GY30-2029-3

OS/MFT and OS/MVT TCAM Logic

Systems

Program No. 360S-CQ-548

OS Release No. 21.0

Preface

The *Organization and Use of the TCAM Program Logic Manual* section of this book defines the audience for which this program logic manual was intended, explains how the book is organized, and suggests how the reader might best familiarize himself with its contents. In order to understand the logic of TCAM, the reader must have a general understanding of the System/360 Operating System. In addition, the following prerequisite publications are applicable:

- *OS TCAM Concepts and Facilities,* Order No. GC30-2022, to gain familiarity with the overall concepts and structure of TCAM.
- *OS TCAM Programmer's Guide,* Order No. GC30-2024, to learn how to construct and modify a TCAM MCP and a TCAM-compatible application program.

The *OS TCAM User's Guide,* Order No. GC30-2025, provides supplementary debugging information.

The *OS System Control Blocks* publication, Order No. GC28-6628, provides corequisite information on system control blocks that are used by TCAM.

Fourth Edition (July 1972)

This publication is a major revision of, and obsoltes, GY30-2029-2; it provides function support of Component Release 4 of TCAM, and maintenance support of TCAM contained in Release 21.0 of OS.

Significant new material has been added throughout, and existing material has been changed extensively; therefore, no vertical lines or bullets appear in the margins, and the manual should be reread in its entirety.

The contents of this publication are subject to change from time to time. Changes will be reflected in periodically updated editions. Before using this publication, consult the latest *System/360 SRL Newsletter,* GN20-0360, for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form is provided at the back of this publication for your comments. If the form is missing, comments may be addressed to IBM Corporation, Publications Center, Dept. E01, P. O. Box 12275, Research Triangle Park, North Carolina 27709. Comments become the property of IBM.

Summary of Changes

This complete revision of the *OS TCAM Logic* publication obsoletes the previous edition, GY30-2029-2. In this edition the method of operation text and charts are replaced by revised and improved method of operation charts and text. The *Executable TCAM Modules Microfiche Directory* has been replaced because of changes in the method of operation chart identification and for operator control and TOTE on-line test. Additional data area layouts are included for TOTE.

This revised edition also incorporates information on the following TCAM support:

- 2790 Data Communications System
- 3270 Information Display System
- 3670 Brokerage Terminal
- 7770 enhancements
- disk error handling
- general poll for 2260 and 3270
- reverse interrupt (RVI)
- TOTE II On-Line Test (OLT)
- TSO/TCAM mixed environment

Contents

Figures

Organization and Use of the TCAM Program Logic Manual

This publication covers the internal logic of the IBM System/360 OS Telecommunications Access Method (TCAM). The *TCAM PLM* is directed to the IBM program system representatives and system engineers who provide program maintenance and who need information on the internal organization of TCAM.

Section 1, the *Introduction*, provides general information that is basic to an understanding of TCAM. This information places TCAM in the proper perspective to the operating system (OS).

Section 2 describes the *Basic TCAM Concepts*. These should be understood before approaching the specifics of the internal logic. Concepts described are the TCAM Dispatcher, buffer management, and queue management.

Section 3, the *Method of Operation* section, describes the functional flow of each operation in the TCAM system. The method of operation diagrams present the internal logic of a basic TCAM system without relying on extensive textual descriptions. The diagrams provide four kinds of information:

- Basic function (provided in the picture area).
- Module interfaces (shown as input to and output from the function being performed in the picture).
- Procedures that support the function (description provided).
- Pointers into the listings and flowcharts (cross-references provided with the description to lead to the proper routine, method of operation chart, flowchart, or listing).

Section 4 covers program organization and operation of TCAM. This information is provided in a series of tables that describe the functions of the various TCAM modules. Also included are:

- Charts of message handling macros, parameter lists, and the module linkages and functions that result from each TCAM macro coded.
- Tables of operator control commands and the action of the system as a result of each command.
- Tables of error recovery procedures to trace and record I/O errors.
- Flowcharts of some TCAM modules to show line and queue control. The flowchart identification is the same as the last two characters of the module name. When multiple flowcharts are necessary for a module, these two or three characters are followed by a dash and then a number (HM1-1). In addition, duplicate identifications are assigned to these flowcharts to facilitate off-page linkage.

Section 5 is a composite of the data areas used by TCAM. Each description includes the purpose, internal references, allocation, and initialization information. Both a visual and a tabular description of the DSECT for each area are also given.

Section 6 contains tables of information to aid in debugging and analyzing the activity of TCAM.

The seventh section consists of information, in four appendixes, to aid in the use of TCAM: a list of TCAM queues and QCBs, a list of TCAM modules by library, a list of TCAM relative priorities, and the TCAM channel programs.

(This page left blank intentionally)

Section 1: Introduction

TCAM is a component of the IBM System/360 Operating System. The primary purpose of TCAM is to provide a high-level access method to communicate with telecommunications equipment while maintaining the greatest possible device independence. In addition to supporting the transfer of data (messages) between both local and remote terminals and the system, TCAM provides a flexible message control language that can be used to direct the processing of the data. By using the TCAM macro instructions, installation-oriented message control is achieved.

TCAM operates under OS MFT or MVT in System/360 Model 40 or larger processors. The minimum main-storage requirement is 128K bytes. In addition to the system timer and normal OS requirements, TCAM requires a 2701, 2702, or 2703 on a multiplexer channel (unless only the 7770 or 2260 Local terminals are used, in which case the 7770 or 2848 is attached to the channel). Secondary storage for libraries and main or secondary storage for queuing are also required.

This section describes the various parts of TCAM and explains what they are, where they come from, how they get into the system, their relationships to each other, and how they pass control back and forth.

Figure 1 shows the steps necessary to begin processing in the TCAM environment.

System Generation

When TCAM is called for during a system generation procedure (by the ACSMETH=TCAM operand in the DATAMGT system generation macro instruction), the TCAM modules are included in four libraries: SYS1.MACLIB, SYS1.TELCMLIB, SYS1.SVCLIB, and SYS1.LINKLIB. An Attention routine and a Type I SVC module (the AQCTL SVC 102 routine) are incorporated in the Supervisor Nucleus (SYS1.NUCLEUS). There is an entry in the SVC table in the nucleus for Type 4 SVC 104 TOPCTL, which is resident in SYS1.SVCLIB. Using these modules, the user can assemble, linkage edit, and execute TCAM message control and application programs.

TCAM Macro Definitions

The operating system macro definition library (SYS1.MACLIB) includes the macro definitions necessary for the assembly of TCAM message control and application programs.

TCAM Resident Modules

When performing a system generation to include TCAM, the user must define a special library area named SYS1.TELCMLIB. During the generation run, modules that can later be linkage edited with message control and application program object modules are copied from SYS1.CQ548 into SYS1.TELCMLIB. In this publication, these modules are defined as the TCAM *resident modules. Appendix A* contains a list of the modules in SYS1.TELCMLIB.

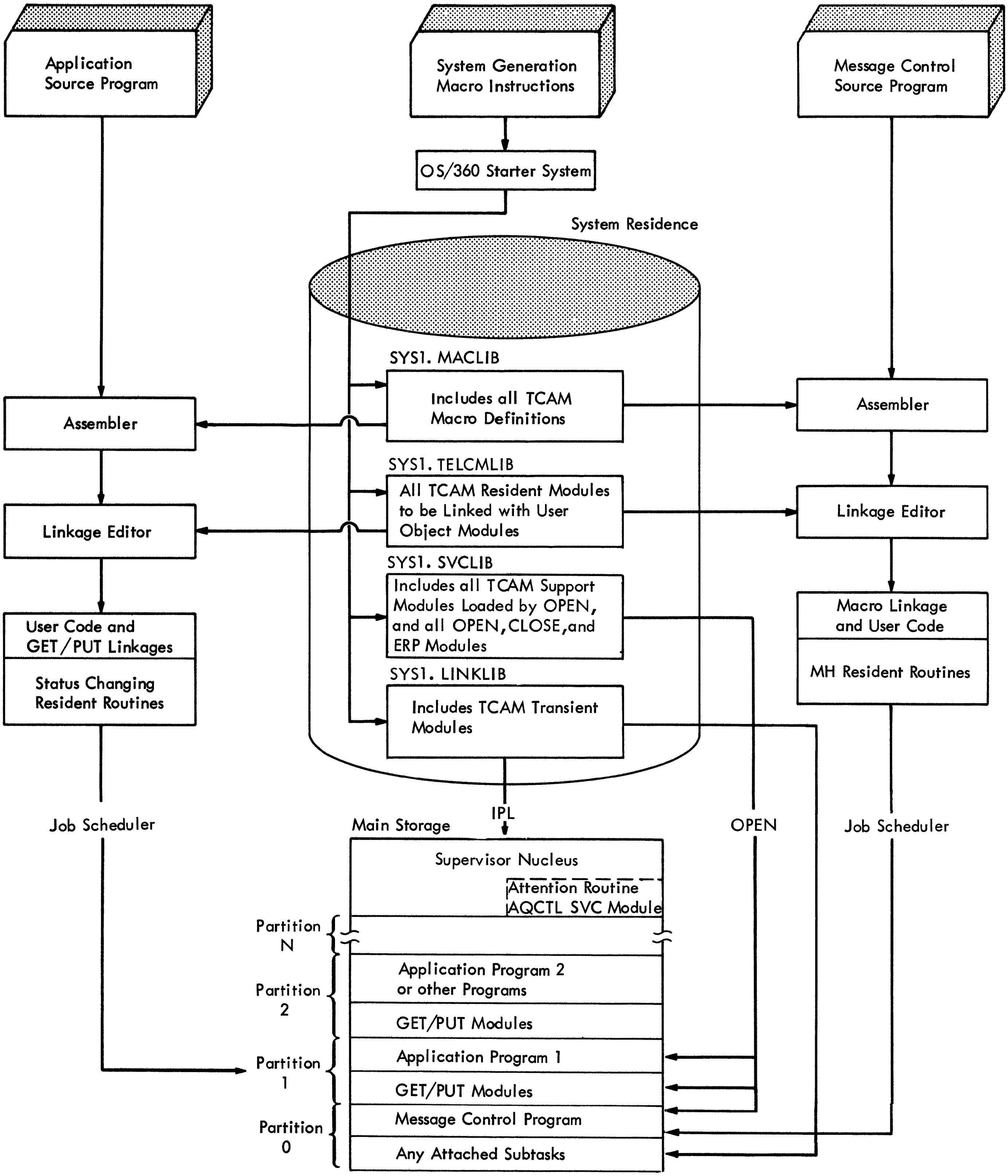

Application Source Program
System Generation Macro Instructions
Message Control Source Program
OS/360 Starter System
System Residence
SYS1. MACLIB
Includes all TCAM Macro Definitions
SYS1. TELCMLIB
All TCAM Resident Modules to be Linked with User Object Modules
SYS1. SVCLIB
Includes all TCAM Support Modules Loaded by OPEN, and all OPEN, CLOSE, and ERP Modules
SYS1. LINKLIB
Includes TCAM Transient Modules
Assembler
Linkage Editor
User Code and GET/PUT Linkages
Status Changing Resident Routines
Assembler
Linkage Editor
Macro Linkage and User Code
MH Resident Routines
Job Scheduler
IPL
Main Storage
OPEN
Job Scheduler
Supervisor Nucleus
Attention Routine
AQCTL SVC Module
Partition N
Partition 2
Application Program 2 or other Programs
GET/PUT Modules
Partition 1
Application Program 1
GET/PUT Modules
Partition 0
Message Control Program
Any Attached Subtasks

Figure 1. Physical Organization of TCAM

TCAM Support Modules

During the system generation run, all modules that are loaded into main storage by the various system open executors, and the TCAM open and close executors are copied from SYS1.CQ548 into SYS1.SVCLIB. The TCAM Dispatcher, the Command Scheduler, the Type IV SVC modules, and the Error Recovery Procedure routines are also placed in SYS1.SVCLIB. In this publication, these modules are defined as TCAM *support modules. Appendix A* contains a list of the TCAM support modules in SYS1.SVCLIB.

The Error Recovery Procedure routines and the TCAM Open and Close routines can, at the option of the user at system generation, be resident or transient during program execution. In either case, these routines reside in SYS1.SVCLIB.

TCAM Transient Modules

At system generation time, modules that can be called into main storage for a limited length of time during the execution of a TCAM message control or application program are copied from SYS1.CQ548 into SYS1.LINKLIB. In this publication, these modules are defined as TCAM *transient modules. Appendix A* contains a list of the modules in SYS1.LINKLIB.

The Operator Control, Checkpoint, and On-line Test routines stored in SYS1.LINKLIB can optionally be specified to be resident during program execution. However, in this publication they are defined as transient modules.

System Nucleus Modules

At system generation time, the Attention routine and the AQCTL SVC 102 routine (a Type I SVC) are copied from SYS1.CQ548 into SYS1.NUCLEUS. In this publication these two modules are defined as the *system nucleus modules.*

The Message Control Program in the System

Assembling and Linkage Editing a Message Control Program

The user codes the TCAM macro instructions necessary to design a Message Control Program. When these instructions are entered for assembly, the output of this assembly includes: several tables and control blocks, linkages to TCAM resident and support routines, Message Handler (MH) macro instruction expansions, and any user-written routines that were included.

The assembled object module is then linkage edited to include the desired resident routines from SYS1.TELCMLIB. These resident routines are the MCP routines used to process header information, to translate from one transmission code to another, to direct messages to the proper lines and queues, to manage system resources, etc.

The resulting load module is stored in a system library to be loaded for execution.

Execution of a Message Control Program

The TCAM Message Control Program (MCP) is normally executed as the highest-priority task in the highest-priority partition or region in the system. The OS Initiator/Terminator routine loads and transfers control to the MCP. The first TCAM macro instruction executed must be INTRO. The initial functions of INTRO are to establish the TCAM address vector table (AVT), addressability and entry linkages for the MCP, the cross-reference table, the channel program block (CPB) pool, the buffer unit pool, and main-storage queues. INTRO also attaches the Operator Control, FE Common Write, and On-line Test tasks and enables the user to override some INTRO parameters through the system console.

The MCP runs under the control of the OS task management routines. It is scheduled and dispatched according to the priorities included in the task control block (TCB) in the partition in which it is being executed.

The Application Program in the System

Assembling and Linkage Editing an Application Program

A TCAM application program processes messages obtained from a TCAM MCP. The application program can run in a partition or region different from the MCP, or it can run as an attached task in the same partition or region.

An application program needs only the OPEN, CLOSE, GET, and PUT macro instructions and some data set definition macro instructions. No resident routines need to be linkage edited with the object module. However, the user may wish to write application programs that use the following macro instructions to examine and modify the status of the MCP:

- CHECK
- CKREQ
- ICOPY
- ICHNG
- MCOUNT
- POINT
- QRESET
- TCHNG
- TCOPY
- TPDATE

When any of these macro instructions are used, the linkage editor includes the corresponding resident modules in the load module. The load module is stored in a system library from which it is loaded for execution.

Execution of an Application Program

It is possible to run an MCP with no application program, but one or more application programs are usually being executed asynchronously with the MCP.

In most cases an application program is loaded into the next highest-priority partition to the MCP. However, application programs may also be executed in the same partition as the MCP after being brought in by the system ATTACH facility.

Application programs, like the MCP, run under the control of the OS task management routines. They are scheduled and dispatched according to the priorities indicated in the task control blocks (TCBs) for the partitions in which they are being run.

The primary difference between a TCAM application program and any other processing program is the requirement for and the implementation of inter-partition communication.

The various macro instructions that can be used in an application program are as follows:

1. *TCOPY, ICOPY, QCOPY, and TPDATE*. The corresponding resident routine for each of these macro instructions copies the requested information from the MCP partition, using address pointers stored in the AVT and in the terminal table. These tables are located by the communications vector table (CVT).
2. *All other macro instructions*. The routines invoked by the remaining macro instructions cause SVC Type I interruptions to the supervisory routines.

A module within a partition can move data or control information from another partition into its own partition; however, that module must use an SVC either to move data from its own partition into another partition or to move data within another partition.

Relationship of the OS Dispatcher to TCAM

The operating system (OS) gains control from the TCAM task when the TCAM MCP has no work to perform and issues an OS WAIT macro. When OS gains control, it examines all the ready tasks in the system and passes control to the one with the highest priority.

When a TCAM appendage has work for the MCP, it invokes the OS Post routine by branching to an entry point to post the MCP event control block (ECB). This indicates to the OS Dispatcher that the MCP now has work to do and is vying for control of the system. OS can pass control to the TCAM task when it is the highest-priority task that is ready to be activated. TCAM resumes execution at the instruction following the WAIT that gave control to OS.

TCAM posts the ECBs for its attached tasks when they are to be activated. When TCAM subsequently issues a WAIT, the attached tasks can vie to gain control from OS.

Selected Options

TCAM has certain optional features available. These features are optional in one of three possible ways:

1. Some of the functions of the feature are optional.
2. The presence or absence of the feature itself is optional.
3. The feature may be either resident or transient.

The following sections discuss each of the optional features of TCAM.

Operator Control

The TCAM Operator Control facility provides a way for the user to dynamically examine or alter the status of his telecommunications network. A detailed description of the functions of this facility is included in the *Operator Control* section of the *OS TCAM Programmer's Guide*, Order No. GC30-2024.

The TCAM user specifies at SYSGEN time whether he wants the Operator Control facility in his system to be supported by resident or transient routines. The control module of the Operator Control facility is always resident. If the user indicates that he wants the operator control support routines to be transient, these routines are called in whenever they are needed. If the routines are specified to be resident, they are all present in the system at all times.

Application Program Processing

The application program services of TCAM enable a programmer to process messages from a telecommunications network with the same macro instructions that he uses for local input/output devices. Because the TCAM MCP performs the I/O operations, a completely device-independent application program can be written. The programmer need not be concerned with the time and device-dependent aspects of the telecommunications environment.

A TCAM MCP can operate in the system without an application program or programs. However, if the user wishes to examine and process the data coming in from his terminals to a greater extent than is allowed by the macro instructions of the MCP, he must use one or more application programs. The macros specific to application programs are discussed in detail in the *OS TCAM Programmer's Guide*, Order No. GC30-2024.

Line Queuing Options

The TCAM user has the option of queuing either by line or by terminal, as specified in the TERMINAL macro for each terminal or group of terminals. Queuing by terminal is required for buffered terminals and for dial lines. Since queuing by terminal requires one destination QCB per terminal rather than one per line group, this method requires more main-storage space.

Message Queuing Options

There are three types of queuing for messages:

- Main-storage queuing
- Reusable disk queuing
- Nonreusable disk queuing

The message queues may be maintained by any one of the three methods or by a combination of main-storage queuing with backup on either reusable or nonreusable disk.

In an MCP there are at most two message queues data sets: reusable disk with or without main-storage queues, and nonreusable disk with or without main-storage queues. The user specifies the type of queuing for a given data set by coding specified keyword operands of the macros that build the terminal table. The way in which the types of queuing are specified is discussed in detail in the *OS TCAM Programmer's Guide*, Order No. GC30-2024. The way that the various queuing types function is discussed under *Queue Management* in the *Basic TCAM Concepts* section of this publication.

Logging

The logging option allows the user to maintain a record of incoming or outgoing message traffic on a sequential medium. Message segments or full messages, as determined by the placement of LOG macros in an MH, are placed on an output device. The various types of logs and the corresponding MH subgroups in which a LOG macro appears, are:

1. Incoming header segments only (Inheader)
2. All incoming segments (Inbuffer)
3. Complete incoming messages (Inmessage)
4. Outgoing header segments only (Outheader)
5. All outgoing segments (Outbuffer)
6. Complete outgoing messages (Outmessage)

Checkpoint/Restart

Checkpoint/Restart is provided as an optional facility for the TCAM MCP at user-specified intervals (every 30 seconds to 65,535 seconds). By using the TCAM Checkpoint/Restart facility for the MCP and other TCAM facilities, such as sequence numbers, an effective restart can be accomplished in an application program.

The checkpoint routines store tables and other control information necessary for a restart subsequent to a system failure or normal closedown. Restart of the TCAM job after a system failure is accomplished by initial program loading (IPL) the system again (if necessary), and loading the TCAM MCP. TCAM reinitializes the tables and pointers from the latest checkpoint record on the disk (unless CY is specified on the STARTUP parameter of the INTRO macro to suppress continuation start-up). After a system failure, the STARTUP=C or STARTUP=W operand on the INTRO macro causes TCAM to perform a continuation restart with a scan of the message queues. If STARTUP=WY is specified, a continuation restart with no message queues scan is performed.

