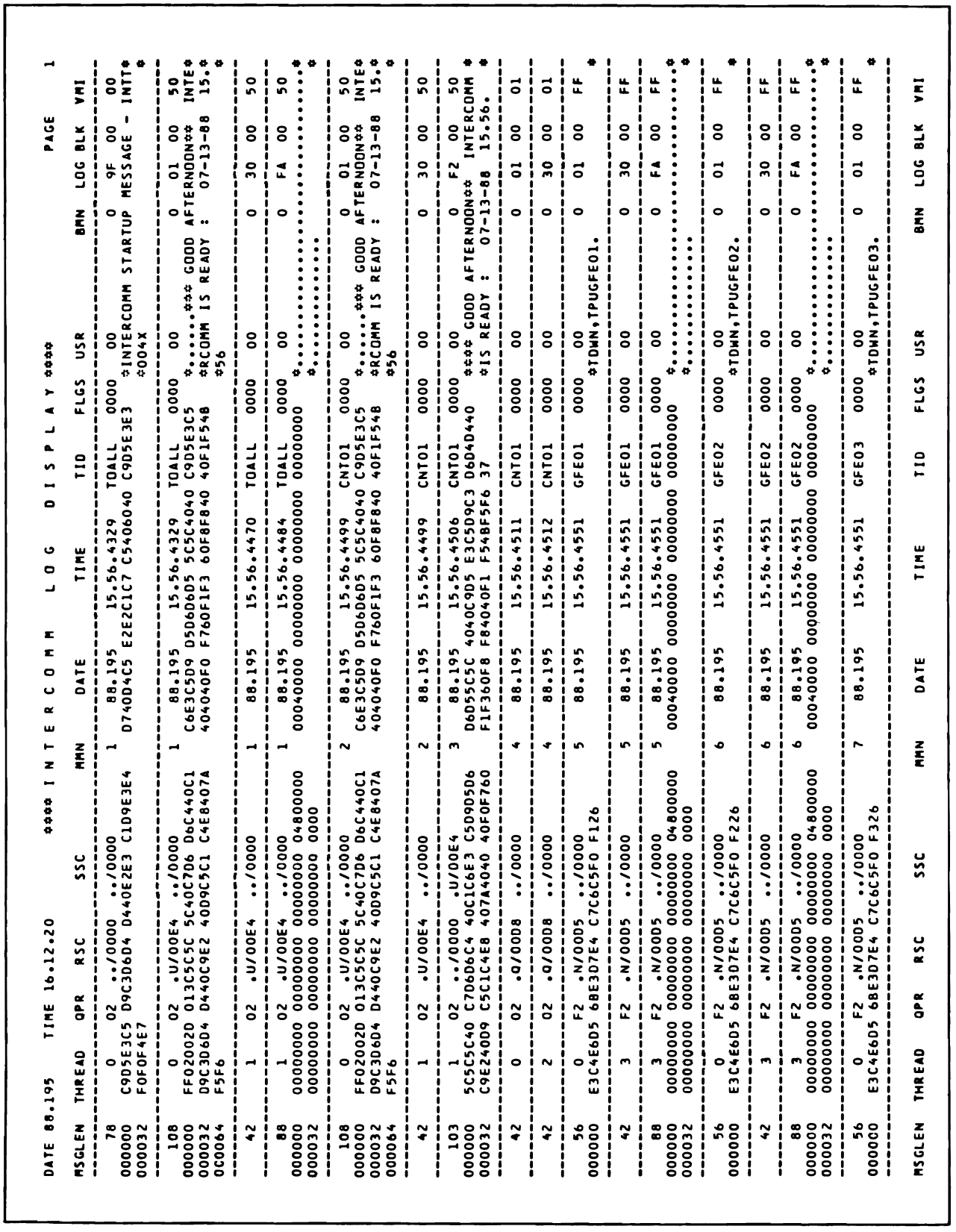
```
TEST1  INPUT                    15.58.29  088195

0000 0018  7D404011 40C1D4E4 D9C111C5 4FF1F2F3 F4F211C6 5EF1F0F0                    *'    AMURA E 12342 F;100        *

                                                                                      AID=7D CURSOR=4040 (01,01)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  .AMURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:J12342Q                                                           .
06  .     YWHS NO:J100Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_                                                    Q        .
11  .     QORDER UNITS:_     Q             QPRICE:_         Q                        .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_                       Q                                        .
16  .     QON HAND:_       Q               QAS OF:_        Q                        .
17  .     QON ORDER:_       Q              QAS OF:_        Q                        .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                              YERROR MESSAGE:                                   .
24  .        HPART NUMBER 12349 NOT FOUND.                    Q                     .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

       ATTRIBUTE CHAR DECODING:
          (40) =   UNP,ALP,DIS/NDT
        A (C1) =   UNP,ALP,DIS/NDT,MDT
        H (C8) =   UNP,ALP,IDS/DET
        J (D1) =   UNP,NUM,DIS/NDT,MDT
        Y (E8) =   PRO,ALP,IDS/DET
        0 (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 19 of 20)



```
TEST1  OUTPUT-F5                    15.58.38  088195

0000 0182  C3114040 1DC1D4E4 D9C11DF0 C5D5E3C5 D940E3D9 C1D5E2C1 C3E3C9D6 D540C3D6  *C    AMURA OENTER TRANSACTION CO*
0020 0162  C4C511C2 F51DF0C5 D5E3C5D9 40C4C1E3 C17A11C5 C51DE8D7 C1D9E340 D5D67A1D  *DE B5 OENTER DATA: EE YPART NO: *
0040 0142  50F1F2F3 F4F21DF0 11C6D51D E8E6C8E2 40D5D67A 1D50F1F0 F01DF011 C9C51DF0  *£12342 0 FN YWHS NO: £100 0 IE 0*
0060 0122  E2E3D6C3 D240E2E3 C1E3E4E2 7A114BD5 1DF0C4C5 E2C3D9C9 D7E3C9D6 D57A1D40  *STOCK STATUS: .N ODESCRIPTION:  *
0080 0102  F361F840 C9D540C6 D3C1E340 C8C5C1C4 40C2D3C1 C3D240C3 C1D9D9C9 C1C7C540  *3/8 IN FLAT HEAD BLACK CARRIAGE *
00A0 00E2  C2D6D3E3 3C4CD940 114C5A1D F0114CE5 1DF0D6D9 C4C5D940 E4D5C9E3 E27A1D40  *BOLT <R  <  0 <V OORDER UNITS:  *
00C0 00C2  C4D6E940 401DF011 4DC61DF0 D7D9C9C3 C57A1D40 5BF5F0F6 4BF1F6F1 F61DF011  *DOZ   0 (F OPRICE:  $506.1616 0 *
00E0 00A2  4FD51DF0 E2E3D6C3 D240E2E3 C1E3E4E2 40C1E340 E6C1D9C5 C8D6E4E2 C57A11D1  * N OSTOCK STATUS AT WAREHOUSE: J*
0100 0082  E51DF0D3 D6C3C1E3 C9D6D57A 1D40D4C9 C1D4C96B 40C6D3C1 4B3CD2C7 401DF011  *V OLOCATION:  MIAMI, FLA. KG  0 *
0120 0062  D2F51DF0 D6D540C8 C1D5C47A 1D40F6F1 F5F0F6F1 F41DF011 D3D61DF0 C1E240D6  *K5 OON HAND:  6150614 0 LO OAS 0*
0140 0042  C67A1D40 F0F361F0 F561F8F2 1DF011D4 C51DF0D6 D540D6D9 C4C5D97A 1D40F5F0  *F:  03/05/82 0 ME OON ORDER:  50*
0160 0022  F4F0F4F0 F41DF011 D4E61DF0 C1E240D6 C67A1D40 F1F061F1 F161F8F2 1DF01140  *40404 0 MW OAS OF:  10/11/82 0  *
0180 0002  C1130000                                                                 *A                                *

                                                                                     AID=7D CURSOR=40C1 (01,02)
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
01  ._MURAQENTER TRANSACTION CODE                                                    .
02  .                                                                                .
03  .                    QENTER DATA:                                                .
04  .                                                                                .
05  .     YPART NO:£12342Q                                                           .
06  .     YWHS NO:£100Q                                                              .
07  .                                                                                .
08  .                    QSTOCK STATUS:                                              .
09  .                                                                                .
10  .     QDESCRIPTION:_3/8 IN FLAT HEAD BLACK CARRIAGE BOLT                  Q      .
11  .     QORDER UNITS:_DOZ  Q              QPRICE:_$506.1616Q                       .
12  .                                                                                .
13  .                    QSTOCK STATUS AT WAREHOUSE:                                 .
14  .                                                                                .
15  .     QLOCATION:_MIAMI, FLA.             Q                                      .
16  .     QON HAND:_6150614Q                QAS OF:_03/05/82Q                        .
17  .     QON ORDER:_5040404Q               QAS OF:_10/11/82Q                        .
18  .                                                                                .
19  .                                                                                .
20  .                                                                                .
21  .                                                                                .
22  .                                                                                .
23  .                                                                                .
24  .                                                                                .
     ....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8

        ATTRIBUTE CHAR DECODING:
           (40) =   UNP,ALP,DIS/NDT
          £ (50) =   UNP,NUM,DIS/NDT
          Y (E8) =   PRO,ALP,IDS/DET
          O (F0) =   PRO,NUM,DIS/NDT
```

Figure 54. SIM3270 Printout from Simulation Mode Execution (Page 20 of 20)



```
DATE 88.195    TIME 16.12.20        **** I N T E R C O M M   L O G   D I S P L A Y ****                    PAGE     1

MSGLEN  THREAD   QPR   RSC      SSC         MMN     DATE        TIME       TID    FLGS  USR          BMN  LOG BLK  VMI
-----------------------------------------------------------------------------------------------------------------------
   78      0     02  ../0000  ../0000         1     88.195    15.56.4329   TOALL   0000   00            0   9F  00   00
000000   C9D5E3C5 D9C3D6D4 D440E2E3 C1D9E3E4    D740D4C5 E2E2C1C7 C5406040 C9D5E3E3   *INTERCOMM STARTUP MESSAGE - INTT*
000032   F0F0F4E7                                                                      *004X                            *
-----------------------------------------------------------------------------------------------------------------------
  108      0     02  .U/00E4  ../0000         1     88.195    15.56.4329   TOALL   0000   00            0   01  00   50
000000   FF02002D 013C5C5C 5C40C7D6 D6C440C1    C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *......*** GOOD AFTERNOON**  INTE*
000032   D9C3D6D4 D440C9E2 40D9C5C1 C4E8407A    404040F0 F760F1F3 60F8F840 40F1F54B   *RCOMM IS READY :   07-13-88  15.*
0C0064   F5F6                                                                          *56                              *
-----------------------------------------------------------------------------------------------------------------------
   42      1     02  .U/00E4  ../0000         1     88.195    15.56.4470   TOALL   0000   00            0   30  00   50
-----------------------------------------------------------------------------------------------------------------------
   88      1     02  .U/00E4  ../0000         1     88.195    15.56.4484   TOALL   0000   00            0   FA  00   50
000000   00000000 00000000 00000000 04800000    00040000 00000000 00000000 00000000   *................................*
000032   00000000 00000000 00000000 0000                                                *..............                  *
-----------------------------------------------------------------------------------------------------------------------
  108      0     02  .U/00E4  ../0000         2     88.195    15.56.4499   CNT01   0000   00            0   01  00   50
000000   FF02002D 013C5C5C 5C40C7D6 D6C440C1    C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *......*** GOOD AFTERNOON**  INTE*
000032   D9C3D6D4 D440C9E2 40D9C5C1 C4E8407A    404040F0 F760F1F3 60F8F840 40F1F54B   *RCOMM IS READY :   07-13-88  15.*
000064   F5F6                                                                          *56                              *
-----------------------------------------------------------------------------------------------------------------------
   42      1     02  .U/00E4  ../0000         2     88.195    15.56.4499   CNT01   0000   00            0   30  00   50
-----------------------------------------------------------------------------------------------------------------------
  103      1     02  ../0000  .U/00E4         3     88.195    15.56.4506   CNT01   0000   00            0   F2  00   50
000000   5C5C5C40 C7D6D6C4 40C1C6E3 C5D9D5D6    D6D55C5C 4040C9D5 E3C5D9C3 D6D4D440   **** GOOD AFTERNOON**  INTERCOMM *
000032   C9E240D9 C5C1C4E8 407A4040 40F0F760    F1F360F8 F84040F1 F54BF5F6 37         *IS READY :   07-13-88  15.56.   *
-----------------------------------------------------------------------------------------------------------------------
   42      0     02  .Q/00D8  ../0000         4     88.195    15.56.4511   CNT01   0000   00            0   01  00   01
-----------------------------------------------------------------------------------------------------------------------
   42      2     02  .Q/00D8  ../0000         4     88.195    15.56.4512   CNT01   0000   00            0   30  00   01
-----------------------------------------------------------------------------------------------------------------------
   56      0     F2  .N/00D5  ../0000         5     88.195    15.56.4551   GFE01   0000   00            0   01  00   FF
000000   E3C4E6D5 6BE3D7E4 C7C6C5F0 F126                                                *TDWN,TPUGFE01.                  *
-----------------------------------------------------------------------------------------------------------------------
   42      3     F2  .N/00D5  ../0000         5     88.195    15.56.4551   GFE01   0000   00            0   30  00   FF
-----------------------------------------------------------------------------------------------------------------------
   88      3     F2  .N/00D5  ../0000         5     88.195    15.56.4551   GFE01   0000   00            0   FA  00   FF
000000   00000000 00000000 00000000 04800000    00040000 00000000 00000000 00000000   *................................*
000032   00000000 00000000 00000000 0000                                                *..............                  *
-----------------------------------------------------------------------------------------------------------------------
   56      0     F2  .N/00D5  ../0000         6     88.195    15.56.4551   GFE02   0000   00            0   01  00   FF
000000   E3C4E6D5 6BE3D7E4 C7C6C5F0 F226                                                *TDWN,TPUGFE02.                  *
-----------------------------------------------------------------------------------------------------------------------
   42      3     F2  .N/00D5  ../0000         6     88.195    15.56.4551   GFE02   0000   00            0   30  00   FF
-----------------------------------------------------------------------------------------------------------------------
   88      3     F2  .N/00D5  ../0000         6     88.195    15.56.4551   GFE02   0000   00            0   FA  00   FF
000000   00000000 00000000 00000000 04800000    00040000 00000000 00000000 00000000   *................................*
000032   00000000 00000000 00000000 0000                                                *..............                  *
-----------------------------------------------------------------------------------------------------------------------
   56      0     F2  .N/00D5  ../0000         7     88.195    15.56.4551   GFE03   0000   00            0   01  00   FF
000000   E3C4E6D5 6BE3D7E4 C7C6C5F0 F326                                                *TDWN,TPUGFE03.                  *
-----------------------------------------------------------------------------------------------------------------------

MSGLEN  THREAD   QPR   RSC      SSC         MMN     DATE        TIME       TID    FLGS  USR          BMN  LOG BLK  VMI
```

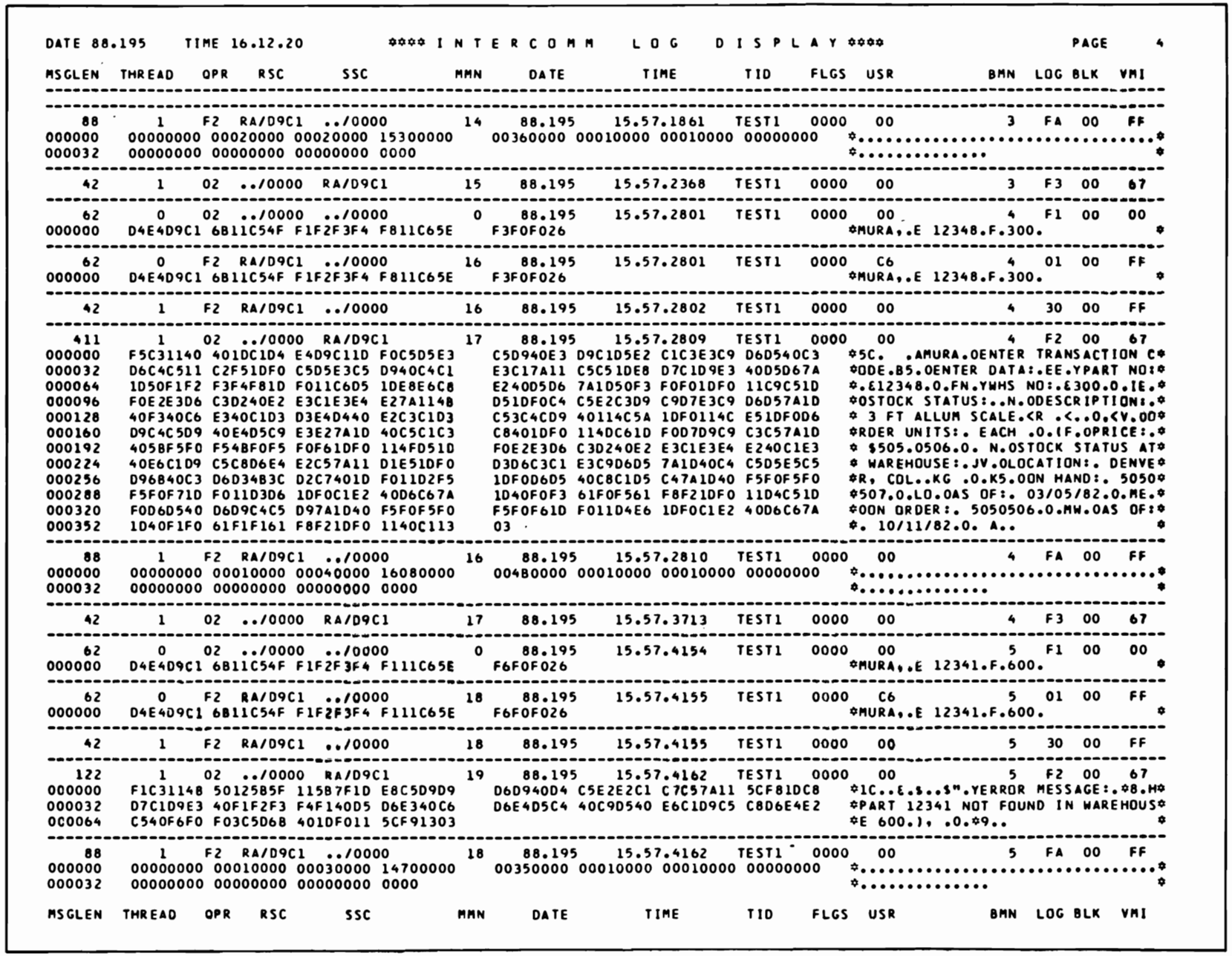
Figure 55. Simulation Mode Execution Log Printout (Page 1 of 8)

```
DATE 88.195    TIME 16.12.20        **** I N T E R C O M M   L O G   D I S P L A Y ****                    PAGE     2

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------------------
    42      3    F2  .N/00D5  ../0000          7    88.195    15.56.4551   GFE03   0000   00            0   30  00   FF
------------------------------------------------------------------------------------------------------------------------
    88      3    F2  .N/00D5  ../0000          7    88.195    15.56.4551   GFE03   0000   00            0   FA  00   FF
000000   00000000 00000000 00000000 04800000     00040000 00000000 00000000 00000000   *.....................................*
000032   00000000 00000000 00000000 0000                                               *................                     *
------------------------------------------------------------------------------------------------------------------------
    88      1    02  .U/00E4  ../0000          2    88.195    15.56.4708   CNT01   0000   00            0   FA  00   50
000000   00000000 00000000 00010000 12600000     00170000 00010000 00010000 00000000   *.............-.....................*
000032   00000000 00000000 00000000 0000                                               *................                     *
------------------------------------------------------------------------------------------------------------------------
   108      0    02  .U/00E4  ../0000          8    88.195    15.56.4708   TBROD   0000   00            0   01  00   50
000000   FF02002D 013C5C5C 5C40C7D6 D6C440C1     C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *.......*** GOOD AFTERNOON**  INTE*
000032   D9C3D6D4 D440C9E2 40D9C5C1 C4E8407A     404040F0 F760F1F3 60F8F840 40F1F54B   *RCOMM IS READY :   07-13-88  15.*
000064   F5F6                                                                          *56                                  *
------------------------------------------------------------------------------------------------------------------------
    42      1    02  .U/00E4  ../0000          8    88.195    15.56.4709   TBROD   0000   00            0   30  00   50
------------------------------------------------------------------------------------------------------------------------
   103      1    02  ../0000  .U/00E4          9    88.195    15.56.4709   TBROD   0000   00            0   F2  00   50
000000   5C5C5C40 C7D6D6C4 40C1C6E3 C5D9D5D6     D6D55C5C 4040C9D5 E3C5D9C3 D6D4D440   **** GOOD AFTERNOON**  INTERCOMM *
000032   C9E240D9 C5C1C4E8 407A4040 40F0F760     F1F360F8 F84040F1 F54BF5F6 37         *IS READY :   07-13-88  15.56.   *
------------------------------------------------------------------------------------------------------------------------
    88      1    02  .U/00E4  ../0000          8    88.195    15.56.4710   TBROD   0000   00            0   FA  00   50
000000   00000000 00000000 00020000 12600000     00160000 00010000 00010000 00000000   *.............-.....................*
000032   00000000 00000000 00000000 0000                                               *................                     *
------------------------------------------------------------------------------------------------------------------------
    65      2    02  ../0000  ../0000          0    88.195    15.56.4823   .....   0000   00            0   00  00   00
000000   15564799 00000006 00000009 00000000     00000000 000080                       *.........................            *
------------------------------------------------------------------------------------------------------------------------
    88      2    02  .0/00D8  ../0000          4    88.195    15.56.4835   CNT01   0000   00            0   FA  00   01
000000   00000000 00050000 00000000 03400000     00290000 00000000 00000000 00000000   *.............. ....................*
000032   00000000 00000000 00000000 0019                                               *................                     *
------------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  .U/00E4          3    88.195    15.56.5109   CNT01   0000   00            0   F3  00   50
------------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  .U/00E4          9    88.195    15.56.5109   TBROD   0000   00            0   F3  00   50
------------------------------------------------------------------------------------------------------------------------
    67      0    02  ../0000  ../0000          0    88.195    15.56.5212   TEST1   0000   00            1   F1  00   00
000000   D4D4E4C3 6BE2C8D6 E66B4DE2 E3D2E2E3     C1E36BD4 C1D7F15D 26                  *MMUC,SHOW,(STKSTAT,MAP1).           *
------------------------------------------------------------------------------------------------------------------------
    67      0    F2  MM/D4D4  ../0000         10    88.195    15.56.5212   TEST1   0000   00            1   01  00   FF
000000   D4D4E4C3 6BE2C8D6 E66B4DE2 E3D2E2E3     C1E36BD4 C1D7F15D 26                  *MMUC,SHOW,(STKSTAT,MAP1).           *
------------------------------------------------------------------------------------------------------------------------
    42      1    F2  MM/D4D4  ../0000         10    88.195    15.56.5213   TEST1   0000   00            1   30  00   FF
------------------------------------------------------------------------------------------------------------------------
   350      1    02  ../0000  MM/D4D4         11    88.195    15.56.5249   TEST1   0000   00            1   F2  00   67
000000   F5C31140 401DC140 4040401D F0C5D5E3     C5D940E3 D9C1D5E2 C1C3E3C9 D6D540C3   *5C.  .A    .OENTER TRANSACTION C*
000032   D6C4C511 C2F51DF0 C5D5E3C5 D940C4C1     E3C17A11 C5C51DE8 D7C1D9E3 40D5D67A   *ODE.B5.OENTER DATA:.EE.YPART NO:*
000064   1D5011C5 D41DF011 C6D51DE8 E6C8E240     D5D67A1D 5011C661 1DF011C9 C51DF0E2   *.&.EM.O.FN.YWHS NO:.&.F/.O.IE.OS*
000096   E3D6C3D2 40E2E3C1 E3E4E27A 114BD51D     F0C4C5E2 C3D9C9D7 E3C9D6D5 7A1D4011   *TOCK STATUS:..N.ODESCRIPTION:. .*
000128   4C5A1DF0 114CE51D F0D6D9C4 C5D940E4     D5C9E3E2 7A1D4011 4CF81DF0 114DC61D   *<..O.<V.OORDER UNITS:. .<8.O.(F.*
000160   F0D7D9C9 C3C57A1D 40114DD7 1DF0114F     D51DF0E2 E3D6C3D2 40E2E3C1 E3E4E240   *OPRICE:. .(P.O. N.CSTOCK STATUS *

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
```

Figure 55. Simulation Mode Execution Log Printout (Page 2 of 8)

```
DATE 88.195    TIME 16.12.20        **** I N T E R C O M M   L O G   D I S P L A Y ****                    PAGE     3