After a normal closedown, TCAM can either reconstruct the environment that existed before closedown (a warm restart) or it can reinitialize the system (a cold restart). A warm restart is specified by STARTUP=W on INTRO; a cold restart is specified by STARTUP=C.

To include the Checkpoint/Restart facility in an MCP, the user has only to specify an OPEN for the checkpoint data set. As a result of this, the Checkpoint Executor is attached in the same region as the MCP. The other checkpoint modules can be either resident or transient, depending on what the user specifies at SYSGEN time.

TCAM as a Startable Procedure

The user has the option of starting a TCAM MCP or application program either by JCL in the system input device or by the START operator command at the system console. If the START command is to be used, the JCL for the MCP and the different TCAM problem programs must be cataloged on SYS1.PROCLIB under individual procedure names. The user may then enter START and the *procname* for the program he wants, and job management immediately fetches the JCL at the *procname* and starts the program.

Error Recovery Procedures

The Error Recovery Procedure (ERP) routines are designed to diagnose and recover, if possible, from line errors occurring during a telecommunications operation. The error routines provide the following:

- Automatic retry of all errors not involving data transfer. Data transfer errors are also handled by the EOB/ETB Handling subtask, if specified in the MH.
- Automatic retry of text errors during a receive operation when the data is still available; that is, the PCI Appendage has not tposted the buffers containing the data following the last good EOB/ETB.
- Statistical recording of all terminal errors.
- Error messages to the primary TCAM operator console for all permanent errors.

The ERP routines are optional in that they may be either resident or transient. The user specifies this option at SYSGEN time.

Subtask Trace

The Subtask Trace facility maintains a time-sequential table of the dispatching activity of the TCAM Dispatcher. Each time the Dispatcher activates a subtask, it completes an entry in the subtask trace table.

The presence of the Subtask Trace facility in the TCAM system is determined by the DTRACE operand of the INTRO macro in the MCP. If the operand is coded DTRACE=0, the facility is not included. If the operand is coded with a numerical value, that value determines the number of four-word entries reserved for the subtask trace table.

The format of the subtask trace table is shown in the *OS TCAM User's Guide,* Order No. GC30-2025.

Cross-Reference Table

The TCAM cross-reference table is formatted if the CROSSRF=integer operand of the INTRO macro is assembled with a nonzero value. The numerical value of *integer* determines the number of four-word entries reserved for this table. Each time that a line is successfully opened, the Line Group Open routine (IGG01940) completes an entry in the table.

The format of the cross-reference table is shown in the *OS TCAM User's Guide,* Order No. GC30-2025.

TCAM in a Multiprocessing Environment

TCAM operating in a multiprocessing environment increases throughput, availability, and flexibility. All TCAM appendages and SVC 102 cause the TCAM task to become ineligible to be dispatched in order to prevent TCAM disabled code from modifying TCAM control blocks while enabled TCAM code is executing. These modules set a flag in the TCAM TCB to indicate that the task is not eligible to be dispatched and then call the OS Task Removal routine. When the Task Removal routine issues an external interrupt to lock the other CPU, the other CPU loops on the lock. When the TCAM module completes its functions, it resets the TCB flag and zeros the lock before exiting. The other CPU then obtains the lock and dispatches the task of the highest priority on its ready queue.

To prevent two enabled tasks from attempting to enqueue/dequeue on the same resource at the same time, each task issues a test-and-set instruction on a specific byte in the QCB before referring to the queue. The byte must be equal to zero before the task can update the queue, and the task must reset the byte to zero after completing the update.

Time Sharing Option

TCAM provides terminal support for the Time Sharing Option (TSO) under MVT when this option is requested on the INTRO macro. There are special macros to generate an MCP with MH routines to handle TSO messages. TCAM also supports application programs that are run under TSO in the foreground region. If the TSO option is specified, TCAM provides a conversational approach to terminal support—this includes support of the transmit and receive interrupt features, modifications to the scheduling of I/O operations, and editing of the data in TSO messages to make the data compatible with disk or tape.

TCAM and the TSO control program run in different partitions. Modified message flow allows TCAM to route the messages to the TSO region.

TCAM support for TSO also includes the ability to use 1050s and 2741s on the same dial line, the ability to simulate receive interrupts when they are not a feature of the hardware, and the ability to have the transmission code dynamically determined.

In a mixed environment, time-sharing supported terminals can be shared by time-sharing applications and message-switching applications.

General Poll

Three types of polling are available for device invitation. The most common is specific poll, which invites each device to transmit. The next most common is the Auto Poll feature, which uses 2702-2703 hardware to perform specific polling without I/O interruption or CPU uitlization. The general poll is desired for a remote cluster of devices. It allows any device, if ready, to transmit without a specific invitation.

General poll is a remote input technique in which special invitation characters are sent to a 2260 or 3270 device control unit to solicit transmission of data from all attached devices that are ready to send. General poll may be conducted with programmed poll or Auto Poll, both of which invite each individual device to send.

General poll begins with transmission of the invitation characters. If a positive response is received, TCAM determines the identity of the device terminal originating the transmission message and puts this information in LCBTTCIN. An entire message is read from one device until an ETX is received. Each device can send only one message at a poll. When the ETX is entered a complete message has been received, and all buffers are tposted to the Message Handler, and the message is processed.

Standard scheduling is performed as for any receive operation on a multipoint line. To receive the next message from the control unit, TCAM begins a new input operation; however, invitation characters are not re-sent (as in programmed poll or Auto Poll); the next message is read. This cycle continues until the device control unit indicates, by sending an EOT, there is no more data to be sent. General poll may also be terminated by the receipt by the control unit of a response other than an ACK, NAK, or ENQ.

No interruptions are allowed during general poll except for conversational processing. Once the EOT is received, TCAM either transmits or polls the next entry in the invitation list. The user should be aware of the time constraints of the hardware involved.

Teleprocessing On-Line Test Executive (TOTE)

The Teleprocessing On-Line Test Executive (TOTE) is an attached subtask of TCAM, designed to control the selection, loading, and execution of on-line tests (OLTs). The on-line test function consists of three parts: TOTE, an on-line device configurator, and the individual teleprocessing device tests (OLTs). TOTE is the interface between TCAM and the on-line tests.

The individual OLTs are intended to diagnose hardware errors, verify repairs, verify engineering changes, and test devices. TOTE conveys messages to the user about the test, schedules and controls the test, and prompts the user when requested or when an error in the format of a Test Request Message (TRM) is detected. The OLTs are transient and reside in a library on a system direct access device.

Test selection is achieved by entering a Test Request Message (TRM) from a TCAM station, operator control terminal, or the system console. Test results are sent to the terminal controlling the test, unless an alternate printer is designated as a parameter or the option field of a TRM.

Configuration Data Set

TOTE also allows the user to enter changes to configuration data stored in a Configuration Data Set (CDS). The configuration data set contains descriptive data about the I/O units attached to the system: this includes telephone numbers, what devices are attached to which channel addresses, the features installed, and any other data the OLTs might need to test all the equipment installed in a particular location.

After the data set is generated it may be dynamically altered by answering questions presented by configuration request messages (CRMs). A CRM can be entered from either the system console (through the operator control facility) or a TCAM station.

TOTE Requirements

The following requirements must be met before executing TOTE:

- The TCAM operator control facility must be initialized.
- The OLT modules must have been placed in a library.
- The configuration data set must have been built by a stand-alone, on-line test, support program (SOSP).
- The terminals must be represented in the TCAM JCL by a DD card.

The following requirements must be met before executing a device test:

- The devices and communications lines used or tested must have been configured.
- The devices used as the control terminal or alternate printer for the OLT, as well as the devices to be tested, must be on opened communication lines.
- The communication lines to be tested must have been opened.

All I/O for the OLTs is done by the EXIO macro. Upon receiving this request, TOTE usually builds an IOB using the data in the parameter list passed with the request. This request is linked to the test DCB, an ECB, a DEB, and the OLTCB. When all these blocks are properly prepared, an EXCP macro initiates the channel program.

Abnormal Termination Recovery

There is an entry point in the TOTE resident module (IEDQWA) that is entered at OLT ABEND. This module will set a flag to indicate that areas used by the OLT are to be cleaned up or freed when the control module next gains control. The flag also indicates that the OLT has terminated, and the reason for the termination is displayed on the system console.

All lines and terminals allocated to the OLT are returned to the state in which they were found when the OLT was started; normal TCAM operations are resumed.

Section 2: Basic TCAM Concepts

This section discusses each of the three basic concepts that influence the control and functions of TCAM. The first concept, the method by which the TCAM Dispatcher manages the TCAM resources, determines the flow of control among the TCAM subtasks. The second and third concepts are the management of the queues and of the buffers, respectively. An understanding of these three concepts will help to clarify the charts in the *Method of Operation* section of this publication.

The TCAM Dispatcher

The TCAM Dispatcher is the control module of the TCAM system. The primary purpose of this module is to allocate and schedule system resources. The following sections describe how the TCAM Dispatcher allocates and schedules the system resources, for example, CPU processing time, main storage, I/O paths, and elements (primarily buffers and lines). The key to the mechanism is the *ready queue,* through which a resource is allocated to a subtask.

The mechanisms of allocation are the *twait* and *tpost* functions performed by the TCAM subtasks. A *twait* schedules a subtask to be activated when a specific resource is available; a *tpost* passes an available resource to the ready queue. The actual implementation of twait and tpost are not exclusive functions of the subtasks; rather, the subtasks return to specific entry points in the TCAM Dispatcher to indicate the status of the resource. *Dispatching* is the process of providing a routine with an element and giving the routine control to handle the element.

Elements, Queues, and Subtasks

The physical resources of the system are composed of *elements* (for example, the buffer pool, a *resource,* is broken into individual buffers, the elements) with each element represented by a resource control block (RCB). An RCB is an 8-byte prefix to an element. The first four bytes are a pointer to the queue control block (QCB) that the element is to be associated with; the last four bytes contain a priority byte and a link field.

There is at least one *subtask* that works with every type of element in the system. Each subtask is represented by a subtask control block (STCB), which contains the data necessary to activate the subtask it represents.

The elements, and the subtasks that operate on these elements, are associated with one another by a third control block, the queue control block (QCB). Thus, a QCB has a pointer to the chain of elements under its control and a pointer to the chain of STCBs for the subtasks waiting to operate on these elements. The chains are referred to as *queues*. Figure 2 illustrates the linkage of these queues to a QCB.

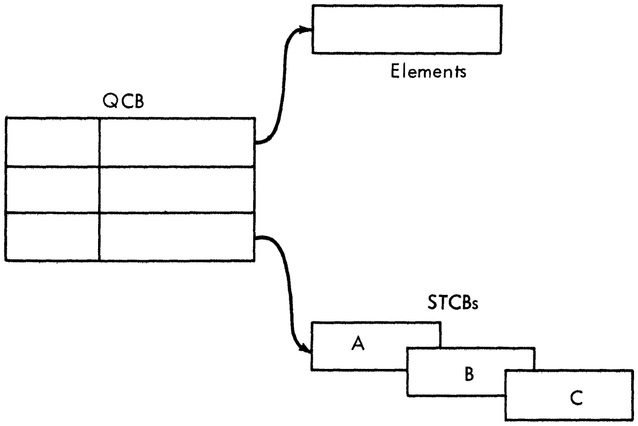


Figure 2. TCAM QCB Linkage

When a subtask needs an element, it can do one of two things: (1) request an element from the QCB that handles that particular element by tposting a request element to that QCB, or (2) insert its STCB into the STCB chain of the QCB to twait for the element. When the element is available, the subtask is dispatched.

When a subtask has finished using an element, it gives (tposts) the element to the appropriate QCB. The TCAM Dispatcher gives this element to the first (highest-priority) subtask in the STCB chain of the QCB. In this case, subtask A in Figure 3 is dispatched. The subtask associated with STCB B in Figure 3 can be dispatched if subtask A indicates to the TCAM Dispatcher that it does not need to process the element. The STCB chain ends with a permanent STCB. STCB C in Figure 3 remains the last STCB in the chain. STCB C might point to a routine that does nothing more than chain elements into the QCB element chain. Subtask C has a lower priority than any other subtask that might use the element and, therefore, is dispatched only if each of the higher-priority subtasks bypasses processing.

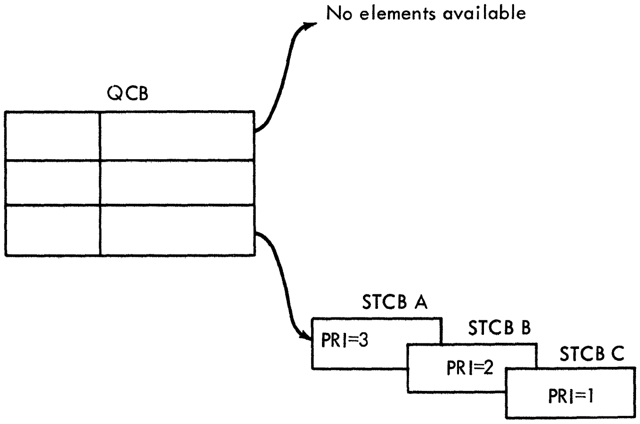


Figure 3. Priority of Subtasks on a QCB

Figure 4 demonstrates the linkage when an element processed by subtask X is tposted to the QCB and placed on the element chain by subtask C. Subtask C can place the element in the QCB element chain only if subtask A and subtask B do not need the element and pass it down the chain to subtask C.

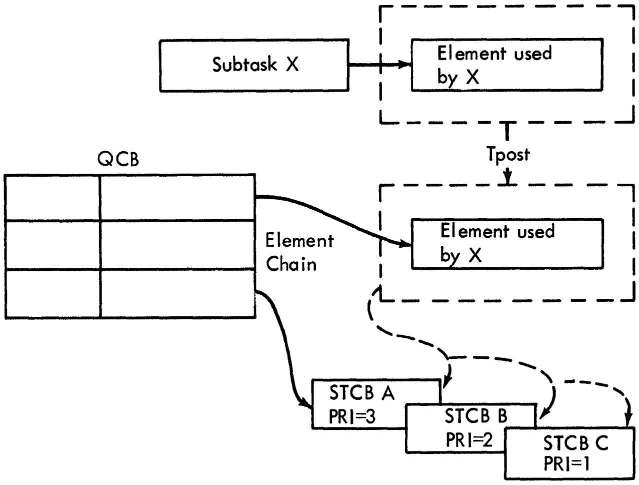


Figure 4. Passing Elements to a QCB

The Ready Queue

The previous discussion points out that subtasks gain control from the TCAM Dispatcher depending on:

1. The availability of elements, and
2. The priority of the STCB for the subtask.

The TCAM Message Control Program is responsible for allocating CPU processing time to the various tasks under its control; it does so by using the ready queue.

The *ready queue* is a chain of elements that represent all the work to be done in the TCAM system. The work to be done is represented by the various elements (RCBs) that appear on the ready queue in priority order. The purpose of the ready queue is to ensure that all elements are processed and dispatched with respect to priority and without one impacting the resources of another.

To support dispatching while enabled for interruption, TCAM actually uses two ready queues. One is designated to be used by disabled appendages or by the disabled AQCTL SVC 102 routine for tposting elements, while the other is used by enabled routines. Although the two ready queues are not managed by the same technique, each is a ready queue because it contains elements (RCBs) to be processed by the various subtasks.

TCAM manages the *disabled ready queue* by the first-in-first-out (FIFO) technique. The queue itself consists of two words: a one-word pointer to the first and a one-word pointer to the last element on the queue. Disabled appendages place an element (RCB) on the disabled ready queue by linking the new element to the element pointed to by the second word of the queue and by then updating the second word to point to the new element.

TCAM manages the *enabled ready queue* by the priority-FIFO technique. The TCAM Dispatcher has the responsibility for merging the disabled into the enabled ready queue just before dispatching. The enabled ready queue handles dispatching, and unless specified otherwise, it is the one usually referred to as the ready queue.

The TCAM Dispatcher manages the ready queue by executing the subtask associated with the highest-priority element on its chain. Since the element has an RCB as its prefix, the Dispatcher can refer to the correct QCB in order to pass control to the first subtask represented in the STCB chain of the QCB. The subtask processes the element and then returns control to the TCAM Dispatcher, which can then examine the next element on the ready queue. Figure 5 illustrates the chain of linkage from the ready queue to a subtask when an element is on the ready queue.

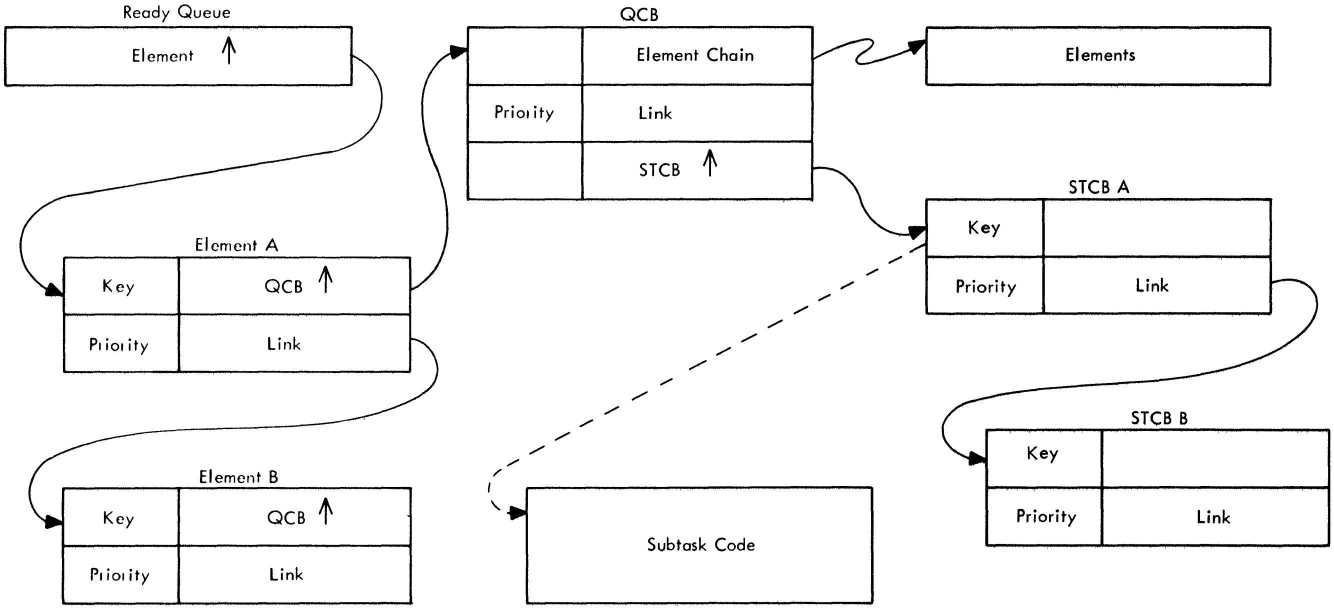


Figure 5. Linkage from the Ready Queue to Subtask Code

When the Dispatcher gains control it removes the highest-priority element from the ready queue by placing the address of the element in register 1. The Dispatcher then inserts the link field of the element in the ready queue to point to the next element. When there are no elements for the ready queue, it points to the "dummy last element" in the AVT (AVTDELEM). This element has a priority of zero. Figure 6 demonstrates the change in linkage between the ready queue and its elements during an update of the ready queue by the Dispatcher.

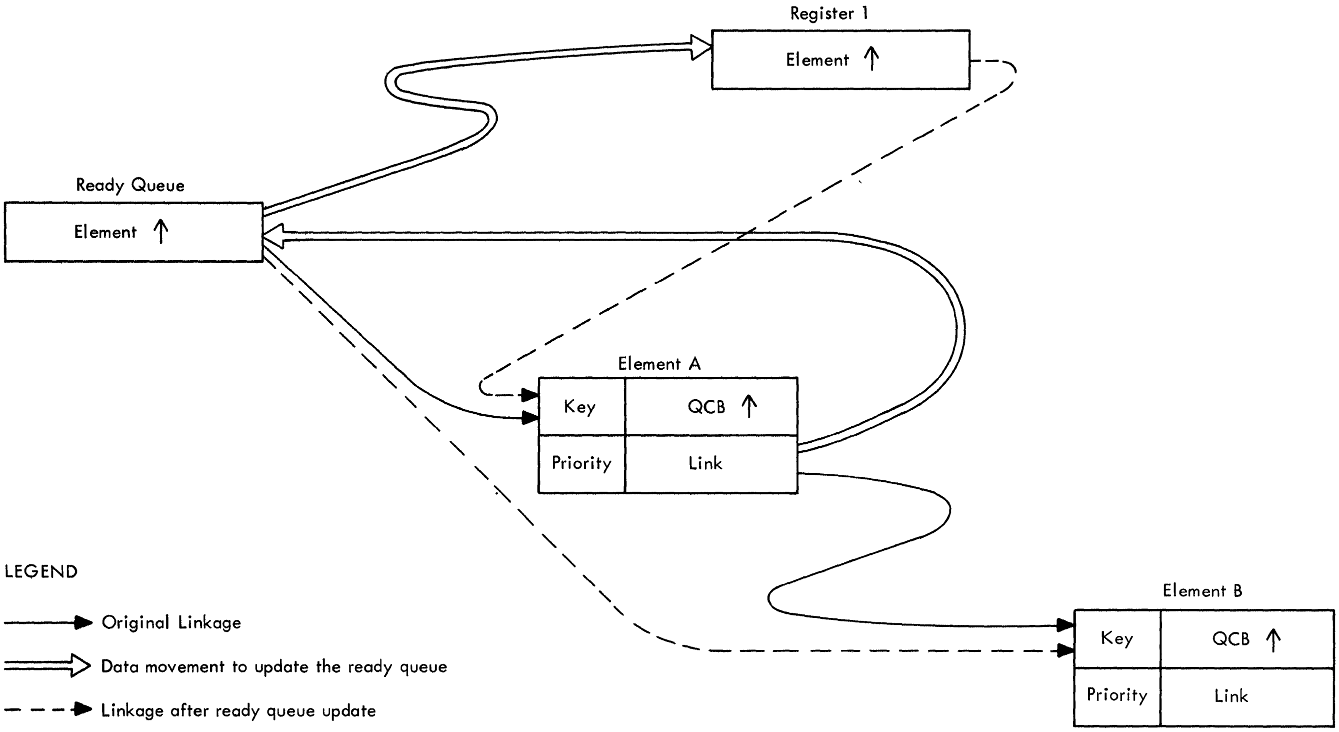


Figure 6. Pointers during a Ready Queue Update

Principle of Tpost and Twait

The technique of passing an element from one queue to another queue is called *tposting.* When the subtask that an STCB points to finishes processing an element and wishes to allow another routine to process that same element, the subtask tposts the element to the second routine. The subtask achieves the tpost by placing in the RCB of the element a pointer to the QCB that controls the STCB for the new routine, and then returning to the TCAM Dispatcher with an indication that the element is to be placed on the ready queue.

The second technique for handling resources is called *twaiting.* When a subtask needs elements to process, it returns control to the TCAM Dispatcher indicating that it has finished the processing that it can do at this time. The twait is implemented by the TCAM Dispatcher. The Dispatcher places the STCB for this subtask in the STCB chain of the QCB to which the resource that the subtask needs to complete processing will be tposted. When an STCB is in the STCB chain of a QCB and the subtask for that STCB does not have control, the subtask is twaiting.

When an application program needs either to place an element on the disabled ready queue, to post an event control block (ECB) as complete, or to move data from one partition to another, a special technique is used. This technique is performed by the AQCTL SVC 102 routine, which uses pointers in the AVT to refer to the disabled ready queue. Since AQCTL is a resident Type I SVC, the actual processing occurs in the OS Supervisor, out of the control of either the application program or the MCP.

Buffer Management

The TCAM network has one buffer unit pool that contains buffer units of one size. These buffer units are the basic building blocks from which *buffers* are constructed. Henceforth, in this publication *unit* refers to a buffer unit.

Messages entering a TCAM network are placed in *buffers,* which are user-defined areas of main storage used for handling, queuing, and transferring message segments between all lines and queuing media. (A *message segment* is that portion of a message contained in one buffer.) A buffer has two parts, one that contains control information (the *buffer prefix*) and the other that contains all or part of the message. Buffers must be at least 35 bytes long, and may be no longer than 65,535 bytes.

The size of a unit is specified in the UNITSZ= operand of the INTRO macro of an MCP, and the number of units in the buffer unit pool is equal to the sum of the numbers specified by the LNUNITS and MSUNITS operands of INTRO. For internal management purposes, TCAM adds 12 bytes as a prefix to the user-specified unit size. These 12 bytes are called a *unit control area.* Thus, if a user defines a unit size of 60 bytes (UNITSZ=60), the size of the unit is actually 72 bytes.

The size of a buffer for a line group is specified by the BUFSIZE= operand of the DCB macro for a line group data set. All buffers used by a given line group are the same size, but each line group may use buffers that differ in size from those assigned to other line groups. (The buffer size can be overridden on a terminal basis for send operations by using the BUFSIZE= operand of the TERMINAL macro.)

TCAM constructs buffers by linking together the number of units necessary to create a buffer that contains a number of usable bytes equal to or greater than that specified by the BUFSIZE= operand of the DCB macro for a given line group. (The 12 bytes added to each unit by TCAM are not considered in defining the size of the buffer; the user should consider only the number of bytes he specified in the UNITSZ= operand of INTRO.) For example, if UNITSZ=60 in the INTRO macro and BUFSIZE=120 in a line group DCB macro are specified, TCAM links together two units in building each buffer for that line group.

There are two types of buffers—header buffers and text buffers. A *header buffer* contains all or part of a message header. A *text buffer* contains message text only.

A *buffer prefix* is a control area contained within each buffer of the system. The user must allow room for the buffer prefix in defining his buffers. TCAM fills the buffer prefix area with buffer control information.

There are two kinds of buffer prefix. The *first-buffer prefix* is 30 bytes long and is contained within the first buffer of a message. Any *subsequent-buffer prefix* is 23 bytes long and is contained within all buffers after the first.

Thus, there are two kinds of control areas associated with buffers: the 12-byte unit control area associated with each *buffer unit* and assigned automatically by TCAM, and the 30-byte or 23-byte buffer prefix assigned to each *buffer* by TCAM in an area defined by the user. Each unit must be big enough to contain a header prefix plus three bytes of message text (35 bytes) and may be no larger than 255 bytes. A subsequent buffer contains more bytes of actual message than the first buffer, since a subsequent-buffer prefix is 7 bytes shorter than the first-buffer prefix.

The 12-byte unit control area that TCAM assigns to each unit is used to manage multi-unit buffers. This control area has different functions dependent on the status of its buffer. It may contain pointers, be used as an RCB, or be used to generate a channel program. The initial format of this 12-byte area is defined in Figure 7.

Offset

0	1	4	8
Key	QCB address	Address of the first unit of the next logical buffer that is assigned	Address of the next unit of this buffer

Figure 7. Unit Control Area

Figure 8 shows how two buffers assigned to a line group look at the time of an initial request if the user specifies the following:

```
INTRO   UNITSZ=60
DCB     BUFSIZE=100,BUFIN=2
```

In Figure 8, each buffer consists of two units linked together by the pointer in the third word of the 12-byte unit control area. The two buffers are linked together by the second word of the 12-byte unit control area. Note that in this situation the first eight bytes of the unit control area of the first unit in each buffer is functioning as an RCB.

When the user's program requests and obtains buffers, they look like the ones in Figure 8. However, when a line is ready to read or write, the function of the 12-byte control area changes. TCAM then uses the area to contain the channel program that operates on the unit. TCAM places a CCW in each RCB field, and the pointer in the third word becomes a TIC to the next unit. The 30-byte prefix contains a count of the number of units in a logical buffer; this indicates where one buffer stops and another starts.

To tpost a buffer, TCAM places only the first unit of that buffer on the ready queue. All other units can be located through the chain created in the TIC field of the unit control area.

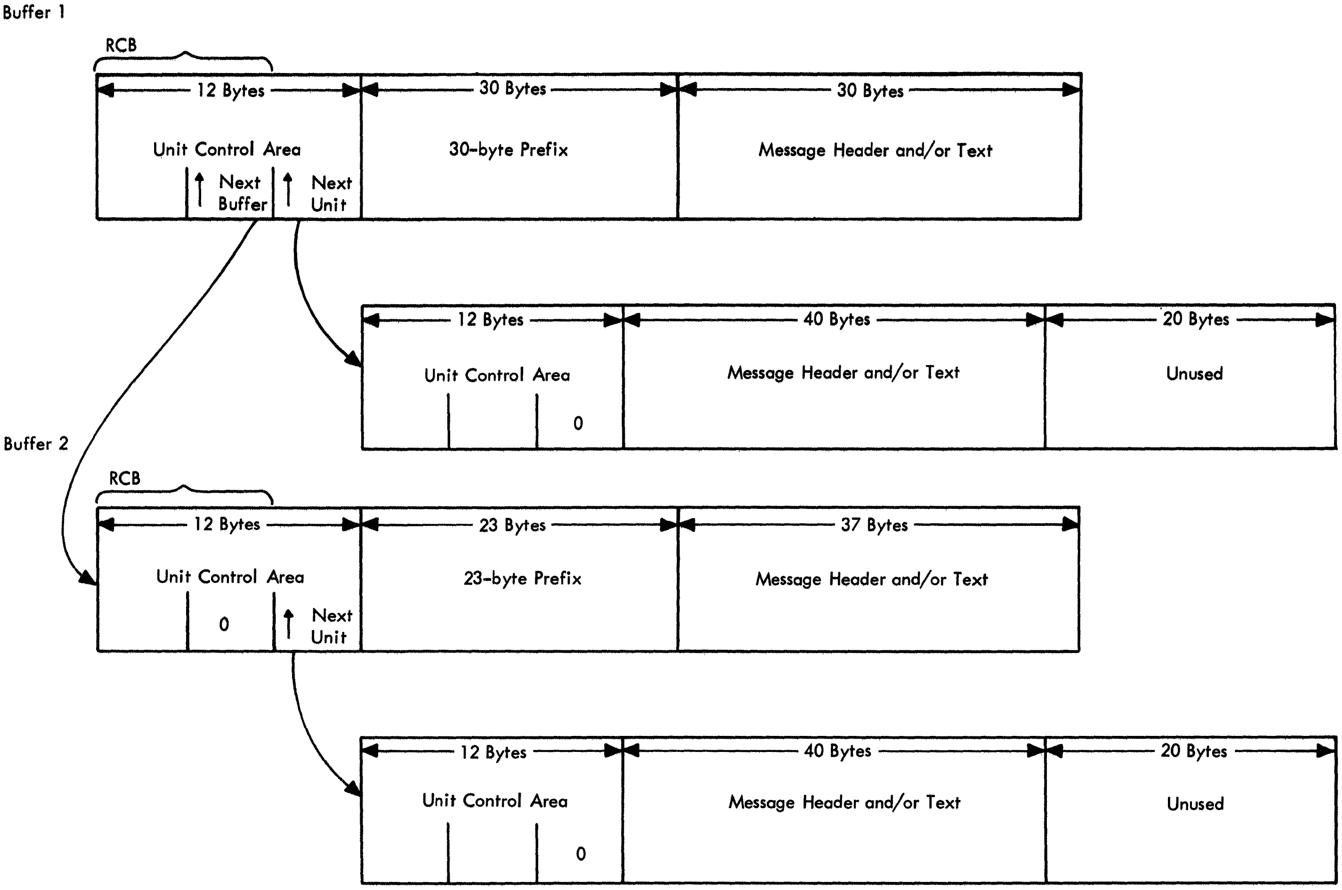


Figure 8. Buffer Units Chained to Form Buffers

TCAM uses an element request block (ERB) to make requests for buffers for a line group. Initial requests for buffers for a line are made when a scheduler tposts its ERB, which contains the number of buffers requested, to the buffer request QCB for a receive operation, or to the disk I/O QCB for a send operation.

Subsequent requests for buffers are handled by the TCAM Program-Controlled Interruption (PCI) Appendage. When the PCI= operand of the DCB for a line group is coded to allow program-controlled interruption, a PCI may occur during the filling or emptying of the first and each subsequent buffer assigned to that line group. When the PCI is received, the PCI Appendage gains control.

When PCI=A is coded on the DCB macro and the first interruption occurs, PCI Appendage assigns to the line group a number of buffers equal to the difference between the maximum number assigned to the line group (specified by the BUFMAX= operand of the DCB) and the number initially assigned to the line group (specified by the BUFIN= operand of the line group DCB for a receiving operation and by the BUFOUT= operand for a sending operation). On subsequent PCIs, the appendage deallocates the buffer immediately preceding the one being filled or emptied and requests a new buffer in order to keep the number of buffers assigned to the line group equal to that specified by the BUFMAX= operand. (For a sending operation, the buffer units are returned by the buffer return QCB to the buffer unit pool—the element chain of the buffer request QCB; for a receiving operation, the buffer is sent to the Message Handler for the line group for that DCB.)

When PCI=R is coded, the appendage deallocates the previous buffer when the second and subsequent PCIs occur, but makes no requests for additional buffers. If program-controlled interruptions are not permitted (PCI=N) or additional allocation is not allowed (PCI=R), the number of buffers assigned must be sufficient to handle the entire transmission, since no new buffers are allocated until the transmission is complete. If PCI=N, there is no deallocation of buffers until the transmission is complete.

Figure 9 shows the result of tposting an ERB with a count of three to the buffer request QCB. The ERB chain of the LCB points to the first buffer. This figure demonstrates the change in linkage after units have been transferred from the buffer unit pool to form a buffer chain off the requesting ERB. The physical location of the units in main storage does not change—the various pointers are changed to reflect the new organization.

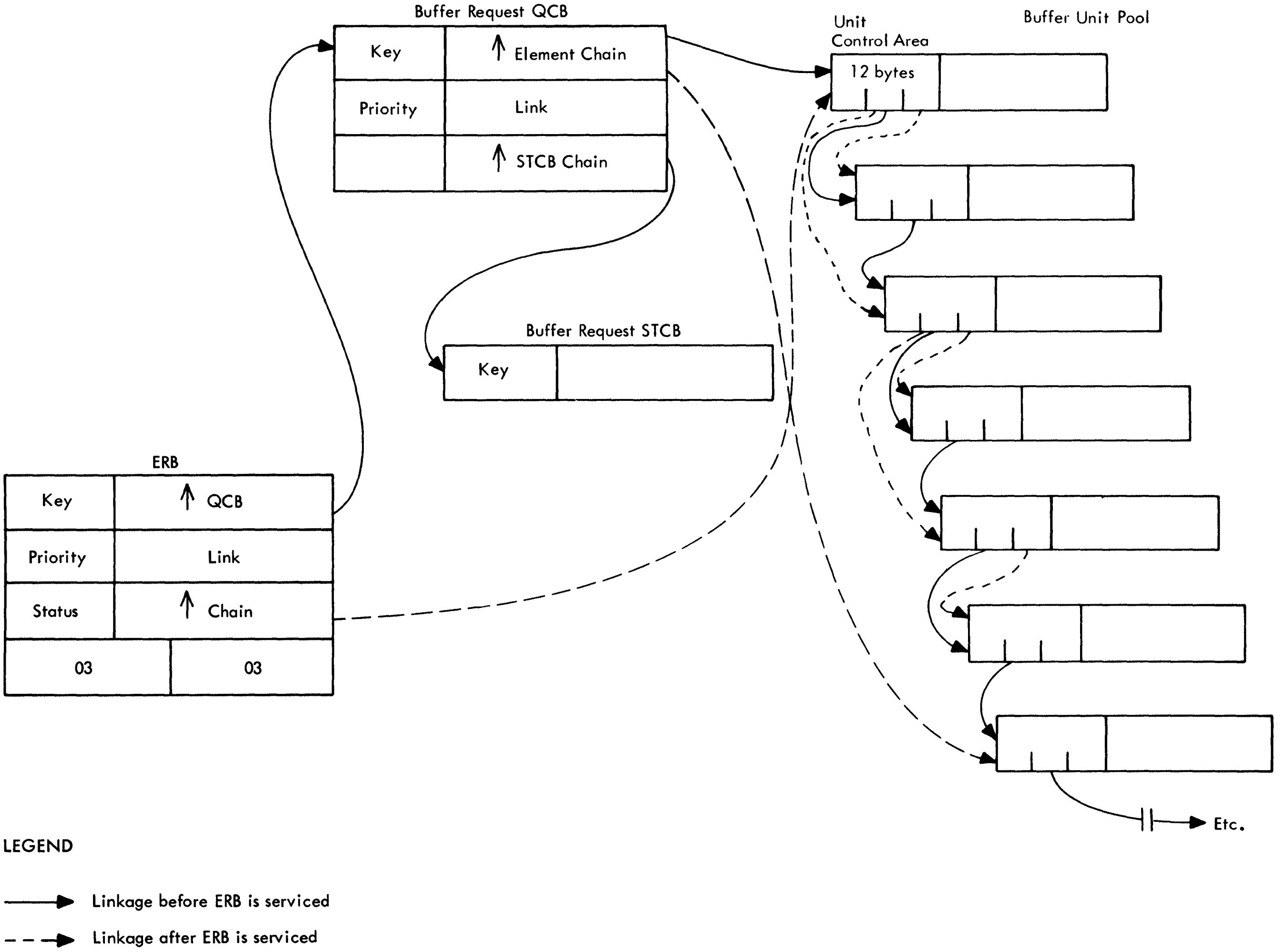


Figure 9. Effect of an ERB on Buffer Unit Linkage

Queue Management

The incoming group of an MH performs user-specified functions in a buffer that contains a message segment. After these functions are completed, the segment is tposted to a destination QCB, which represents a line, terminal, or application program.

Each destination QCB in a TCAM MCP is assigned to one or more specific message queues data sets. When a buffer is tposted to its destination QCB, it is placed on the appropriate message queue in the associated message queues data set to wait its turn to be sent to the specified destination.

The message queues data set to which a message segment is to be directed may be in main storage or on a direct-access storage device. Each message queue within a data set contains segments that are to be transmitted on a certain line or to a certain terminal, or that are to be processed in a specific application program.

TCAM supports five types of queuing to a message queues data set:

- Nonreusable disk queuing
- Reusable disk queuing
- Main-storage queuing
- Main-storage queuing with nonreusable disk backup
- Main-storage queuing with reusable disk backup

The following sections discuss the functions of these types of queuing.

Nonreusable Disk Queuing

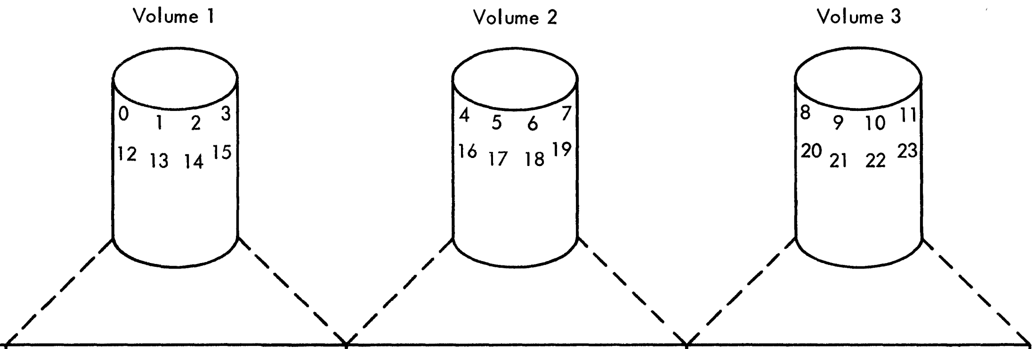
Queuing a message on a direct-access storage device is referred to in this publication as disk queuing. The fields AVTNADDR and AVTRADDR in the AVT contain the index to the nonreusable and reusable disk relative record numbers, respectively, of the next record to be assigned.

In nonreusable disk queuing, the Destination Scheduler initiates a closedown when a user-specified percentage of the disk message queues data set has been filled. If, before the closedown is completed, there are more messages in the system than the data set has room to accommodate, TCAM issues an ABEND.