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME       TID    FLGS  USR          BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------------
000192   C1E340E6 C1D9C5C8 D6E4E2C5 7A11D1E5    1DF0D3D6 C3C1E3C9 D6D57A1D 4011D2C7   *AT WAREHOUSE:.JV.OLOCATION:. .KG*
000224   1DF011D2 F51DF0D6 D540C8C1 D5C47A1D    4011D3C6 1DF011D3 D61DF0C1 E240D6C6   *.0.K5.OON HAND:. .LF.0.LO.OAS OF*
000256   7A1D4011 D3E61DF0 11D4C51D F0D6D540    D6D9C4C5 D97A1D40 11D4D71D F011D4E6   *:. .LW.0.ME.OON ORDER:. .MP.0.MW*
000288   1DF0C1E2 40D6C67A 1D4011D4 F61DF011    40C11303                              *.OAS OF:. .M6.0. A..          *
------------------------------------------------------------------------------------------------------------------
    88       1    F2  MM/D4D4  ../0000        10    88.195     15.56.5249   TEST1   0000   00          1   FA  00   FF
000000   00000000 00020000 00070000 22F00000    002E0000 00010000 00010000 00010000   *.............0.................*
000032   00000000 00000000 00000000 0000                                              *.............                  *
------------------------------------------------------------------------------------------------------------------
    42       1    02  ../0000  MM/D4D4        11    88.195     15.57.0053   TEST1   0000   00          1   F3  00   67
------------------------------------------------------------------------------------------------------------------
    62       0    02  ../0000  ../0000         0    88.195     15.57.0488   TEST1   0000   00          2   F1  00   00
000000   D4E4D9C1 6B11C54F F1F2F3F4 F511C65E    F2F0F026                              *MURA,.E 12345.F.200.          *
------------------------------------------------------------------------------------------------------------------
    62       0    F2  RA/D9C1  ../0000        12    88.195     15.57.0488   TEST1   0000   C6          2   01  00   FF
000000   D4E4D9C1 6B11C54F F1F2F3F4 F511C65E    F2F0F026                              *MURA,.E 12345.F.200.          *
------------------------------------------------------------------------------------------------------------------
    42       1    F2  RA/D9C1  ../0000        12    88.195     15.57.0488   TEST1   0000   00          2   30  00   FF
------------------------------------------------------------------------------------------------------------------
   413       1    02  ../0000  RA/D9C1        13    88.195     15.57.0562   TEST1   0000   00          2   F2  00   67
000000   F5C31140 401DC1D4 E4D9C11D F0C5D5E3    C5D940E3 D9C1D5E2 C1C3E3C9 D6D540C3   *5C.  .AMURA.OENTER TRANSACTION C*
000032   D6C4C511 C2F51DF0 C5D5E3C5 D940C4C1    E3C17A11 C5C51DE8 D7C1D9E3 40D5D67A   *ODE.B5.OENTER DATA:.EE.YPART NO:*
000064   1D50F1F2 F3F4F51D F011C6D5 1DE8E6C8    E240D5D6 7A1D50F2 F0F01DF0 11C9C51D   *.&12345.0.FN.YWHS NO:.&200.0.IE.*
000096   F0E2E3D6 C3D240E2 E3C1E3E4 E27A114B    D51DF0C4 C5E2C3D9 C9D7E3C9 D6D57A1D   *OSTOCK STATUS:..N.ODESCRIPTION:.*
000128   40F161F2 40C9D540 E2E3C5C5 D340E6C1    E2C8C5D9 3C4CD940 114C5A1D F0114CE5   * 1/2 IN STEEL WASHER.<R .<...0.<V*
000160   1DF0D6D9 C4C5D940 E4D5C9E3 E27A1D40    C7D9E240 401DF011 4DC61DF0 D7D9C9C3   *.ORDER UNITS:. GRS  .0.(F.OPRIC*
000192   C57A1D40 5BF5F0F5 4BF0F5F0 F71DF011    4FD51DF0 E2E3D6C3 D240E2E3 C1E3E4E2   *E:. $505.0507.0. N.OSTOCK STATUS*
000224   40C1E340 E6C1D9C5 C8D6E4E2 C57A11D1    E51DF0D3 D6C3C1E3 C9D6D57A 1D40D4C9   * AT WAREHOUSE:.JV.OLOCATION:. MI*
000256   C1D4C96B 40C6D3C1 4B3CD2C7 401DF011    D2F51DF0 D6D540C8 C1D5C47A 1D40F6F1   *AMI, FLA..KG .0.K5.OON HAND:. 61*
000288   F6F1F5F0 F61DF011 D3D61DF0 C1E240D6    C67A1D40 F0F361F0 F561F8F2 1DF011D4   *61506.0.LO.0AS OF:. 03/05/82.0.M*
000320   C51DF0D6 D540D6D9 C4C5D97A 1D40F4F0    F4F0F6F1 F71DF011 D4E61DF0 C1E240D6   *E.OON ORDER:. 4040617.0.MW.0AS O*
000352   C67A1D40 F1F061F1 F161F8F2 1DF01140    C11303                                *F:. 10/11/82.0. A..           *
------------------------------------------------------------------------------------------------------------------
    88       1    F2  RA/D9C1  ../0000        12    88.195     15.57.0563   TEST1   0000   00          2   FA  00   FF
000000   00000000 00010000 00090000 16900000    00510000 00010000 00020000 00010000   *................................*
000032   00000000 00000000 00000000 0000                                              *.............                  *
------------------------------------------------------------------------------------------------------------------
    42       1    02  ../0000  RA/D9C1        13    88.195     15.57.1416   TEST1   0000   00          2   F3  00   67
------------------------------------------------------------------------------------------------------------------
    62       0    02  ../0000  ../0000         0    88.195     15.57.1849   TEST1   0000   00          3   F1  00   00
000000   D4E4D9C1 6B11C54F F5F5F5F5 F511C65E    F2F0F026                              *MURA,.E 55555.F.200.          *
------------------------------------------------------------------------------------------------------------------
    62       0    F2  RA/D9C1  ../0000        14    88.195     15.57.1850   TEST1   0000   C6          3   01  00   FF
000000   D4E4D9C1 6B11C54F F5F5F5F5 F511C65E    F2F0F026                              *MURA,.E 55555.F.200.          *
------------------------------------------------------------------------------------------------------------------
    42       1    F2  RA/D9C1  ../0000        14    88.195     15.57.1850   TEST1   0000   00          3   30  00   FF
------------------------------------------------------------------------------------------------------------------
   112       1    02  ../0000  RA/D9C1        15    88.195     15.57.1861   TEST1   0000   00          3   F2  00   67
000000   F1C3114B 5012585F 115B7F1D E8C5D9D9    D6D940D4 C5E2E2C1 C7C57A11 5CF81DC8   *1C..&.$..$".YERROR MESSAGE:.*8.M*
000032   D7C1D9E3 4CD5E4D4 C2C5D940 3C5D4AF5    40D5D6E3 40C6D6E4 D5C44B3C 5D6B401D   *PART NUMBER .).5 NOT FOUND..), .*
000064   F0115CF9 1303                                                                *0.*9..                        *

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME       TID    FLGS  USR          BMN  LOG BLK  VMI
```

Figure 55. Simulation Mode Execution Log Printout (Page 3 of 8)



```
DATE 88.195    TIME 16.12.20        **** I N T E R C O M M    L O G    D I S P L A Y ****                    PAGE     4

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID    FLGS  USR          BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------------------

   88      1     F2  RA/D9C1  ../0000         14    88.195     15.57.1861   TEST1   0000   00            3   FA  00   FF
000000   00000000 00020000 00020000 15300000     00360000 00010000 00010000 00000000   *.....................................*
000032   00000000 00000000 00000000 0000                                               *................                    *
------------------------------------------------------------------------------------------------------------------------

   42      1     02  ../0000  RA/D9C1         15    88.195     15.57.2368   TEST1   0000   00            3   F3  00   67
------------------------------------------------------------------------------------------------------------------------

   62      0     02  ../0000  ../0000          0    88.195     15.57.2801   TEST1   0000   00            4   F1  00   00
000000   D4E4D9C1 6B11C54F F1F2F3F4 F811C65E     F3F0F026                              *MURA,,E 12348.F.300.                *
------------------------------------------------------------------------------------------------------------------------

   62      0     F2  RA/D9C1  ../0000         16    88.195     15.57.2801   TEST1   0000   C6            4   01  00   FF
000000   D4E4D9C1 6B11C54F F1F2F3F4 F811C65E     F3F0F026                              *MURA,,E 12348.F.300.                *
------------------------------------------------------------------------------------------------------------------------

   42      1     F2  RA/D9C1  ../0000         16    88.195     15.57.2802   TEST1   0000   00            4   30  00   FF
------------------------------------------------------------------------------------------------------------------------

  411      1     02  ../0000  RA/D9C1         17    88.195     15.57.2809   TEST1   0000   00            4   F2  00   67
000000   F5C31140 401DC1D4 E4D9C11D F0C5D5E3     C5D940E3 D9C1D5E2 C1C3E3C9 D6D540C3   *5C.  .AMURA.0ENTER TRANSACTION C*
000032   D6C4C511 C2F51DF0 C5D5E3C5 D940C4C1     E3C17A11 C5C51DE8 D7C1D9E3 40D5D67A   *ODE.B5.0ENTER DATA:.EE.YPART NO:*
000064   1D50F1F2 F3F4F81D F011C6D5 1DE8E6C8     E240D5D6 7A1D50F3 F0F01DF0 11C9C51D   *.&12348.0.FN.YWHS NO:.&300.0.IE.*
000096   F0E2E3D6 C3D240E2 E3C1E3E4 E27A114B     D51DF0C4 C5E2C3D9 C9D7E3C9 D6D57A1D   *0STOCK STATUS:..N.0DESCRIPTION:.*
000128   40F340C6 E340C1D3 D3E4D440 E2C3C1D3     C53C4CD9 40114C5A 1DF0114C E51DF0D6   * 3 FT ALLUM SCALE.<R .<..0.<V.00*
000160   D9C4C5D9 40E4D5C9 E3E27A1D 40C5C1C3     C8401DF0 114DC61D F0D7D9C9 C3C57A1D   *RDER UNITS:. EACH .0.(F.0PRICE:.*
000192   405BF5F0 F54BF0F5 F0F61DF0 114FD51D     F0E2E3D6 C3D240E2 E3C1E3E4 E240C1E3   * $505.0506.0. N.OSTOCK STATUS AT*
000224   40E6C1D9 C5C8D6E4 E2C57A11 D1E51DF0     D3D6C3C1 E3C9D6D5 7A1D40C4 C5D5E5C5   * WAREHOUSE:.JV.0LOCATION:. DENVE*
000256   D96840C3 D6D34B3C D2C7401D F011D2F5     1DF0D6D5 40C8C1D5 C47A1D40 F5F0F5F0   *R, COL..KG .0.K5.00N HAND:. 5050*
000288   F5F0F71D F011D3D6 1DF0C1E2 40D6C67A     1D40F0F3 61F0F561 F8F21DF0 11D4C51D   *507.0.LO.0AS OF:. 03/05/82.0.ME.*
000320   F0D6D540 D6D9C4C5 D97A1D40 F5F0F5F0     F5F0F61D F011D4E6 1DF0C1E2 40D6C67A   *00N ORDER:. 5050506.0.MW.0AS OF:*
000352   1D40F1F0 61F1F161 F8F21DF0 1140C113     03                                    *. 10/11/82.0. A..                   *
------------------------------------------------------------------------------------------------------------------------

   88      1     F2  RA/D9C1  ../0000         16    88.195     15.57.2810   TEST1   0000   00            4   FA  00   FF
000000   00000000 00010000 00040000 16080000     004B0000 00010000 00010000 00000000   *.....................................*
000032   00000000 00000000 00000000 0000                                               *................                    *
------------------------------------------------------------------------------------------------------------------------

   42      1     02  ../0000  RA/D9C1         17    88.195     15.57.3713   TEST1   0000   00            4   F3  00   67
------------------------------------------------------------------------------------------------------------------------

   62      0     02  ../0000  ../0000          0    88.195     15.57.4154   TEST1   0000   00            5   F1  00   00
000000   D4E4D9C1 6B11C54F F1F2F3F4 F111C65E     F6F0F026                              *MURA,,E 12341.F.600.                *
------------------------------------------------------------------------------------------------------------------------

   62      0     F2  RA/D9C1  ../0000         18    88.195     15.57.4155   TEST1   0000   C6            5   01  00   FF
000000   D4E4D9C1 6B11C54F F1F2F3F4 F111C65E     F6F0F026                              *MURA,,E 12341.F.600.                *
------------------------------------------------------------------------------------------------------------------------

   42      1     F2  RA/D9C1  ../0000         18    88.195     15.57.4155   TEST1   0000   00            5   30  00   FF
------------------------------------------------------------------------------------------------------------------------

  122      1     02  ../0000  RA/D9C1         19    88.195     15.57.4162   TEST1   0000   00            5   F2  00   67
000000   F1C31148 50125B5F 115B7F1D E8C5D9D9     D6D940D4 C5E2E2C1 C7C57A11 5CF81DC8   *1C..&.$..$".YERROR MESSAGE:.*8.H*
000032   D7C1D9E3 40F1F2F3 F4F140D5 D6E340C6     D6E4D5C4 40C9D540 E6C1D9C5 C8D6E4E2   *PART 12341 NOT FOUND IN WAREHOUS*
000064   C540F6F0 F03C5D6B 401DF011 5CF91303                                           *E 600.), .0.*9..                    *
------------------------------------------------------------------------------------------------------------------------

   88      1     F2  RA/D9C1  ../0000         18    88.195     15.57.4162   TEST1   0000   00            5   FA  00   FF
000000   00000000 00010000 00030000 14700000     00350000 00010000 00010000 00000000   *.....................................*
000032   00000000 00000000 00000000 0000                                               *................                    *

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID    FLGS  USR          BMN  LOG BLK  VMI
```

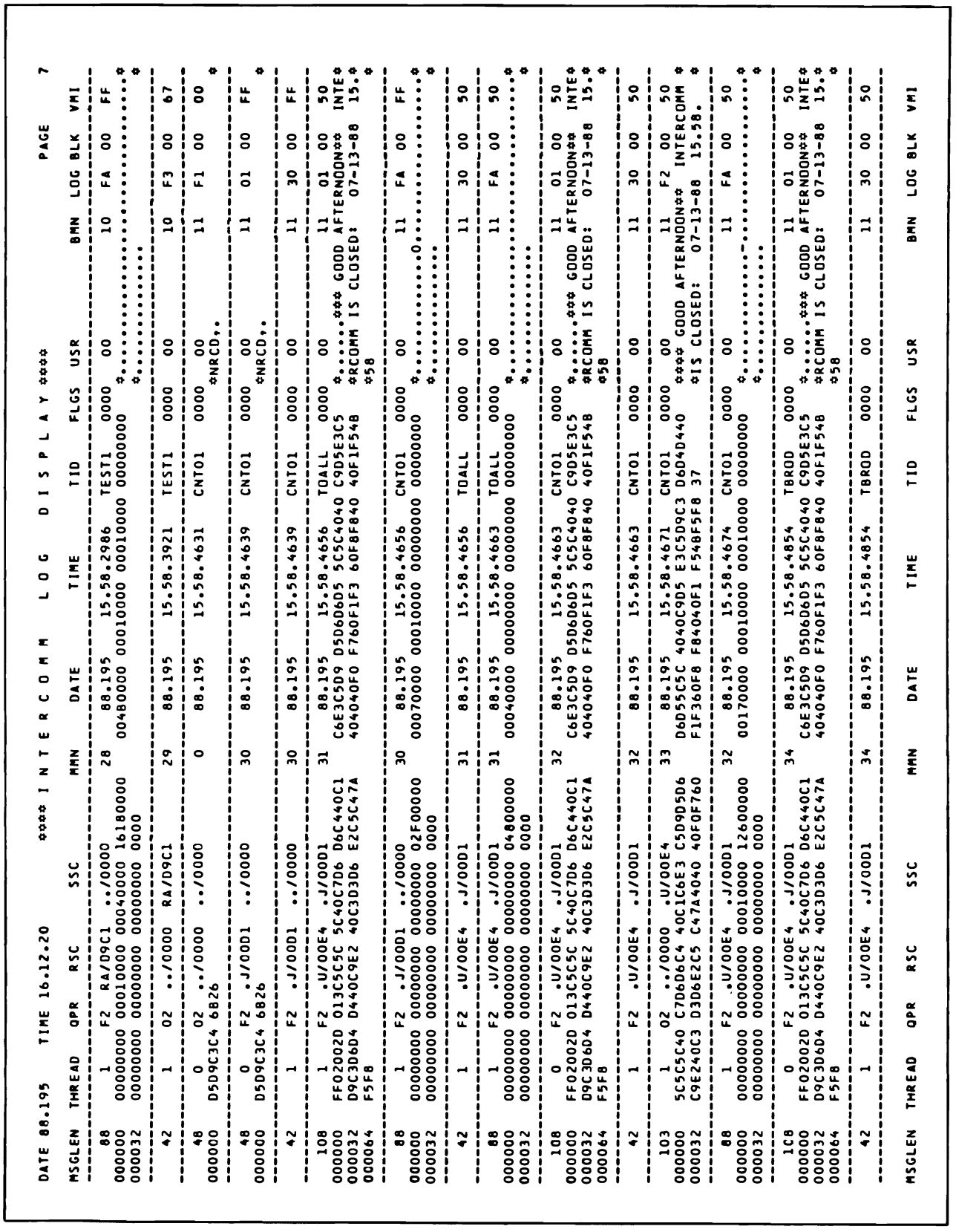
Figure 55. Simulation Mode Execution Log Printout (Page 4 of 8)

```
DATE 88.195    TIME 16.12.20        **** I N T E R C O M M   L O G   D I S P L A Y ****                    PAGE     5

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID    FLGS  USR          BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  RA/D9C1        19    88.195     15.57.4684    TEST1  0000   00            5   F3  00   67
------------------------------------------------------------------------------------------------------------------------
    62      0    02  ../0000  ../0000         0    88.195     15.57.5125    TEST1  0000   00            6   F1  00   00
000000   D4E4D9C1 6B11C54F C1F2F3F4 F511C65E    F2F0F026                                 *MURA,.E A2345.F.200.          *
------------------------------------------------------------------------------------------------------------------------
    62      0    F2  RA/D9C1  ../0000         20    88.195     15.57.5126    TEST1  0000   C6            6   01  00   FF
000000   D4E4D9C1 6B11C54F C1F2F3F4 F511C65E    F2F0F026                                 *MURA,.E A2345.F.200.          *
------------------------------------------------------------------------------------------------------------------------
    42      1    F2  RA/D9C1  ../0000         20    88.195     15.57.5126    TEST1  0000   00            6   30  00   FF
------------------------------------------------------------------------------------------------------------------------
   121      1    02  ../0000  RA/D9C1        21    88.195     15.57.5127    TEST1  0000   00            6   F2  00   67
000000   F1C3114B 50125B5F 115B7F1D E8C5D9D9    D6D940D4 C5E2E2C1 C7C57A11 5CF81DC8   *1C..&.$..$".YERROR MESSAGE:.*8.H*
000032   C9D5E5C1 D3C9C440 C4C1E3C1 7A40D7C1    D9E3D5D6 40D4E4E2 E340C2C5 40D5E4D4   *INVALID DATA: PARTNO MUST BE NUM*
000064   C5D9C9C3 3C5D6B40 1DF0115C F91303                                            *ERIC.), .0.*9..               *
------------------------------------------------------------------------------------------------------------------------
    88      1    F2  RA/D9C1  ../0000         20    88.195     15.57.5128    TEST1  0000   00            6   FA  00   FF
000000   00000000 00000000 00020000 14700000    00280000 00010000 00010000 00000000   *.....................................*
000032   00000000 00000000 00000000 0000                                              *................               *
------------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  RA/D9C1        21    88.195     15.57.5653    TEST1  0000   00            6   F3  00   67
------------------------------------------------------------------------------------------------------------------------
    62      0    02  ../0000  ../0000         0    88.195     15.58.0088    TEST1  0000   00            7   F1  00   00
000000   D4E4D9C1 6B11C54F F1F2F3F4 F511C65E    C2F0F026                                 *MURA,.E 12345.F.B00.          *
------------------------------------------------------------------------------------------------------------------------
    62      0    F2  RA/D9C1  ../0000         22    88.195     15.58.0088    TEST1  0000   C6            7   01  00   FF
000000   D4E4D9C1 6B11C54F F1F2F3F4 F511C65E    C2F0F026                                 *MURA,.E 12345.F.B00.          *
------------------------------------------------------------------------------------------------------------------------
    42      1    F2  RA/D9C1  ../0000         22    88.195     15.58.0088    TEST1  0000   00            7   30  00   FF
------------------------------------------------------------------------------------------------------------------------
   120      1    02  ../0000  RA/D9C1        23    88.195     15.58.0090    TEST1  0000   00            7   F2  00   67
000000   F1C3114B 50125B5F 115B7F1D E8C5D9D9    D6D940D4 C5E2E2C1 C7C57A11 5CF81DC8   *1C..&.$..$".YERROR MESSAGE:.*8.H*
000032   C9D5E5C1 D3C9C440 C4C1E3C1 7A40E6C8    E2D5D640 D4E4E2E3 40C2C540 D5E4D4C5   *INVALID DATA: WHSNO MUST BE NUME*
000064   D9C9C33C 5D6B401D F0115CF9 1303                                              *RIC.), .0.*9..                *
------------------------------------------------------------------------------------------------------------------------
    88      1    F2  RA/D9C1  ../0000         22    88.195     15.58.0090    TEST1  0000   00            7   FA  00   FF
000000   00000000 00000000 00020000 14680000    00280000 00010000 00010000 00000000   *.....................................*
000032   00000000 00000000 00000000 0000                                              *................               *
------------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  RA/D9C1        23    88.195     15.58.0615    TEST1  0000   00            7   F3  00   67
------------------------------------------------------------------------------------------------------------------------
    62      0    02  ../0000  ../0000         0    88.195     15.58.1046    TEST1  0000   00            8   F1  00   00
000000   D4E4D9C1 6B11C54F F1F2F3F4 E711C65E    F2F0E826                                 *MURA,.E 1234X.F.20Y.          *
------------------------------------------------------------------------------------------------------------------------
    62      0    F2  RA/D9C1  ../0000         24    88.195     15.58.1046    TEST1  0000   C6            8   01  00   FF
000000   D4E4D9C1 6B11C54F F1F2F3F4 E711C65E    F2F0E826                                 *MURA,.E 1234X.F.20Y.          *
------------------------------------------------------------------------------------------------------------------------
    42      1    F2  RA/D9C1  ../0000         24    88.195     15.58.1047    TEST1  0000   00            8   30  00   FF
------------------------------------------------------------------------------------------------------------------------
   131      1    02  ../0000  RA/D9C1        25    88.195     15.58.1048    TEST1  0000   00            8   F2  00   67
000000   F1C3114B 50125B5F 115B7F1D E8C5D9D9    D6D940D4 C5E2E2C1 C7C57A11 5CF81DC8   *1C..&.$..$".YERROR MESSAGE:.*8.H*

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID    FLGS  USR          BMN  LOG BLK  VMI
```

Figure 55. Simulation Mode Execution Log Printout (Page 5 of 8)

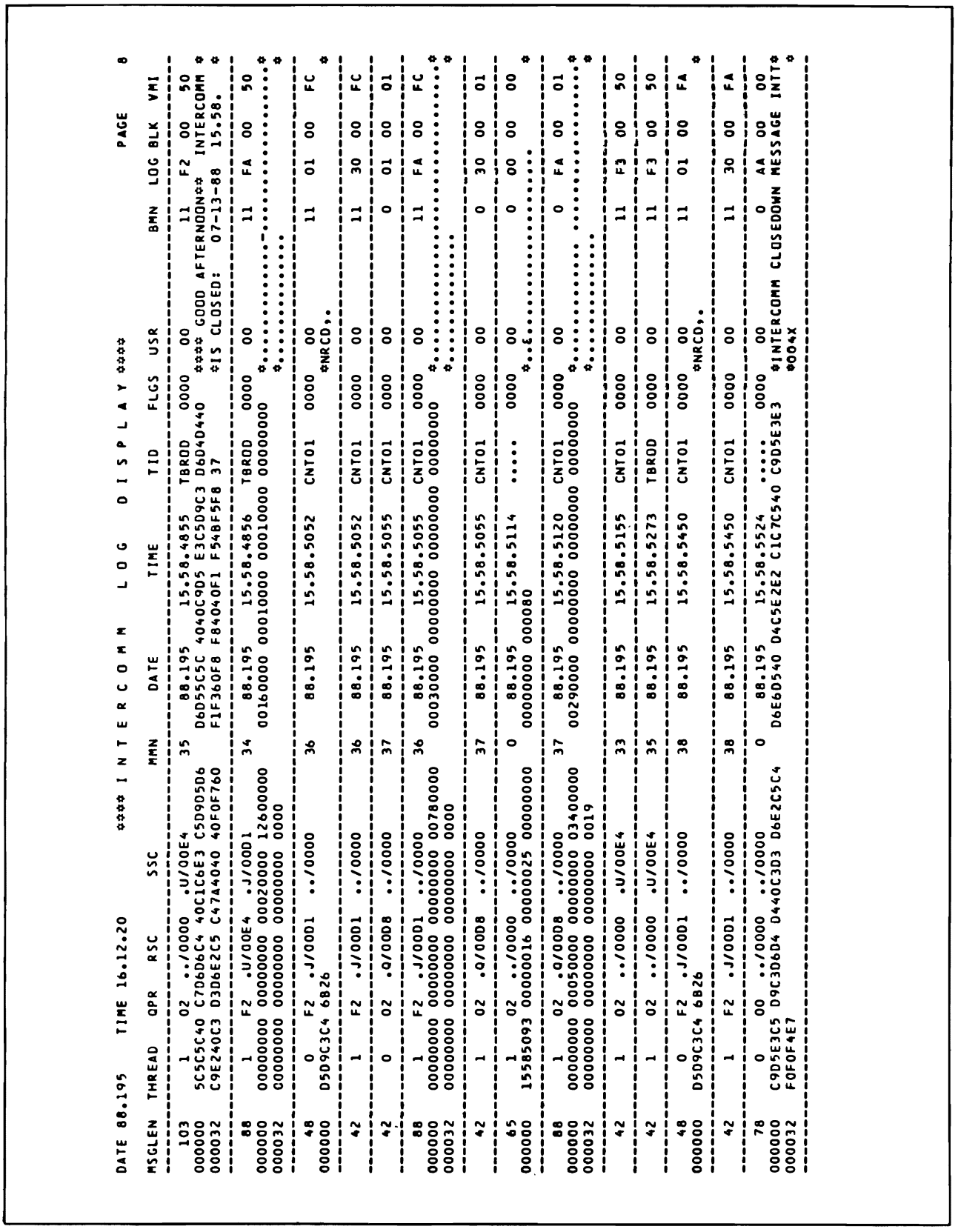
```
DATE 88.195    TIME 16.12.20        **** I N T E R C O M M    L O G    D I S P L A Y ****                    PAGE     6

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
-------------------------------------------------------------------------------------------------------------------
000032  C9D5E5C1 D3C9C440 C4C1E3C1 7A40D7C1    D9E3D5D6 40C1D5C4 40E6C8E2 D5D640D4   *INVALID DATA: PARTNO AND WHSNO M*
000064  .E4E2E340 C2C540D5 E4D4C5D9 C9C34040   40401DF0 115CF913 03                   *UST BE NUMERIC    .0.*9..       *
-------------------------------------------------------------------------------------------------------------------
    88      1    F2  RA/D9C1  ../0000        24    88.195    15.58.1048   TEST1   0000   00           8   FA  00   FF
000000  00000000 00000000 00020000 14780000    00280000 00010000 00010000 00000000   *.............................*
000032  00000000 00000000 00000000 0000                                              *.............               *
-------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  RA/D9C1        25    88.195    15.58.1589   TEST1   0000   00           8   F3  00   67
-------------------------------------------------------------------------------------------------------------------
    62      0    02  ../0000  ../0000         0    88.195    15.58.2026   TEST1   0000   00           9   F1  00   00
000000  D4E4D9C1 6B11C54F F1F2F3F4 F911C65E    F1F0F026                              *MURA,.E 12349.F.100.          *
-------------------------------------------------------------------------------------------------------------------
    62      0    F2  RA/D9C1  ../0000        26    88.195    15.58.2026   TEST1   0000   C6           9   01  00   FF
000000  D4E4D9C1 6B11C54F F1F2F3F4 F911C65E    F1F0F026                              *MURA,.E 12349.F.100.          *
-------------------------------------------------------------------------------------------------------------------
    42      1    F2  RA/D9C1  ../0000        26    88.195    15.58.2026   TEST1   0000   00           9   30  00   FF
-------------------------------------------------------------------------------------------------------------------
   113      1    02  ../0000  RA/D9C1        27    88.195    15.58.2040   TEST1   0000   00           9   F2  00   67
000000  F1C31148 50125B5F 115B7F1D E8C5D9D9    D6D940D4 C5E2E2C1 C7C57A11 5CF81DC8   *1C..&.$..$".YERROR MESSAGE:.*8.H*
000032  D7C1D9E3 40D5E4D4 C2C5D940 F1F2F3F4    F940D5D6 E340C6D6 E4D5C44B 3C5D6B40   *PART NUMBER 12349 NOT FOUND..), *
000064  1DF0115C F91303                                                             *.0.*9..                       *
-------------------------------------------------------------------------------------------------------------------
    88      1    F2  RA/D9C1  ../0000        26    88.195    15.58.2040   TEST1   0000   00           9   FA  00   FF
000000  00000000 00010000 00020000 14680000    00310000 00010000 00010000 00000000   *.............................*
000032  00000000 00000000 00000000 0000                                              *.............               *
-------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  RA/D9C1        27    88.195    15.58.2543   TEST1   0000   00           9   F3  00   67
-------------------------------------------------------------------------------------------------------------------
    62      0    02  ../0000  ../0000         0    88.195    15.58.2977   TEST1   0000   00          10   F1  00   00
000000  D4E4D9C1 6B11C54F F1F2F3F4 F211C65E    F1F0F026                              *MURA,.E 12342.F.100.          *
-------------------------------------------------------------------------------------------------------------------
    62      0    F2  RA/D9C1  ../0000        28    88.195    15.58.2977   TEST1   0000   C6          10   01  00   FF
000000  D4E4D9C1 6B11C54F F1F2F3F4 F211C65E    F1F0F026                              *MURA,.E 12342.F.100.          *
-------------------------------------------------------------------------------------------------------------------
    42      1    F2  RA/D9C1  ../0000        28    88.195    15.58.2977   TEST1   0000   00          10   30  00   FF
-------------------------------------------------------------------------------------------------------------------
   430      1    02  ../0000  RA/D9C1        29    88.195    15.58.2986   TEST1   0000   00          10   F2  00   67
000000  F5C31140 401DC1D4 E4D9C11D F0C5D5E3    C5D940E3 D9C1D5E2 C1C3E3C9 D6D540C3   *5C.  .AMURA.0ENTER TRANSACTION C*
000032  D6C4C511 C2F51DF0 C5D5E3C5 D940C4C1    E3C17A11 C5C51DE8 D7C1D9E3 40D5D67A   *0DE.B5.0ENTER DATA:.EE.YPART NO:*
000064  1D50F1F2 F3F4F21D F011C6D5 1DE8E6C8    E240D5D6 7A1D50F1 F0F01DF0 11C9C51D   *.&12342.0.FN.YWHS NO:.&100.0.IE.*
000096  F0E2E3D6 C3D240E2 E3C1E3E4 E27A114B    D51DF0C4 C5E2C3D9 C9D7E3C9 D6D57A1D   *0STOCK STATUS:..N.0DESCRIPTION:.*
000128  40F361F8 40C9D540 C6D3C1E3 40C8C5C1    C440C2D3 C1C3D240 C3C1D9D9 C9C1C7C5   * 3/8 IN FLAT HEAD BLACK CARRIAGE*
000160  40C2D6D3 E33C4CD9 40114C5A 1DF0114C    E51DF0D6 D9C4C5D9 40E4D5C9 E3E27A1D   * BOLT.<R .<..0.<V.0ORDER UNITS:.*
000192  40C4D6E9 40401DF0 114DC61D F0D7D9C9    C3C57A1D 405BF5F0 F64BF1F6 F1F61DF0   * DOZ  .0.(F.0PRICE:. $506.1616.0*
000224  114FD51D F0E2E3D6 C3D240E2 E3C1E3E4    E240C1E3 40E6C1D9 C5C8D6E4 E2C57A11   *. N.0STOCK STATUS AT WAREHOUSE:.*
000256  D1E51DF0 D3D6C3C1 E3C9D6D5 7A1D40D4    C9C1D4C9 6B40C6D3 C14B3CD2 C7401DF0   *JV.0LOCATION:. MIAMI, FLA..KG .0*
000288  11D2F51D F0D6D540 C8C1D5C4 7A1D40F6    F1F5F0F6 F1F41DF0 11D3D61D F0C1E240   *.K5.0ON HAND:. 6150614.0.LO.0AS *
000320  D6C67A1D 40F0F361 F0F561F8 F21DF011    D4C51DF0 D6D540D6 D9C4C5D9 7A1D40F5   *OF:. 03/05/82.0.ME.0ON ORDER:. 5*
000352  F0F4F0F4 F0F41DF0 11D4E61D F0C1E240    D6C67A1D 40F1F061 F1F161F8 F21DF011   *040404.0.MW.0AS OF:. 10/11/82.0.*
000384  40C11303                                                                    * A..                          *
-------------------------------------------------------------------------------------------------------------------