The EXCP Driver routine assigns disk relative addresses across the volumes of a multivolume disk message queues data set in such a way that the next relative record address after the last record on a track is on a different volume. The routine numbers all the records for a given track consecutively before assigning addresses on a track of a different volume. In addition, the routine numbers all the tracks of a cylinder before assigning addresses on a different cylinder. Figure 10 illustrates the disk record numbering scheme for a data set that has four records per track on three volumes.

At MCP assembly or restart time, each destination QCB is assigned a unique relative record number for the first buffer segment tposted to it. As a result, when the first message enters the TCAM system, the AVT value is one greater than the total number of destination QCBs.

The Destination Scheduler stores the address to be used for the first unit of the first buffer of the next message received in the QCBDNHDR field of the destination QCB—this is referred to as the *next-message* location. The routine stores the address for the first unit of the next buffer of the current message in the SCBNTXT field of the SCB—this is referred to as the *next-buffer* location.



Cylinder	Track	Relative Record Number				Relative Record Number				Relative Record Number			
0	0	0	1	2	3	4	5	6	7	8	9	10	11
	1	12	13	14	15	16	17	18	19	20	21	22	23
	2	24	25	26	27	28	29	30	31	32	33	34	35
	3	36	37	38	39	40	41	42	43	44	45	46	47
	4	48	49	50	51	52	53	54	55	56	57	58	59
	5	60	61	62	63	64	65	66	67	68	69	70	71
	6	72	73	74	75	76	77	78	79	80	81	82	83
	7	84	85	86	87	88	89	90	91	92	93	94	95
	8	96	97	98	99	100	101	102	103	104	105	106	107
	9	108	109	110	111	112	113	114	115	116	117	118	119
1	0	120	121	122	123	124	125	126	127	128	129	130	131
	1	132...											

Figure 10. Assignment of Disk Message Queues Data Set Relative Record Numbers Across Three Volumes

The principle of assigning next-message and next-buffer values allows queuing ahead on the disk. Records for buffer units are assigned before the buffer is received.

In the example in Figure 11, there are five possible destinations. For each of these, the MCP assembly has preassigned record addresses (marked A through E) with relative record addresses zero to four. The applicable code for this example is:

```
             INTRO UNITSZ=100

    LINEA    DCB    BUFSIZE=300,PCI=(A,A)

    LINEC    DCB    BUFSIZE=800,PCI=(A,A)
```

Three messages arrive in the following order:
1. 500 characters—from Line A to Line D
2. 3000 characters—from Line C to Line B
3. 30 characters—from Line A to Line B

Figure 11 shows the situation in which TCAM reads a buffer (the first buffer of the first message) from line A. The 30-byte prefix contains the information that this message is to be sent to line D. The message segment consists of three units (since BUFSIZE=300 and UNITSZ=100) and does not contain an end-of-message (EOM) indicator. The Destination Scheduler assigns the first unit of this header buffer to the preassigned location for destination D, record 3. The Scheduler then preassigns the next-message location for destination D to the next available disk location at record 5, and places a pointer to record 5 in the prefix of the buffer that will start in disk record 3. The Scheduler then assigns two additional units to the next available disk locations at records 6 and 7. The Scheduler inserts a pointer to the first of these records in the prefix of the buffer that will start in disk record 3.

Since the 300-byte buffer does not contain an EOM indicator, the Destination Scheduler preassigns a record number (8) for the first unit of the next buffer to arrive for this message. The Scheduler places a pointer to record 8 in the prefix of the buffer that will start in disk record 3. The records are actually written after the three pointers are included in the prefix of record 3. Figure 11 shows the records and pointers after they are written on disk.

In this queuing scheme the additional records are always contiguous, and the first unit of a subsequent buffer of a message is always contiguous to the last unit of the previous buffer.

In Figure 12 the first buffer of the 3000-byte message from line C for line B is queued. The buffer consists of eight units since BUFSIZE for line C is 800 bytes. The Destination Scheduler places the first unit of the message in the preassigned slot for destination line B. The scheduler then preassigns a location for the first unit of the next message for line B to record 9, the next available disk location. The scheduler places the additional records (units) for the current message segment in disk locations 10 through 16. Since this buffer does not contain an EOM indicator, the scheduler preassigns the next-buffer location to record 17.

In Figure 13, the second buffer of the message for line D is queued. This is a three-unit buffer with an EOM character in the last unit. The DestinationScheduler places the first unit in the next-buffer slot of line D at record 8 and places the two additional records in the next available disk locations, records 18 and 19. No preassignment for the next-buffer location is made because of the EOM character in this buffer. The scheduler preassigned the next-message slot for line D to record 5 when the first buffer of this message was queued (see Figure 11).

In Figure 14, the 30-byte message from line A to line B is queued. Since this message is contained within a single unit, only that unit must be written on disk. The Destination Scheduler places this unit in the preassigned next-message location for destination B, record 9. No next-buffer location needs to be preassigned, but the scheduler changes the next-message location for line B to disk record 20. The next available disk location is now record 21.

Figures 11 through 14 do not illustrate all the disk record pointers. However, Figure 15 shows the pointers mentioned above, as well as the pointers from each subsequent buffer of a message to the first buffer of the message. These pointers are the base for the *queue-back chain* to be discussed next.

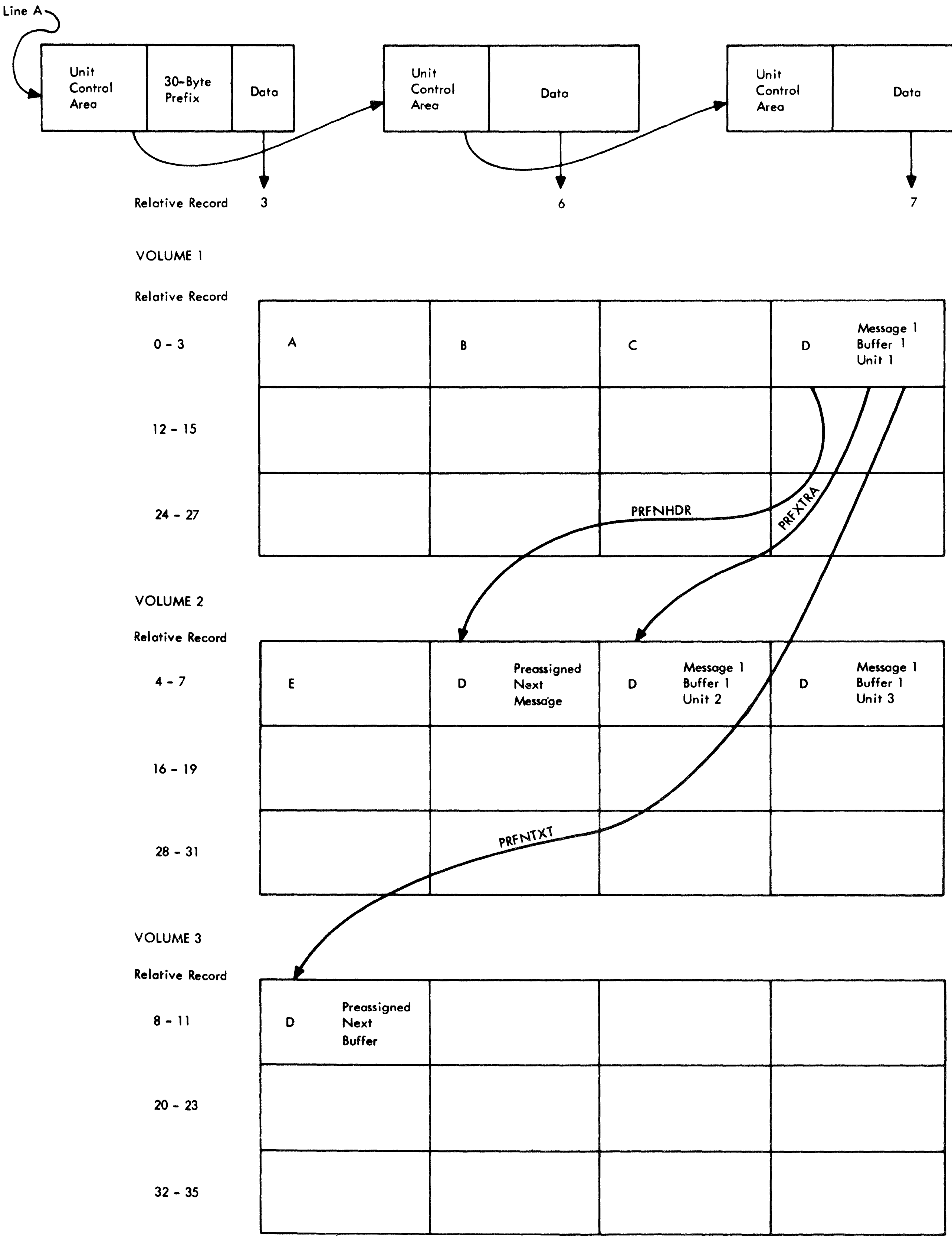
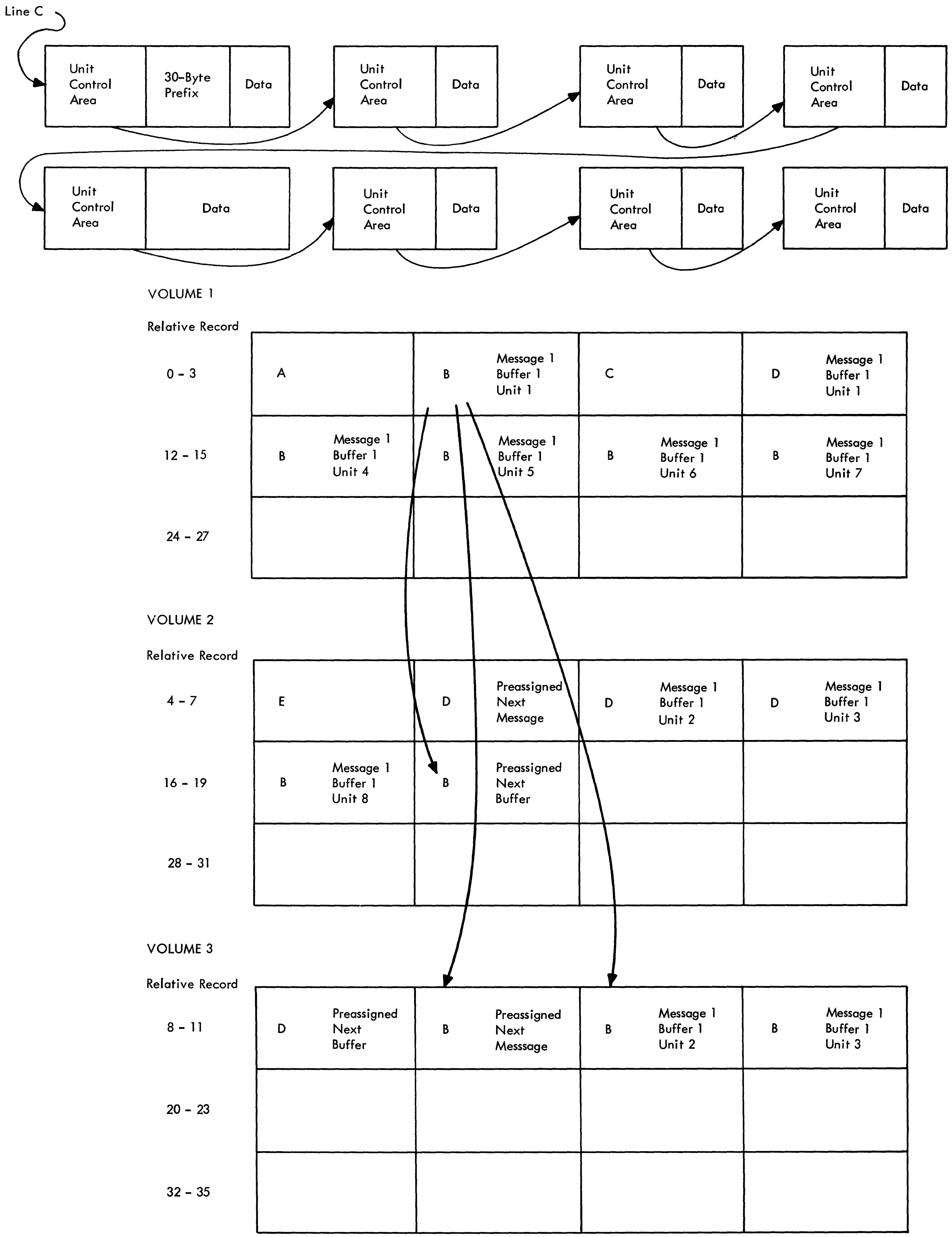


Figure 11. Disk Queuing a Three-Unit Buffer



VOLUME 1

Relative Record				
0 - 3	A	B Message 1 Buffer 1 Unit 1	C	D Message 1 Buffer 1 Unit 1
12 - 15	B Message 1 Buffer 1 Unit 4	B Message 1 Buffer 1 Unit 5	B Message 1 Buffer 1 Unit 6	B Message 1 Buffer 1 Unit 7
24 - 27				

VOLUME 2

Relative Record				
4 - 7	E	D Preassigned Next Message	D Message 1 Buffer 1 Unit 2	D Message 1 Buffer 1 Unit 3
16 - 19	B Message 1 Buffer 1 Unit 8	B Preassigned Next Buffer		
28 - 31				

VOLUME 3

Relative Record				
8 - 11	D Preassigned Next Buffer	B Preassigned Next Messsage	B Message 1 Buffer 1 Unit 2	B Message 1 Buffer 1 Unit 3
20 - 23				
32 - 35				

Figure 12. Disk Queuing an Eight-Unit Buffer

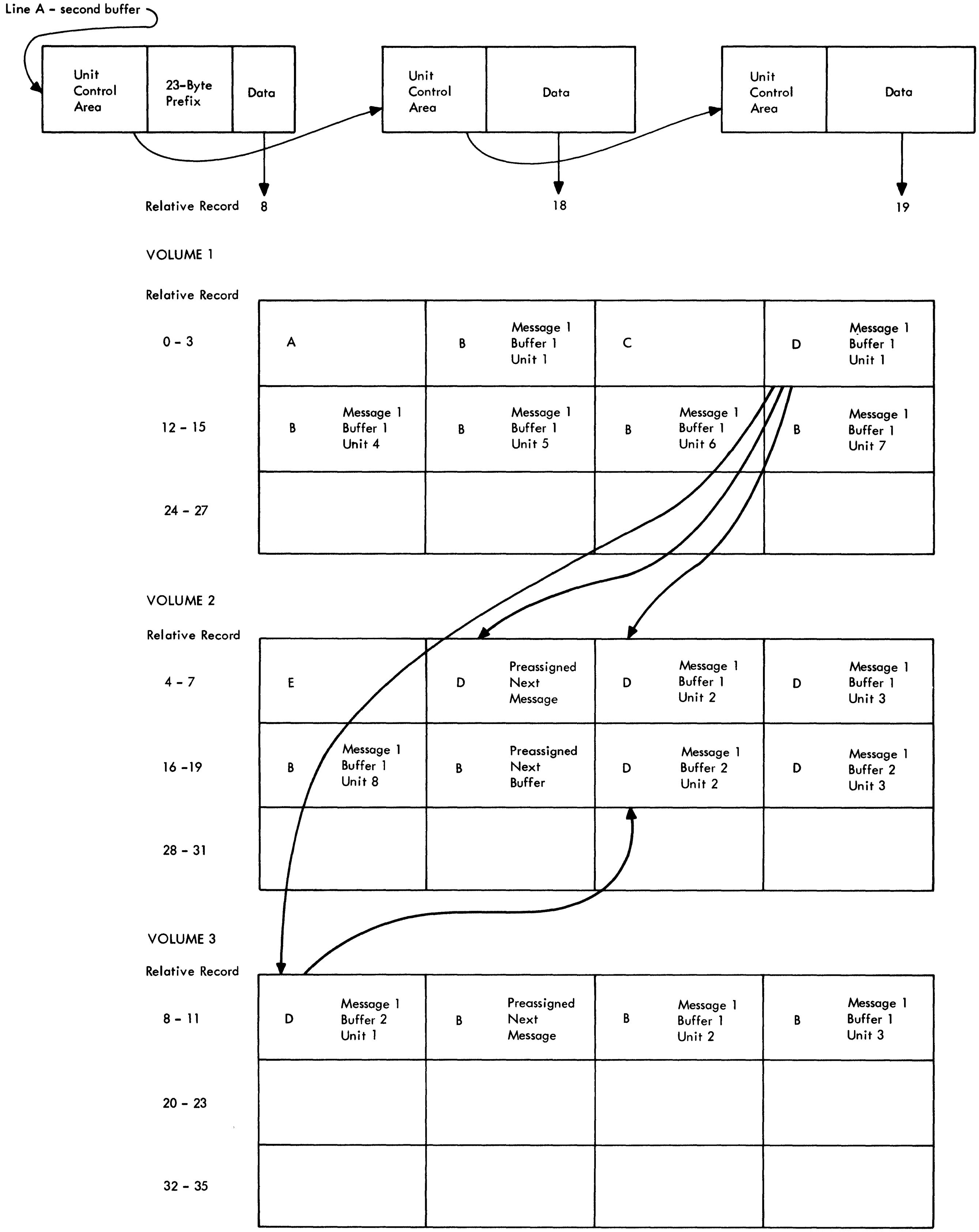
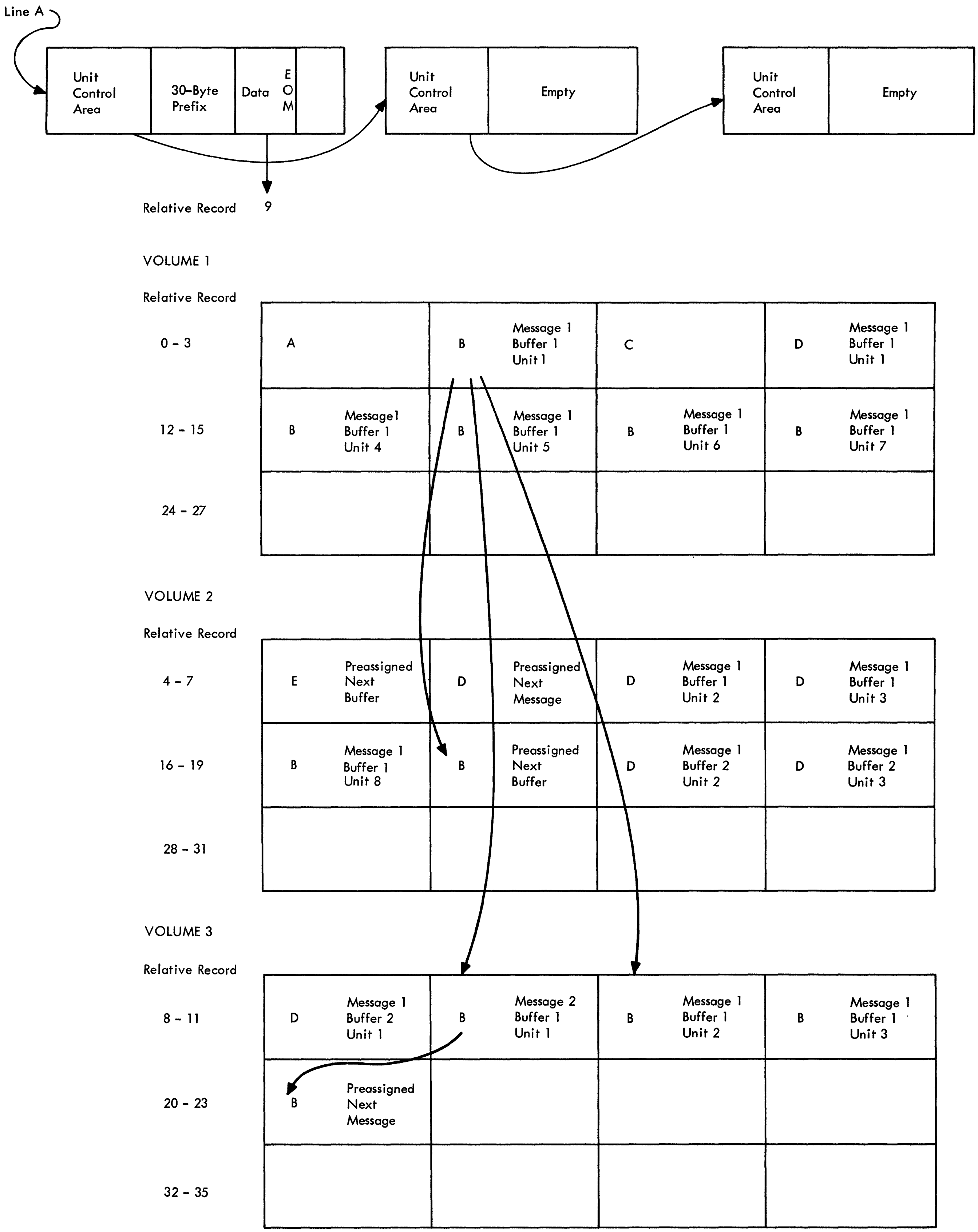