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
```

Figure 55. Simulation Mode Execution Log Printout (Page 6 of 8)



```
DATE 88.195   TIME 16.12.20        **** I N T E R C O M M   L O G   D I S P L A Y ****                    PAGE    7

MSGLEN  THREAD  QPR  RSC     SSC          MMN    DATE      TIME      TID     FLGS  USR        BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------
   88     1     F2  RA/D9C1  ../0000       28    88.195   15.58.2986  TEST1   0000  00         10   FA  00   FF
000000  00000000 00010000 00040000 16180000      00480000 00010000 00010000 00000000   *.....................*
000032  00000000 00000000 00000000 0000                                                  *.....................*
                                                                                        *                     *
------------------------------------------------------------------------------------------------------------
   42     1     02  ../0000  RA/D9C1       29    88.195   15.58.3921  TEST1   0000  00         10   F3  00   67
------------------------------------------------------------------------------------------------------------
   48     0     02  ../0000  ../0000        0    88.195   15.58.4631  CNT01   0000  00         11   F1  00   00   *
000000  D5D9C3C4 6826                                                                  *NRCD..                    *
------------------------------------------------------------------------------------------------------------
   48     0     F2  .J/00D1  ../0000        30    88.195   15.58.4639  CNT01   0000  00         11   01  00   FF   *
000000  D5D9C3C4 6826                                                                  *NRCD..                    *
------------------------------------------------------------------------------------------------------------
   42     1     F2  .J/00D1  ../0000        30    88.195   15.58.4639  CNT01   0000  00         11   30  00   FF
------------------------------------------------------------------------------------------------------------
  108     1     F2  .U/00E4  .J/00D1        31    88.195   15.58.4656  TOALL   0000  00         11   01  00   50
000000  FF02002D 013C5C5C 5C40C7D6 D6C440C1      C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *.....*** GOOD AFTERNOON** INTE*
000032  D9C3D6D4 D440C9E2 40C3D3D6 E2C5C47A      40404040 F760F1F3 60F8F840 40F1F548   *RCOMM IS CLOSED:  07-13-88  15.*
000064  F5F8                                                                          *58                        *
------------------------------------------------------------------------------------------------------------
   88     1     F2  .J/00D1  ../0000        30    88.195   15.58.4656  CNT01   0000  00         11   FA  00   FF
000000  00000000 00000000 00000000 02F00000      00070000 00010000 00000000 00000000   *.....................*
000032  00000000 00000000 00000000 0000                                                  *.....................*
------------------------------------------------------------------------------------------------------------
   42     1     F2  .U/00E4  .J/00D1        31    88.195   15.58.4656  TOALL   0000  00         11   30  00   50
------------------------------------------------------------------------------------------------------------
   88     1     F2  .U/00E4  .J/00D1        31    88.195   15.58.4663  TOALL   0000  00         11   FA  00   50
000000  00000000 00000000 00000000 04800000      00040000 00000000 00000000 00000000   *.....................*
000032  00000000 00000000 00000000 0000                                                  *.....................*
------------------------------------------------------------------------------------------------------------
  108     0     F2  .U/00E4  .J/00D1        32    88.195   15.58.4663  CNT01   0000  00         11   01  00   50
000000  FF02002D 013C5C5C 5C40C7D6 D6C440C1      C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *.....*** GOOD AFTERNOON** INTE*
000032  D9C3D6D4 D440C9E2 40C3D3D6 E2C5C47A      40404040 F760F1F3 60F8F840 40F1F548   *RCOMM IS CLOSED:  07-13-88  15.*
000064  F5F8                                                                          *58                        *
------------------------------------------------------------------------------------------------------------
   42     1     F2  .U/00E4  .J/00D1        32    88.195   15.58.4663  CNT01   0000  00         11   30  00   50
------------------------------------------------------------------------------------------------------------
  103     1     02  ../0000  .U/00E4        33    88.195   15.58.4671  CNT01   0000  00         11   F2  00   50
000000  5C5C5C40 C7D6D6C4 40C1C6E3 C5D9D5D6      D6D55C5C 4040C9D5 E3C5D9C3 D6D4D440   **** GOOD AFTERNOON** INTERCOMM *
000032  C9E240C3 D3D6E2C5 C47A4040 40F0F760      F1F360F8 F8404040 F1F548F5 F8 37      *IS CLOSED:  07-13-88  15.58.   *
------------------------------------------------------------------------------------------------------------
   88     1     F2  .U/00E4  .J/00D1        32    88.195   15.58.4674  CNT01   0000  00         11   FA  00   50
000000  00000000 00000000 00010000 12600000      00170000 00010000 00010000 00000000   *.....................*
000032  00000000 00000000 00000000 0000                                                  *.....................*
                                                                                        *                     *
------------------------------------------------------------------------------------------------------------
  1C8     0     F2  .U/00E4  .J/00D1        34    88.195   15.58.4854  TBROD   0000  00         11   01  00   50
000000  FF02002D 013C5C5C 5C40C7D6 D6C440C1      C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *.....*** GOOD AFTERNOON** INTE*
000032  D9C3D6D4 D440C9E2 40C3D3D6 E2C5C47A      40404040 F760F1F3 60F8F840 40F1F548   *RCOMM IS CLOSED:  07-13-88  15.*
000064  F5F8                                                                          *58                        *
------------------------------------------------------------------------------------------------------------
   42     1     F2  .U/00E4  .J/00D1        34    88.195   15.58.4854  TBROD   0000  00         11   30  00   50
------------------------------------------------------------------------------------------------------------

MSGLEN  THREAD  QPR  RSC     SSC          MMN    DATE      TIME      TID     FLGS  USR        BMN  LOG BLK  VMI
```

Figure 55. Simulation Mode Execution Log Printout (Page 7 of 8)



```
DATE 88.195    TIME 16.12.20        **** I N T E R C O M M   L O G   D I S P L A Y ****                    PAGE     8

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID    FLGS  USR          BMN  LOG BLK  VMI
-----------------------------------------------------------------------------------------------------------------------------
   103      1    02  ../0000  .U/00E4        35    88.195    15.58.4855   TBROD   0000   00           11   F2  00   50
000000   5C5C5C40 C7D6D6C4 40C1C6E3 C5D9D5D6    D6D55C5C 4040C9D5 E3C5D9C3 D6D4D440   **** GOOD AFTERNOON**  INTERCOMM *
000032   C9E240C3 D3D6E2C5 C47A4040 40F0F760    F1F360F8 F84040F1 F54BF5F8 37         *IS CLOSED:   07-13-88  15.58.   *
-----------------------------------------------------------------------------------------------------------------------------
    88      1    F2  .U/00E4  .J/00D1        34    88.195    15.58.4856   TBROD   0000   00           11   FA  00   50
000000   00000000 00000000 00020000 12600000    00160000 00010000 00010000 00000000   *.............-...................*
000032   00000000 00000000 00000000 0000                                               *..............                   *
-----------------------------------------------------------------------------------------------------------------------------
    48      0    F2  .J/00D1  ../0000        36    88.195    15.58.5052   CNT01   0000   00           11   01  00   FC
000000   D5D9C3C4 6B26                                                                *NRCD,.                          *
-----------------------------------------------------------------------------------------------------------------------------
    42      1    F2  .J/00D1  ../0000        36    88.195    15.58.5052   CNT01   0000   00           11   30  00   FC
-----------------------------------------------------------------------------------------------------------------------------
    42      0    02  .Q/00D8  ../0000        37    88.195    15.58.5055   CNT01   0000   00            0   01  00   01
-----------------------------------------------------------------------------------------------------------------------------
    88      1    F2  .J/00D1  ../0000        36    88.195    15.58.5055   CNT01   0000   00           11   FA  00   FC
000000   00000000 00000000 00000000 00780000    00030000 00000000 00000000 00000000   *...............................*
000032   00000000 00000000 00000000 0000                                               *..............                   *
-----------------------------------------------------------------------------------------------------------------------------
    42      1    02  .Q/00D8  ../0000        37    88.195    15.58.5055   CNT01   0000   00            0   30  00   01
-----------------------------------------------------------------------------------------------------------------------------
    65      1    02  ../0000  ../0000         0    88.195    15.58.5114   .....   0000   00            0   00  00   00
000000   15585093 00000016 00000025 00000000    00000000 000080                       *..£....................         *
-----------------------------------------------------------------------------------------------------------------------------
    88      1    02  .Q/00D8  ../0000        37    88.195    15.58.5120   CNT01   0000   00            0   FA  00   01
000000   00000000 00050000 00000000 03400000    00290000 00000000 00000000 00000000   *............. ...................*
000032   00000000 00000000 00000000 0019                                               *..............                   *
-----------------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  .U/00E4        33    88.195    15.58.5155   CNT01   0000   00           11   F3  00   50
-----------------------------------------------------------------------------------------------------------------------------
    42      1    02  ../0000  .U/00E4        35    88.195    15.58.5273   TBROD   0000   00           11   F3  00   50
-----------------------------------------------------------------------------------------------------------------------------
    48      0    F2  .J/00D1  ../0000        38    88.195    15.58.5450   CNT01   0000   00           11   01  00   FA
000000   D5D9C3C4 6B26                                                                *NRCD,.                          *
-----------------------------------------------------------------------------------------------------------------------------
    42      1    F2  .J/00D1  ../0000        38    88.195    15.58.5450   CNT01   0000   00           11   30  00   FA
-----------------------------------------------------------------------------------------------------------------------------
    78      0    00  ../0000  ../0000         0    88.195    15.58.5524   .....   0000   00            0   AA  00   00
000000   C9D5E3C5 D9C3D6D4 D440C3D3 D6E2C5C4    D6E6D540 D4C5E2E2 C1C7C540 C9D5E3E3   *INTERCOMM CLOSEDOWN MESSAGE INTT*
000032   F0F0F4E7                                                                     *004X                            *
-----------------------------------------------------------------------------------------------------------------------------
```

Figure 55. Simulation Mode Execution Log Printout (Page 8 of 8)

Chapter 13

SUBSYSTEM TESTING IN TEST MODE

13.1 INTRODUCTION

All of the testing functions may be performed using the Intercomm Test Mode of operation without a Front End defined. Rather than receiving messages from a terminal, the Test Monitor reads messages into the system from a card-image data set. Snaps of input (snap id=15) and output (snap id=20) messages constitute a history of Test Mode execution. Essentially, the Front End is replaced by the Test Monitor (PMITEST) to drive the Back End as usual. In this way, subsystem testing can be going on in one or more regions or address spaces without affecting the on-line system. Figure 56 illustrates a sample reentrant Assembler subsystem (SQASM) designed for the same purpose as SQASMA, but using the Edit, Output and Change/Display Utilities.

13.2 TESTING A SUBSYSTEM IN TEST MODE

To add and test an application subsystem in Test Mode, do the following:

NOTE: Steps preceded by an asterisk (*) may often be performed for the application programmer by an installation's Intercomm System Manager. Appendix C summarizes the Intercomm Table entries.

1. Compile and linkedit the application program. Appendix A describes Intercomm-supplied Assembler JCL procedures.

*2. Create or add to a USRSCTS member on a user test library to contain a Subsystem Control Table Entry (SYCTTBL macro) which describe the subsystem. Reassemble and link INTSCT which copies the USRSCTS member from the test library (see Figure 57).

*3. Create or add to a USRVERBS member on the user test library to contain an Edit Control Table (VERBTBL) entry for editing of input test messages by the Edit Utility. Reassemble and link PMIVERBS which copies the USRVERBS member from the test library (see Figure 57).

*4. If a Fixed Format output message (VMI=X'72') is created for processing by the Change/Display Utility, code an entry for the CHNGTB (see Figure 57) to define the DES000 data set entry number for the File Description Record (DES00001--see Figure 58). The PMIEXLD utility must be used to load the FDR to the DES000 file (see the Utilities Users Guide and the Operating Reference Manual).

5. Code, assemble and link and add an INCLUDE statement for the OFT load module RPTnnnnn (RPT00100 and RPT00501--see Figure 58) to the Output Format Table (PMIRCNTB) in the Test Mode Intercomm linkedit for output message formatting by the Output Utility.

6. Prepare test messages via the SIMCRTA utility or as direct card-image input data (SYSIN data set). An input test message consists of a header card, detail cards, and a trailer card, grouped together as illustrated in Figure 60. Figure 59 details the required card formats. The message area in the Test Monitor will accomodate a message text up to 958 bytes long. Longer messages would require a modification to the Test Monitor (PMITEST), as described in the Operating Reference Manual.

*7. Add control cards to the linkedit deck for the user program, unless the subsystem is dynamically loadable (see Figure 61).

*8. Linkedit to create an Intercomm Test Mode load module (see Figure 61).

9. Create test data sets and add DD statements for them to the execution JCL.

10. Execute in Test Mode with test messages in card-image format:

    a. Single-thread test the subsystem; to test a reentrant subsystem, initially specify MNCL=1 in the subsystem's SYCTTBL macro.

    b. Multithread test a reentrant subsystem (change MNCL) using several test messages.

    Test Mode execution is activated by the parameter 'TEST' on the Intercomm EXEC statement. Figure 61 illustrates a sample execution deck with test message input (DD statement SYSIN) for the sample inquiry program and JCL to print the system log.

    The resulting snaps for the test mode execution of the sample inquiry subsystem are illustrated in Figure 62.

    The System Log printed after executing in Test Mode with the sample inquiry subsystem is shown in Figure 63.

11. Test the subsystem concurrently with other application subsystems.

Note: to implement the sample subsystem for on-line execution, it would be necessary to code a BTVERB macro (in USRBTVRB--see Chapter 12) as follows:

```
BTVERB VERB=RTRA,SSCH=R,SSC=A,CONV=18000,EDIT=YES
```

```
 2 SQASM    CSECT
 3 *
 4 *     SAMPLE REENTRANT ASSEMBLER SUBSYSTEM USING THE FILE HANDLER TO
 5 *     ACCESS  BDAM AND VSAM FILES. THE EDIT UTILITY IS USED FOR INPUT
 6 *     MESSAGE EDITING, THE OUTPUT UTILITY FOR OUTPUT ERROR MESSAGE
 7 *     FORMATTING, AND THE DISPLAY UTILITY FOR OUTPUT MESSAGE TEXT
 8 *     CONVERSION TO AN OUTPUT UTILITY FORMATTING MESSAGE.
 9 *
10 ***********************   REGISTER USAGE  ******************************
11 ***  R2    PARAMETER SAVE REGISTER                                    ***
12 ***  R3    SPA ADDRESS                                                ***
13 ***  R4    INPUT MESSAGE ADDRESS                                      ***
14 ***  R5    BASE REGISTER FOR OUTPUT MESSAGE DSECT                     ***
15 ***  R6    HOLD OUTPUT MESSAGE LENGTH                                 ***
16 ***  R7    -  UNUSED  -                                               ***
17 ***  R8    -  UNUSED  -                                               ***
18 ***  R9    WORK REGISTER                                              ***
19 ***  R10   RETURN CODE                                                ***
20 ***  R11   BAL REGISTER                                               ***
21 ***  R12   BASE REGISTER                                              ***
22 ***  R13   SAVEAREA(WORKAREA DSECT)                                   ***
23 ************************************************************************


25          REGS
26+*  EQUATE RN NAMES TO ALL GENERAL PURPOSE REGISTERS
27+*                                LAST REVISION    12/10/68
28+R0       EQU   0
29+R1       EQU   1
30+R2       EQU   2
31+R3       EQU   3
32+R4       EQU   4
33+R5       EQU   5
34+R6       EQU   6
35+R7       EQU   7
36+R8       EQU   8
37+R9       EQU   9
38+R10      EQU   10
39+R11      EQU   11
40+R12      EQU   12
41+R13      EQU   13
42+R14      EQU   14
43+R15      EQU   15
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 1 of 10)

```
 45          PRINT NOGEN
 46 SQASM    CSECT
 47 ********************************************************************************
 48 *                  PROVIDE LINKAGE AND OBTAIN CORE                             *
 49 ********************************************************************************
 50          USING INMSG,R4
 51          USING OUTMSG,R5
 52          USING WORKAREA,R13
 53          LINKAGE BASE=(R12),LEN=DYNLEN,PARM=(R2),SPA=(R3),MSG=(R4),     X
                DSECTS=(SCT,MSG,R13)
 55+         PRINT NOGEN
 69+         PRINT NOGEN
 74+         PRINT GEN
 75+         PUSH  PRINT               TURN OFF PRINT GENERATION
 76+         PRINT NOGEN
789+         POP   PRINT               RESUME PRINT GENERATION
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 2 of 10)

```
816+          PRINT NOGEN
917           MVI   ERRMSG,C' '          BLANK ERROR MESSAGE AREA
918           MVC   ERRMSG+1(L'ERRMSG-1),ERRMSG
919           SR    R10,R10              SET RETURN CODE TO ZERO


921 ********************************************************************
922 *                          INVOKE EDITCTRL                         *
923 ********************************************************************
924 MAPINPUT DS    0H
925           CALL  EDITCTRL,((R4),(R3),0),VL,MF=(E,PARMSAVE)
937           LTR   R15,R15              EDIT OK ?
938           BZ    MSGOKAY              YES
939           LA    R10,4                NO - SET RETURN CODE
940 *                        NOTE: EDIT UTILITY RETURNS ERROR MESSAGE
941           B     RETURN               EXIT (NO MSG TO FREE)
942 MSGOKAY  DS    0H
943           L     R4,PARMSAVE          LOAD EDITED MSG ADDRESS
944           ST    R4,IMSGADDR          SAVE MSG ADDR FOR LATER FREE
945 ********************************************************************
946 *              PREPARE TO SELECT AND ACCESS BDAM FILE              *
947 ********************************************************************
948           MVC   CURRFILE,=C'PARTFILE'  DD NAME OF BDAM FILE
949           BAL   R11,SELECT           SELECT THE FILE-EXIT IF NO GOOD
950           PACK  DBLWORD,RBNBYTE      PACK RBN DIGIT INTO DOUBLEWORD
951           CVB   R9,DBLWORD           CONVERT TO BINARY
952           ST    R9,FULLRBN           RBN FOR ACCESSING BDAM FILE
953           CALL  READ,(EXTDSCT,FHCW,BDAMFILE,RBN),VL,MF=(E,PARMSAVE)
967           CLI   FHCW,C'0'            WAS READ SUCCESSFUL?
968           BNE   BDAMERR              NO
969           BAL   R11,RELEASE          RELEASE THE FILE
970           CLC   PARTNO,PARTNUM       DO WE HAVE THE CORRECT RECORD?
971           BNE   NOTFOUND             NO
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 3 of 10)

```
 973 ********************************************************************
 974 *               PREPARE TO SELECT AND ACCESS VSAM FILE             *
 975 ********************************************************************
 976          XC     EXTDSCT,EXTDSCT       CLEAR EXTDSCT
 977          MVC    CURRFILE,=C'STOKFILE'   DDNAME OF VSAM FILE
 978          BAL    R11,SELECT            SELECT THE FILE-EXIT IF NO GOOD
 979          MVC    KEYPART,PARTNO        FORMAT KEY FOR VSAM GET
 980          MVC    KEYWHS,WHSNO          VSAM KEY NOW COMPLETE
 981          CALL   GETV,(EXTDSCT,FHCW,VSAMFILE,VSAMKEY),VL,MF=(E,PARMSAVE)
 995          CLI    FHCW,C'0'             WAS GETV SUCCESSFUL?
 996          BNE    VSAMERR               NO
 997          BAL    R11,RELEASE           ROUTINE TO RELEASE FILE
 998          LA     R6,OUTMLEN            SET OUTPUT MSG AREA LENGTH
 999          BAL    R11,GETOMSG           GET AND INIT OUTPUT MSG AREA
1000          B      MOVEINFO              SET OUTPUT TEXT AREAS


1002 ********************************************************************
1003 *         SELECT, RELEASE, AND OUTPUT MESSAGE 'STORAGE' ROUTINES     *
1004 ********************************************************************
1005 SELECT   DS     0H
1006          MVC    FHCW,BLANKS           CLEAR FILE HANDLER CONTROL WORD
1007          CALL   SELECT,(EXTDSCT,FHCW,CURRFILE),VL,MF=(E,PARMSAVE)
1019          CLI    FHCW,C'0'             WAS SELECT SUCCESSFUL?
1020          BNE    SLCTERR               NO
1021          MVC    FHCW,BLANKS           CLEAR FHCW FOR I/O
1022          BR     R11                   BRANCH BACK


1024 RELEASE  DS     0H
1025          CALL   RELEASE,(EXTDSCT,FHCW),VL,MF=(E,PARMSAVE)
1036          BR     R11                   BRANCH BACK

1038 GETOMSG  DS     0H
1039          STORAGE ADDR=OMSGADDR,LEN=(6),LIST=PARMSAVE,SPA=(3)
1050          LTR    R15,R15              WAS STORAGE ACQUIRED ?
1051          BZ     CONT                 YES
1052          LA     R10,8                NO CORE RETURN CODE
1053          B      FREEIN               NOTHING CAN BE DONE-GO BACK
1054 CONT     DS     0H
1055          L      R4,IMSGADDR          RELOAD INPUT MSG ADDRESS
1056          L      R5,OMSGADDR          LOAD OUTPUT MSG ADDRESS
1057          MVC    MSGHLEN(MSGHLNTH),0(R4)  MOVE INPUT MSG HEADER
1058          STH    R6,MSGHLEN           STORE OUTPUT MSG LENGTH
1059          MVI    MSGHQPR,C'2'         SET QPR - SINGLE SEGMENT MSG
1060          MVC    MSGHSSCH,MSGHRSCH    SET HI-ORDER SENDING SS CODE
1061          MVC    MSGHSSC,MSGHRSC      SET LO-ORDER SENDING SS CODE
1062          MVI    MSGHRSCH,0           SET HI-ORDER RECEIVING SS CODE
1063          BR     R11                  BRANCH BACK
```

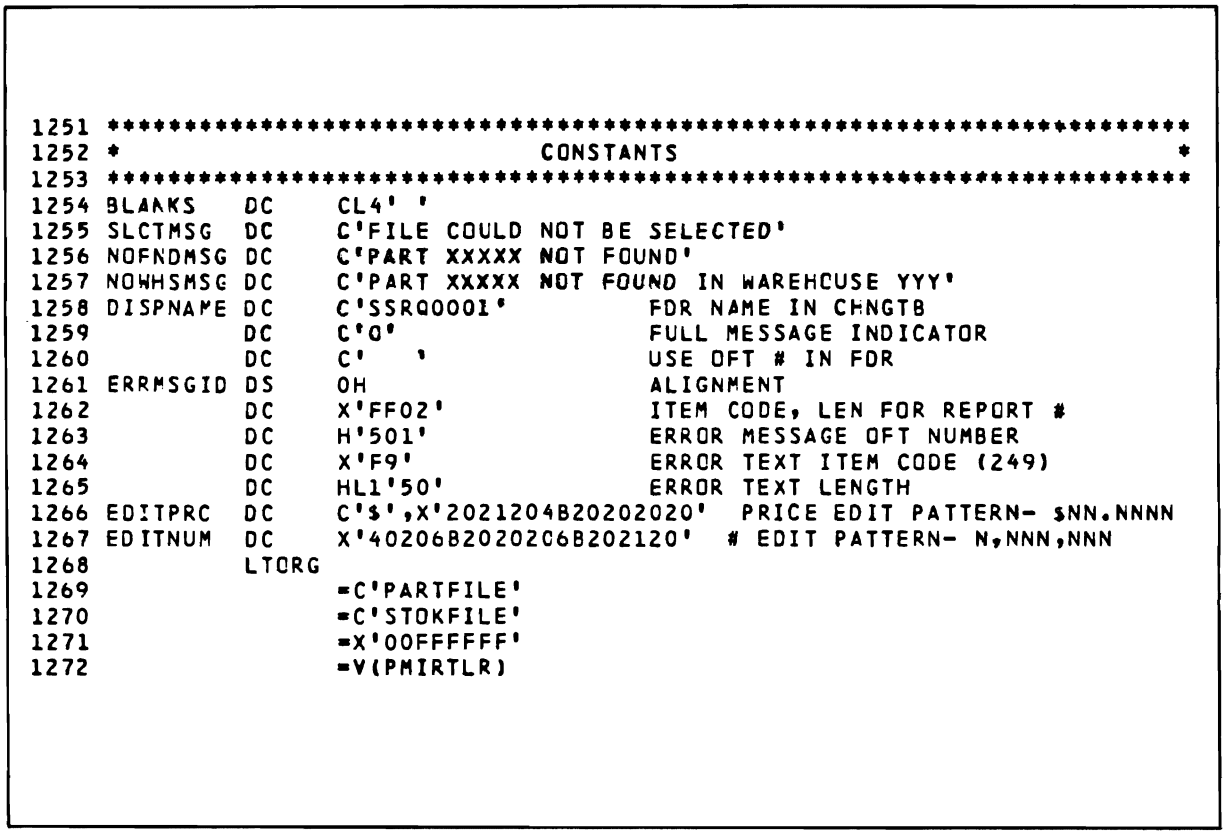
Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 4 of 10)

```
1065 ********************************************************************
1066 *              MOVE DATA FROM FILES TO OUTPUT MESSAGE AREA          *
1067 *     PACKED INTERNAL FIELDS ARE PRE-EDITED FOR DISPLAY UTILITY     *
1068 ********************************************************************
1069 MOVEINFO DS    0H
1070          MVI   MSGHRSC,C'H'           SET RSC FOR DISPLAY SS
1071          MVI   MSGHVMI,X'72'          SET VMI FOR DISPLAY SS
1072          MVC   FMTNAME,DISPNAME       FILE DESCRIPTION RECORD NAME
1073          MVC   OUTWHSNO(2),BLANKS     LEADING BLANKS NEEDED
1074          MVC   OUTWHSNO+2(3),WHSNO    WAREHOUSE NUMBER
1075          MVC   PRTDATA,PARTNUM        PART #, DESCRIPTION, UNITS
1076          MVC   PRCEDIT,EDITPRC        MOVE EDIT PATTERN TO WORK AREA
1077          ED    PRCEDIT,PRICE          EDIT UNIT PRICE
1078          MVC   PRTPRC,PRCEDIT         PRICE
1079          MVC   WHSLOC,WHS             WAREHOUSE
1080          MVC   NUMEDIT,EDITNUM        MOVE EDIT PATTERN TO WORK AREA
1081          ED    NUMEDIT,SLEV           EDIT STOCK ON HAND
1082          MVC   STKLEV,NUMEDIT+1       STOCK LEVEL
1083          MVC   LEVDATE(2),SDATE       MONTH
1084          MVI   LEVDATE+2,C'/'
1085          MVC   LEVDATE+3(2),SDATE+2   DAY
1086          MVI   LEVDATE+5,C'/'
1087          MVC   LEVDATE+6(2),SDATE+4   YEAR
1088          MVC   NUMEDIT,EDITNUM        MOVE EDIT PATTERN TO WORK AREA
1089          ED    NUMEDIT,OLEV           EDIT STOCK ON ORDER
1090          MVC   STKORD,NUMEDIT+1       STOCK ORDER
1091          MVC   ORDDATE(2),ODATE       MONTH
1092          MVI   ORDDATE+2,C'/'
1093          MVC   ORDDATE+3(2),ODATE+2   DAY
1094          MVI   ORDDATE+5,C'/'
1095          MVC   ORDDATE+6(2),ODATE+4   YEAR
1096 ********************************************************************
1097 *              INVOKE MSGCOL TO QUEUE MESSAGE FOR DISPLAY UTILITY   *
1098 ********************************************************************
1099          CALL  MSGCOL,((5),(3)),VL,MF=(E,PARMSAVE)
1110          LTR   R15,R15              SUCCESSFULLY QUEUED?
1111          BZ    FREEIN               YES
1112          LA    R10,8                NO CORE RETURN CODE
1113          B     FREEIN               EXIT
1114          PRINT NOGEN
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 5 of 10)

```
1116 ************************************************************************
1117 *                FREE INPUT MESSAGE AREA AND RETURN                    *
1118 ************************************************************************
1119 FREEIN   DS    0H
1120          L     R1,IMSGADDR          RELOAD INPUT MSG ADDRESS
1121          LH    R0,0(R1)             LOAD MESSAGE LENGTH
1122          STORFREE ADDR=(1),LEN=(0),SPA=(3)
1123+         PRINT NOGEN                                                  CH
1128+         PRINT GEN                                                    CH
1129+         N     0,=X'0CFFFFFF'       ENSURE SUBPOOL BYTE IS CLEARED.
1130+         L     15,SPAFREE-SPALST(3)  .   LOAD STORFREE ROUTINE ADDR  SK
1131+         BALR  14,15                CALL ROUTINE.
1132 RETURN   DS    0H
1133          PRINT NOGEN
1134          RTNLINK ADDR=(R13),LEN=DYNLEN,RC=(R10)     RETURN CONTROL
1144+         PRINT NOGEN
1180+*,GETSPA - V7.0 - 11/76 - SM


1204 ************************************************************************
1205 *                          ERROR PROCESSING                            *
1206 ************************************************************************
1207 SLCTERR  DS    0H
1208          MVC   ERRMSG(8),CURRFILE         MOVE FILE NAME
1209          MVI   ERRMSG+9,C'-'
1210          MVC   ERRMSG+11(L'SLCTMSG),SLCTMSG  SELECT-ERROR MESSAGE
1211          B     DOERRMSG
1212 NOTFOUND DS    0H
1213          MVC   ERRMSG(L'NOFNDMSG),NOFNDMSG   ELUSIVE-PART MESSAGE
1214          MVC   ERRMSG+5(5),PARTNO      MOVE IN INVALID PART NUMBER
1215          B     DOERRMSG
1216 VSAMERR  DS    0H
1217          CLI   FHCW,C'2'                RECORD NOT FOUND ?
1218          BNE   BDAMERR                  NO - I/O ERROR
1219          BAL   R11,RELEASE
1220          MVC   ERRMSG(L'NOWHSMSG),NOWHSMSG   WRONG-WAREHOUSE MESSAGE
1221          MVC   ERRMSG+5(5),PARTNO      MOVE IN INVALID PART NUMBER
1222          MVC   ERRMSG+34(3),WHSNO      MOVE IN INVALID WHS NUMBER
1223          B     DOERRMSG
1224 BDAMERR  DS    0H
1225          BAL   R11,RELEASE
1226          LA    R10,12                   I/O ERROR RETURN CODE
1227          B     FREEIN
1228 DOERRMSG DS    0H
1229          LA    R6,ERRMLEN               SET ERROR MSG LENGTH
1230          BAL   R11,GETOMSG              GET AND INIT OUTPUT MESSAGE AREA
1231          MVI   MSGHRSC,C'U'             SET OUTPUT UTILITY SS CODE
1232          MVI   MSGHVMI,X'50'            SET FORMAT MSG VMI
1233          MVC   ERRORFMT,ERRMSGID        MOVE REPORT #, ITEM CODE, LENGTH
1234          MVC   ERRORTXT,ERRMSG          MOVE ERROR MESSAGE TEXT
1235          CALL  MSGCOL,((5),(3)),VL,MF=(E,PARMSAVE)
1246          LTR   R15,R15                  WAS MSG QUEUING SUCCESSFUL?
1247          BZ    FREEIN                   YES - FREE INPUT MSG, EXIT
1248          LA    R10,8                    NO CORE RETURN CODE
1249          B     FREEIN                   NOTHING CAN BE DONE-GOBACK
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 6 of 10)



```
1251 ********************************************************************************
1252 *                              CONSTANTS                                       *
1253 ********************************************************************************
1254 BLANKS   DC    CL4' '
1255 SLCTMSG  DC    C'FILE COULD NOT BE SELECTED'
1256 NOFNDMSG DC    C'PART XXXXX NOT FOUND'
1257 NOWHSMSG DC    C'PART XXXXX NOT FOUND IN WAREHOUSE YYY'
1258 DISPNAME DC    C'SSRQ0001'          FDR NAME IN CHNGTB
1259          DC    C'0'                 FULL MESSAGE INDICATOR
1260          DC    C'   '               USE DFT # IN FDR
1261 ERRMSGID DS    0H                   ALIGNMENT
1262          DC    X'FF02'              ITEM CODE, LEN FOR REPORT #
1263          DC    H'501'               ERROR MESSAGE DFT NUMBER
1264          DC    X'F9'                ERROR TEXT ITEM CODE (249)
1265          DC    HL1'50'              ERROR TEXT LENGTH
1266 EDITPRC  DC    C'$',X'2021204B20202020'  PRICE EDIT PATTERN- $NN.NNNN
1267 EDITNUM  DC    X'40206B2020206B202120'  # EDIT PATTERN- N,NNN,NNN
1268          LTORG
1269                =C'PARTFILE'
1270                =C'STOKFILE'
1271                =X'00FFFFFF'
1272                =V(PMIRTLR)
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 7 of 10)

```
1274 ********************************************************************************
1275 *                                DSECTS                                        *
1276 ********************************************************************************
1277 WORKAREA DSECT
1278 SAVEAREA DS    18F
1279 IMSGADDR DS    F                    ADDRESS OF INPUT MESSAGE
1280 OMSGADDR DS    F                    ADDRESS OF OUTPUT MESSAGE
1281 PARMSAVE DS    6F                   6 IS MAX NUMBER OF PARMS PASSED
1282 DBLWORD  DS    D                    USED IN CREATING RBN
1283 FHCW     DS    F                    FILE HANDLER CONTROL WORD
1284 EXTDSCT  DS    12F                  BDAM AND THEN VSAM CONTROL AREA
1285 FULLRBN  DS    0F
1286          DS    XL1
1287 RBN      DS    XL3                  USED FOR ACCESSING THE BDAM FILE
1288 VSAMKEY  DS    0CL8
1289 KEYWHS   DS    CL3
1290 KEYPART  DS    CL5
1291 CURRFILE DS    CL8
1292 BDAMFILE DS    0CL100               100 BYTE BDAM RECORD BEGINS HERE
1293 PARTNUM  DS    CL5
1294 DESCRIPT DS    CL54
1295 UNITS    DS    CL5
1296 PRICE    DS    PL4
1297 MFR      DS    CL15
1298 UNUSED1  DS    CL17
1299 VSAMFILE DS    0CL80                80 BYTE VSAM RECORD BEGINS HERE
1300 DELETE   DS    CL1
1301 KEY      DS    CL8
1302 UNUSED2  DS    CL28
1303 WHS      DS    CL23
1304 SLEV     DS    PL4
1305 SDATE    DS    CL6
1306 OLEV     DS    PL4
1307 ODATE    DS    CL6
1308 ERRMSG   DS    CL50
1309          ORG   ERRMSG               REUSE AREA FOR NUMBER EDITING
1310 NUMEDIT  DS    CL10
1311 PRCEDIT  DS    CL9
1312          ORG
1313 DYNLEN   EQU   *-WORKAREA           TOTAL DYNAMIC WORKAREA LENGTH


1315 INMSG    DSECT
1316          DS    CL42                 MESSAGE HEADER
1317 PARTNO   DS    0CL5                 PART NUMBER
1318          DS    CL4
1319 RBNBYTE  DS    CL1                  FOR BDAM FILE ACCESS
1320 WHSNO    DS    CL3                  WAREHOUSE NUMBER
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 8 of 10)

```
1322 CLTMSG   DSECT
1323          COPY  MSGHDRC              MESSAGE HEADER
1324 *
1325 *       MESSAGE HEADER LAYOUT
1326 *       ********************
1327 *                                LAST REVISION 10/20/82-RELEASE 9.0
1328 *                                LAST REVISION 07/30/85-LU 6.2 SUPPORT
1329 *
1330 MSGHLEN  DS    BL2        LENGTH OF MESSAGE
1331 MSGHQPR  DS    BL1        CTAM/BTAM I/O PREFIX BLANK IF SS MSG
1332 MSGHRSCH DS    XL1        HI-ORDER BYTE OF RECEIVING SUBSYSTEM CODE
1333 MSGHRSC  DS    CL1        RECEIVING SUBSYSTEM CODE
1334 MSGHSSC  DS    CL1        SENDING SUBSYSTEM CODE
1335 MSGHMMN  DS    OBL3       MONITOR SEQUENCE NUMBER                    X1078
1336 MSGHTXTL DS    BL2              RECORD LENGTH (FILE RECOVERY)        X1078
1337 MSGHKEYL DS    CL1              KEY LENGTH (FILE RECOVERY)           X1078
1338 MSGHDAT  DS    OCL6       DATE (YY.DDD)                              X1078
1339 MSGHYR   DS    CL2              YEAR                                 X1078
1340 MSGHTHRD DS    BL1              THREAD NUMBER                        X1078
1341 MSGHDAY  DS    CL3              DAY                                  X1078
1342 MSGHTIM  DS    CL8        TIME   (HH.MM.SS)
1343          ORG   MSGHTIM            FIELDS USED IN SCANVERB DURING     JA
1344 *                          CONSTRUCTION OF MESSAGE IN LINE HANDLERS   JA
1345 MSGHVFLG DS    B                FLAGS                                 JA
1346 MSGHVFND EQU   X'80'          VERB WAS ANALYZED BEFORE CALLING BTSEARCH JA
1347 MSGHVBA  DS    AL3        A(BTVERB ENTRY) IF MSGHVFND FLAG ON         JA
1348          ORG   MSGHTIM+L'MSGHTIM                                      JA
1349 MSGHTID  DS    CL5        TERMINAL ID   (AAANN)  AAA=CITY,NN=DEVICE ID
1350 MSGHMRDX DS    0X         INDEX TO MULTIREGION MCT ENTRY
1351 MSGHCON  DS    BL2        COMPANY NUMBER
1352 *        SPECIAL VALUES OF MSGHCON                                    JA
1353 MSGHCFLA EQU   X'BB01'       FLUSH-ALL CHASER MSG                     JA
1354 MSGHCP12 EQU   X'BB03'   3270 COPY FORM 1 (REM.-SAME CU),2 (3275-HR) JA
1355 *  MSGHCP12:  COPY TYPE 1 OR 2, ISSUING TERM REQUEST RESPONSE     SM1124
1356 MSGHCN12 EQU   X'BB13'   COPY TYPE 1 OR 2, NO RESPONSE TO ISSUER SM1124
1357 MSGHCP3  EQU   X'BB02'   3270 COPY FORM3 (READ FULL BUF REQUEST)      JA
1358 MSGHR129 EQU   X'BB04'   IBM129 CARD READER RESET I/P INHIBITED MSG   JA
1359 MSGHFEVR EQU   X'BB'     SET IN MSGHCON+1 OF RESPONSES TO F.E.VERBS   JA
1360          ORG   MSGHCON+1
1361 MSGHRETN DS    BL1        RETURN CODE.
1362 MSGHCONV EQU   C'C'             30 LOGGED FROM CONVERSE.
1363 MSGHFLGS DS    OFL2       MESSAGE INDICATOR FLAGS                    SM1166
1364 MSGHFLG1 DS    FL1        MESSAGE INDICATOR FLAG-BYTE-1              SM1166
1365 MSGHFSDR EQU   X'80'      ASK FOR DEFINITE RESPONSE                 VTAM
1366 MSGHFSER EQU   X'40'      ASK FOR EXCEPTION RESPONSE                VTAM
1367 *              IF MSGHFSDR+MSGHFSER=0 THEN NO RESPONSE              VTAM
1368 *              SPECIFICATION, USE OTHER SOURCES TO DETERMINE.       VTAM
1369 MSGHFRSP EQU   MSGHFSDR+MSGHFSER   MASK TO CHECK 'SRESP'            VTAM
1370 MSGHFSR1 EQU   X'20'      1 -> RESPONSE TYPE 1 (FME)                VTAM
1371 MSGHFSR2 EQU   X'10'      1 -> RESPONSE TYPE 2 (RRN)                VTAM
1372 MSGHFSEB EQU   X'08'      SEND EB WITH THIS MESSAGE                 VTAM
1373 MSGHNCON EQU   X'04'      DO NOT CANCEL CONVERSATION TIMEOUT         XM0215
1374 MSGHFNF3 EQU   X'02'      1 -> DONT WRITE X'F3' LOG RECORD FOR MSG
1375 MSGHFRLS EQU   X'01'      RELEASE NEXT OUTPUT MESSAGE                SM1166
1376 *                                                                    SM1166
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 9 of 10)

```
1377 MSGHFLG2 DS    FL1       MESSAGE INDICATOR FLAG-BYTE-2              SM1166
1378 MSGHFTRM EQU   X'80'     MSGHADDR POINTS TO SOURCE BTERM/LUC        SM1166
1379 MSGHSRST EQU   X'40'     SERIALLY RESTARTED MESSAGE INDICATOR  (9.0) CH
1380 MSGHSYSC EQU   X'20'     QUEUE THIS MSG TO A 6.2 SESSION EVEN        51MD
1381 *                        IF NO CONVERSATION CURRENTLY ACTIVE         51MD
1382 MSGHFMHI EQU   X'10'     THIS MESSAGE CONTAINS 6.2 FMHDR             51MD
1383 *                                                                     JS
1384 MSGHBMN  DS    BL3       BTAM SEQUENCE NUMBER                        JS
1385 *                                                                     JS
1386 MSGHPMN  EQU   *
1387 MSGHSSCH DS    XL1       HI/ORDER BYTE OF SENDING SUBSYSTEM
1388 MSGHUSR  DS    XL1       AVAILABLE TO USER
1389          ORG   MSGHUSR                                               JS
1390 MSGHADDR DS    AL3       ADDRESS OF AN AUXILIARY AREA (FE ONLY)      JS
1391          ORG   MSGHTID             FOR FILE RECOVERY               X1078
1392 MSGHBKID DS    CL8       BDAM BLOCK ID (FILE RECOVERY)             X1078
1393 MSGHDD   DS    CL8       FILE DDNAME (FILE RECOVERY)               X1078
1394 MSGHLOG  DC    C'0'      LOG TYPE CODE -SEE MONITOR WRITEUP
1395 RVZONE   EQU   X'80'          FILE REVERSAL ENTRY.
1396 RCZONE   EQU   X'90'          FILE RECREATION ENTRY
1397 MSGHXFIL EQU   RVZONE+15      CHECKPOINT RECORD.
1398 RCSTUP   EQU   RCZONE+15      STARTUP RECORD.
1399 MSGHRQST EQU   X'A0'          LOGPROC REQUEUEING STARTED.
1400 MSGHRQND EQU   X'A1'          LOGPROC REQUEUEING ENDED.
1401 MSGHRBUF DS    0H        BUFFER LENGTH (BDAM FILE RECOVERY)        X1078
1402 MSGHMACR DS    0BL1      FILE HANDLER MACRO #                        JT
1403 MSGHBLK  DS    CL1               BLANK  (BINARY ZERO)
1404 MSGHVMI  DS    BL1       VERB/MSG ID
1405 MSGHFFVM EQU   X'67'          SPECIAL VMI FOR FULLY FORMATTED MSGS     JA
1406 DDQVMI   EQU   X'EE'          SPECIAL VMI FOR DYN. DATA QUEING         MM
1407 *
1408 MSGHEND  EQU   *
1409 MSGHLNTH EQU   MSGHEND-MSGHLEN          LENGTH OF MESSAGE HEADER
1410 *
1411 OUTTEXT  DS    0CL147                  TEXT AREA
1412 FMTNAME  DS    CL12                    FIXED FORMAT AREA
1413 PRTDATA  DS    CL64                    PART #, DESCRIPTION, UNIT TYPE
1414 PRTPRC   DS    CL9                     PART PRICE (EDITED)
1415 OUTWHSNO DS    CL5                     WAREHOUSE NUMBER (LEADING BLANKS)
1416 OUTSDATA DS    0CL57                   STOCK DATA
1417 WHSLOC   DS    CL23                    WAREHOUSE SITE
1418 STKLEV   DS    CL9                     WAREHOUSE IN STOCK (EDITED)
1419 LEVDATE  DS    CL8                     WAREHOUSE LEVEL DATE
1420 STKORD   DS    CL9                     WAREHOUSE ON ORDER (EDITED)
1421 ORDDATE  DS    CL8                     WAREHOUSE ORDER DATE
1422          DS    0D                      ROUND UP MSG AREA
1423 OUTMLEN  EQU   *-OUTMSG                OUTPUT MSG AREA LENGTH
1424          ORG   OUTTEXT
1425 ERRORFMT DS    CL6                     ERROR MSG OFT #, ITEM CODE, LEN
1426 ERRORTXT DS    CL50                    ERROR TEXT AREA
1427          DS    0D                      ROUND UP MSG AREA
1428 ERRMLEN  EQU   *-OUTMSG                OUTPUT MSG AREA LENGTH
1429          END
```

Figure 56. Sample Inquiry Subsystem; Reentrant Assembler (Page 10 of 10)

PAGE 179 INTENTIONALLY MISSING

```
//TABLES      JOB
//*
//*                 DEFINE SYCTTBL FOR SUBSYSTEM
//*
//STEP1       EXEC  LIBELINK,Q=TEST,NAME=INTSCT,LMOD=INTSCT
//LIB.SYSIN   DD    *
./ ADD NAME=USRSCTS
./ NUMBER     NEW1=100,INCR=100
USRSCTS       DS    0H
RA            SYCTTBL SUBH=R,SUBC=A,SBSP=SQASM,LANG=RBAL,OVLY=0,        X
                    NUMCL=10,MNCL=1,TCTV=60
/*
//ASM.SYSIN   DD    DSN=INT.SYMREL(INTSCT),DISP=SHR
//*
//*                 DEFINE EDIT CONTROL TABLE ENTRY
//*
//STEP2       EXEC  LIBELINK,Q=TEST,NAME=PMIVERBS,LMOD=PMIVERBS
//LIB.SYSIN   DD    *
./ ADD  NAME=USRVERBS
./ NUMBER     NEW1=100,INCR=100
USRVERBS      DS    0H
RTRAECT       VERB  RTRA,D9,256,2,FIX=YES
              PARM  P/N,1,7,5,10000111
              PARM  WHS,2,7,3,10000111
/*
//ASM.SYSIN   DD    DSN=INT.SYMREL(PMIVERBS),DISP=SHR
//*
//*                 DEFINE CHANGE/DISPLAY TABLE
//*
//STEP3       EXEC  LIBELINK,Q=TEST,NAME=CHNGTB,LMOD=CHNGTB
//LIB.SYSIN   DD    *
./ ADD  NAME=CHNGTB
./ NUMBER     NEW1=100,INCR=100
CHTB          TITLE 'CHNGTB - FIXED FORMAT OUTPUT-DESCRIPTOR NAME TABLE'
CHNGTB        CSECT
              DC    CL8'SSRQ0001' USED ONLY TO TEST BAL PGM. GUIDE S/S
              DC    F'0'
              PMISTOP
              END
//
```

Figure 57. Table Updates to Implement Test Mode Testing

```
* OUTPUT FORMAT TABLE FOR SAMPLE INQUIRY SUBSYSTEM                00000100
*                                                                 00000200
OFT100   REPORT NUM=100,LINES=8                                   00000300
         LINE   NUM=1,ITEMS=1                                     00000400
         ITEM   CODE=255,DATA='STCCK STATLS REQUEST',FROM=6,TO=25 00000500
         LINE   NUM=2,ITEMS=2                                     00000600
         ITEM   CCDE=255,DATA='PART NUMBER',FROM=1,TO=11          00000700
C12PNO   ITEM   CODE=12,FROM=13,TO=17                             00000800
         LINE   NUM=3,ITEMS=2                                     00000900
         ITEM   CCDE=255,DATA='DESCRIPTION',FROM=1,TO=11          00001000
C21DES   ITEM   CCDE=21,FROM=13,TO=66                             00001100
         LINE   NUM=4,ITEMS=4                                     00001200
         ITEM   CODE=255,DATA='ORDER UNITS',FROM=1,TO=11          00001300
C18UNT   ITEM   CODE=18,FROM=13,TO=17                             00001400
         ITEM   CCDE=255,DATA='PRICE',FROM=19,TO=23               00001500
C19PRC   ITEM   CCDE=19,FROM=25,TO=33                             00001601
         LINE   NUM=5,ITEMS=2                                     00001700
         ITEM   CCDE=255,DATA='STCCK STATUS AT WAREHOUSE',FROM=1,TC=25 00001800
C8WHS    ITEM   CODE=8,FROM=27,TO=31                              00001900
         LINE   NUM=6,ITEMS=2                                     00002000
         ITEM   CODE=255,DATA='LOCATION',FRCM=4,TO=11             00002100
C10WLC   ITEM   CODE=10,FROM=13,TO=35                             00002200
         LINE   NUM=7,ITEMS=4                                     00002300
         ITEM   CODE=255,DATA='ON HANC',FROM=6,TO=12              00002400
C13LEV   ITEM   CODE=13,FROM=15,TO=23                             00002500
C14LDT   ITEM   CODE=14,FROM=38,TO=45                             00002600
         ITEM   CCDE=255,DATA='AS OF',FROM=31,TO=35               00002700
         LINE   NUM=8,ITEMS=4                                     00002800
         ITEM   CODE=255,CATA='ON ORDER',FROM=6,TO=13             00002900
C15ORD   ITEM   CODE=15,FROM=15,TO=23                             00003000
         ITEM   CCDE=255,DATA='AS OF',FROM=31,TO=35               00003100
C16OAT   ITEM   CCDE=16,FROM=38,TO=45                             00003200
         END                                                      00003300
```

Figure 58. Utilities Table Coding for Test Mode Subsystem (Page 1 of 2)

```
* OUTPUT FORMAT TABLE FOR ERROR MESSAGES FROM INQUIRY SUBSYSTEM      00000100
*                                                                   00000200
OFT5C1   REPORT NUM=501,LINES=1                                     00000300
         LINE NUM=1,ITEMS=2                                         00000400
         ITEM CODE=255,FROM=1,TO=10,DATA='**ERROR** '               00000500
         ITEM CODE=249,FROM=12,TO=62                                00000600
         END                                                        00000700
* FILE DESCRIPTION RECORD FOR FIXED FORMAT CUTPUT                   00000100
*        FROM SAMPLE INQUIRY SUBSYSTEM                              00000200
DES0C001 CSECT                                                      00000300
SSRQ100  FDHDR NAME=SSRQC001,RPTNO=100,FIELDS=10                    0C000400
PNO12    FDETL OFSET=0,LEN=5,NAME=P/NXX,CODE=12                     00000500
DES21    FDETL OFSET=5,LEN=54,NAME=DESXX,CODE=21                    00000600
UNT18    FDETL OFSET=59,LEN=5,NAME=UNTXX,CODE=18                    00000700
PRC19    FDETL OFSET=64,LEN=9,NAME=PRCXX,CODE=19                    00000801
WHS08    FDETL OFSET=72,LEN=5,NAME=WHSXX,CODE=8                     00000900
WLC10    FDETL OFSET=77,LEN=23,NAME=WLCXX,CCDE=10                   00001000
LEV13    FDETL OFSET=100,LEN=9,NAME=LEVXX,CODE=13                   00001100
LDT14    FDETL OFSET=109,LEN=8,NAME=LDTXX,CODE=14                   00001200
ORD15    FDETL OFSET=117,LEN=9,NAME=ORDXX,CCDE=15                   00001300
ODT16    FDETL OFSET=126,LEN=8,NAME=ODTXX,CCDE=16                   0C001400
         END                                                        00001500
```

Figure 58. Utilities Table Coding for Test Mode Subsystem (Page 2 of 2)

Card		Contents
HEADER	1-3	MSG
	*6-8	Low-order byte of S/S code (MSGHRSC) (or 8)
	*9-11	Hi-order byte of S/S code (MSGHRSCH) (or 11)
	20-24	Sending terminal ID (MSGHTID)
	50-53	Front-end Message Number (MSGHBMN)
	*55-57	VMI value (MSGHVMI); leave blank if EDIT required; code 255 if no editing by Edit Utility (or 57).
DETAIL(s)	1-64**	Data for one line of input message. If VMI in header card is left blank, a new line character is inserted at end of text on every card except last one. If the last non-blank character is a $ sign (X'5B'), it will be replaced by a NL; the preceding character (usually a blank) is kept as part of the input. All NL's are suppressed if editing is not required.
TRAILER	1-3	Generates End of Transmission character following the last non-blank character of the previous detail card. Contents of Card / Ending Character EMS — EOT (X'37') EOT — EOT (X'37') ETX — ETX (X'03') ETB — ETB (X'26')

*3-digit integer values (from 000 to 255) or a corresponding single alpha-numeric character in low-order field position.

**64 is default maximum. See the Operating Reference Manual if necessary to alter this specification.

Figure 59. Test Mode Message Card Formats

```
MSG     A  R          TEST1                          0001
RTRA
P/N 12345
WHS 200
EMS
MSG     A  R          TEST1                          0001
RTRA
P/N 55555
WHS 200
EMS
MSG     A  R          TEST1                          0001
RTRA
P/N 12345
WHS 300
EMS
MSG     A  R          TEST1                          0001
RTRA
P/N 12349
WHS 200
EMS
MSG     A  R          TEST1                          0001
RTRA
P/N 12341
WHS 100
EMS
MSG     A  R          TEST1                          0001
RTRA
P/N A2345
WHS 400
EMS
```

Figure 60. Sample Input Test Messages for Test Mode