Line A - second buffer
Unit Control Area
23-Byte Prefix
Data
Unit Control Area
Data
Unit Control Area
Data
Relative Record
8
18
19
VOLUME 1
Relative Record
0 - 3
A
B
Message 1 Buffer 1 Unit 1
C
D
Message 1 Buffer 1 Unit 1
12 - 15
B
Message 1 Buffer 1 Unit 4
B
Message 1 Buffer 1 Unit 5
B
Message 1 Buffer 1 Unit 6
B
Message 1 Buffer 1 Unit 7
24 - 27
VOLUME 2
Relative Record
4 - 7
E
D
Preassigned Next Message
D
Message 1 Buffer 1 Unit 2
D
Message 1 Buffer 1 Unit 3
16 -19
B
Message 1 Buffer 1 Unit 8
B
Preassigned Next Buffer
D
Message 1 Buffer 2 Unit 2
D
Message 1 Buffer 2 Unit 3
28 - 31
VOLUME 3
Relative Record
8 - 11
D
Message 1 Buffer 2 Unit 1
B
Preassigned Next Message
B
Message 1 Buffer 1 Unit 2
B
Message 1 Buffer 1 Unit 3
20 - 23
32 - 35

Figure 13. Disk Queuing the Second Buffer of a Message



VOLUME 1

Relative Record				
0 - 3	A	B Message 1 Buffer 1 Unit 1	C	D Message 1 Buffer 1 Unit 1
12 - 15	B Message1 Buffer 1 Unit 4	B Message 1 Buffer 1 Unit 5	B Message 1 Buffer 1 Unit 6	B Message 1 Buffer 1 Unit 7
24 - 27				

VOLUME 2

Relative Record				
4 - 7	E Preassigned Next Buffer	D Preassigned Next Message	D Message 1 Buffer 1 Unit 2	D Message 1 Buffer 1 Unit 3
16 - 19	B Message 1 Buffer 1 Unit 8	B Preassigned Next Buffer	D Message 1 Buffer 2 Unit 2	D Message 1 Buffer 2 Unit 3
28 - 31				

VOLUME 3

Relative Record				
8 - 11	D Message 1 Buffer 2 Unit 1	B Message 2 Buffer 1 Unit 1	B Message 1 Buffer 1 Unit 2	B Message 1 Buffer 1 Unit 3
20 - 23	B Preassigned Next Message			
32 - 35				

Figure 14. Disk Queuing a One-Unit Message

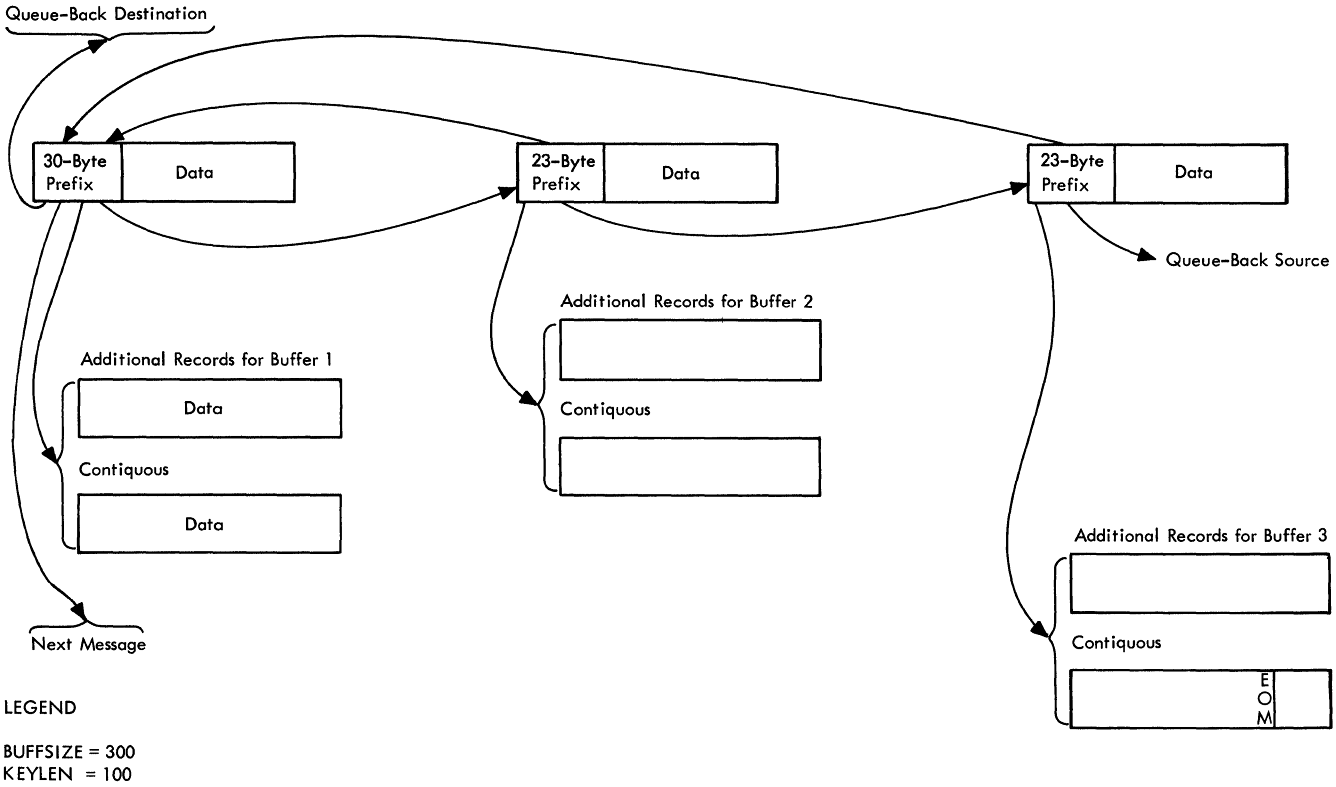


Figure 15. Disk Queuing Pointers

Queue-back Chain: A queue-back chain is a time-sequential record of the sending and receiving message traffic for the terminal or terminals of a specific destination QCB. TCAM maintains this chain for the message retrieval function of application programs. A message that has already been sent can be retrieved by source (input) or by destination (output) sequence number.

When the first buffer of a message is tposted to its destination QCB, the Destination Scheduler moves the current queue-back chain pointer (QCBQBACK) from the destination QCB to the PRFHQBCK field in the buffer prefix and then stores the disk relative record number assignment of the first unit of the buffer in the queue-back chain field of that destination QCB (QCBQBACK). The presence of a relative record number for the first buffer of a message in the queue-back chain of the destination QCB indicates that the message is to be queued for the terminal or terminals of the destination QCB.

When the last buffer of a message is tposted to its destination QCB, the Destination Scheduler uses the source destination offset in the buffer prefix (PRFSRCE) to gain access to the associated terminal entry. The location of the destination QCB for the sending (source) terminal is in this terminal entry. The scheduler then places the current destination QCB queue-back chain pointer (QCBQBACK) in the text queue-back field in the buffer prefix (PRFTQBCK) and places the disk relative record number (address) of the first unit of the last buffer-in the queue-back chain of the destination QCB (QCBQBACK) for the source terminal. The presence of a relative record number for the last buffer of a message in the queue-back chain of the destination QCB indicates that the message was sent from the terminal or terminals represented by that destination QCB.

An examination of the queue-back chain of a specific destination QCB indicates exactly which messages were sent from or received by the related terminal or terminals. If the value in the chain is for the first buffer of a message, the message was received by this terminal; if the value is for the last buffer of a message, the message was sent by this terminal. Since the prefix of a first buffer points to its subsequent buffer segment (PRFNTXT) and the prefix of a subsequent buffer segment points to its first buffer (PRFCHDR), the entire message is available from the queue-back chain pointers.

Note that if a message is only one buffer long, its relative record number location goes in both queue-back chains.

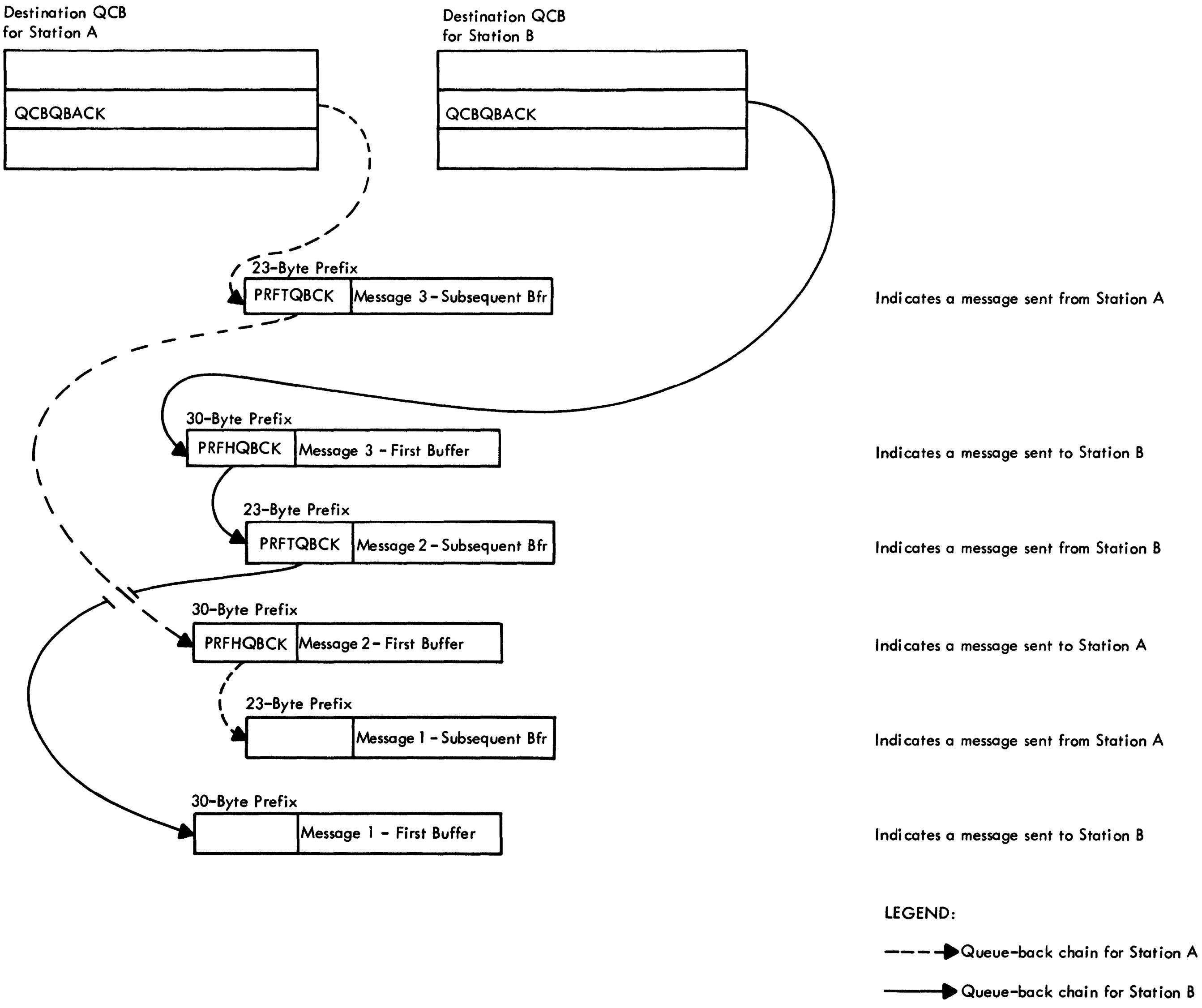

Destination QCB for Station A
QCBQBACK
Destination QCB for Station B
QCBQBACK
23-Byte Prefix
PRFTQBCK
Message 3 - Subsequent Bfr
Indicates a message sent from Station A
30-Byte Prefix
PRFHQBCK
Message 3 - First Buffer
Indicates a message sent to Station B
23-Byte Prefix
PRFTQBCK
Message 2 - Subsequent Bfr
Indicates a message sent from Station B
30-Byte Prefix
PRFHQBCK
Message 2 - First Buffer
Indicates a message sent to Station A
23-Byte Prefix
Message 1 - Subsequent Bfr
Indicates a message sent from Station A
30-Byte Prefix
Message 1 - First Buffer
Indicates a message sent to Station B
LEGEND:
Queue-back chain for Station A
Queue-back chain for Station B

Figure 16. Example of Two Queue-Back Chains

Figure 16 illustrates the queue-back chains for two destination QCBs. The following message sequence applies to this example:

Message 1—sent from Station A to Station B
Message 2—sent from Station B to Station A
Message 3—sent from Station A to Station B

Duplicate-Header Messages: When a message is identical to a message sent previously (as in multiple routing), it is called a *duplicate-header message.* This condition is indicated by a flag in bit 4 of the status field (PRFSTAT1) of the 30-byte buffer prefix. The Destination Scheduler handles a duplicate-header message just like any other message except that no additional record locations and no next-buffer location are assigned. The first unit of the first segment of a duplicate-header message contains the same pointers that are in the first unit of the first segment of the original message. TCAM modules use these pointers to obtain any additional units and buffers in the message.

FEFO Queuing: FEFO (first-ended-first-out) queuing is used in sending messages from the message queues data sets to destinations. This queuing allows TCAM to send the messages that end first, rather than the messages that begin transmission first.

Since the segments of a message cannot be kept in main storage until the message is complete, they must be queued (placed on the disk) as they are received. This results in a FIFO (first-in-first-out) message queue.

To create a chain of messages in FEFO order, the message with the previous EOM received for a destination QCB must chain to the message with the current EOM, regardless of which message began transmission first. This chaining pointer cannot be written until after the current EOM is received. When the current EOM is received, one message is completely on disk and the other is on disk except for the last segment. A chain of first-buffer prefixes is all that is required; therefore, the FEFO pointer can be written in the data field (at DATFEFO) of the record that contains the first-buffer (30-byte) prefix of the message already on disk at the same time the EOM segment of the current message is written.

When the first-ended message is to be sent and its first segment is read from disk, the FEFO pointer is read from the data field of the record and placed in the FEFO field of the SCB. When the first buffer is passed to the outgoing MH, the STARTMH subtask updates the FEFO field in the destination QCB. The message-serviced flag (X'40') is written in the disk data field along with the FEFO pointer when the EOM is successfully sent.

The destination QCB contains two FEFO pointers: the disk record address of the first FEFO message to send to the destination (QCBFFEFO) and the disk record address of the last message completely received (QCBLFEFO).

Figure 17 illustrates FEFO queuing for five messages routed to the same destination. Messages 1, 3, and 4 require two buffers, and messages 2 and 5 require one buffer. The first buffers of the messages arrive in the order in which the messages are numbered. The messages complete transmission in the following order: 2, 4, 3, 1, 5.

In this example, assume that the first buffers of messages 1, 2, 3, and 4 are already written on disk, message 2 is complete, and the first buffer of message 5 is currently being transmitted. The FEFO queuing activity proceeds as follows:

- Message 2 is written out on the line. No FEFO pointers were written when message 2 completed because it was the first message for the destination.
- Message 4 is completely received. Message 2 is still sending. QCBFFEFO and QCBLFEFO are updated to point to disk address 8. A FEFO pointer to message 4 is written in the disk data field of the first unit of the first buffer of message 2.
- Message 3 is completely received. A FEFO pointer to message 3 is written in the disk data field of the first unit of the first buffer of message 4. The destination QCB field QCBLFEFO is updated to point to disk address 7.
- Message 2 is completely sent. Message 4 is to be sent out. When the first buffer of message 4 is sent to the MH, its disk data field is used to update the QCBFFEFO field of the destination QCB to point to disk record 7.
- Message 1 is completely received. A FEFO pointer to message 1 is written in the disk data field of the first buffer of message 3. The destination QCB field QCBLFEFO is updated to point to disk address 1, the location of the first buffer of message 1.
- Message 5 is completely received. A FEFO pointer to message 5 is written in the disk data field of the first buffer of the last message received, message 1. The QCBLFEFO field is updated to disk address 10, the location of the first unit of the first buffer of message 5.
- Message 4 is completely sent. Message 3 is the next message to be sent. When the first buffer of message 3 is sent to the MH, its disk data field is used to update QCBFFEFO to point to message 1 in disk location 1, the next message to be sent.
- Message 3 is completely sent. Message 1 is the next message to be sent. When the first buffer of message 1 is sent to the MH, its disk data field is used to update QCBFFEFO to point to message 5 in disk location 10, the next message to be sent.
- Message 1 is completed, and message 5 is sent out. The QCBFFEFO pointer is cleared.

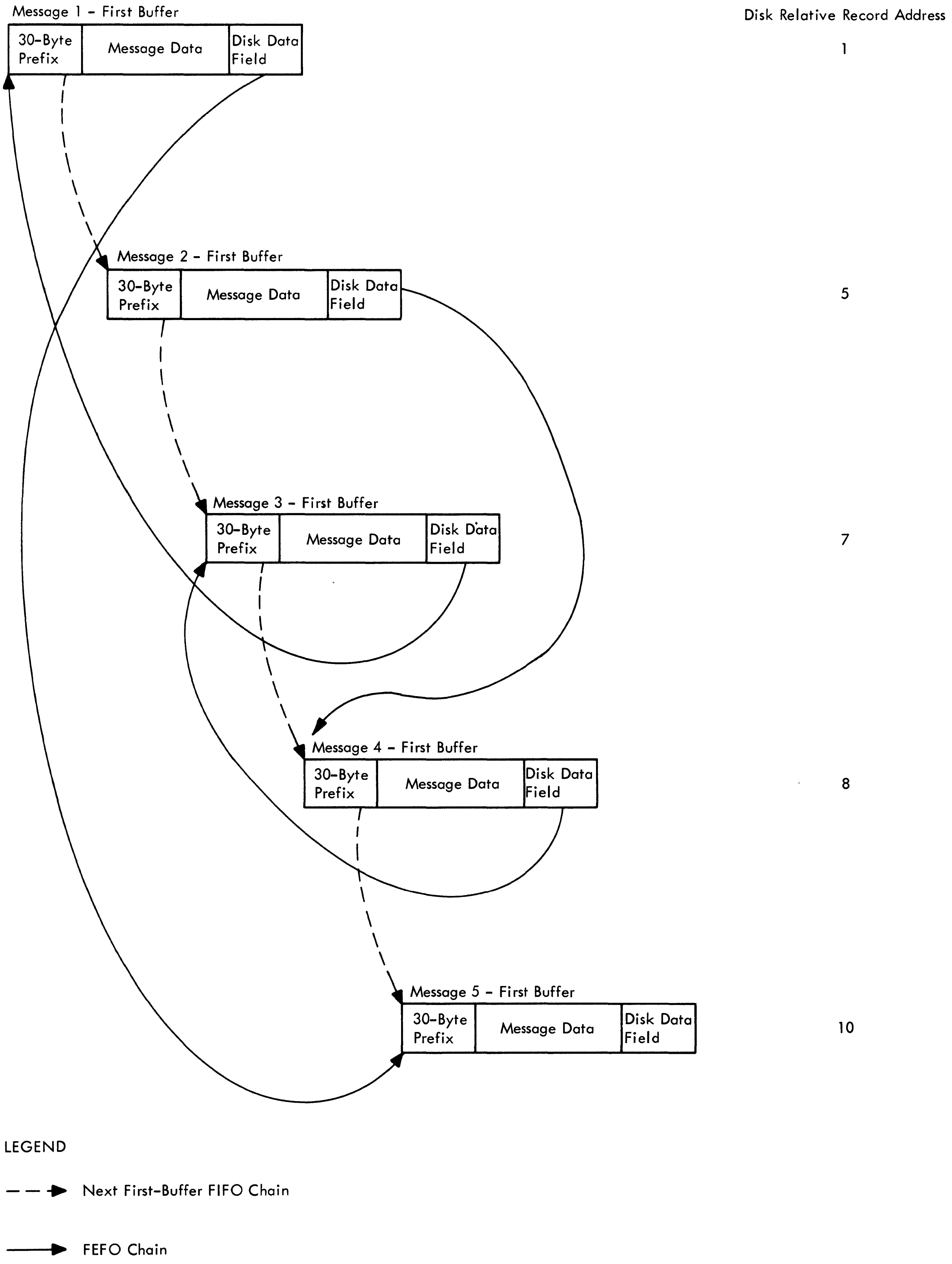


Figure 17. Disk Queuing—FIFO and FEFO Pointers

Hold Queues: When the HOLD macro is issued in the outgoing section of an MH, a special *hold queue* is built for multipoint terminals on a line that is queued by line.