```
//EXECTEST JOB (ICOMTEST,,,2C),'ICOM TEST SCASM',CLASS=A,
//   RESTART=(GENLINK.ASM)
//PRCCLIB DD DSN=INT.PROCLIB,DISP=SHR        (AS NEEDED)
//*********************************************************************
//*  THE RESTART PARM IN THE JOB STATEMENT RESTARTS THE TEST AT THE   *
//*  BEGINNING.  IF YOU WISH TO RESTART AT A DIFFERENT STEP, CODE     *
//*  RESTART=STEPNAME  OR  RESTART=STEPNAME.PROCSTEPNAME              *
//*                                                                   *
//*  NOTE: WHEN USING A VSAM FILE, IT MAY BE NECESSARY TO EXECUTE     *
//*        IDCAMS TO VERIFY THE FILE IF A PREVIOUS EXECUTION ABENDED. *
//*********************************************************************
//*
//*********************************************************************
//*  STEP GENLINK GENERATES A STANDARD TEST MODE LINKEDIT DECK         *
//*   VIA ASSEMBLY OF THE ICOMLINK MACRO.                             *
//*  THE GENERATED DECK (TESTLINK) IS PLACED ON INT.SYMTEST.          *
//*********************************************************************
//GENLINK  EXEC  ASMPC,Q=LIB,L=REL,DECK=DECK
//ASM.SYSIN DD *
         ICOMLINK TEST=YES,MMU=NO,STCRFCH=NO
         END
//SYSPLNCH DD DSN=INT.SYMTEST(TESTLINK),DISP=SHR
//*
//*********************************************************************
//*  STEPS SCRSCR AND ALLOCSCR DELETE AND RE-ALLOCATE THE LOAD        *
//*   MODULE LIBRARY USED IN THE TEST   (ALSO USED FOR DYNLLIB)       *
//*********************************************************************
//SCRSCR   EXEC PGM=IEFBR14
//FILE1    DD   DSN=INT.MODSCR,DISP=(OLD,DELETE)
//ALLOCSCR EXEC PGM=IEFBR14
//A        DD   DSN=INT.MODSCR,DISP=(,CATLG),UNIT=SYSDA,
//    DCB=INT.MODREL,VOL=SER=INT001,SPACE=(CYL,(3,,7))
//*
//*********************************************************************
//*  STEP GENINCL CREATES INCLUDE CARDS USED BY THE LINK EDIT STEP    *
//*  THE ADDED INCLUDE STATEMENTS ARE FOR THE SAMPLE SUBSYSTEM AND    *
//*       THE REFERENCED OFTS (INCLUDE AFTER PMIRCNTB).               *
//*  IF THE TEST1 TERMINAL IS NOT IN THE SYSTEM PMISTATB TABLE, ADD:  *
//*       INCLUDE MODREL(PMISTATB)                                    *
//*       INCLUDE MODREL(PMIDEVTB)                                    *
//*       INCLUDE MODREL(PMIBROAD)                                    *
//*     THE ABOVE ASSUMES THE CONTROL TERMINAL IS NAMED CNT01.        *
//* *** BEFORE THIS STEP, SEQUENCE NUMBER THE TESTLINK SOURCE. *****  *
//*********************************************************************
//GENINCL  EXEC PGM=IEBUPDTE
//SYSPRINT DD   SYSOUT=A
//SYSUT1   DD   DSN=INT.SYMTEST,DISP=SHR
//SYSUT2 DD DSN=&&INCL,DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1,1)),
//    DCB=(BLKSIZE=80,LRECL=80)
//SYSIN    DD   *
./  CHANGE NAME=TESTLINK,LIST=ALL
         INCLUDE SYSLIB(SQASM)            SAMPLE SUBSYSTEM              00000010
         INCLUDE SYSLIB(RPT00100)         DISPLAY OFT FOR SUBSYSTEM     01981000
         INCLUDE SYSLIB(RPT00501)         ERROR MESSAGES OFT            01982000
```

NOTE: JCL requirements vary by installation requirements. The above example illustrates representative JCL. The installation System Manager should verify JCL to use.

Figure 61. Linkedit and Execution JCL for Test Mode (Page 1 of 3)

```
//*********************************************************************
//*   LINK EDIT THE TEST INTERCOMM SYSTEM                             *
//*   NOTE: THE INTERCOMM PROC 'LKEDT' LINKEDITS MODULES FROM THE     *
//*         SYSLIB CONCATENATION STREAM AS FOLLOWS -                  *
//*         THE LOAD LIBRARY SPECIFIED BY THE C= PARAMETER,           *
//*         FOLLOWED BY MODULES FOUND IN MODUSR, MODLIB, THEN MODREL. *
//*         THE INTERCOMM LOAD MODULE IS PLACED ON INT.MODSCR.        *
//*   IT IS NOT NECESSARY TO RE-DO THE WHOLE LINK TO REPLACE 1 MODULE *
//*   IN THIS CASE, ALL YOU SHOULD DO IS:                             *
//*    1)  REASSEMBLE OR RECOMPILE THE CHANGED/NEW MODULE INTO A      *
//*        SEPARATE LOAD LIBRARY                                      *
//*    2)  OVERRIDE THE SYSLIN DD STMT TO //LKED.SYSLIN DD *          *
//*        FOLLOW IT WITH INCLUDE CARDS                               *
//*        FOR THE MODULES YOU WISH TO REPLACE                        *
//*    3)  FOLLOW THOSE INCLUDES WITH THE FOLLOWING 3 CARDS:          *
//*             INCLUDE SYSLMOD(TESTICOM)                             *
//*             ENTRY   PMISTUP                                       *
//*             NAME    TESTICOM(R)                                   *
//*     4) INSERT A DD STMT FOR THE LOAD LIBRARY ON WHICH THE         *
//*        REPLACEMENT MODULES RESIDE                                 *
//*     5) CHANGE THE RESTART PARM ON THE JCB STATEMENT               *
//*          TO POINT TO THE LKED STEP                                *
//*********************************************************************
//LKED     EXEC LKEDT,LMOD=TESTICOM,Q=TEST,
//      PARM.LKED='LIST,LET,XREF,NCAL,SIZE=(250K,100K)'
//LKED.SYSLIN DD DSN=&&INCL(TESTLINK),DISP=(OLD,PASS)
//MODREL   DD   DSN=INT.MODREL,DISP=SHR
```

Figure 61. Linkedit and Execution JCL for Test Mode (Page 2 of 3)

```
//*********************************************************************
//*    EXECUTE INTERCOMM IN TESTMODE                                  *
//*********************************************************************
//GC       EXEC PGM=TESTICOM,PARM='TEST',TIME=(,30)
//STEPLIB  DD   DSN=INT.MODSCR,DISP=(OLD,PASS)          (DYNLLIB)
//         DD   DSN=INT.MODUSR,DISP=SHR          (USER LOAD LIBRARY)
//         DD   DSN=INT.MODLIB,DISP=SHR          (SYSTEM UPDATE LIBRARY)
//         DD   DSN=INT.MODREL,DISP=SHR          (SYSTEM RELEASE LIBRARY)
//INTERLOG DD   DSN=&&INTLOG,DISP=(NEW,PASS),
//  SPACE=(TRK,(10,5)),VOL=SER=INT001,UNIT=SYSDA,
//  DCB=(DSORG=PS,RECFM=VB,BLKSIZE=4096,LRECL=4092,NCP=8,OPTCD=C)
//STSLOG   DD SYSOUT=A,DCB=(DSORG=PS,BLKSIZE=120,RECFM=FA)
//SMLOG    DD SYSOUT=A,DCB=(DSORG=PS,BLKSIZE=120,RECFM=FA)
//SYSPRINT DD SYSOUT=A,DCB=(DSORG=PS,RECFM=VA,BLKSIZE=141,LRECL=137)
//RCT000   DD DSN=INT.RCT000,DISP=SHR,
//          DCB=(DSORG=DA,OPTCD=RF)              OUTPUT FORMATS
//PMIQUE   DD DISP=OLD,DSN=INT.PMIQUE,
//          DCB=(DSORG=DA,OPTCD=R)               SUBSYSTEM DISK QUEUE
//STCKFILE DD DSN=VSAMSD1.STCKFILE.CLUSTER,DISP=OLD,
//          AMP=(AMORG,'RECFM=F')                VSAM TEST FILE
//PARTFILE DD DSN=INT.TEST.PARTFILE,DISP=OLD,
//          DCB=(DSORG=DA,OPTCD=R)               BDAM TEST FILE
//DES000   DD DSN=INT.DES000,DISP=SHR,
//          DCB=(DSORG=DA,OPTCD=RF)        FILE DESCRIPTION RECORDS
//SYSIN    DD DSN=INT.SYMTEST(TESTMSGS),DISP=SHR,
//          DCB=DSORG=PS                   TEST MODE INPUT MESSAGES
//PMISTOP  DD DUMMY
//ICOMIN   DD  *
INTSTORO,ICOMBDAMXCTRL
/*
//STEPCAT  DD DSN=VSAMSD1,DISP=SHR                 VSAM CATALOG
//DYNLPRNT DD SYSOUT=A
//DYNLWORK DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(,PASS)
//DYNLLIB  DD DSN=INT.MODSCR,DISP=(OLD,PASS)
//*
//SNAPDD   DD SYSOUT=A
//SYSSNAP  DD SYSOUT=A                    SNAP INPUT TEST MESSAGES
//SYSSNAP2 DD SYSOUT=A                    SNAP OUTPUT TEST MESSAGES
//SYSUDUMP DD SYSOUT=A
//*
//ABNLIGNR DD DUMMY  FORCE ABEND-AID TO IGNORE DUMP (PRODUCE IBM DUMP)
//*
//*********************************************************************
//*   PRINT INTERCOMM LOG FROM TEST MODE RUN                          *
//*********************************************************************
//INTERLOG EXEC PGM=LOGPRINT,COND=EVEN
//STEPLIB  DD   DSN=INT.MODREL,DISP=SHR
//SYSPRINT DD   SYSOUT=A,DCB=(DSORG=PS,BLKSIZE=121)
//INTERLOG DD   DSN=&&INTLOG,DISP=SHR,DCB=BLKSIZE=5000
//SYSIN    DD   DUMMY,DCB=BLKSIZE=80
//
```

Figure 61. Linkedit and Execution JCL for Test Mode (Page 3 of 3)

```
JOB INTT003X          STEP GO              TIME 125632   DATE 89199    ID = 015    CPUID = 001101783081   PAGE 00000001

PSW AT ENTRY TO SNAP   07802000 0C0074BA        ILC 2   INTC 0033

-STORAGE

00006F20                    004102D9 C1000000    00000000 00000000 00C00000 0CC000E3   *        ...RA...................T*
00006F40 C5E2E3F1 00010000 00000100 C0000100    0000D9E3 D9C115D7 61D540F1 F2F3F4F5   *EST1..............RTRA.P.N 12345*
00006F60 15E6C8E2 40F2F0F0 37000000                                                   *.WHS 200....                          *


-STORAGE

00006F20                    004102D9 C1000000    00000000 00000000 00C00000 000000E3   *        ...RA...................T*
00006F40 C5E2E3F1 00010000 00000100 C0000100    C000D9E3 D9C115D7 61D540F5 F5F5F5F5   *EST1..............RTRA.P.N 55555*
00006F60 15E6C8E2 40F2F0F0 37000000                                                   *.WHS 200....                          *


-STORAGE

00006F20                    00410209 C100000C    00000000 00000000 00000000 000000E3   *        ...RA...................T*
00006F40 C5E2E3F1 00010000 00000100 C0000100    0000D9E3 D9C115D7 61D540F1 F2F3F4F5   *EST1..............RTRA.P.N 12345*
00006F60 15E6C8E2 40F3F0F0 37000000                                                   *.WHS 300....                          *


-STORAGE

00006F20                    004102D9 C100000C    00000000 00000000 00000000 000000E3   *        ...RA...................T*
00006F40 C5E2E3F1 00010000 00000100 C0000100    0000D9E3 D9C115D7 61D540F1 F2F3F4F9   *EST1..............RTRA.P.N 12349*
00006F60 15E6C8E2 40F2F0F0 37000000                                                   *.WHS 200....                          *


-STORAGE

00006F20                    004102D9 C100000C    00000000 00000000 00000000 000000E3   *        ...RA...................T*
00006F40 C5E2E3F1 000100C0 00000100 00000100    0000D9E3 D9C115D7 61D540F1 F2F3F4F1   *EST1..............RTRA.P.N 12341*
00006F60 15E6C8E2 40F1F0F0 37000000                                                   *.WHS 100....                          *


-STORAGE

00006F20                    00410209 C1000000    00000000 00000000 00C00000 000000E3   *        ...RA...................T*
00006F40 C5E2E3F1 000100C0 00000100 0000010C    0000D9E3 D9C115D7 61D540C1 F2F3F4F5   *EST1..............RTRA.P.N A2345*
00006F60 15E6C8E2 40F4F0F0 37000000                                                   *.WHS 400....                          *
```

Figure 62. Sample Test Mode Execution Snaps (Page 1 of 3)

```
JOB INTT003X          STEP GO              TIME 125632   DATE 89199    ID = 020    CPUID = 001101783C81   PAGE 00000001

PSW AT ENTRY TC SNAP   078D2E00 00007AF6        ILC 2   INTC 0033

-STORAGE

0001D740 C0690200 E4E40000 08F8F900 F1F9F9F1    F2F5F6F3 F2F6F1C3 D5E3F0F1 00010C00   *....UU...89.19912563261CNT01....*
0C01D760 00000000 000000F2 0050155C 5C5C40C7    D6D6C440 C1C6E3C5 D9D5D6D6 D55C5C40   *.......2...... GOOD AFTERNOON.. *
0001D780 40C9D5E3 C5D9C3D6 D4D440C9 E240D9C5    C1C4E840 7A404040 F0F760F1 F860F8F9   * INTERCOMM IS READY .   C7.18.89*
0001C7AC 4040F1F2 4BF5F615 26000000                                                   *  12.56.....                    *

-STORAGE

0001D620                                        C0670200 E4E40000 09F8F900 F1F9F9F1   *                ....UU...89.1991*
0001D640 F2F5F6F3 F3F0F9E3 C5E2E3F1 000100CC    00000000 000000F2 00505C5C 5C40C7D6   *2563309TEST1...........2..... GO*
0001D660 D6C440C1 C6E3C5D9 D5D6D6D5 5C5C404C    C9D5E3C5 D9C3D6D4 D440C9E2 40D9C5C1   *OD AFTERNOON..  INTERCOMM IS REA*
0001D680 C4E8407A 404040F0 F760F1F8 60F8F940    40F1F24B F5F63700                     *DY .   07.18.89  12.56..        *

-STORAGE

0001FAC0 C12D0200 E4E40000 0BF8F901 F1F9F9F1    F2F5F6F3 F3F9F8E3 C5E2E3F1 00010000   *....UU...89.19912563398TEST1....*
0001FAE0 00000100 000001F2 00504C40 404040E2    E3D6C3D2 40E2E3C1 E3E4E240 D9C5D8E4   *.......2..     STOCK STATUS REQU*
0001FB00 C5E2E315 D7C1D9E3 4CD5E4D4 C2C5D940    F1F2F3F4 F515C4C5 E2C3D9C9 D7E3C9D6   *EST.PART NUMBER 12345.DESCRIPTIO*
0001FB2C D540F161 F240C9C5 40E2E3C5 C5D340E6    C1E2C8C5 D915D6D9 C4C5D940 E4D5C9E3   *N 1.2 IN STEEL WASHER.ORDER UNIT*
0001FB40 E240C7D9 E2404040 D7D9C9C3 C5405BF5    F0F548F0 F5F0F715 E2E3D6C3 D240E2E3   *S GRS   PRICE .505.0507.STOCK ST*
0001FB60 C1E3E4E2 40C1E340 E6C1D9C5 C8D6E4E2    C540F740 40F2FC15 404040D3 D6C3C1E3   *ATUS AT WAREHOUSE 7  20.   LOCAT*
0001FB80 C9D6D540 F0D4C9C1 D4C96B40 C6D3C14B    15404040 4040D6D5 40C8C1D5 C44040F6   *ION OMIAMI. FLA..     ON HAND  6*
0001FBA0 6BF1F6F1 6BF5F04C 40404040 404040C1    E240D6C6 4040F6F0 F361F0F5 61F81540   *.161.50        AS OF  603.05.8. *
0001FBC0 40404040 D6D540D6 D9C4C5D9 4CF2F46B    F0F4F06B F6F14040 40404040 40C1E240   *    ON ORDER 24.040.61       AS *
0C01FBE0 C6C64040 F7F1F061 F1F161F8 37BBBBBB                                          *OF  710.11.8....                *

-STCRAGE

0001D1E0 004A0200 E4E40000 0CF8F902 F1F9F9F1    F2F5F6F3 F4F0F2E3 C5E2E3F1 00010000   *....UU...89.19912563402TEST1....*
0001D2C0 C0000100 000001F2 00505C5C C5D9D9D6    D95C5C40 40D7C1D9 E340F5F5 F5F5F540   *.......2....ERROR..  PART 55555 *
0001C220 D5D6E340 C6D6E4C5 C437BBBB                                                   *NOT FOUND...                    *

-STORAGE

0001D100                                        005B02C0 E4E40000 0DF8F901 F1F9F9F1   *                ....UU...89.1991*
0001D120 F2F5F6F3 F4F0F8E3 C5E2E3F1 0001000C    00000100 000001F2 00505C5C C5D9D9D6   *2563408TEST1...........2....ERRO*
0001D140 D95C5C40 40D7C1D9 E340F1F2 F3F4F540    D5D6E340 C6D6E4D5 C440C9D5 40E6C1D9   *R..  PART 12345 NOT FOUND IN WAR*
0001C160 C5C8D6E4 E2C540F3 F0F03700                                                   *EHOUSE 300..                    *

-STCRAGE

0001D240                    004A0200 E4E40000   0EF8F901 F1F9F9F1 F2F5F6F3 F4F1F5E3   *        ....UU...89.19912563415T*
0001D260 C5E2E3F1 00010000 00000100 C00001F2    00505C5C C5D9D9D6 D95C5C40 40D7C1D9   *EST1...........2....ERROR..  PAR*
0001D28C E340F1F2 F3F4F940 D5D6E340 C6D6E4D5    C437BBBB                              *T 12349 NOT FOUND...            *
```

Figure 62. Sample Test Mode Execution Snaps (Page 2 of 3)

```
JOB INTT003X          STEP GO              TIME 125634   DATE 89199    ID = 020    CPUID = 001101783081   PAGE 00000001

PSW AT ENTRY TC SNAP   078D2E00 00007AF6        ILC 2   INTC 0033

-STORAGE

0001DDC0                                                        00B90200 D5E40000   *                          ....NU..*
0001DDE0 10F8F902 F1F9F9F1 F2F5F6F3 F4F1F8E3    C5E2E3F1 0001000C 00000100 000001F2   *.89.19912563418TEST1...........2*
0001DEC0 0050F0F0 C5F840F0 F0F0F2F9 15D5D6D5    6CD5E4D4 C5D9C9C3 40C3C8C1 D9C1C3E3   *..00E8 00029.NON.NUMERIC CHARACT*
0001DE20 C5D940C7 C9E5C5D5 40D6D540 D761D540    D7C1D9C1 D4C5E3C5 D940C6D6 D940D9E3   *ER GIVEN ON P.N PARAMETER FOR RT*
0001DE40 D9C140E5 C5D9C248 40C1D3D3 40C3C8C1    D9C1C3E3 C5D9E240 E2C8D6E4 D3C440C2   *RA VERB. ALL CHARACTERS SHOULD B*
0001DE60 C540D5E4 D4C5D9C9 C34B1540 D4C5E2E2    C1C7C540 D5D64840 F0F0F0F0 F0F0F0F6   *E NUMERIC.. MESSAGE NO. 00C00006*
0001DE80 40C6D9D6 D440E3D7 E440E3C5 E2E3F14B    37F161F8                              * FROM TPU TEST1..1.8            *

-STORAGE

0001DDC0                                                        00B60200 D5E40000   *                          ....NU..*
0001DDE0 11F8F902 F1F9F9F1 F2F5F6F3 F4F1F8E3    C5E2E3F1 00010000 00000100 000001F2   *.89.19912563418TEST1...........2*
0001DE00 0050F0F0 C5F840F0 F0F0F2F2 15D9C5D8    E4C9D9C5 C440D7C1 D9C1D4C5 E3C5D940   *..00E8 00022.REQUIRED PARAMETER *
0001DE20 D761D540 E6C1E240 D6D4C9E3 E3C5C44D    D6D940C7 C9E5C5D5 40C9D540 C5D9D9D6   *P.N WAS OMITTED.OR GIVEN IN ERRO*
0001DE40 D95DD6D5 40E3C8C5 4040D9E3 D9C140E5    C5D9C24B 40E5C5D9 C240E6C1 E240C3C1   *R.ON THE  RTRA VERB. VERB WAS CA*
0001DE60 D5C3C5D3 D3C5C415 40D4C5E2 E2C1C7C5    40D5D64B 40F0F0F0 F0F0F0F0 F640C6D9   *NCELLED. MESSAGE NO. 00000006 FR*
0001DE80 D6D440E3 D7E440E3 C5E2E3F1 4B37F14B                                          *OM TPU TEST1..1.                *

-STORAGE

0002C780                                        014102C0 E4E40000 12F8F901 F1F9F9F1   *                ....UU...89.1991*
000207A0 F2F5F6F3 F4F2F8E3 C5E2E3F1 0001000C    00000100 000001F2 C0504040 404040E2   *2563428TEST1...........2..     S*
000207C0 E3D6C3D2 40E2E3C1 E3E4E240 D9C5D8E4    C5E2E315 D7C1D9E3 40D5E4D4 C2C5D940   *TOCK STATUS REQUEST.PART NUMBER *
000207E0 F1F2F3F4 F115C4C5 E2C3D9C9 D7E3C9D6    D540F161 F440C9D5 40C3C8D9 D6D4C540   *12341.DESCRIPTION 1.4 IN CHROME *
00020800 C3C5D9D9 C1E3C5C4 40D3D6C3 D240D5E4    E315D6D9 C4C5D940 E4D5C9E3 E240C4D6   *CERRATED LOCK NUT.ORDER UNITS DO*
00020820 E9404040 D7D9C9C3 C5405BF6 F1F64BF1    F6F1F615 E2E3D6C3 D240E2E3 C1E3E4E2   *Z   PRICE .616.1616.STOCK STATUS*
00020840 40C1E340 E6C1D9C5 C8D6E4E2 C540F640    40F1F015 404040D3 D6C3C1E3 C9D6D540   * AT WAREHOUSE 6  10.   LOCATION *
00020860 F0D5C5E6 40E8D6D9 D240C3C9 E3E86B40    D54BE84B 15404040 4040D6D5 40C8C1D5   *ONEW YORK CITY. N.Y..     ON HAN*
00020880 C44040F5 6BF0F5F0 6BF5F040 40404040    404040C1 E240D6C6 4040F4F0 F361F0F5   *D  5.050.50        AS OF  403.05*
000208A0 61F8154C 40404040 D6D540D6 D9C4C5D9    40F2F56B F0F5F06B F5F0404C 40404040   *.8.     ON ORDER 25.050.50      *
0002C8C0 40C1E24C D6C64040 F4F1F061 F1F161F8    37000000                              * AS OF  410.11.8....            *

-STORAGE

0001D7CC                    00690200 E4E40000    15F8F900 F1F9F9F1 F2F5F6F3 F7F5F7C3   *        ....UU...89.19912563757C*
0001D7E0 D5E3F0F1 00000000 00000000 C00000F2    0050155C 5C5C40C7 D6D6C440 C1C6E3C5   *NT01...........2...... GOOD AFTE*
0001D800 D9D5D6D6 D55C5C40 40C9D5E3 C5D9C3D6    D4D440C9 E240C3D3 D6E2C5C4 7A404040   *RNOON..  INTERCOMM IS CLOSED.   *
0001D820 F0F760F1 F860F8F9 4040F1F2 4BF5F615    26000000                              *07.18.89  12.56.....            *

-STORAGE

0001D740 C0670200 E4E40000 16F8F900 F1F9F9F1    F2F5F6F3 F8F0F5E3 C5E2E3F1 00000000   *....UU...89.19912563805TEST1....*
0001D760 00000000 000000F2 00505C5C 5C40C7D6    D6C440C1 C6E3C5D9 D5D6D6D5 5C5C4040   *.......2..... GOOD AFTERNOON..  *
0001D780 C9D5E3C5 D9C3D6D4 D440C9E2 40C3D3D6    E2C5C47A 404040F0 F760F1F8 60F8F940   *INTERCOMM IS CLOSED.   C7.18.89 *
0001D7A0 40F1F24B F5F63700                                                            * 12.56..                        *
```

Figure 62. Sample Test Mode Execution Snaps (Page 3 of 3)

```
DATE 89.199   TIME 12.56.41          **** I N T E R C O M M   L O G   D I S P L A Y ****          PAGE    1

MSGLEN THREAD  CPR  RSC     SSC       MMN    DATE      TIME       TID   FLGS  USR          BMN LOG BLK VMI
------------------------------------------------------------------------------------------------------------
    78     0   02  ../0CC0  ../0CC0     1   89.199   12.56.3225  TEST1 0000  00           1  9F  00  00
000000 C9D5E3C5 D9C3D6C4 D440E2E3 C1D9E3E4   D7400C5 E2E2C1C7 C54C6040 C9D5E3E3  *INTERCOMM STARTUP MESSAGE - INIT*
000032 F0F0F3E7                                                                  *003X                *
------------------------------------------------------------------------------------------------------------
    65     0   02  RA/D9C1 ../0CCC     1   89.199   12.56.3225  TEST1 0000  00           1  01  00  00
000000 D9E3D9C1 15D761D5 40F1F2F3 F4F515E6   C8E240F2 F0F037            *RTRA.P/N 12345.WHS 200.          *
------------------------------------------------------------------------------------------------------------
    65     0   02  RA/D9C1 ../CC00     2   89.199   12.56.3226  TEST1 0000  C0           1  01  00  00
000000 D9E3D9C1 15D761D5 40F5F5F5 F5F515E6   C8E240F2 F0F037            *RTRA.P/N 55555.WHS 200.          *
------------------------------------------------------------------------------------------------------------
    65     0   02  RA/D9C1 ../0000     3   89.199   12.56.3227  TEST1 0000  00           1  01  00  00
0C0000 D9E3D9C1 15D761D5 40F1F2F3 F4F515E6   C8E240F3 F0F037            *RTRA.P/N 12345.WHS 300.          *
------------------------------------------------------------------------------------------------------------
    65     0   C2  RA/D9C1 ../00C0     4   89.199   12.56.3228  TEST1 0000  00           1  01  00  00
000000 D9E3D9C1 15D761D5 40F1F2F3 F4F915E6   C8E240F2 F0F037            *RTRA.P/N 12349.WHS 200.          *
------------------------------------------------------------------------------------------------------------
    65     0   02  RA/D9C1 ../0000     5   89.199   12.56.3228  TEST1 0000  00           1  01  00  00
0C0000 D9E3D9C1 15D761D5 40F1F2F3 F4F115E6   C8E240F1 F0F037            *RTRA.P/N 12341.WHS 100.          *
------------------------------------------------------------------------------------------------------------
    65     0   02  RA/D9C1 ../0000     6   89.199   12.56.3229  TEST1 0000  00           1  01  00  00
000000 D9E3D9C1 15D761D5 40C1F2F3 F4F515E6   C8E240F4 F0F037            *RTRA.P/N A2345.WHS 400.          *
------------------------------------------------------------------------------------------------------------
   108     0   02  .U/00E4 ../0000     7   89.199   12.56.3256  TOALL 0000  00           0  01  00  50
0C0000 FF02002D 013C5C5C 5C40C7D6 D6C440C1   C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5  *.....*** GOOD AFTERNOON** INTE*
000032 D9C3D6D4 D440C9E2 40D9C5C1 C4E8407A   404040F0 F760F1F8 60F8F940 40F1F24B  *RCCMM IS READY :  07-18-89 12.*
000064 F5F6                                                                      *56                  *
------------------------------------------------------------------------------------------------------------
    42     1   02  .U/00E4 ../0000     7   89.199   12.56.3257  TOALL 0000  00           0  30  00  50
    42     1   02  .U/CCE4 ../CC00     7   89.199   12.56.3257  TOALL 0000  00           0  FA  00  50
------------------------------------------------------------------------------------------------------------
    42     1   02  RA/D9C1 ../0000     1   89.199   12.56.3257  TEST1 0000  00           1  30  00  00
------------------------------------------------------------------------------------------------------------
   1C8     0   02  .U/00E4 ../0000     8   89.199   12.56.3261  CNT01 0000  00           0  01  00  50
000000 FF02002D 013C5C5C 5C40C7D6 D6C440C1   C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5  *.....*** GOOD AFTERNOON** INTE*
000032 D9C3D6D4 D440C9E2 4CD9C5C1 C4E8407A   404040F0 F760F1F8 60F8F940 40F1F24B  *RCCMM IS READY :  C7-18-89 12.*
000064 F5F6                                                                      *56                  *
------------------------------------------------------------------------------------------------------------
    42     2   02  .U/CCE4 ../0C00     8   89.199   12.56.3261  CNT01 0000  00           0  30  00  50
------------------------------------------------------------------------------------------------------------
   105     2   02  .U/00E4 .U/00E4     8   89.199   12.56.3274  CNT01 0000  00           0  40  00  50
000000 155C5C5C 40C7D6D6 C44CC1C6 E3C5D9D5   D6D6D55C 5C4040C9 D5E3C5D9 C3D6D4D4  *.*** GOOD AFTERNOON** INTERCOMM*
000032 40C9E240 D9C5C1C4 E8407A40 4040F0F7   60F1F860 F8F94040 F1F248F5 F61526    * IS READY :  07-18-89 12.56..*
------------------------------------------------------------------------------------------------------------
    42     2   02  .U/CCE4 ../00C0     8   89.199   12.56.3274  CNT01 0000  00           0  FA  00  50
------------------------------------------------------------------------------------------------------------
   108     0   02  .U/00E4 ../0000     9   89.199   12.56.3309  TEST1 0000  00           0  01  00  50
0C0000 FF02002D 013C5C5C 5C40C7D6 D6C440C1   C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5  *.....*** GOOD AFTERNOON** INTE*
000032 D9C3D6D4 D440C9E2 40D9C5C1 C4E8407A   404040F0 F760F1F8 60F8F940 40F1F24B  *RCCMM IS READY :  07-18-89 12.*
0C0064 F5F6                                                                      *56                  *
------------------------------------------------------------------------------------------------------------
```

Figure 63. Test Mode Execution Log Printout (Page 1 of 6)