When queuing multidrop terminals by line, the messages for the different terminals are intermixed on the destination queue. The Send Scheduler uses the FEFO chain to read one "first buffer of a message" after another. When a message for a held terminal is reached, it is placed in the hold queue chain.

A pointer to the first held message is placed in the QCBINTFF field of the destination QCB. When the next message to be held is encountered, no changes are made for the message in the QCB in the FEFO chain; QCBFFEFO is merely updated to point to the next FEFO message.

When a release occurs, the QCBINTFF field is moved into the QCBFFEFO field. The FEFO chain is again followed, skipping over those messages already marked serviced.

Queuing by terminal must be specified for dial lines, and messages are not intermixed on a message queue. In this case, only one message is in the hold queue, because the Send Scheduler determines that the terminal is held and does not request any more messages.

Reusable Disk Queuing

Reusable disk queuing uses a "wrapped" message queues data set, on which serviced messages are overlaid by new messages entering the system.

The Destination Scheduler activates the Reusability–Copy subtask to keep the data set "cleaned up" to avoid losing messages that have not been serviced. Message units are queued until 3/8 of the data set is full. At this point, the Reusability–Copy subtask examines the next-message field in each destination QCB for this data set. If any next-message field has a location value that falls within the scope of the first quarter of the data set, the subtask writes a dummy cancel message record at the specified next-message address and updates the next-message field in the QCB to the current adjusted address value at AVTRADDR in the AVT. This keeps new messages in close proximity on the data set.

The Reusability–Copy subtask performs the next-message update process each quarter of the way through the data set from this point on. For example, after 5/8 of the data set has been assigned to units, the Reusability–Copy subtask compares the address values in the second quarter to the next-message location specified in each destination QCB for this data set.

The Reusability–Copy subtask also handles log data sets. Although the LOGTYPE macro contains no ALTDEST= operand, the original destination is automatically designated as an alternate destination, which allows the subtask to perform zone reorganizations on log data sets on reusable disk.

The Reusability–Copy subtask sends to the specified alternate destination any unserviced messages located in the quarter that precedes the part of the data set that is getting dummy cancel record messages. The subtask does this by reading the old message from its current location and enqueuing the message to its alternate destination, thus causing the message to be written in the current zone of the data set.

If a duplicate-header message is more than a quarter of the data set away from the first unit of the first segment of the original message, the Reusability–Copy subtask copies the entire message.

The Reusability–Copy subtask gains control each time the address value reaches a zone mid-point (the middle of a quarter) of the data set. The only exception is that the first time through the data set, it is not activated until the address value is 3/8 of the way through the data set.

Figure 18 illustrates the part of the disk message queues data set that has had canceled messages issued to it and the part in which messages are sent to alternate destinations when the address value is at a specific zone mid-point.

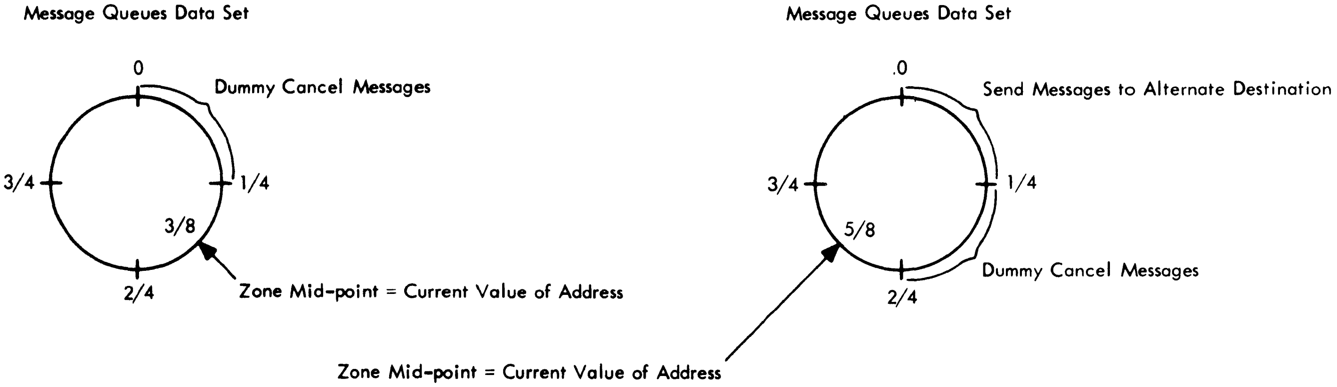

Message Queues Data Set
0
Dummy Cancel Messages
3/4
1/4
3/8
2/4
Zone Mid-point = Current Value of Address
Message Queues Data Set
.0
Send Messages to Alternate Destination
3/4
1/4
5/8
Dummy Cancel Messages
2/4
Zone Mid-point = Current Value of Address

Figure 18. Zones for Servicing and Updating a Reusable Disk Message Queues Data Set

Main-Storage Queuing

Main-storage queuing chains the actual main storage addresses of message units, rather than using relative record numbers. Once an entire message is queued, all the fields in the buffer prefix look the same as in disk queuing, except that the Destination Scheduler uses the additional units field (PRFXTRA) of the buffer prefix to hold the main-storage address of this unit and the current record field (PRFCRCD) to hold the disk address if disk backup is used. The scheduler uses the TIC field of the 12-byte unit control area that precedes each unit to chain units together.

Main-storage queuing does not assign locations ahead; rather, the destination QCB contains the address of the previous first-buffer segment, and the SCB contains the address of the previous subsequent-buffer segment. When the first segment of a message is received, the address of the previous first-buffer segment is inserted in the destination QCB in the previous first-buffer field (QCBCPVHD). When a message segment other than the first-buffer segment is received, its address is placed in the previous subsequent-buffer field of the SCB.

The Destination Scheduler does not build a queue-back chain for a main-storage message queues data set.

Main-Storage Queuing with Disk Backup

If the user specifies main-storage queuing with backup on either reusable or nonreusable disk, the message segments are first queued as described under *Main-Storage Queuing* and then the data is copied into buffers for the disk message queues data set and queued as described in the sections on disk queuing.

If the Destination Scheduler finds that the main-storage message queues data set does not contain enough free units to queue a message, the scheduler queues the message on disk only. Main-storage queuing resumes as soon as space is available. The CPB Initialization routine retrieves the messages queued on disk just as if they were placed in the main-storage data set.

Special Queuing Considerations

Duplicate-Header Message that Spans Queue-Type: A duplicate header message that spans queue-type is one that is tposted to a destination QCB that is to be queued in a manner other than that of the original message. For example, the original message is directed to a destination QCB that uses reusable disk queuing and the duplicate-header message is directed to a destination QCB that uses main-storage queuing with no disk backup.

If the entire message does not have to be copied, the Destination Scheduler moves the send scheduler STCB to the STCB chain of the LCB (if it is not already there) to service the message. If the message has to be copied, the Reusability–Copy subtask is activated.

Destination QCB for Main-Storage Queuing with Disk Backup: In this situation all recalls are from disk; therefore, the duplicate-header message is written on the disk data set only.

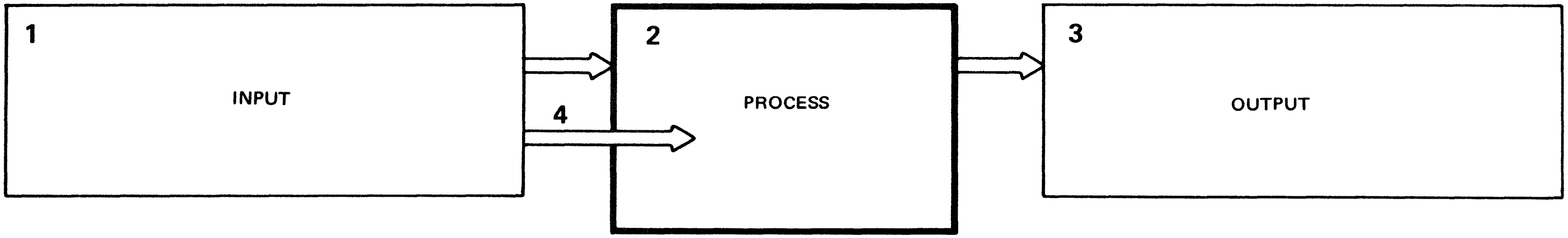
Main-Storage Queuing when Units Run Out: If a main-storage message queues data set fills up with data and there is a message segment unit to be queued, the Destination Scheduler acts according to the type of unit being processed. If the unit is not the first unit of the first segment of a message, the scheduler gets the first segment of the message, flags the message lost, and frees all the queued units except the first one.

If the unit to be queued is the first unit of the first segment of a message and one unit is available in the data set, the scheduler queues the unit and flags the message as lost by setting a flag in that unit. If no unit is available or if the count of units in the main-storage queue exceeds or equals the MSMAX= value specified on the INTRO macro, the scheduler queues the buffer unit that contains the first unit of the message into the data set, does not return a unit to the buffer unit pool in its place, and sets a flag to stop receiving activity. Receiving is resumed when enough messages have been sent to remove enough units from the message queues data set to lower the number of units used to or below the MSMIN= value specified on the INTRO macro.

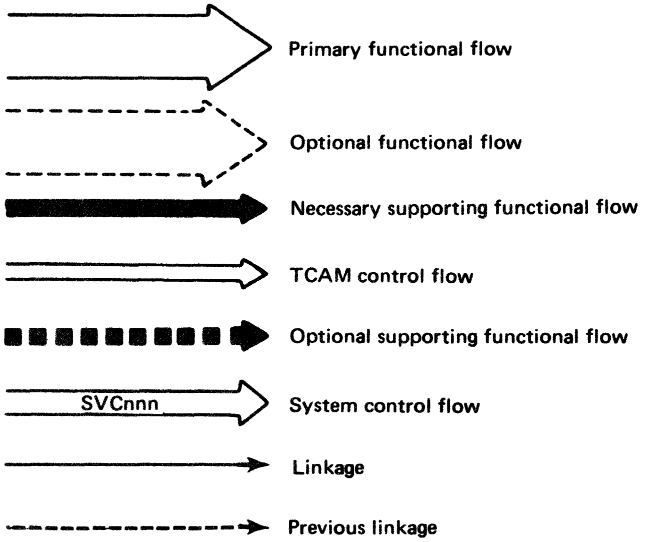
(This page left blank intentionally)

Section 3: Method of Operation

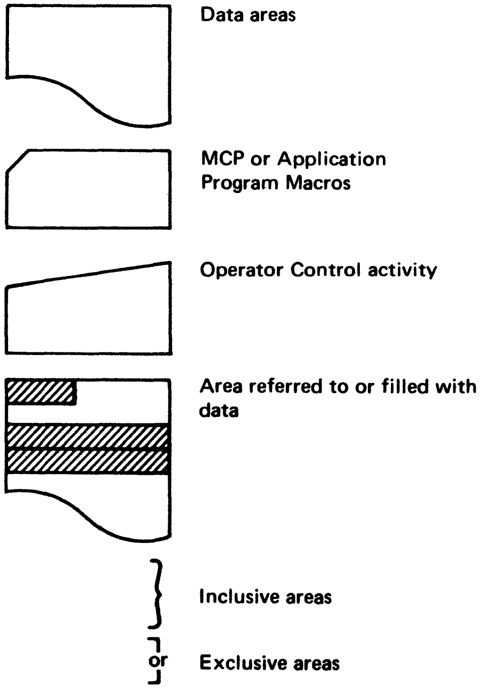
The diagrams in this section provide an overview of the basic functional structure of a TCAM system. The diagrams alone provide general concepts and can be used for quick reference. Extended descriptions accompany the diagrams to provide more detailed information—bit settings, field descriptions, names of modules performing the functions, and register usage. The diagrams are divided into three general areas:



LEGEND:



Conventions Used:


Data areas
MCP or Application Program Macros
Operator Control activity
Area referred to or filled with data
Inclusive areas
or
Exclusive areas

Method of Operation Introduction

Item Description

1. The left area of the diagram contains the input required to perform a given function. This input can be data areas, registers, parameter lists, and the like. When more than one field in a data area is refered to, the fields are illustrated schematically, rather than contiguously. Contiguity is shown when possible. In like manner, these fields are represented sequentially whenever possible.

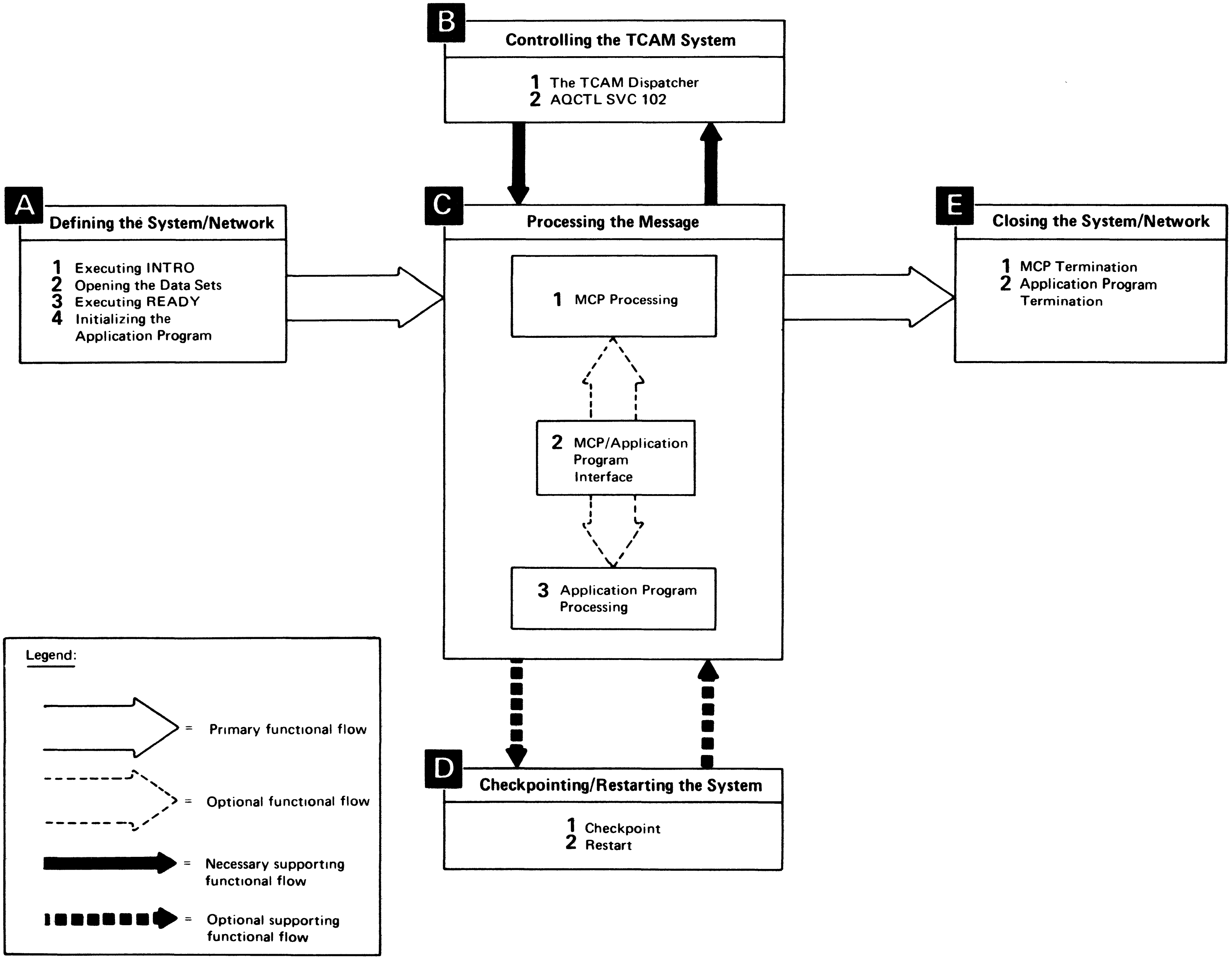
2. The central area of the diagram contains the processing steps required to perform a given function. The numbering of these steps does not necessarily indicate sequence, as some steps are executed concurrently. In some instances, processing steps are further subdivided into substeps, indicated by lowercase alphabets.

   The information in this processing area is presented on a high level; see the accompanying extended description for each diagram for more detailed information. The numbers associated with the processing steps correspond to the numbers in the extended description section.

3. The right area of the diagram contains the output resulting from the processing step. Just as with the input, the output can be in the form of data areas, parameter lists, addresses in registers, and the like. The same conventions are applicable to both input and output.

4. When input is for a particular substep of processing, the data flow arrow from the input area penetrates the processing box. When input is for an entire step, the arrow does not penetrate the box.

Note: *Sometimes the input to or output from a particular step will be represented in a detailed manner the first time, while subsequent references to the same input or output will be less detailed.*


B
Controlling the TCAM System
1 The TCAM Dispatcher
2 AQCTL SVC 102
A
Defining the System/Network
1 Executing INTRO
2 Opening the Data Sets
3 Executing READY
4 Initializing the Application Program
C
Processing the Message
1 MCP Processing
2 MCP/Application Program Interface
3 Application Program Processing
E
Closing the System/Network
1 MCP Termination
2 Application Program Termination
D
Checkpointing/Restarting the System
1 Checkpoint
2 Restart
Legend:
= Primary functional flow
= Optional functional flow
= Necessary supporting functional flow
= Optional supporting functional flow

Method of Operation Charts Overview

A. DEFINING THE SYSTEM/NETWORK shows the operations that must be performed before TCAM can be activated. These operations include defining and initializing control blocks and work areas, and opening data sets. Once this is done, TCAM is ready to process a message.

B. CONTROLLING THE TCAM SYSTEM describes the TCAM dispatcher and SVC 102 as tools used by the message control program (MCP) to process a message. Control passes from the MCP to the TCAM dispatcher and/or SVC 102 as their functions are needed.

C. PROCESSING A MESSAGE traces a message through the receiving, queuing, and sending functions. Application-program message processing is shown as an optional subset of MCP message processing.

D. CHECKPOINTING/RESTARTING THE SYSTEM describes the TCAM option that provides recovery from a system failure or restart after normal closedown. If checkpoint/restart is activated, it receives control from and passes control back to the message processing routines.

E. CLOSING THE SYSTEM/NETWORK describes the MCP closedown, and if application programs are active, their closedown. This is the last function performed in the TCAM system.