```
DATE 89.199    TIME 12.56.41        **** I N T E R C O M M   L C G   D I S P L A Y ****                    PAGE    2

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------------
    42      2    02  .U/00E4  ../0000          9    89.199     12.56.3309   TEST1   0000   00            0   30  00   50
------------------------------------------------------------------------------------------------------------------
   103      2    02  .U/CCE4  .U/0CE4          9    89.199     12.56.331C   TEST1   0000   00            0   40  00   50
0CC0C0   5C5C5C40 C7D6D6C4 40C1C6E3 C5D9D5D6     D6D55C5C 4040C9D5 E3C5D9C3 D6D4D440   **** GOOD AFTERNOON**  INTERCOMM *
000032   C9E240D9 C5C1C4E8 407A4040 40F0F760     F1F860F8 F94040F1 F24BF5F6 37         *IS READY :   07-18-89  12.56.   *
------------------------------------------------------------------------------------------------------------------
    42      2    02  .U/00E4  ../0000          9    89.199     12.56.331C   TEST1   0000   00            0   FA  00   5C
------------------------------------------------------------------------------------------------------------------
   192      1    F2  .H/00C8  RA/D9C1        10    89.199     12.56.331C   TEST1   0000   00            1   01  00   72
0000C0   E2E2D9D8 F0F0F0F1 F0404040 F1F2F3F4     F540F161 F240C9D5 40E2E3C5 C5D34CE6   *SSRQ00010   12345 1/2 IN STEEL W*
000032   C1E2C8C5 D940404C 40404040 40404040     40404040 4040404C 40404040 40404040   *ASHER                            *
0C0064   40404C4C 404040C7 D9E24040 5BF5F0F5     4BF0F5F0 F74040F2 F0F0D4C9 C1D4C96B   *       GRS  $505.0507  200MIAMI,*
000096   40C6D3C1 4B40404C 40404040 4040404C     40F66BF1 F6F16BF5 F0F6F0F3 61F0F561   * FLA.            6,161,50603/05/*
000128   F8F2F46B F0F4F06B F6F1F7F1 F061F1F1     61F8F200 0000                        *824,040,61710/11/82...           *
------------------------------------------------------------------------------------------------------------------
    42      1    02  RA/D9C1  ../0000          1    89.199     12.56.3310   TEST1   0000   00            1   FA  00   00
------------------------------------------------------------------------------------------------------------------
    42      1    F2  .H/00C8  RA/D9C1        10    89.199     12.56.331C   TEST1   0000   00            1   30  00   72
------------------------------------------------------------------------------------------------------------------
    42      2    02  RA/D9C1  ../0C00          2    89.199     12.56.3393   TEST1   0000   00            1   30  00   00
------------------------------------------------------------------------------------------------------------------
   200      1    F2  .U/00E4  .H/00C8        11    89.199     12.56.3398   TEST1   0000   00            1   01  00   50
000000   0C05F1F2 F3F4F515 35F161F2 40C9D540     E2E3C5C5 D340E6C1 E2C8C5D9 40404040   *..12345..1/2 IN STEEL WASHER     *
000032   40404040 40404040 40404040 40404040     40404040 40404040 40404040 40401205   *                               ..*
0C0064   C7D9E240 40130958 F5F0F548 F0F5F0F7     C8C5F740 40F2F00A 17F0D4C9 C1D4C96B   *GRS  ..$505.0507..7  20..OMIAMI,*
000096   40C6D3C1 4B404040 40404040 40404040     0D08F66B F1F6F16B F5F00E08 F6F0F361   * FLA.            ..6,161,50..603/*
000128   F0F561F8 0F09F2F4 6BF0F4F0 6BF6F110     08F7F1F0 61F1F161 F8FF0200 6400       *05/8..24,040,61..710/11/8.....   *
------------------------------------------------------------------------------------------------------------------
    42      1    F2  .H/CCC8  RA/D9C1        10    89.199     12.56.3398   TEST1   0000   00            1   FA  00   72
------------------------------------------------------------------------------------------------------------------
    42      1    F2  .U/0CE4  .H/00C8        11    89.199     12.56.3398   TEST1   0000   00            1   30  00   50
------------------------------------------------------------------------------------------------------------------
   3C1      1    02  .U/00E4  .U/00E4        11    89.199     12.56.3402   TEST1   0000   00            1   40  00   50
0000CC   40404C40 40E2E3D6 C3D240E2 E3C1E3E4     E240D9C5 D8E4C5E2 E315D7C1 D9E340D5   *     STOCK STATUS REQUEST.PART N*
000032   E4D4C2C5 D940F1F2 F3F4F515 C4C5E2C3     D9C9D7E3 C9D6D540 F161F240 C9D54CE2   *UMBER 12345.DESCRIPTION 1/2 IN S*
0C0G64   E3C5C5D3 40E6C1E2 C8C5D915 D6D9C4C5     D940E4D5 C9E3E240 C7D9E240 4040D7D9   *TEEL WASHER.ORDER UNITS GRS   PR*
000096   C9C3C540 5BF5F0F5 4BF0F5F0 F715E2E3     D6C3D240 E2E3C1E3 E4E240C1 E340E6C1   *ICE $505.0507.STOCK STATUS AT WA*
000128   D9C5C8D6 E4E2C54C F74040F2 F0154040     40D3D6C3 C1E3C9D6 D540F0D4 C9C1D4C9   *REHOUSE 7  20.   LOCATION CMIAMI*
000160   6B40C6D3 C14B154C 40404040 D6D540C8     C1D5C440 40F66BF1 F6F16BF5 F0404040   *, FLA..      ON HAND  6,161,50   *
000192   4040404C 40C1E240 D6C64040 F6F0F361     F0F561F8 15404040 4040D6D5 40D6D9C4   *     AS OF  603/05/8.      ON ORD*
000224   C5D940F2 F46BF0F4 F06BF6F1 40404040     404040C1 E240D6C6 4040F7F1 FC61F1F1   *ER 24,040,61        AS OF  710/11*
000256   61F837                                                                      */8.                              *
------------------------------------------------------------------------------------------------------------------
    42      1    F2  .U/00E4  .H/00C8        11    89.199     12.56.3402   TEST1   0000   00            1   FA  00   50
------------------------------------------------------------------------------------------------------------------
   1C4      2    F2  .U/CCE4  RA/D9C1        12    89.199     12.56.3402   TEST1   00C0   00            1   01  00   50
000000   FF0201F5 F932D7C1 D9E340F5 F5F5F5F5     40D5D6E3 40C6D6E4 D5C44040 40404040   *...59.PART 55555 NOT FOUND       *
000032   40404040 40404040 40404040 40404040     40404040 40404040 00000000 0000       *                           ......  *
------------------------------------------------------------------------------------------------------------------
    42      2    02  RA/D9C1  ../CCC0          2    89.199     12.56.3402   TEST1   0000   00            1   FA  00   00
------------------------------------------------------------------------------------------------------------------

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
```

Figure 63. Test Mode Execution Log Printout (Page 2 of 6)

```
DATE 89.199   TIME 12.56.41          **** I N T E R C O M M   L C G   D I S P L A Y ****          PAGE   3

MSGLEN  THREAD  QPR  RSC     SSC       MMN   DATE      TIME      TID   FLGS  USR                    BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------------
    42       1   F2  .U/00E4 RA/D9C1    12   89.199   12.56.3402  TEST1 0000  00                     1   30  00   50
------------------------------------------------------------------------------------------------------------------
    74       1   02  .U/CCE4 .U/0CE4    12   89.199   12.56.3403  TEST1 0000  00                     1   40  00   50
000000  5C5CC5D9 D9D6D95C 5C404007 C1D9E34C  F5F5F5F5 F540D5D6 E340C6D6 E4D5C437  ***ERROR**  PART 55555 NOT FOUND.*
------------------------------------------------------------------------------------------------------------------
    42       1   F2  .U/CCE4 RA/D9C1    12   89.199   12.56.3403  TEST1 0000  00                     1   FA  00   50
------------------------------------------------------------------------------------------------------------------
    42       1   02  RA/D9C1 ../C000     3   89.199   12.56.3403  TEST1 0000  00                     1   30  00   00
------------------------------------------------------------------------------------------------------------------
    1C4      1   F2  .U/CCE4 RA/D9C1    13   89.199   12.56.3408  TEST1 0000  00                     1   01  00   50
000000  FF0201F5 F932D7C1 D9E340F1 F2F3F4F5  40D5D6E3 40C6D6E4 D5C440C9 D540E6C1  *...59.PART 12345 NOT FOUND IN WA*
000032  C9C5C8D6 E4E2C540 F3F0F4C0 40404040  40404040 40404040 00000000 0000      *REHOUSE 300            .......  *
------------------------------------------------------------------------------------------------------------------
    42       1   02  RA/D9C1 ../0CC0     3   89.199   12.56.3408  TEST1 0000  00                     1   FA  00   00
------------------------------------------------------------------------------------------------------------------
    42       1   F2  .U/CCE4 RA/D9C1    13   89.199   12.56.341C  TEST1 0000  00                     1   30  00   50
------------------------------------------------------------------------------------------------------------------
    91       1   C2  .U/00E4 .U/00E4    13   89.199   12.56.3411  TEST1 0000  00                     1   40  00   5C
0C0000  5C5CC5D9 D9D6D95C 5C404007 C1D9E340  F1F2F3F4 F540D5D6 E340C6D6 E4D5C440  ***ERROR**  PART 12345 NOT FOUND *
000032  C9D540E6 C1D9C5C8 D6E4E2C5 40F3F0FC  37                                  *IN WAREHOUSE 300.      *
------------------------------------------------------------------------------------------------------------------
    42       1   F2  .U/00E4 RA/D9C1    13   89.199   12.56.3411  TEST1 0000  00                     1   FA  00   50
------------------------------------------------------------------------------------------------------------------
    42       1   02  RA/D9C1 ../0CC0     4   89.199   12.56.3411  TEST1 0000  00                     1   30  00   00
------------------------------------------------------------------------------------------------------------------
    104      1   F2  .U/00E4 RA/D9C1    14   89.199   12.56.3415  TEST1 0000  00                     1   01  00   50
0000C0  FF0201F5 F932D7C1 D9E340F1 F2F3F4F5  40D5D6E3 40C6D6E4 D5C44040 40404040  *...59.PART 12349 NOT FOUND     *
000032  40404C4C 40404040 40404040 40404040  40404040 40404040 00000000 0000      *                       *
------------------------------------------------------------------------------------------------------------------
    42       1   02  RA/D9C1 ../0000     4   89.199   12.56.3415  TEST1 0000  00                     1   FA  00   00
------------------------------------------------------------------------------------------------------------------
    42       1   F2  .U/CCE4 RA/D9C1    14   89.199   12.56.3415  TEST1 0000  00                     1   30  00   50
------------------------------------------------------------------------------------------------------------------
    74       1   02  .U/CCE4 .U/0CE4    14   89.199   12.56.3416  TEST1 0000  00                     1   40  00   50
000000  5C5CC5D9 D9D6D95C 5C404007 C1D9E340  F1F2F3F4 F940D5D6 E340C6D6 E4D5C437  ***ERROR**  PART 12349 NOT FOUND.*
------------------------------------------------------------------------------------------------------------------
    42       1   F2  .U/CCE4 RA/D9C1    14   89.199   12.56.3416  TEST1 0000  00                     1   FA  00   50
------------------------------------------------------------------------------------------------------------------
    42       1   02  RA/D9C1 ../0CC0     5   89.199   12.56.3416  TEST1 0000  00                     1   30  00   00
------------------------------------------------------------------------------------------------------------------
    192      1   F2  .H/CCC8 RA/D9C1    15   89.199   12.56.3418  TEST1 0000  00                     1   01  00   72
000000  E2E2D9D8 F0F0F0F1 F0404040 F1F2F3F4  F140F161 F440C9D5 40C3C8D9 D6D4C54C  *SSR000010  12341 1/4 IN CHROME *
000032  C3C5D9D9 C1E3C5C4 40D3D6C3 D240D5E4  E3404040 40404040 40404040 40404040  *CERRATED LOCK NUT      *
000064  40404C4C 40404040 D6E94040 5BF6F1F6  48F1F6F1 F6404040 F0F0D5C5 E640E8D6  *  DOZ  $616.1616  100NEW YO*
0C0096  D9D240C3 C9E3E8B8 4CD548E8 4B404040  40F568F0 F5F06BF5 F0F4F0F3 61F0F561  *RK CITY, N.Y.   5,050,50403/05/*
000128  F8F2F5F6 FCF5F06B F5F0F4F1 F061F1F1  61F8F200 0000                       *825.050,50410/11/82...  *
------------------------------------------------------------------------------------------------------------------
    42       1   02  RA/D9C1 ../0CC0     5   89.199   12.56.3418  TEST1 0CC0  00                     1   FA  00   00
------------------------------------------------------------------------------------------------------------------
    42       1   F2  .H/CCC8 RA/D9C1    15   89.199   12.56.3418  TEST1 0000  00                     1   30  00   72
------------------------------------------------------------------------------------------------------------------
```

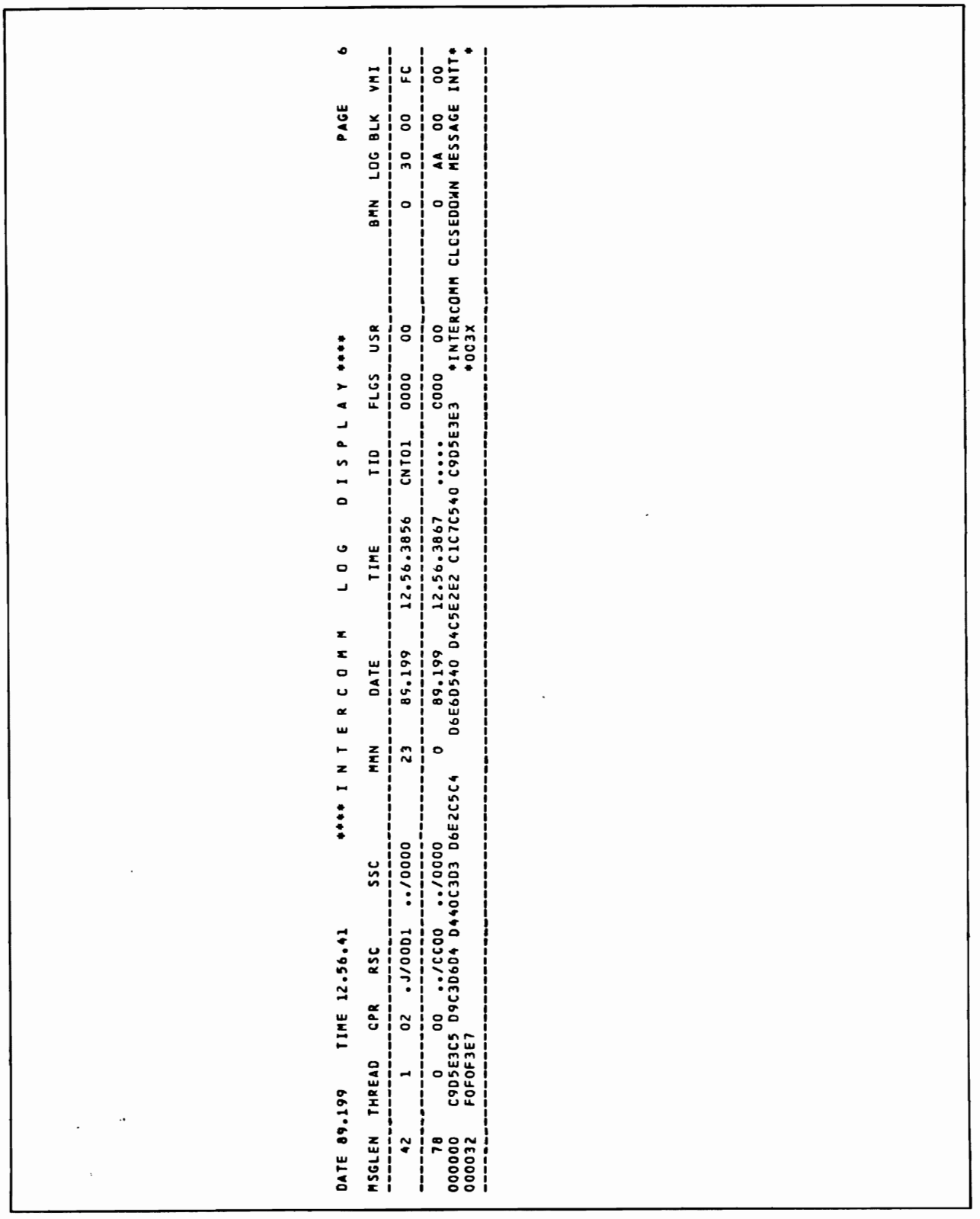
Figure 63. Test Mode Execution Log Printout (Page 3 of 6)

```
DATE 89.199    TIME 12.56.41        **** I N T E R C C M M    L C G    D I S P L A Y ****                    PAGE     4

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
------------------------------------------------------------------------------------------------------------------------
    42      2    02  RA/D9C1  ../000C          6    89.199    12.56.3418   TEST1   0000   00             1   30  00   00
------------------------------------------------------------------------------------------------------------------------
    81      2    02  .N/CCD5  .Y/0CE8         16    89.199    12.56.3418   TEST1   0000   00             1   01  00   50
0000C0   C505C1F2 F3F4F504 03D761C5 0304D9E3    D9C10105 E3C5E2E3 F1FF0200 1D0208F0   *..A2345..P/N..RTRA..TEST1.....0*
000032   F0F0F0F0 F0F0F6                                                              *0000006                        *
------------------------------------------------------------------------------------------------------------------------
    74      2    02  .N/00D5  .Y/00E8         17    89.199    12.56.3418   TEST1   0000   00             1   01  00   50
000000   0403D761 D5030409 E3D9C101 05E3C5E2    E3F1FF02 00160208 F0F0F0F0 F0F0F0F6   *..P/N..RTRA..TEST1.....00000006*
------------------------------------------------------------------------------------------------------------------------
    42      2    02  RA/D9C1  ../000C          6    89.199    12.56.3418   TEST1   0000   00             1   FA  00   00
------------------------------------------------------------------------------------------------------------------------
    42      2    02  .N/CCD5  .Y/0CE8         16    89.199    12.56.3419   TEST1   0000   00             1   30  00   50
------------------------------------------------------------------------------------------------------------------------
    42      3    02  .N/00D5  .Y/00E8         17    89.199    12.56.3428   TEST1   0000   00             1   30  00   50
------------------------------------------------------------------------------------------------------------------------
   2C0      1    F2  .U/00E4  .H/C0C8         18    89.199    12.56.3428   TEST1   0000   00             1   01  00   50
000000   CC05F1F2 F3F4F115 35F161F4 40C9D540    C3C8D9D6 D4C540C3 C5D9D9C1 E3C5C440   *..12341..1/4 IN CHROME CERRATED *
000032   D3D6C3D2 4CD5E4E3 40404C40 40404040    40404040 40404040 40404040 40401205   *LOCK NUT                        ..*
000064   C4D6E940 40130958 F6F1F64B F1F6F1F6    08C5F640 40F1F00A 17F0D5C5 E640E8D6   *DOZ  ..$616.1616..6  10..ONEW YO*
0C0096   D9D240C3 C9E3E86B 40D54BE8 4B404040    0D08F56B F0F5F06B F5F00EC8 F4F0F361   *RK CITY, N.Y.    ..5,C5C,50..403/*
000128   F0F561F8 0F09F2F5 6BF0F5F0 6BF5F010    08F4F1F0 61F1F161 F8FF02C0 6400       *05/8..25,050,50..410/11/8.....  *
------------------------------------------------------------------------------------------------------------------------
    42      1    F2  .H/00C8  RA/D9C1         15    89.199    12.56.3428   TEST1   0000   00             1   FA  00   72
------------------------------------------------------------------------------------------------------------------------
    42      1    F2  .U/00E4  .H/00C8         18    89.199    12.56.3429   TEST1   0000   00             1   30  00   50
------------------------------------------------------------------------------------------------------------------------
   185      2    02  .N/CCD5  .U/00E4         16    89.199    12.56.3433   TEST1   0000   00             1   40  00   50
000000   F0F0C5F8 40F0F0F0 F2F915D5 D6D560D5    E4D4C5D9 C9C340C3 C8C1D9C1 C3E3C5D9   *00E8 00029.NON-NUMERIC CHARACTER*
000032   40C7C9E5 C5D540D6 D540D761 D540D7C1    D9C1D4C5 E3C5D940 C6D6D940 D9E3D9C1   * GIVEN ON P/N PARAMETER FOR RTRA*
000064   40E5C5D9 C24B40C1 D3D340C3 C8C1D9C1    C3E3C5D9 E240E2C8 D6E4D3C4 40C2C540   * VERB. ALL CHARACTERS SHOULD BE *
000096   D5E4D4C5 D9C9C34B 1540D4C5 E2E2C1C7    C540D5D6 4B40F0F0 F0F0F0F0 F0F640C6   *NUMERIC.. MESSAGE NO. 00000006 F*
000128   D9D6D440 E3D7E440 E3C5E2E3 F14B37                                            *ROM TPU TEST1..                 *
------------------------------------------------------------------------------------------------------------------------
    42      2    02  .N/00D5  .Y/00E8         16    89.199    12.56.3433   TEST1   0000   00             1   FA  00   50
------------------------------------------------------------------------------------------------------------------------
   182      3    02  .N/CCD5  .U/00E4         17    89.199    12.56.3437   TEST1   0000   00             1   40  00   50
000000   F0F0C5F8 40F0F0F0 F2F215D9 C5D8E4C9    D9C5C440 D7C1D9C1 D4C5E3C5 D940D761   *00E8 00022.REQUIRED PARAMETER P/*
000032   D540E6C1 E24CD6D4 C9E3E3C5 C44DD6D9    40C7C9E5 C5D540C9 D54CC5D9 D9D6D95D   *N WAS OMITTED(OR GIVEN IN ERROR)*
0C0064   D6D540E3 C8C54040 D9E3D9C1 40E5C5D9    C24B40E5 C5D9C240 E6C1E240 C3C1D5C3   *ON THE  RTRA VERB. VERB WAS CANC*
000096   C5D3D3C5 C4154004 C5E2E2C1 C7C540D5    C64B40F0 F0F0F0F0 F0F0F640 C6D9D6D4   *ELLED. MESSAGE NO. 00000006 FROM*
000128   40E3D7E4 40E3C5E2 E3F14B37                                                   * TPU TEST1..                    *
------------------------------------------------------------------------------------------------------------------------
    42      3    02  .N/CCD5  .Y/00E8         17    89.199    12.56.3437   TEST1   0000   00             1   FA  00   50
------------------------------------------------------------------------------------------------------------------------
   321      1    02  .U/00E4  .U/00E4         18    89.199    12.56.3438   TEST1   0000   00             1   40  00   50
000000   40404C40 40E2E3D6 C3D240E2 E3C1E3E4    E240D9C5 D8E4C5E2 E315D7C1 D9E340D5   *     STOCK STATUS REQUEST.PART N*
000032   E4D4C2C5 D940F1F2 F3F4F115 C4C5E2C3    D9C9D7E3 C9D6D54C F161F440 C9D540C3   *UMBER 12341.DESCRIPTION 1/4 IN C*
000064   C8D9D6D4 C540C3C5 D9D9C1E3 C5C440D3    D6C3D240 D5E4E315 D6D9C4C5 D940E4D5   *HROME CERRATED LOCK NUT.ORDER UN*
0C0096   C9E3E240 C4D6E940 4C4CD7D9 C9C3C540    5BF6F1F6 4BF1F6F1 F615E2E3 D6C3D240   *ITS DOZ   PRICE $616.1616.STOCK *
000128   E2E3C1E3 E4E240C1 E340E6C1 D9C5C8D6    E4E2C540 F64040F1 F0154040 40D3D6C3   *STATUS AT WAREHOUSE 6  10.   LOC*
000160   C1E3C9D6 D540F0D5 C5E640E8 D6D9D240    C3C9E3E8 6B40D54B E84B1540 40404040   *ATION ONEW YORK CITY, N.Y..     *

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
```

Figure 63. Test Mode Execution Log Printout (Page 4 of 6)

```
DATE 89.199    TIME 12.56.41        **** I N T E R C O M M    L C G    D I S P L A Y ****                    PAGE     5

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
-------------------------------------------------------------------------------------------------------------------------
000192   C6D540C8 C1D5C44C 40F56BF0 F5F06BF5    F0404C40 40404C40 40C1E24C D6C64040   *ON HAND  5,050,50        AS CF  *
000224   F4F0F361 F0F561F8 15404040 4040D6D5    40D6D9C4 C5D940F2 F56BF0F5 F06BF5F0   *403/05/8.     ON ORDER 25,050,50*
000256   40404C4C 404040C1 E240D6C6 4C40F4F1    F061F1F1 61F837                       *       AS OF  410/11/8.          *
-------------------------------------------------------------------------------------------------------------------------
    42       1    F2  .U/00E4  .H/CCC8        18    89.199    12.56.3438   TEST1   0000   00            1   FA  00   50
-------------------------------------------------------------------------------------------------------------------------
    47       0    02  .J/CCD1  ../CC00        19    89.199    12.56.3757   CNT01   0000   00            0   01  00   FF
000000   D5D9C3C4 37                                                                  *NRCD.                              *
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .J/CCD1  ../CCCC        19    89.199    12.56.3757   CNT01   0000   00            0   30  00   FF
-------------------------------------------------------------------------------------------------------------------------
   108       1    02  .U/00E4  .J/00D1        20    89.199    12.56.3757   TOALL   0000   00            0   C1  00   50
000000   FF02002D 013C5C5C 5C40C7D6 D6C440C1    C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *......*** GOOD AFTERNOON**  INTE*
000032   D9C3D6D4 D440C9E2 40C3D3D6 E2C5C47A    404040F0 F760F1F8 60F8F940 40F1F24B   *RCOMM IS CLOSED:   07-18-89  12.*
0C0064   F5F6                                                                        *56                                 *
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .J/00D1  ../00C0        19    89.199    12.56.3757   CNT01   0000   00            0   FA  00   FF
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .U/CCE4  .J/C0D1        20    89.199    12.56.3757   TOALL   0000   00            0   30  00   50
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .U/00E4  .J/00D1        20    89.199    12.56.3757   TOALL   0000   00            0   FA  00   50
-------------------------------------------------------------------------------------------------------------------------
   1C8       0    02  .U/00E4  .J/C0D1        21    89.199    12.56.3757   CNT01   0000   00            0   01  00   50
0000C0   FF02002D 013C5C5C 5C40C7D6 D6C440C1    C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *......*** GOOD AFTERNOON**  INTE*
000032   D9C3D6D4 D440C9E2 40C3D3D6 E2C5C47A    404040F0 F760F1F8 60F8F940 40F1F24B   *RCOMM IS CLOSED:   07-18-89  12.*
000064   F5F6                                                                        *56                                 *
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .U/00E4  .J/00D1        21    89.199    12.56.3757   CNT01   0000   00            0   30  00   50
-------------------------------------------------------------------------------------------------------------------------
   105       1    02  .U/CCE4  .U/00E4        21    89.199    12.56.3760   CNT01   0000   00            0   40  00   50
000000   155C5C5C 40C7D6D6 C440C1C6 E3C5D9D5    D6D6D55C 5C4040C9 D5E3C5D9 C3D6D4D4   *.*** GOOD AFTERNOON**  INTERCOMM*
000032   4CC9E240 C3D3D6E2 C5C47A40 4040F0F7    60F1F860 F8F94040 F1F24BF5 F61526     * IS CLOSED:   07-18-89  12.56.. *
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .U/CCE4  .J/C0D1        21    89.199    12.56.376C   CNT01   0000   00            0   FA  00   50
-------------------------------------------------------------------------------------------------------------------------
   108       0    02  .U/00E4  .J/00D1        22    89.199    12.56.3805   TEST1   0000   00            0   01  00   50
000000   FFC2002D 013C5C5C 5C4CC7D6 D6C44CC1    C6E3C5D9 D5D6D6D5 5C5C4040 C9D5E3C5   *......*** GOOD AFTERNOON**  INTE*
000032   D9C3D6D4 D440C9E2 40C3D3D6 E2C5C47A    404040F0 F760F1F8 60F8F940 40F1F24B   *RCOMM IS CLOSED:   07-18-89  12.*
000064   F5F6                                                                        *56                                 *
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .U/00E4  .J/00D1        22    89.199    12.56.3805   TEST1   0000   00            0   30  00   5C
-------------------------------------------------------------------------------------------------------------------------
   1C3       1    02  .U/00E4  .U/C0E4        22    89.199    12.56.3808   TEST1   0000   00            0   40  00   50
0000C0   5C5C5C40 C7D6D6C4 40C1C6E3 C5D9D5D6    D6D55C5C 4040C9D5 E3C5D9C3 D6D4D440   **** GOOD AFTERNOON**  INTERCOMM *
000032   C9E240C3 D3D6E2C5 C47A4040 40F0F760    F1F860F8 F94040F1 F24BF5F6 37         *IS CLOSED:   07-18-89  12.56.   *
-------------------------------------------------------------------------------------------------------------------------
    42       1    02  .U/CCE4  .J/CCD1        22    89.199    12.56.3808   TEST1   0000   00            0   FA  00   50
-------------------------------------------------------------------------------------------------------------------------
    47       0    02  .J/00D1  ../0000        23    89.199    12.56.3856   CNT01   0000   00            0   01  00   FC
0C00C0   C5D9C3C4 37                                                                  *NRCD.                              *
-------------------------------------------------------------------------------------------------------------------------

MSGLEN  THREAD   QPR   RSC      SSC          MMN     DATE        TIME        TID     FLGS  USR          BMN  LOG BLK  VMI
```

Figure 63. Test Mode Execution Log Printout (Page 5 of 6)



```
                                                
```

Appendix A

ASSEMBLER LANGUAGE JCL PROCEDURES

The following JCL procedures, which use Assembler H, are supplied on the Intercomm release library, SYMREL. Check with your System Manager before using them to ensure they reside on your installation's system procedure library (SYS1.PROCLIB) and to verify parameters to code. When appropriate, SYSLIB references Intercomm libraries.

ASMPC:	Assemble BAL source code
Example:	// EXEC ASMPC,Q=TEST,NAME=BALPROG
ASMPCL:	Assemble BAL source code and linkedit it to produce a load module. (The linkedit step, PARM overrides AMODE=31 and RMODE=ANY, cause the program to be loaded above the 16meg line).
Example:	// EXEC ASMPCL,Q=TEST,NAME=BALPROG,LMOD=BALPROG, // PARM.LKED='LIST,XREF,LET,NCAL,AMODE=31,RMODE=ANY'
LIBEASM:	IEBUPDTE step, followed by Assembly of updated source code. Add: //LIB.SYSIN to specify IEBUPDTE control and change cards.
Example:	// EXEC LIBEASM,Q=TEST,NAME=BALPROG
LIBELINK:	IEBUPDTE step, followed by same JCL as ASMPCL.
Note:	LKED override parms for 31 Amode, as in ASMPCL, may be used. Add //LIB.SYSIN statement to specify IEBUPDTE input.

Figure A-1. Intercomm-supplied Assembler JCL Procedures

Refer to the Intercomm Operating Reference Manual for further details on JCL parameter requirements, and other BAL procedures for Assembler F.

For Assembler Language programs eligible for loading above the 16meg line under MVS/XA or ESA, add the following to procedures with a linkedit step:

```
//LKED.SYSIN  DD *
       INCLUDE  SYSLIB(INTLOAD)
       ENTRY    user-program-name
```

For those to be dynamically loaded below the 16meg line, or if executing under MVS/370, an include for INTLOAD will save dynamic linkedit time at Intercomm Startup for direct calls to system routines, and for system macro-generated calls.

Appendix B

DSECTS FOR ASSEMBLER LANGUAGE PROGRAMS

The following lists members in the Intercomm SYMREL source library which contain source statement code for Intercomm Dsects which can be inserted in a BAL program simply by coding COPY member-name at the desired source line, or by coding the listed macro. Generating Dsects for the MCB and EXTDSCT areas are listed for debugging purposes only. SYMREL must be named in the DD statement concatenation for the SYSLIB data set for assemblies (automatic if Intercomm-supplied Assembler procedures used). For the Dsect label, the word user means the programmer must choose a name and prefix the COPY or macro statement with a labeled DSECT statement.

NOTE: except for the user-generated MMU Symbolic Map area, and selected Message Header fields (see Figure 7), none of the other areas (field values, bit settings, etc.) may be changed by a user program during Intercomm execution. System commands are available to change some tables dynamically as described in System Control Commands. Dsects for all table areas are listed in an appendix of the Operating Reference Manual.

Area	Generated by	Dsect label	Description
Message Header	COPY MSGHDRC MSGHDR macro LINKAGE macro	user user MSGHEAD	Message Header fields (see Figure 7)
Parameter List	LINKAGE macro PARMLIST macro	PARMLIST user	3-word parameter list passed to subsystem
SCT entry	LINKAGE macro COPY SCTLISTC	SCTLIST SCTLIST	SYCTTBL macro-generated fields in SCT
SCT extension	automatic - after SCT entry	SCTEXTLT	SYCTTBL (SCT) extension for dynamic load
SPA	LINKAGE macro SPALIST macro	SPALIST user	SPA Csect fields followed by USERSPA (if any)
SPAEXT	automatic - after SPA, USERSPA	SPAEXT	SPAEXT (SPA extension) Csect fields
MMU map area	COPY user (MMU SYMGEN JCL proc)	user	MMU user-generated symbolic map fields
MCB area	COPY MCBDSECT	MCB	MMU mapping control block (MCB) fields
EXTDSCT area	IXFDSCTA macro	DSCT	File Handler EXTDSCT fields

Figure B-1 Intercomm Dsects for Assembler Programs

PAGES 201-204 INTENTIONALLY MISSING

Appendix C

INTERCOMM TABLE SUMMARY

Basic tables are included in the Intercomm release library (SYMREL) and must be modified (added to) for each installation. An asterisk (*) indicates optional tables which may be generated individually at each installation according to application program requirements. A complete list is provided in an Appendix of the Operating Reference Manual.

TABLE or CSECT Name	Description	Created by	SYMREL and MODREL Member Name
BROADCST	*Output Broadcast Table	BCGROUP macro	PMIBROAD
BTAMSCTS	Front End Queue Table (BTAM/TCAM/GFE only)	SYCTTBL macro	BTAMSCTS
BTVRBTB	Front End Verb Table	BTVERB macro	BTVRBTB
(User-name)	Front End Network Configuration Table	LINEGRP, BLINE BTERM macros, etc. VCT, LUNIT, LCOMP macros, etc.	FENETWRK (BTSAMP) (VTSAMP)
CHNGTB	*Change Table for Change/Display Utilities	DC's	None
File Description Records (DESnnnnn)	*File Descriptions Data Set (DES000); generated by file load utility PMIEXLD (for Change/ Display Utility)	FDHDR, FDETL macros	None
IXFDSCTn	File Handler Data Set Control Table	IXFDSCTA macro	IXFDSCT1 (50 DDs) IXFDSCT2 (100 DDs) IXFDSCT3 (200 DDs)

Figure C-1. Table Names and Associated Macro Instructions (Page 1 of 2)

TABLE or CSECT Name	Description	Created by	SYMREL and MODREL Member Name
KEYTABLE	*Display Utility Key Transformation Routing Table	DC's	None
PADDTBLE	*Edit Utility Pad Table	PADD macro	PADDTBLE
PAGETBL	*Page Facility Table	PAGETBL macro	PAGETBLE
PMIALTRP	*Output Utility Alternate Format Table	PMIALTRN macro	None
PMIDEVTB	Back End Device Table	DEVICE macro	PMIDEVTB
PMIFILET	*Change/Display File Table	GENFTBLE macro	PMIFILET
PMIRCNTB	*Output Utility Format Table	CSECT	PMIRCNTB
		REPORT, LINE ITEM macros	RPTnnnnn
		PMISTOP macro	PMIRCEND
PMIRPTAB	*Output Utility Company/ Report/Terminal Table	DC's	None
PMISTATB	Back End Station Table	STATION macro	PMISTATB
PTRNTBLE	*Display Utility Symbol Edit Pattern Table	PATRN macro	None
REENTSBS	Subroutine Entries List	SUBMODS macro	REENTSBS
REPTAPE	*Output Utility Batch Report Table	DC's	None
SPA/SPAEXT	System Parameter Table (SPA)	SPALIST macro	INTSPA
SCT	Subsystem Control Table (SCT)	SYCTTBL macro	INTSCT
VERBTBL	*Edit Control Table	VERBGEN, VERB, PARM, PMIELIN macros	PMIVERBS

Figure C-1. Table Names and Associated Macro Instructions (Page 2 of 2)

Component Name	Tables Used
Change/Display Utility	CHNGTB File Description Records KEYTABLE PMIFILET PTRNTBLE
Edit Utility	PADDTBLE PMIFILET PMIVERBS PMIDEVTB PMISTATB
File Handler	IXFDSCTn FAR statements
Front/End TP Interface	BTVRBTB Front End Network Table BTAMSCTS
Message Mapping Utilities	MMUVTBL LOGCHARS PMIDEVTB PMISTATB User-coded Maps
Monitor	REENTSBS INTSPA INTSCT BROADCST
Output Utility	PMIALTRP PMIDEVTB PMIFILET PMIRCNTB PMIRPTAB PMISTATB REPTAPE RPTnnnnn (user-coded OFTs)
Page Facility	PAGETBL

Figure C-2. Components and Associated Table Names

Appendix D

SPA AND SPAEXT FIELD NAMES FOR ROUTINE ENTRY POINTS

The following tables list the names of fields in the Intercomm SPA and SPAEXT Csects of the System Parameter List which are labels of Vcons containing service routine entry point names. The field SPAEXTAD in the SPA Csect contains the address of the SPAEXT Csect. Addressability to SPAEXT must be established (see Chapter 9) before referencing any field whose label begins with the letters SEX (see also the SPAEXT parameters of the LINKAGE and SUBLINK macros in Basic System Macros). Entry points for macro-called routines are listed under the related macro description in the Macros manual. Table addresses and other fields of interest to systems programmers writing user exits may be found by studying the SPALIST macro-generated Dsect listing.

D.1 FIELDS IN THE SPA

SPAPEDT	DC	V(EDITCTRL)	Pointer to Edit routine
SPAPMCR	DC	V(MSGCOL)	Pointer to Message Collection
SPAFECRL	DC	V(FECMRLSE)	Address of FECM RLSE routine
SPAFINDB	DC	V(PMIFINDB)	Ptr to routine to find Item Code
SPAFECFB	DC	V(FECMFDBK)	Address of FECM FDBK routine
SPADLTDB	DC	V(PMIDLTDB)	Ptr to routine to Add/Delete field
SPASELCT	DC	V(SELECT)	File Handler SELECT routine
SPARELES	DC	V(RELEASE)	File Handler RELEASE routine
SPAWRITE	DC	V(WRITE)	File Handler WRITE routine
SPAREAD	DC	V(READ)	File Handler READ routine
SPAWHOIT	DC	V(IJKWHOIT)	Csect, etc. name lookup routine
SPACONVR	DC	V(CONVERSE)	Conversational Enviroment saving
SPASORT	DC	V(INTSORT)	In-core Table Sort routine (Rel 10)
SPALOGP	DC	V(LOGPUT)	Logging routine

D.2 FIELDS IN THE SPAEXT

```
SEXGET    DC    V(GET)        File Handler GET entry
SEXPUT    DC    V(PUT)        File Handler PUT entry
SEXLOCAT  DC    V(LOCATE)     File Handler LOCATE entry
SEXRELEX  DC    V(RELEX)      File Handler RELEX entry
SEXGETV   DC    V(GETV)       VSAM GET entry point
SEXPUTV   DC    V(PUTV)       VSAM PUT/ERASE entry point
SEXBINSH  DC    V(BINSRCH)    Binary search - halfword index
SEXDELAY  DC    V(IJKDELAY)   Dispatcher - timed DELAY routine
SEXPRINT  DC    V(IJKPRINT)   Line Print Routine (Rel 10 only)
SEXSECUS  DC    V(SECUSER)    ESS Operator-id Checking (Rel 10)
SEXFESND  DC    V(FESEND)     FESEND entry point
SEXFECDQ  DC    V(FECMDDQ)    Address of FECM DDQ routine
SEXCONVR  DC    V(CONVERSE)   Conversational Environment saving
SEXDBINT  DC    V(DBINT)      Data Base interface
SEXPAGE   DC    V(PAGE)       Page Facility processing
SEXDVASN  DC    V(DVASN)      Output term. control-segmented msgs
SEXGETSG  DC    V(GETSEG)     Input message segment retrieval
SEXBSRC3  DC    V(BINSRCH3)   Binary search - fullword index
SEXDQBLD  DC    V(QBUILD)     DDQ Queue Build entry point
SEXDQOPN  DC    V(QOPEN)      DDQ Queue Open entry point
SEXDQCLS  DC    V(QCLOSE)     DDQ Queue Close entry point
SEXDQRD   DC    V(QREAD)      DDQ Queue Read (normal) entry point
SEXDQWR   DC    V(QWRITE)     DDQ Queue Write (normal) entry point
SEXDQRDX  DC    V(QREADX)     DDQ Queue Read (update) entry point
SEXDQWRX  DC    V(QWRITEX)    DDQ Queue Write (update) entry point
SEXTRACE  DC    V(IJKTRACE)   WQE Trace dump processing
SEXALCTE  DC    V(ALLOCATE)   ALLOCATE routine (DFA)
SEXACCES  DC    V(ACCESS)     ACCESS routine   (DFA)
SEXSFFAD  DC    V(INTFETCH)   FETCH            (Store/Fetch)
SEXSFSAD  DC    V(INTSTORE)   STORE            (Store/Fetch)
SEXSFUAD  DC    V(INTUNSTO)   UNSTORE (delete) (Store/Fetch)
SEXMAPIN  DC    V(MAPIN)      MMU MAPIN entry point
SEXMAPOT  DC    V(MAPOUT)     MMU MAPOUT entry point
SEXMAPEN  DC    V(MAPEND)     MMU MAPEND entry point
SEXMAPPU  DC    V(MAPURGE)    MMU MAPURGE entry point
SEXMAPCL  DC    V(MAPCLR)     MMU MAPCLR entry point
SEXMAPFR  DC    V(MAPFREE)    MMU MAPFREE entry point
SEXBSRC2  DC    V(BINSRCH2)   Table Binary Search entry
SEXFNDC   DC    V(FESENDC)    FESEND-Copy message entry
```

PAGES 211-214 INTENTIONALLY MISSING

Appendix E

NONREENTRANT SUBSYSTEMS

E.1 INTRODUCTION

Nonreentrant subsystems do not use a dynamic save/work area. The LINKAGE and RTNLINK macros are not used. The programmer must save and restore the Monitor's registers using the IBM SAVE and RETURN macros, set base registers, chain a local save area, and generate Dsects as appropriate. At entry, the same parameter list is passed as to a reentrant BAL subsystem. The user must initialize registers (input message, SPA) from the parameter list, and define USING statements for the associated Dsects. As illustrated in Figure E-1, both constant and variable data areas are defined in the program. Therefore, nonreentrant subsystems are always single-threaded; each input message is processed to completion and the subsystem returns to the Monitor before a new message can be processed by the subsystem. On the SYCTTBL macro for the subsystem, MNCL=1 and LANG=NBAL must be coded. Obviously, if messages might be input to the subsystem from more than one terminal concurrently, response time for each message queued after the first will be increasingly slower. Thus, it is recommended that Assembler Language subsystems always be coded as reentrant.

To call a user subroutine, the MODCNTRL macro may not be used. Instead, the subsystem calls the routine directly, and the user subroutines must be resident in the Intercomm load module or in the same overlay segment as the subsystem; they are not defined in REENTSBS. CONVERSE may not be called from a nonreentrant subsystem. The Intercomm environment for a nonreentrant subsystem is illustrated in Figure E-2.

The STORAGE macro must be used to acquire an area for a message passed to MSGCOL. A hard-coded area in the program may be used if FESENDC is called for an output terminal message. Giving up control to the Monitor within the program via the INTENQ, DISPATCH, INTWAIT or SUBTASK macros is not recommended. Before returning control to the Monitor, a STORFREE of the input message is still required.

Nonreentrant programs are not eligible for loading above the 16meg line under XA.