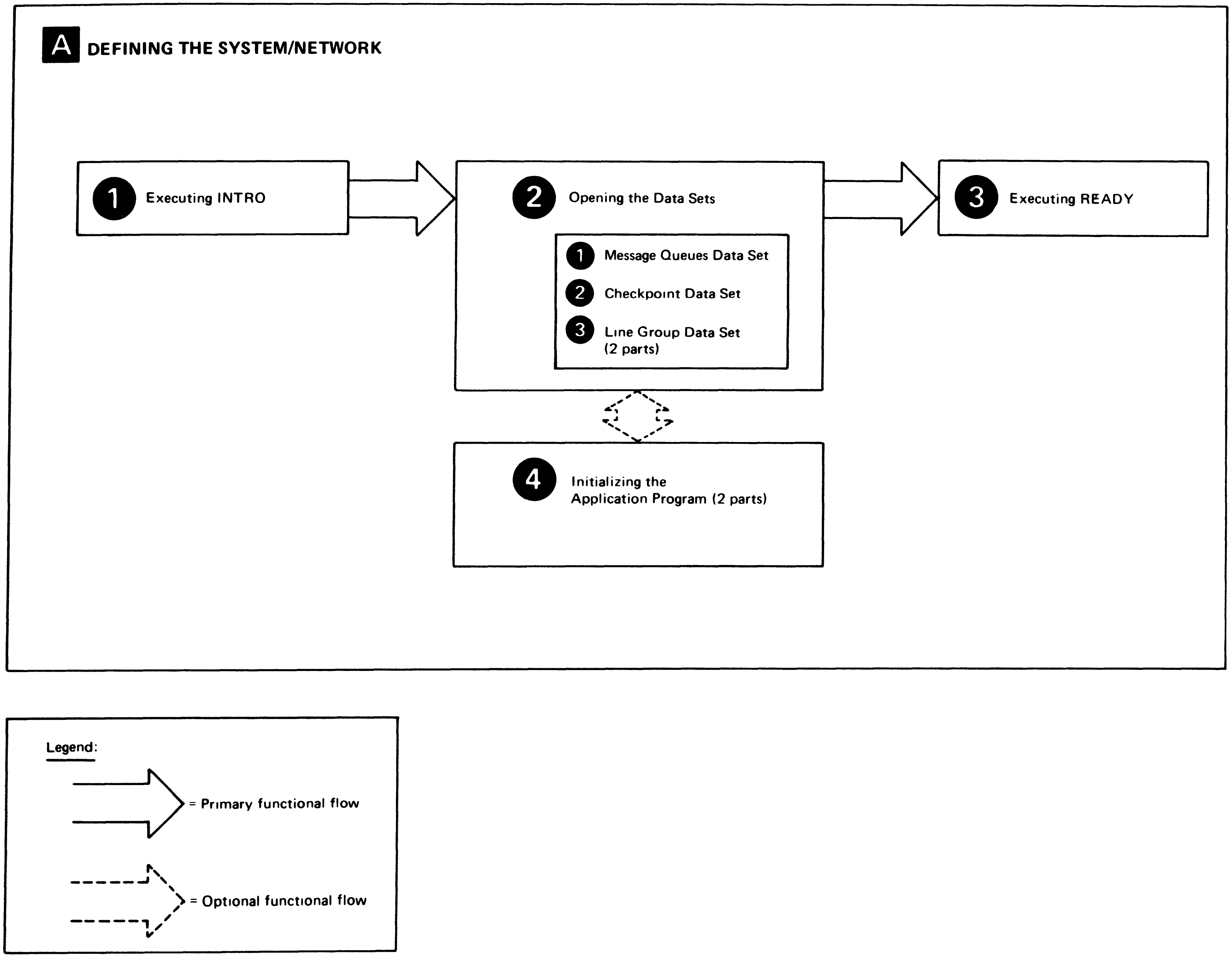

A DEFINING THE SYSTEM/NETWORK
1 Executing INTRO
2 Opening the Data Sets
1 Message Queues Data Set
2 Checkpoint Data Set
3 Line Group Data Set
(2 parts)
3 Executing READY
4 Initializing the
Application Program (2 parts)
Legend:
= Primary functional flow
= Optional functional flow

Chart A Defining the System/Network

Description	**Chart No.**
Executing INTRO	A1
describes using the parameters from the INTRO macro to define and initialize data areas and to create buffers and trace tables.	
Opening the Data Sets	A2-1
Message Queues Data Set	
Checkpoint Data Set	A2-2
Line Group Data Set	A2-3
Application Program Data Set	A3
Executing READY	A4
describes building a parameter list for the TCAM dispatcher and activating the ready queues and destination queues.	

Chart A1 Executing INTRO

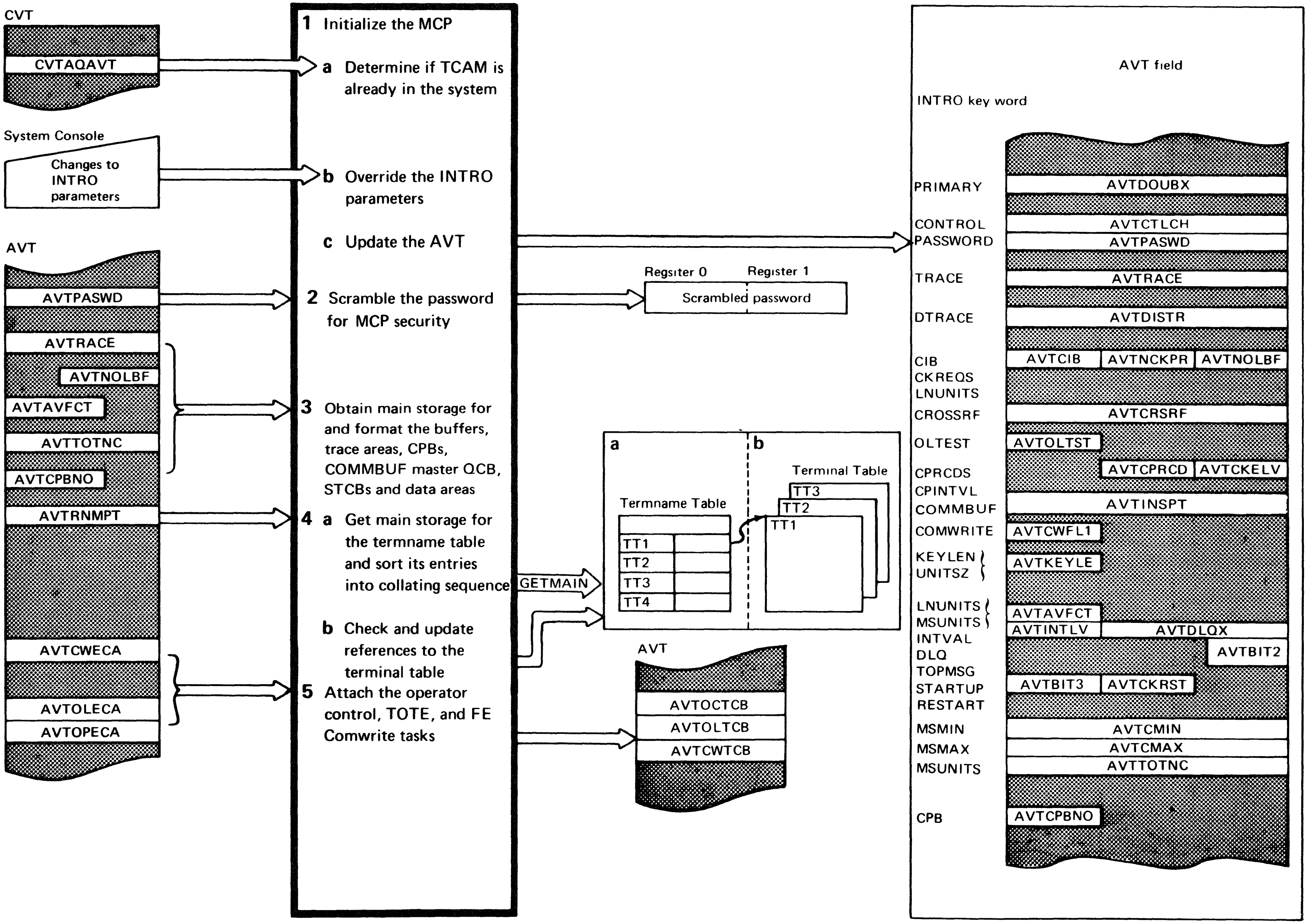


Chart A1 Executing INTRO—Description (1 of 2)

	Description	Routine	Register Usage
1a	If CVT+240 (CVTAQAVT) is nonzero, the MCP is already in the system.Return to IEDQOA with an error return code of 4; otherwise, continue processing.	IEDQOA	R1 I—AVT address R15 0—return code
1b	Check for valid keywords and parameters. Keywords that may be changed are STARTUP, LNUNITS, MSUNITS, KEYLEN, RESTART, UNITSZ, CPINTVL, CONTROL, PRIMARY, INTVAL, PASSWRD, CKREQS, CPB, CPRCDS, CROSSRF, COMWRTE, TRACE, DTRACE, CIB, MSMIN, MSMAX, DLQ, OLTEST, and TOPMSG.		
1c	Store the requested keyword parameters in the AVT. Return to IEDQOA with a return code of 0 indicating successful completion.		
2	Get the password from AVTPASWD (eight bytes) and rearrange the characters of the password.	IEDQE6	R0 and R1 0—scrambled password
3	Get main storage for and initialize the following areas, as requested: Main-storage message queues data set Channel program blocks Trace tables Cross-reference table Line buffers COMMBUF Master QCB STCBs Data areas Return to IEDQOA with a return code of X'00' for successful, or X'08' for unsuccessful, completion of the GETMAIN operation.	IEDQOA	R1 I—AVT address R14 0—IEDQOA address R15 0—return code

(This page left blank intentionally)

Chart A1 Executing INTRO—Description (2 of 2)

	Description	Routine	Register Usage
4a	Get main storage for the termname table and store the table address in AVTRNMPT. Sort the termname table entries into collating sequence. Recalculate the termname table offsets for distribution, cascade, and invitation lists that refer to specific entries in the table, and for alternate destinations.	IEDQOA	R1 I—AVT address R15 O—return code
4b	If requested, store the offsets of the primary operator control terminal in AVTOPCON and the offset of the dead-letter queue in AVTDLQX. If the dead-letter queue is specified as a TSO terminal, issue an error message and place zeros in AVTDLQX. The following return codes are set before returning to IEDQOA: X'00'—routine executed successfully X'12'—insufficient main storage available for GETMAIN macro X'16'—terminal definition error X'20'—primary operator control terminal definition error		R15 O—return code
5	Attach the following tasks: Operator control task—address in AVTOPECA TOTE (Terminal On-Line Test Executive)—address in AVTOLECA COMWRITE (FE Common Write)—address in AVTCWECA Load the following routines, if requested on the INTRO macro: System delay subtask (IEDQHI), if the system delay interval (AVTINTVL)is not equal to zero. Operator Awareness Message Router (IEDQNX), if the system console is not the primary operator control terminal. Place its address in AVTNX.	IEDQOA	

Chart A2-1 Opening the Message Queues Data Set

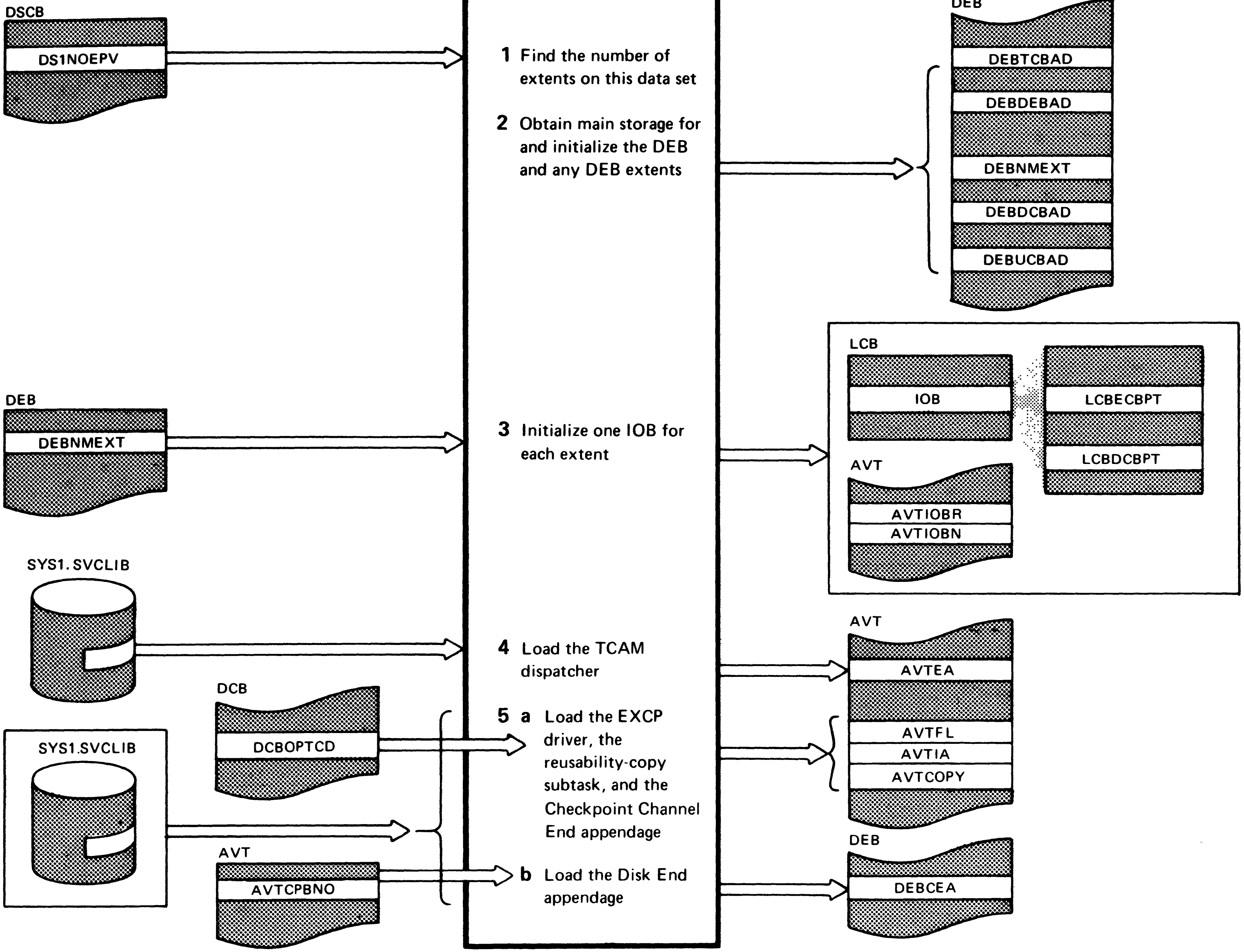
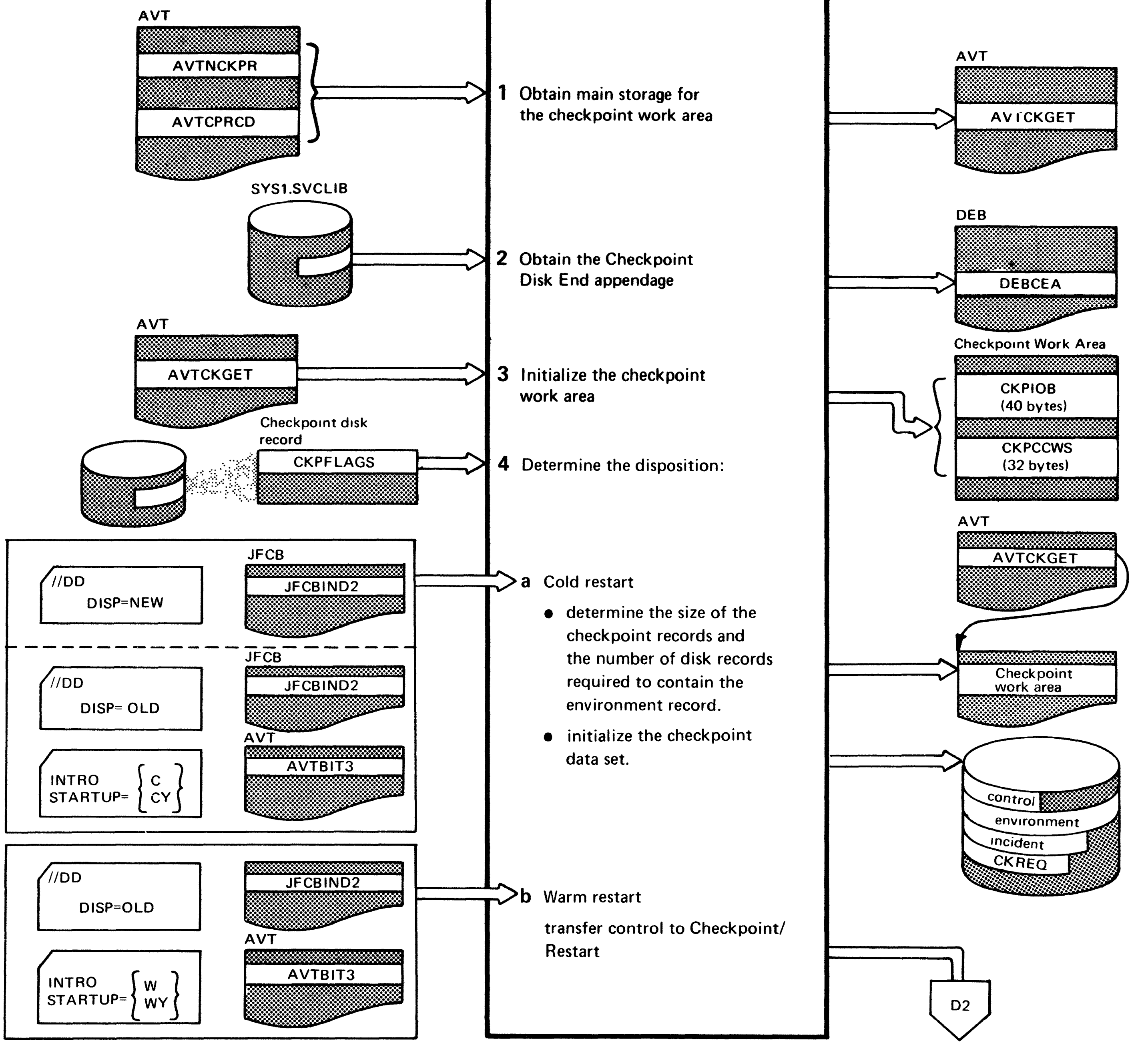

DSCB
DS1NOEPV
1 Find the number of extents on this data set
2 Obtain main storage for and initialize the DEB and any DEB extents
DEB
DEBTCBAD
DEBDEBAD
DEBNMEXT
DEBDCBAD
DEBUCBAD
DEB
DEBNMEXT
3 Initialize one IOB for each extent
LCB
IOB
LCBECBPT
LCBDCBPT
AVT
AVTIOBR
AVTIOBN
SYS1.SVCLIB
4 Load the TCAM dispatcher
AVT
AVTEA
DCB
DCBOPTCD
SYS1.SVCLIB
5 a Load the EXCP driver, the reusability-copy subtask, and the Checkpoint Channel End appendage
AVTFL
AVTIA
AVTCOPY
AVT
AVTCPBNO
b Load the Disk End appendage
DEB
DEBCEA

Chart A2-1 Opening the Message Queues Data Set—Description

	Description	Routine	Register Usage
1	Determine the number of extents from the DS1NOEPV field of the DSCB. DSCBs are built by the system open routines before TCAM open.	IGG01931	R1 I—DSCB address
2	Issue a GETMAIN macro for main storage (from subpool 234) for the DEB and any DEB extents. Initialize the following DEB fields: DEBTCBAD—TCB address for this DEB DEBDEBAD—next DEB address DEBNMEXT—number of DEB extents DEBDCBAD—address of DCB associated with this DEB DEBUCBAD—address of UCB associated with this data set	IGG01930	R1 I—DEB address
3	Build the IOBs in the line control blocks (LCBs). Initialize LCBECBPT with the ECB address and LCBDCBPT with the DCB address. Update the AVTIOBR field (address of a series of IOBs—reusable disk queuing) and AVTIOBN field (address of a series of IOBs— nonreusable disk queuing).	IGG01931	
4	Load the the TCAM dispatcher from SYS1.SVCLIB (IGG019RB or, if the DTRACE= value is greater than zero, IGG019RO). Place the TCAM dispatcher address into AVTEA. Place a pointer to the AVT address at CVT+240.	IGG01934	
5a	If the DCBOPTCD field is X'01' (OPTCD=R), load the reusability-copy subtask. If DCBOPTCD is X'02' (OPTCD=L), determine if the MSUNITS= value is not equal to zero (AVTTOTNC≠0). Load the reusability-copy subtask in AVTIA. If the DCBOPTCD field is X'20' (OPTCD=C), indicating a checkpoint DCB, the open routine loads the Checkpoint Channel End appendage (IGG019RA); the number of pages occupied by the appendage is placed in the high-order byte of the appendage address field (DEBCEA).		
5b	If the AVTCPBNO field (the CPB= value from the INTRO macro) in thethe AVT is equal to 1, load the Disk End appendage for a single CPB (IGG019RK); otherwise, load the Disk End appendage (IGG019R2). Both modules contain the Start I/O appendage for disk.		