```
SUBSYSYY CSECT
         REGS
*SUBSYSTEM ENTRY
         SAVE      ...
          :         set base, parms, chain save areas
          :
*MESSAGE PROCESSING LOGIC
          :
          :
*GET CORE FOR OUTPUT MESSAGE
         STORAGE   ...,RENT=NO
          :         Build Output Message
          :
*PASS THE MESSAGE TO THE FRONT END VIA FESEND, OR
*QUEUE THE OUTPUT MSG VIA MSGCOL
         CALL      ...,VL,MF=(E,PARMSAVE)
          :
          :
*FREE THE INPUT MESSAGE
         STORFREE  ...
          :
          :
*SUBSYSTEM EXIT
         L         R13,4(R13)    restore caller's R13
         RETURN    ...,RC=(r)    register r contains return code
*
REGSAVE  DS        18F
COREADDR DS        F
PARMSAVE DS        5F
SUBSYSWK DS
          :         define subsystem constants and variables
          :
*
INMSG    DSECT
INHDR    DS         CL42
INTEXT   DS
          :         define input text format
          :
INLEN    EQU       *-INHDR
*
OUTMSG   DSECT
OUTHDR   DS        0CL42
         COPY      MSGHDRC
OUTTEXT  DS
          :         define output text format
          :
OUTLEN   EQU       *-OUTHDR
SUBSYSYY CSECT
         END
```

Figure E-1. Nonreentrant Assembler Subsystem Structure

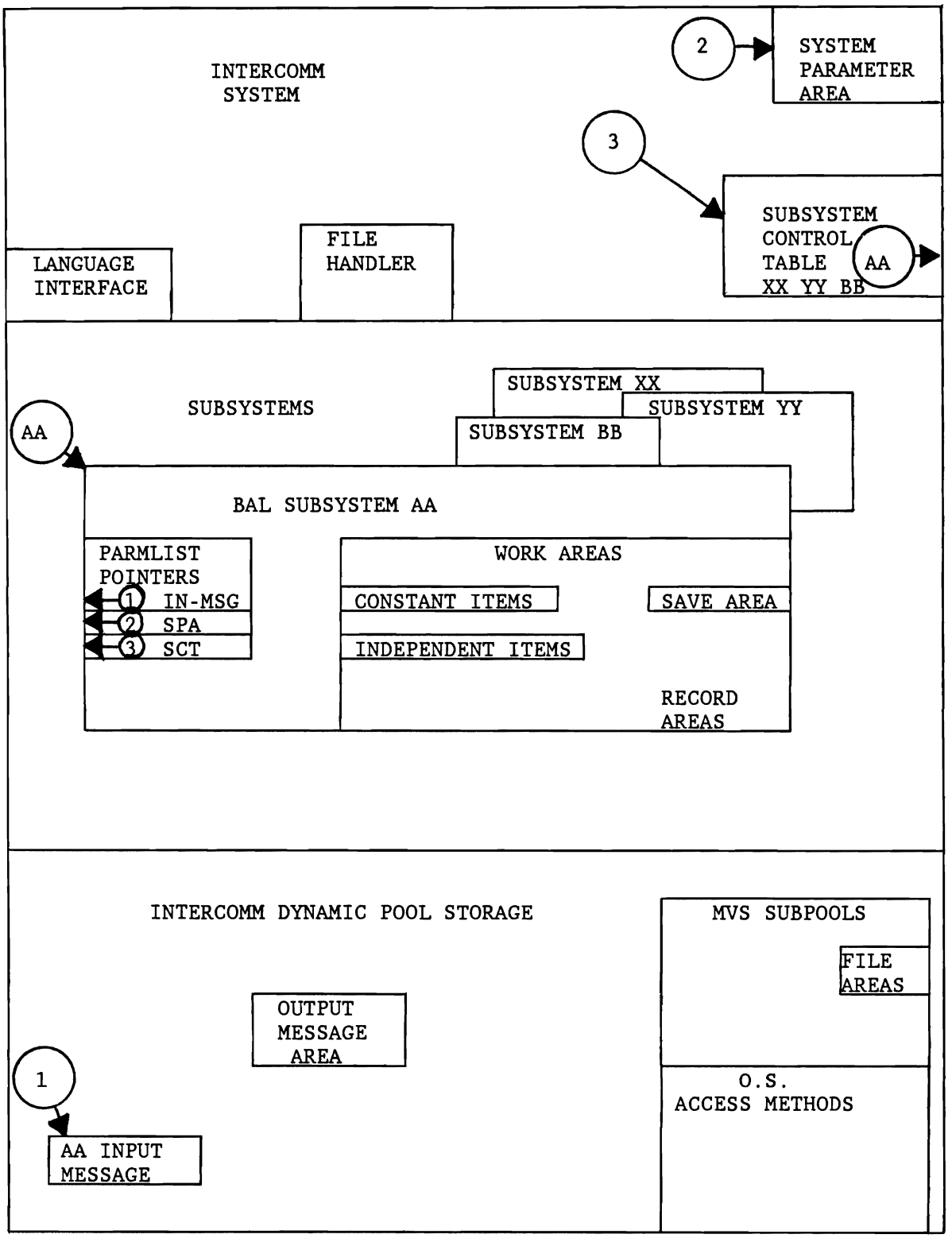


Figure E-2. Nonreentrant Application Program Environment

PAGES 218-220 INTENTIONALLY MISSING

	Page
ACCESS function (DFA)	110.3,210
ADABAS data base	15,52
ALLOCATE function (DFA)	110.3,210
Alternate Format Table	206-207
Alternate Index (VSAM)	71
Alternate Path processing (VSAM)	71
AMODE linkedit parameter	46.2,197
AMP parameter (VSAM JCL)	67,74
Application processing functions	7
Application Programs. See Subsystems.	
ASMLOGCH copy member	123
ASMPC procedure	197
ASMPCL procedure	197
AUTOLOK feature	94
Back End--defined	11
Back End Device Table. See Device Table.	
Back End Station Table. See Station Table.	
Backout-on-the-Fly	6,34
Batch execution	1,3,30
Batch Report Table	206-207
BCGROUP macro	205
BDAM access method	
--DCB	64
--DD statement parameters	52
--and exclusive control	57
--and File Handler Option Codes	65
--and File Handler Parameters	54
--and File Handler return codes	66
--sample inquiry subsystem	123
--service routines	51,64
BDW. See Block Descriptor Word.	
Binary table search	104
BINSRCH	104-105,210
BINSRCH2	104-105,210
BINSRCH3	104-105,210
BISAM access method	
--DD statement parameters	52
--and exclusive control	57
--and ISAM/VSAM compatibility	74
--processing	62
--return codes	63
--service routines	51,62
BITSECT table	46.3
BLDL parameter (SUBMODS macro)	46.3
BLDL parameter (SYCTTBL macro)	46.2
BLHOT module	29
BLINE macro	205
Block Descriptor Word	61
Broadcast Group	20,75
Broadcast Table	205,207
BROADCST. See Broadcast Table.	
BSAM access method	51-52,57,60
BTAM	11
--simulator	123,129-131
BTAMSCTS. See Front End Queue Table.	
BTAMSIM module	131
BTERM macro	205
BTSAMP table	205
BTVERB macro	
--and BTVRBTB	17,18
--and editing	35
--and priority	21
--and sample subsystem	130,132,168
--and subsystem codes	18
--table summary	205
BTVRBTB. See Front End Verb Table.	
CALL macro	10
--and MVS/XA loading	46.2
CALLIF macro	111
CALLOVLY macro	46.2,111
Cancelled messages	34,46.1
CATCH macro	113,114,120
Change/Display Utility	
--described	16
--and formatting required, fixed text messages	76,167
--and message switching	46.1
--and sample subsystem	167
--tables used by	207
Change Table	167,180,205,207
Checkpointing	6,28
CHNGTB. See Change Table.	
Closing files	51,52,57
Commands (Intercomm)	16
Company/Report/Terminal Table	206-207
Conversational subsystem. See Subsystems, conversational.	
Conversational time-out (Front End) and FESEND	99
CONVERSE	
--automatic time-out	90-91
--calling format	90
--and conversation implementation options	81
--described	88
--and LINKAGE macro	90-91

	Page
--and nonreentrant subsystems	215
--return codes	91
--and SPA VCONs	209
--and SPAEXT VCONs	210
--subsystem design	90
--subsystem logic using	89
CREATEGF utility	64
CREATSIM utility	129-130
--JCL for	134-135
Data Base interface	52,110.3
Data field search routines	105
Data Set Control Table	17
Data Set Name Sharing (VSAM)	67,71
Data strings	15,84
Dataspeed 40 terminal	21
DBINT function	110.3,210
DBMS support	15,24,34,110.3
DDNFIND macro	111
DDQ. See Dynamic Data Queuing.	
Debugging program problems	129
DES000 data set	167,205
Device control characters	49
DEVICE macro	206
Device Table	17,98,206-207
DFA. See Dynamic File Allocation.	
DISPATCH macro	46.2,113-119,215
Dispatcher	12
--and IJKDELAY	110,110.1
--and IJKPRINT	110
--and IJKTRACE	110,110.1
DL/I data base	15,52
DSCT. See Data Set Control Table.	
Dsects	39-40,44,83,215
--generation of	199-200
DSN option. See Data Set Name Sharing.	
DVASN service routine	76,109,210
Dumps	129
Dynamic Data Queuing	
--and conversational subsystem	81
--described	15
--and Front End Data Queuing	100
--and Message Mapping Utilities	47
--and segmented input messages	22,30
--and SPAEXT VCONs	210
--subroutine entry names	110.3,210
--types of queues	86
--use of	30,86-87
Dynamic File Allocation	15,30,110.3,210
Dynamic file backout	6,34
Dynamic loaded subsystems	18,131,168
--and dynamic linkedit	30,46.2
--and SCT Extension	200
DYNLLOAD module	46.3

	Page
ECT. See Edit Control Table.	
EDIT. See Edit Utility.	
EDIT parameter (BTVERB macro)	35,37
Edit Control Table	
--associated components	207
--described	49
--macros	206
--and message flow using EDIT and OUTPUT	24
--and required fields	50
--and subsystem testing	123,167
EDITCTRL edit entry	49,209
Edit subroutines	49
Edit Utility	
--concepts	14,49
--and CONVERSE	91
--function	14
--message flow using	24-25
--and message switching	46.1
--and MSGHUSR field	21
--processing results	49-50
--and sample subsystem	123,167
--and segmented input	108
--subsystem logic using	37-38
--and subsystem testing	123,167,180
--tables used by	17,207
--and terminal tables	17
--using	49-50
--and verb message identifier (MSGHVMI)	35,38,46.1
Entry sequenced data sets	54,67
Error messages	
--and calls to service routines	55
--and conversational subsystems	79
--and debugging	129
--and Edit Utility	50
ESA (IBM). See MVS/XA.	
ESDS. See Entry sequenced data sets.	
ESS. See Extended Security System.	
Exclusive Control	
--and File Handler Control Word	53
--for non-VSAM files	57-59
--processing (file/record)	58
--releasing (file/record)	59
--and resource control	113
--and Store/Fetch	85
--for VSAM files	70
EXEC statement parameter	131,168
EXMVE macro	111
EXSS macro	111
EXTDSCT. See External DSCT.	

Page

EXTRT macro 111
Extended Security System 15,29,112
External DSCT 53-54,56-57,59
--Dsect generation 199-200

FAR. See File Attribute Record.
FDETL macro 205
FDHDR macro 205
FDR. See File Description Records.
FECM. 98
See Front End Control Messages.
FECMDDQ. See Front End Data Queuing.
FECMFBK. See Front End Feedback Messages.
FECMRLSE. See Front End Queue Release.
Feedback codes, VSAM 70,73
FENETWRK table 136,205
FESEND
--calling format 98
--described 98-99
--and conversational time-out 99
--and FESENDC entry point 94,98
--and Front End Control Messages 100
--and Front End Feedback Messages 101
--and Front End Queue Release 99,103
--function 14,35
--and INTERLOG 27-29
--and message flow using EDIT and OUTPUT 24-25
--and message flow using MMU 22-23
--and message header altering 21
--and Message Mapping Utilities 22-23,39,47
--and option codes 99
--and Output Utility 24-25,75
--return codes 99
--and SPAEXT VCONs 210
--and storage control 42
--and subsystem coding 39
--and subsystem logic using EDIT and OUTPUT 38
--and subsystem logic using MMU 36
--and VTAM 99,103
FESENDC entry point 94,98-99,210,215
FETCH function 84-85
FHCW. See File Handler Control Word.

Page

FILE command 56,57
File Attribute Record 17,51,56,61,62,67,71,207
File Description Records 76,167,205,207
--illustrated 182
File Handler
--access methods supported 52
--and batch execution 30
--and closing a file 52,57
--described 12,51-52
--and direct access files 64-66
--error checking 55
--exclusive control
--non-VSAM files 57-59
--VSAM files 59
--general concepts 51-52
--and indexed sequential files 62,63
--and ISAM/VSAM compatibility 74
--and message flow using EDIT and OUTPUT 24-25
--and message flow using MMU 22-23
--and MVS/XA 46.2,56
--above 16-meg line restrictions 54
--parameters 54
--return codes 55-56,59-60,63,66,73
--and sample inquiry subsystem 123
--SELECT and RELEASE functions 56
--and sequential access files 60-61
--service routines 51,53-55,215
--and SPA VCONs 209
--and SPAEXT VCONs 210
--subsystem processing 52-53
--tables used 205,207
--and undefined record format 61
--and variable length records 61
--and VSAM 59,67-73
File Handler Control Word
--and closing a file 57
--described 53-54
--and direct access files 64-65
--and exclusive control 57
--and indexed sequential files 62
--and ISAM/VSAM compatibility 74
--return codes 55
--and undefined records 61
--and VSAM files 69-71
File Handler Data Set Control Table 17,205
File Recovery feature 6,51,97
File Table 206-207
Flushed messages 29

Page

Format Description Record.
  See File Description Records.
Front End
 --defined 11
 --and system commands 98
 --and TP Queuing Interface 12
Front End Control Messages 100-103
 --and FESEND 98
Front End Data Queuing 30,100-101,210
Front End Feedback Messages 100-102,209
Front End Network Configuration
  Table 17,205,207
Front End Queue Release 99,100,103,209
Front End Queue Table 205,207
Front End Verb Table (BTVRBTB)
 --and AUTOLOK 94
 --creation of 17,205,207
 --and CONVERSE 90-91
 --and editing 37,49
 --function 17-18
 --and message processing (VMI) 35
 --and USRBTVRB 18,130,132,168

GENFTBLE macro 206
GET service routine
 --calling format 60,62
 --and exclusive control 57-59
 --function 51
 --and indexed sequential files 62
 --and ISAM/VSAM compatibilty 70,74
 --and sequential access
    files 60-61
 --and SPAEXT VCONs 210
 --and subsystem processing 52
GETDATE macro 111
GETSEG service routine 108,210
GETSPA macro 111
GETV service routine
 --calling format 67-68,72
 --and exclusive control 70
 --and FHCW reason codes 70
 --function 51
 --and SPAEXT VCONs 210
 --and subsystem processing 52
 --and VSAM files 67-68
 --and VSAM processing options 69-71
HEXCON macro 110.3,111
HPRTY parameter (BTVERB macro) 21

IAM access method 52,54
IBM 2740 Terminal 21
IBM 3270 Display Station 21,49,131
 --and simulation testing 130

Page

ICOMLINK macro 136,185
IDMS data base 15,52
IJKDELAY service routine 110.1,210
IJKPRINT service routine 110,210
IJKTRACE service routine 110.1,210
IJKWHOIT service routine 110.3,209
Indicative dumps 129
INTDEQ macro 113,115,121
INTENQ macro 113,115,121,215
INTERLOG file
  26,28-29,97,161-166,191-196
 --JCL for 138
INTFETCH function 110.3,210
INTLOAD
 --and mode switching 46.2
 --and MVS/XA 46.3,110.4,197
INTPOST macro 113,115,197
INTSCT module 18,206
 --and USRSCTS 130,132,167,180
INTSORT Facility 110.2,209
INTSPA module 82,206
INTSTORE function 110.3,210
INTTIME macro 111
INTUNSTO function 110.3,210
INTWAIT macro 113,115,117-118,215
ISAM (see also BISAM and QISAM)
 --access by File Handler 52
 --File Handler processing 62
 --File Handler return codes 63
 --File Handler service routine
    parameters 54
 --and VSAM 67,70,74
ITEM macro 206
IXFCHKPT module 28
IXFDSCTA macro 200,205
IXFDSCT1/2/3/n 205,207
IXFLOG (File Recovery logging) 28

JCL
 --Assembler procedures 197
 --CREATSIM utility 134-135
 --DD statement for File
    Handler 51-52
 --for direct access files 64-66
 --for indexed sequential files 62
 --for ISAM/VSAM compatibility 74
 --sample simulation Mode 136-139
 --sample Test Mode 185-187
 --for VSAM files 67,187
JOBCAT DD statement (VSAM) 131
Journaling. See Logging.

	Page
Key Table	206-207
Key sequenced data sets	54,67
KEYCREAT utility	64
KSDS. See Key sequenced data sets.	
LANG parameter (SYCTTBL macro)	18,46.2,215
LAYOUT macro	110.3,111
LCOMP macro	205
LIBEASM procedure	197
LIBELINK procedure	132,180,197
LIMCT parameter (JCL)	52,64
LINE macro	206
Line control	6-7
LINEGRP macro	205
LINKAGE macro	
--and coding techniques	42-43
--and CONVERSE	90-91
--and Dsects	39-40,83,200
--and dynamic storage	40
--and MVS/XA	46.2,110.4
--restrictions	46.3
--and nonreentrant subsystems	215
--and output messages	42
--and subsystem linkage	38
Linkedit	130,136-137,168,185-186,197
--and MVS/XA	46.2-46.3,197
LKEDT procedure	137,186
LNAME parameter (SUBMODS macro)	46.3
LOAD command	30
LOADNAM parameter (SYCTTBL macro)	18,46.2
LOCATE entry point	210
LOCK command	93-94
Log Analysis utility	21,26
Log codes	26-29,46.1,97
Logging. See INTERLOG file.	
LOGCHARS table	207
LOGPROC module	28
LOGPRINT utility	21,26
--sample JCL for	138
LOGPUT module	
--calling format	97
--function	14
--and INTERLOG entries	28-29
--and SPA VCONs	209
--and user entries on INTERLOG	26,97
LOGTROUT table	97
LSR (pools) option (VSAM)	67,71
LTORG statement	45
LUNIT macro	205
Macros (Intercomm)	
--for Dsects	199-200
--listed	111-113
--for Tables	205-206
--and MVS/XA loading	46.2-46.3
MANAGER module	46.3
Map Control Block	199-200
MAPCLR function	110.3,210
MAPEND function	34,36,48,102,110.3,210
MAPFREE function	110.3,210
MAPIN function	36,42,48,110.3,210
MAPOUT function	36,48,110.3,210
MAPURGE function	110.3,210
MCB. See Map Control Block.	
Message Collection	
--calling format	96
--function	14
--and INTERLOG	28-29
--log codes	27
--and message flow using EDIT and OUTPUT	24-25
--and message flow using MMU	22-23
--and message switching	96
--and MSGHMMN	20
--and nonreentrant subsystems	215
--and MSGHUSR	21
--return codes	34,96
--and SPA VCONs	209
--and storage control	42
Message header	
--changing	35
--and CONVERSE	90
--described	19-21
--Dsect generation	40,200
--and Message Collection	96
--and message flow using EDIT and OUTPUT	24-25
--and message flow using MMU	22-23
--and message processing logic	35
--and message routing	18
--and output messages	35,77
--specifications for the Output Utility	77
--and subsystem structure	31,44
--and Test Mode input	183
--and user log entries	97

	Page
Message Mapping Utilities	
--described	14,47
--and Dynamic Data Queuing	39
--and Front End Data Queuing	100-101
--function	14
--and log codes	27
--maps for sample subsystem	133
--message flow using	22-23
--and messages to Front End	39,98
--and MSGHQPR	22
--and MSGHVMI	44
--and MVS/XA	46.2
--and Page Facility	39
--processing	22,30,47-48
--and sample inquiry subsystem	123,130
--service routines	110.3,210
--and SPAEXT VCONs	210
--and subsystem logic	35-36,48
--and subsystem testing	130,131
--symbolic maps	47,200
--tables used by	17,207
Message Processing	7
Message Processing Control	6-7
Message switching	46.1,96
MMU. See Message Mapping Utilities.	
MMUC command	130,141
MMUVTBL table	207
MNCL parameter (SYCTTBL macro)	19,131,168,215
MODCNTRL macro	
--ACTION parameter	46.3
--and MVS/XA loading	46.3,116
--and nonreentrant subsystems	215
--and system control	113,116
--usage	121
--and XASWITCH macro	46.3,116
Model 204 data base	15,52
Monitor Control Functions	7
MRINTER module	28
MRQMNGR module	28-29
MSGAC module	29
MSGCOL. See Message Collection.	
MSGHADDR field	20
MSGHBLK field	20
MSGHBMN field	20,26,183
MSGHCON field	20
MSGHDAT field	20,26,97
MSGHDR Dsect	40,200
MSGHDR macro	111,200
MSGHDRC copy member	44,200
MSGHFLGS field	20
MSGHLEN field	
--described	20
--and Front End Control Messages	102-103
--and output messages	44
--and STORFREE macro	41
--and user log entries	97
MSGHLOG field	20,26,97
MSGHMMN field	20,26
MSGHQPR field	20
--described	22
--and segmented input	76,98,108
MSGHRETN field	20,34
MSGHRSC field	
--and assigning verb to terminal	94
--described	20
--and Message Collection	96
--and message routing	18,35
--and message switching	96
--and output messages	44,77
--and subsystem logic using EDIT and OUTPUT	37
--and Test Mode	183
MSGHRSCH field	
--and assigning verb to terminal	94
--described	20
--and Message Collection	96
--and message routing	18,35
--and message switching	96
--and output messages	44,77
--and subsystem logic using EDIT and OUTPUT	37
--and Test Mode	183
MSGHSSC field	20,44
MSGHSSCH field	20,44
MSGHTID field	
--described	20
--and FESEND	98-99
--and Front End Data Queuing	101
--and Front End Queue Release	103
--and message switching	46.1
--and Output Utility	75
--and segmented output	109
--and Store/Fetch facility	84
--and subsystem coding	44
--and Test Mode	183

	Page
MSGHTIM field	20,26,97
MSGHUSR field	20,21
MSGHVMI field	
--described	20,22
--and Edit Utility	35,38,41
--and FESEND	21,98
--and Front End Control Messages	100
--and Front End data queuing	100
--and input messages	35
--and message processing	35,41
--and message switching	46.1
--and output messages	44,75-77
--and subsystem logic using EDIT and OUTPUT	37
--and Test Mode	183
--values	35
Multiregion System	
--and batch interface	30
--and testing	129
Multithread processing	
--described	4-5
--and exclusive control	57
--and message flow using MMU	22
--and reentrant subsystems	16
--and subsystem testing	131,168
MVS/XA	
--and CONVERSE	90
--extended storage loading requirements	46.2-46.3
--and FECMDDQ	101
--and File Handler calls	54-57
--and INTLOAD	46.2,110.4,197
--and MODCNTRL macro	46.2,116
--and nonreentrant programs	215
--and subroutines interfaces	46.2
--and subsystem coding requirements and restrictions	46.3
--and subsystem loading	45
--and SUBTASK macro	46.2,117
--and XASWITCH macro	46.3,112,116
Network Table	17,205,207
Nonreentrant subsystems	16,215-217
OFT. See Output Format Table.	
OPEN option	51
Opening files	51-52
OTQUEUE user exit (VTAM)	21
Output Format Number	168
Output Format Table	
--associated components	207
--creation of	206
--described	76
--and message flow using EDIT and OUTPUT	24
--and subsystem testing	168,181-182
Output Utility	
--and CONVERSE	90
--described	16,75
--and Front End feedback messages	101
--function	14
--and INTERLOG	26-27
--message flow using	24-25
--message header specifications for	77
--and message switching	30,46.1
--and MSGHQPR	22
--and MSGHVMI	41,77
--processing	24,75-76
--and sample inquiry subsystem	123,167
--and segmented output	109
--subsystem logic using	37-38
--and subsystem testing	168
--tables used by	17,207
Pad character table	206-207
PADD macro	206
PADDTBLE. See Pad character table.	
PAGE entry point	210
Page Facility	15-16,47,110.3,206,207,210
PAGETBL macro	206
PAGETBL table	206-207
PARM macro	206
PARMLIST Dsect	40,200
PARMLIST macro	200
PASS macro	113,116
PATRN macro	206
Pattern Table	206-207
PMIALTRN macro	206
PMIALTRP. See Alternate Format Table.	
PMIBROAD. See Broadcast Table.	
PMICANC. See USRCANC.	
PMIDEVTB. See Device Table.	
PMIDLTDB search routine	105-106,209
PMIELIN macro	206
PMIDVASN	109
PMIEXLD Utility	167,205
PMIFILET. See File Table.	
PMIFINDB search routine	105,107,209

Page

PMIRCEND. See Output Format Table.
PMIRCNTB. See Output Format Table.
PMIRPTAB. See Company/Report/ Terminal Table.
PMISNAP macro 112,116
PMISPA module (Release 8 only) 18
PMISTATB. See Station Table.
PMISTOP DD statement 131
PMISTOP macro 206
PMITEST module 167,168
PMIVERBS. See Edit Control Table.
PMIWTO macro 112,116
PMIWTOR macro 112,117
PTRNTBLE. See Pattern Table.
PUT service routine
 --calling format 60,62
 --and exclusive control 59
 --function 51
 --and indexed sequential files 62-63
 --and sequential access files 60-61
 --and SPAEXT VCONs 210
 --and subsystem processing 52
 --and ISAM/VSAM compatibility 70,74
PUTV service routine
 --calling format 67-68,72
 --and exclusive control 70
 --and FHCW reason codes 70
 --function 51
 --and SPAEXT VCONs 210
 --and subsystem processing 52
 --and VSAM files 67-68
 --and VSAM processing options 69-70

QBUILD function 87,110.3,210
QCLOSE function 87,110.3,210
QISAM access method
 --and exclusive control 57-59
 --and File Handler service routines 51-52,62
 --and ISAM/VSAM compatibility 74
 --return codes 63
QOPEN function 87,110.3,210
QPR. See MSGHQPR.
QREAD function 87,110.3,210
QREADX function 87,110.3,210
QWRITE function 87,110.3,210
QWRITEX function 87,110.3,210
QSAM access method 51-52,57,60
Queues
 --defined 15
 --and Dynamic Data queuing 86-87
 --and Front End 11
 --illustrated 23,25
 --and INTERLOG 26
 --and Message Collection 96
 --and message switching 46.1
 --and TP interface 12
 --types of DDQs 86

RBA. See Relative byte address.
RBN. See Relative block number.
RDW. See Record Descriptor Word.
READ service routine
 --calling format 60,62,64
 --and direct access files 64-66
 --and exclusive control 57-59
 --function 51
 --and indexed sequential files 62-63
 --and ISAM/VSAM compatibility 70,74
 --and sequential access files 60-61
 --and SPA VCONs 209
 --and subsystem processing 52
READONLY option 51,71
Reason codes, VSAM 70,73
Record Descriptor Word 61
Reentrant subsystems
 --environment 33
 --example 46
 --and File Handler service routines 53-54
 --and LANG parameter (SYCTTBL macro) 18
 --and Message Collection 96
 --and MNCL parameter (SYCTTBL macro) 19,131,168
 --and MVS/XA loading 46.3
 --structure 31-32
 --vs. nonreentrant subsystems 16
REENTSBS table
 --creation of 206
 --and MODCNTRL macro 113,116
 --and MVS/XA loading 46.2,116
 --and user subroutines 46.3,215
 --and USRSUBS 130,132
REGA macro 112
REGS macro 112
Relative block number 54,64-66
 --example 123
Relative byte address 54,67-69
Relative record data set 54,67
Relative record number 54,67-70

	Page
Relative track and record number	54
RELEASE service routine	
--calling format	56
--and exclusive control	59
--function	51
--return codes	56
--and SPA VCONs	209
--and subsystem processing	52-53
--use of	56-57
--and VSAM	67,74
RELEX service routine	59,210
REPORT macro	206
REPTAPE. See Batch Report Table.	
Residency	11,92,215
RESOURC parameter (SYCTTBL macro)	19
Resource Audit and Purge	116
RESOURCE macro	19
Resource Manager	12
--and CATCH macro	114
RESTART parameter (SYCTTBL macro)	46.1
Restart/recovery	21,26,46.1
Restarted messages	46.1
Retriever	29
Return codes	
--from Binary Search	105
--from CONVERSE	91
--from Edit Utility	49
--from FECM calls	100
--from FESEND	99
--from File Handler	
--BDAM	66
--ISAM	63
--outline	55
--RELEX	59
--SELECT/RELEASE	56
--sequential access	60
--VSAM	69-70,73
--from Front End Control Message facility	100
--from GETSEG	108
--from INTSORT	110.2
--from LOGPUT	97
--from Message Collection	96
--from queuing routine	44
--from PMIDLTDB	107
--from PMIFINDB	106
--from STORAGE macro	41
--from subroutine	100,123
--from subsystem	22,31
--illustrated	46
--listed	34
RETURN macro (IBM)	
--and nonreentrant subsystems	215
REUSE parameter (SYCTTBL macro)	46.2
RLSE command	103
RMODE linkedit parameter	46.2,197
ROUND macro	41,112
RRDS. See Relative record data set.	
RRN. See Relative record number.	
RTNLINK macro	
--and coding techniques	42-43
--and dynamic storage	40
--and File Handler	52
--and MVS/XA loading	46.2
--and nonreentrant subsystems	215
--and output message area	42,98
--and SUBLINK macro	112
--and subsystem linkage	38
SAM. See System Accounting and Measurement.	
SAM parameter (SYCTTBL macro)	117
Sample subroutine (SQASMB)	123,125-127
Sample subsystems	123,167
Save area	215,217
SAVE macro (IBM)	
--and nonreentrant subsystems	215
SBA sequence (IBM 3270)	21,27,49
SBSP parameter (SYCTTBL macro)	18,46.2
SCT. See Subsystm Control Table.	
SCT Extension	200
SCTEXTLT Dsect label	200
SCTL command	110.3
SCTLIST Dsect	40,200
SCTLISTC copy member	200
SECTEST macro (ESS)	112
SECUSER routine (ESS)	15,110.3,210
Segmented messages	
--and DDQ	15,30
--input (GETSEG)	108
--and MSGHQPR	22
--output (DVASN)	109
--and Output Utility	76
SELECT service routine	
--calling format	56
--function	51
--and ISAM/VSAM compatibility	74
--return codes	56
--and SPA VCONs	209
--and subsystem processing	52-53
--use of	56
--and VSAM	67
Share options (VSAM)	70-71
SIMCRTA utility	168
Simulation Mode	
--and BTAM simulator	123,129-131
--execution JCL	138
--imput messages	134-135

	Page
--linkedit JCL	136-137
--linkedit JCL	136-137
--logging	27
--output from	139-160
SIM3270 module	129,131
--printout from	131,139-158
Single-thread processing	4,131,168,215
SNA	99
Snap 51	99
Snap, issue	112
Snaps, Test Mode	168,187-190
Sort Facility	110.2
SPA. See System Parameter List.	
SPAEXT. See System Parameter List.	
SPALIST Dsect	40,200
SPALIST macro	17,46,200,206
SSCONV macro	112
SSSTART macro	46.3,112
SSSTOP macro	46.3,112
SSTEST macro	46.3,112
STATION macro	206
Station Table	17,98,206-207
Statistics	6,26
STEPCAT DD statement (VSAM)	131
STORAGE macro	
--and coding techniques	42,43
--and DISPATCH macro	119
--and dynamic storage	40
--and function	39
--and input message area	42
--and MVS/XA restrictions	46.2
--and nonreentrant subsystems	215
--and output message area	41,98
--and output message length	41
--and PASS macro	116,120
STORE function	84-85
Store/Fetch facility	
--and conversational subsystems	81
--and CONVERSE	88
--function	15
--service routines	110.3,210
--and SPAEXT VCONs	210
--and subsystem testing	130
--use of	30,84-85
STORFREE macro	
--and coding techniques	42-43
--and DISPATCH macro	119
--and function	39
--and INTLOAD	46.2
--and MVS/XA loading	46.2
--and nonreentrant subsystems	215
--and output message area	41-42,98
STRT command	139-140
SUBLINK macro	112
--and MVS/XA loading	46.2-46.3,110.4
SUBMODS macro	46.2,130,206
Subroutines	
--calling of	10
--and dynamic linkedit	30
--and INTLOAD	46.2
--and MODCNTRL macro	46.3,116,121
--and MVS/XA loading	46.2-46.3
--sample	125-127
Subsystems	
--coding	38-46.1
--conversational	
--and CONVERSE	88-92
--design considerations	93-94
--and Dynamic Data Queuing	86-87
--general concepts	79-80
--implementation	81
--and Store/Fetch	84-85
--and USERSPA	82-83
--defined	9-10
--and dynamic linkedit	30
--entry (Csect or ENTRY name)	39
--and File Handler	52-55
--illustration of	44-46
--Intercomm-supplied	16
--and INTLOAD	46.2,197
--and message processing	35-38
--and message switching	46.1,96
--and MVS/XA	45
--requirements and restrictions	46.2
--nonreentrant	16,215-217
--and output messages	98
--reentrant	16,31-33
--return codes from	34
--sample inquiry	123,167
--structure	31-32
--testing	123,129-196
--time controlled processing	30
See also Reentrant subsystems.	
Subsystem Control Table (SCT)	
--creation of	18,206
--Dsect generation	40,200
--function	17-18
--and message collection	96
--and message flow using EDIT and OUTPUT	24
--and message flow using MMU	22
--and RESOURCE macro	19
--and subsystem entry point	39,44
--and subsystem structure	31-32
--and subsystem testing	130,167

	Page
Subsystem Controller	
--function	12
--and File Handler return codes	55
--and INTERLOG	28-29
--and message processing concepts	35
--and return codes	42
--and subsystem structure	31-34
SUBTASK macro	46.2,113,117,215
SWMODE interface routine	46.3
SYCTTBL macro	
--BLDL parameter	46.2
--and Front End Queue Table	205
--function	17
--and INTSCT	130,132,167,180
--LANG parameter	18,46.2,215
--LOADNAM parameter	18,46.2
--and Message Collection	96
--MNCL parameter	19,215
--and MVS/XA loading	46.2
--and nonreentrant subsystems	215
--parameters	18-19
--and RESOURCE macro	19
--and restarted messages	46.1
--REUSE parameter	46.2
--SAM parameter	117
--and sample inquiry subsystem	123
--and Subsystem Control Table	18,200,206
--TCTV parameter	19
SYCT400 module	29,46.3
SYMGEN proc	200
SYMREL	82,199
SYMUSR	82
SYSPRINT (SYSOUT data set)	110,116
System Accounting and Measurement (SAM)	117
System Components	11
System log. See INTERLOG file.	
System Parameter List (SPA)	
--and associated components	207
--and conversational subsystem	81-83
--creation of	206
--Dsect generation	40,200
--fields in the SPA	209
--fields in the SPAEXT	210
--function	17
--and Message Collection	96
--and MVS/XA loading	110.4
--and subsystem entry point	39
--and subsystem structure	31-32
--and system separator character	49
System Parameter Table. See System Parameter List.	
System 2K data base	15,52
Systems Operation Control	6-7
Table Sort	110.2
Tables (Intercomm)	17,205-207
TCAM	11,130
TCTV parameter (SYCTTBL macro)	19
Teletype Terminal	21
Terminal control	6-7
Test Mode	26,123,129,167
Test input messages	123,130,134-135,168,183-184
Testing Subsystems	123,129
Thread	
--defined	4
--dumps of	129
--and File Handler Control areas	53
--and MNCL	19
--and RELEASE	56
Thread number	21
Time controlled processing	30
Time-out	19,91
TOTAL data base	15,52
TP interface	12
Transaction codes/identifiers	17-18
Transactions	
--described	1-3
--sample inquiry	123
TTR. See Relative Track and Record Number.	
Undefined records	61
UNLK command	93-94
UNSTORE function	85
USAGE parameter (SUBMODS macro)	46.3
User exits	30,34
USERSPA	81-84,88
USRBTLOG module (user exit)	29
USRBTVRB	130,132,168
USRCANC module (user exit)	34,55
USRSCTS	18,130,132,167,180
USRSUBS	130,132
USRTRACK macro	112,117
USRVERBS	167,180
Variable-length record	61
VCT macro	205
VERB macro	206

Page

Verb Table. See Front End Verb Table.
VERBGEN macro 206
VERBTBL. See Edit Control Table.
VMI (Verb Message Identifier). See MSGHVMI Field.
VSAM access method
--DD statement parameters 67
--and Dynamic File Allocation 15
--FAR options for 67
--File Handler processing
--alternate index, keyed files 71
--alternate path processing 71
--call summary 72-73
--example 123
--exclusive control 59,70
--File Handler Control Word reason codes 55,70-73
--file types supported 67
--GETV and PUTV 67-68
--processing options 69-70
--File Handler Service routines 51-52,54,67-68
--ISAM/VSAM compatibility 74
--SHAREOPTIONs 70-71
VTAM 11,99,103,130
VTSAMP table 205

WRITE service routine
--calling format 60,62,64
--and direct access files 64-65
--and exclusive control 57-59
--function 51
--and indexed sequential files 62-63
--and ISAM/VSAM compatibility 70,74
--and sequential access files 60-61
--and SPA VCONs 209
--and subsystem processing 52
WRITEOVER FAR option 56
--and VSAM ESDS files 67
WTOMOD Intercomm module 116,117

XA. See MVS/XA.
XASWITCH macro 46.3,112