Chart A2-2 Opening the Checkpoint Data Set



1 Obtain main storage for the checkpoint work area

2 Obtain the Checkpoint Disk End appendage

3 Initialize the checkpoint work area

4 Determine the disposition:

a Cold restart

- determine the size of the checkpoint records and the number of disk records required to contain the environment record.
- initialize the checkpoint data set.

b Warm restart

transfer control to Checkpoint/ Restart

Chart A2-2 Opening the Checkpoint Data Set—Description (1 of 2)

	Description	Routine	Register Usage
1	Use AVTNCKPR and AVTCPRCD to calculate the storage needed for a checkpoint work area. AVTNCKPR—maximum decimal number of destination queues (obtained from the CKREQ parameter of the INTRO macro) in use at any time for application programs using a CKREQ macro. AVTCPRCD—number of environment records (obtained from the CPRCDS= parameter of the INTRO macro) to be retained in a checkpoint data set at any one time. Issue a GETMAIN macro to obtain the necessary main storage and place the address of the work area into AVTCKGET. For error conditions, IGG01941 sends an error message to the system console, sets AVTCKGET to zero, and passes control to the next module in the where-to-go table. Error conditions: • Insufficient main storage for GETMAIN • Disk I/O error while reading the control record of a checkpoint data set.	IGG01941	R2 O—checkpoint work area address R8 I—address of current entry in where-to-go table O—address of next entry R9 I—AVT address R15 O—return code
2	Load the Checkpoint Disk End appendage from SYS1.SVCLIB. Calculate the amount of main-storage occupied by the appendage and place that value in the high-order byte of DEBCEA.		
3	Get the address of the work area from AVTCKGET and build a 40-byte IOB beginning at CKPIOB. Build a 32-byte channel program beginning at CKPCCWS.		

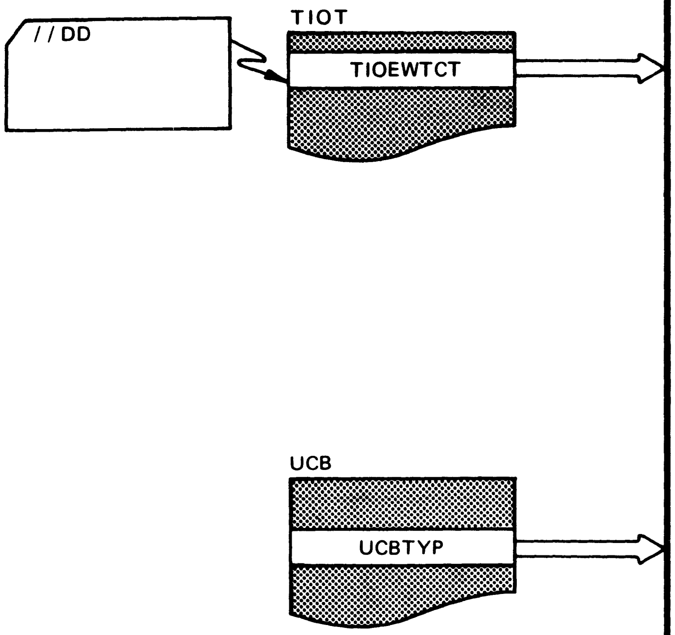
Chart A2-2 Opening the Checkpoint Data Set—Description (2 of 2)

Description	Routine	Register Usage
4 Determine the type of start or restart necessary by examining the following fields: Normal or abnormal closedown—checkpoint disk record +0 (CKPFLAGS) Disposition—JFCB+87 (JFCBIND2) X'40'—OLD data set X'80'—MOD data set X'C0'—NEW data set Startup—AVT+1052 (AVTBIT3) C—cold restart W—warm restart Perform the restart necessary according to the following input specifications: DISP=NEW XCTL to the Checkpoint Disk Allocation routine. DISP=OLD, S=C, normal closedown XCTL to the Checkpoint Disk Allocation routine. DISP=OLD, S=C, abnormal closedown XCTL to the checkpoint/restart modules and scan the message queues. DISP=OLD, S=CY, normal closedown XCTL to the Checkpoint Disk Allocation routine. DISP=OLD, S=CY, abnormal closedown XCTL to the Checkpoint Disk Allocation routine.		

Chart A2-2 Opening the Checkpoint Data Set—Description (2 of 2) *Continued*

	Description	Routine	Register Usage
4a	Scan the TCAM tables to determine the size of the environment record and the number of disk records needed to contain it. Calculate the number of each type of checkpoint record that will fill one track of the checkpoint data set. Use the device type index from the UCBTYP field of the UCB and the I/O device table (address at CVTZDTAB) to calculate the number of tracks in the checkpoint data set. Use the maximum number of priority QCBs to be used for any one application program destination QCB plus the length of the longest option area for any terminal entry to calculate the length of a CKREQ record. The length of an incident record is equal to the length of the longest option area or the length of the operator control data area, whichever is greater. DISP=OLD, S=W, normal closedown XCTL to the checkpoint/restart modules and do not scan the message queues. DISP=OLD, S=W, abnormal closedown XCTL to the checkpoint/restart modules and scan the message queues. DISP=OLD, S=WY, normal closedown XCTL to the checkpoint/restart modules and do not scan the message queues. DISP=OLD, S=WY, abnormal closedown XCTL to the checkpoint/restart modules and do not scan the message queues.	IGG01949	
4b	Format the checkpoint data set. The number of environment records is at AVT+681 (AVTCPRCD); the number of CKREQ records is at AVT+453 (AVTNCKPR). There is one control record, and the remainder of the disk space is used for incident records.		

Chart A2-3 Opening the Line Group Data Set (Part 1 of 2)



1 Initialize the line group:

a Determine the number of lines in the line group

b Obtain main storage for and initialize a DEB

c Initialize channel programs for the device

d Obtain main storage for and initialize an LCB for each line

e Place the send scheduler STCB in the STCB chain of the destination QCB and of the LCB

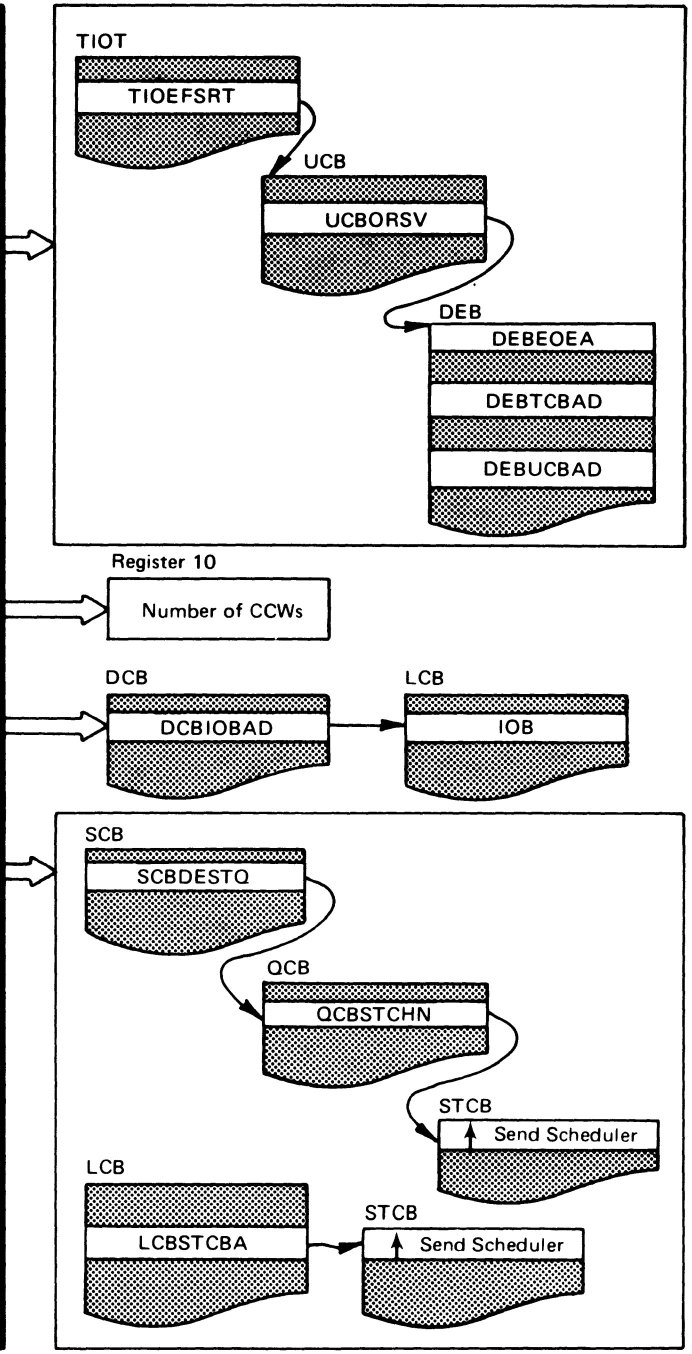
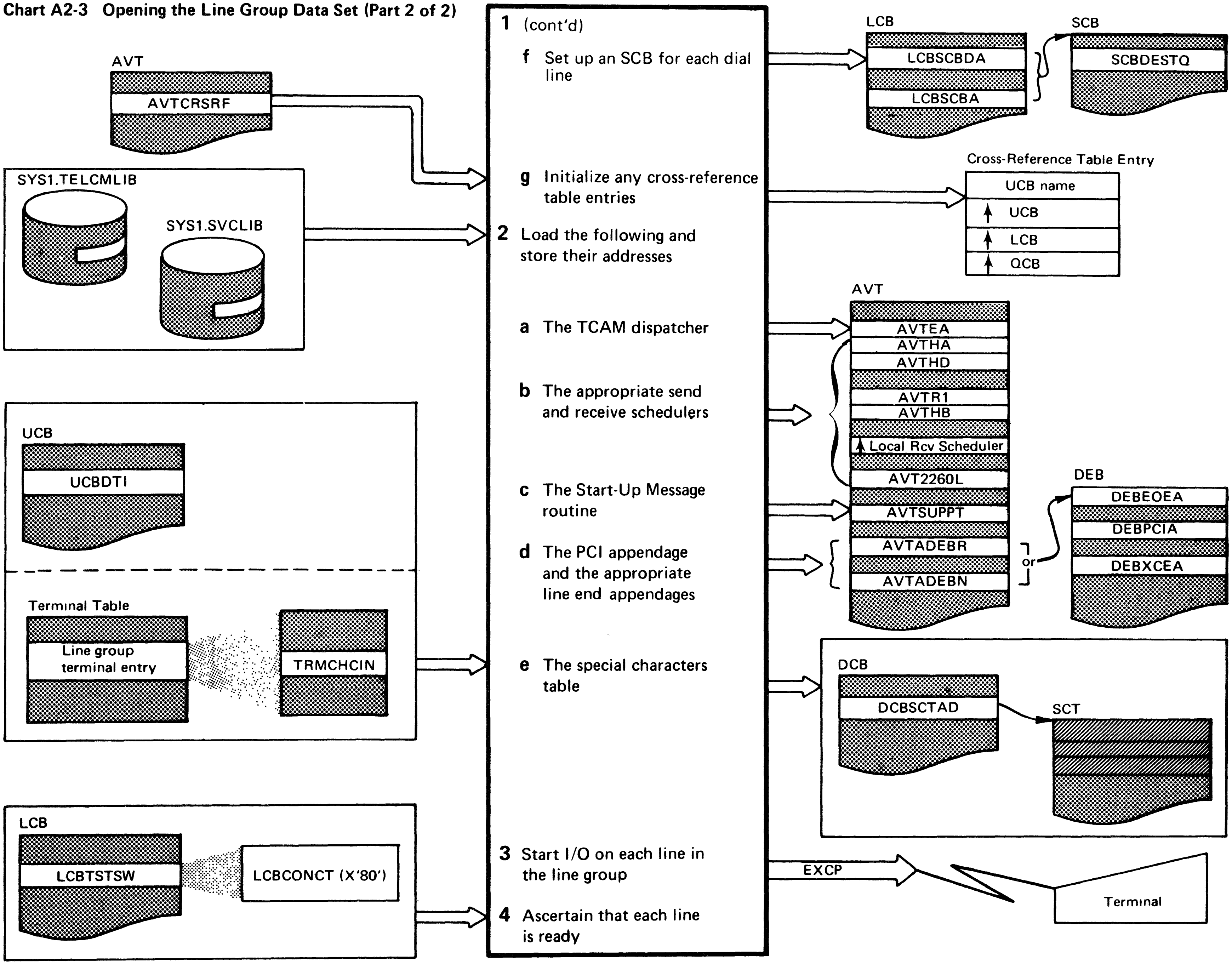


Chart A2-3 Opening the Line Group Data Set—Description (1 of 2)

	Description	Routine	Register Usage
1a	Examine the TIOEWTCT field of the task I/O table to determine the number of lines in this line group.	IGG01935	
1b	Issue a GETMAIN macro to get storage from subpool 234 for the DEB. Initialize the DEBUCBAD field with the UCB address from the TIOT (TIOEFSRT). UCBORSV is the address of the DEB for the first user on the queue for this device. Determine the size of the LCB. If this is the first OPEN for the line group, transfer control to IGG01932. If not the first OPEN, continue.	IGG01936	R2 I—current DCB address R13 O—total number of CCWs
1c	Use the information from the UCBTYP fields of the UCBs to build channel programs for each line in the line group.		
1d	Issue a GETMAIN macro to get an LCB for each line in the line group. Divide the LCB area into individual LCBs, and put the IOB address (LCB+32) into DCBIOBAD.		
1e	Place the Send scheduler STCB in the STCB chain of the destination QCB. If send priority is specified, move the Send scheduler STCB into the STCB chain of the LCB. SCBDESTQ is the address of the destination QCB and QCBSTCHN is the address of the first element in the STCB chain.	IGG01936 IGG01937	

Chart A2-3 Opening the Line Group Data Set (Part 2 of 2)



1 (cont'd)

f Set up an SCB for each dial line

g Initialize any cross-reference table entries

2 Load the following and store their addresses

a The TCAM dispatcher

b The appropriate send and receive schedulers

c The Start-Up Message routine

d The PCI appendage and the appropriate line end appendages

e The special characters table

3 Start I/O on each line in the line group

4 Ascertain that each line is ready

Chart A2-3 Opening the Line Group—Description (2 of 2)

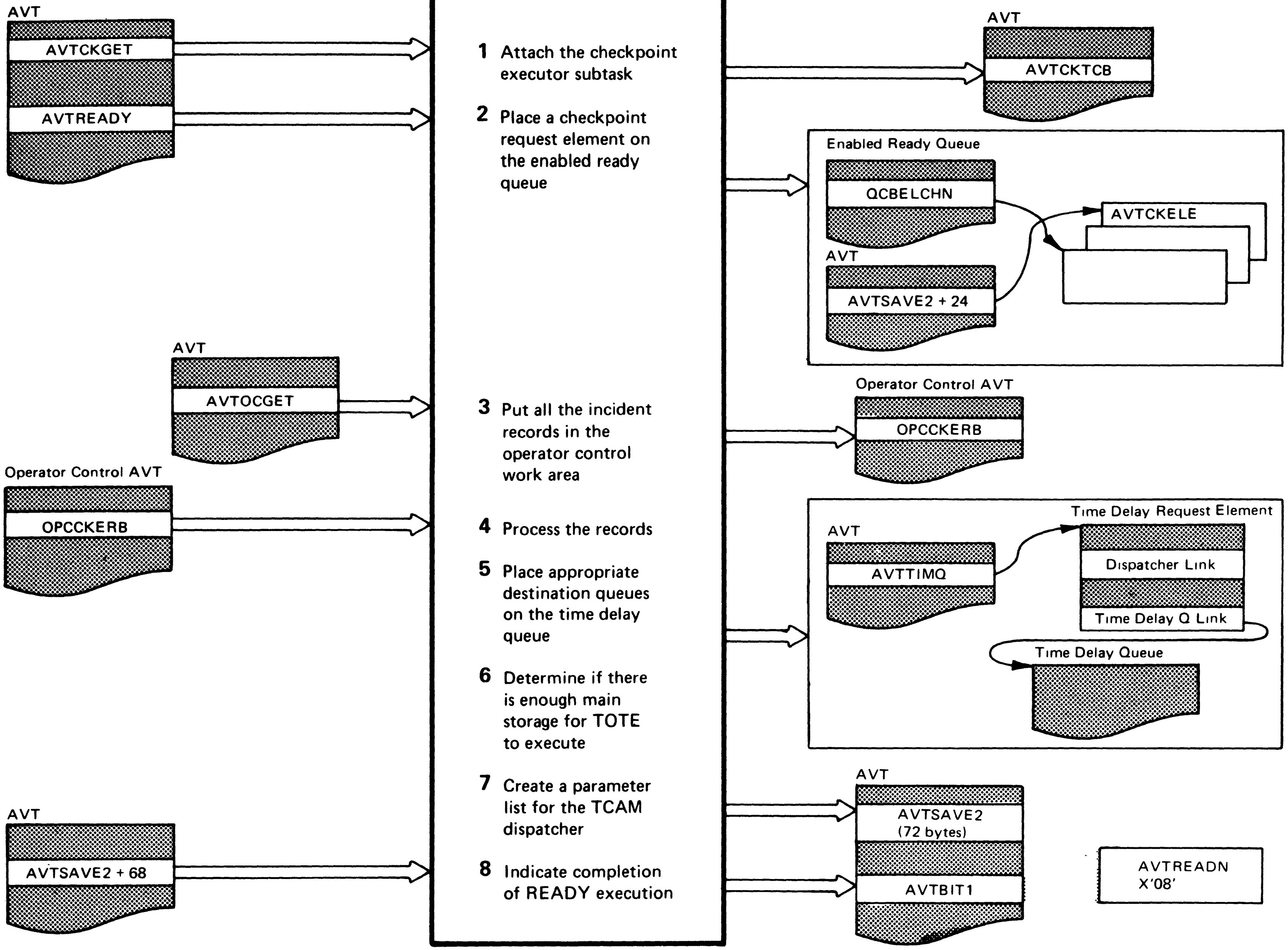
	Description	Routine	Register Usage
1f	Build and initialize SCBs for dial terminals using the address of the current SCB from LCBSCBA and the address of the SCB directory from LCBSCBDA.	IGG01936	
1g	Each time a line is successfully opened, complete the next entry in the cross-reference table. AVTCRSRF contains the address of the cross-reference control table. There is a 4-word entry in the cross-reference table for each open line.	IGG01948	
2a	If the TCAM dispatcher has not previously been loaded, load the appropriate version (IGG019RB or IGG019RO) from SYS1.SVCLIB. Place the address of the TCAM dispatcher into AVTEA. If the I/O supervisor loads the TCAM dispatcher, it also places a pointer to the AVT address at CVT+240. If the TCAM dispatcher has already been loaded, update the use count in the contents directory.	IGG01939	R9 I—AVT address
2b	Load the Send and Receive schedulers that are appropriate for this TCAM system and store their addresses in the AVT: AVTHA—address of the Receive scheduler AVTHD—address of the Send scheduler AVTR1—address of the Dial scheduler AVTHB—address of the Buffer scheduler AVT+588—address of the Local Receive scheduler AVT2260L—address of the 2260 Local Receive scheduler	IGG01939 IGG01940	
2c	Load the Start-Up Message routine (IGG019R6) and place its address into AVTSUPPT.	IGG01939	
2d	Load the PCI appendage (IGG019RN). Store its address in DEBPCIA. Also load one of the following line end appendages: QTAM-compatible system—IEDQKE BSC lines—IEDQKB leased and start-stop lines with no TSO start-stop lines—IEDQKD Fields used for this operation are: AVTADEBR—address of the DEBEOEA field for reusable disk message queues data sets. AVTADEBN—address of the DEBEOEA field for nonreusable disk message queues data sets. DEBEOEA—address of the End-of-Extent appendage DEBPCIA—address of the PCI appendage DEBEXCEA—address of the Abnormal End appendage	IGG01940	R9 I—AVT address

(This page left blank intentionally)

Chart A2-3 Opening the Line Group—Description (2 of 2) *Continued*

	Description	Routine	Register Usage
2e	Use information from the UCB and the terminal entry to load the special characters table (SCT) from SYS1.SVCLIB. UCBDTI is an index into the device table and TRMCHCIN is an index into the device characteristics table. Store the SCT address in DCBSCTAD.		
3	Issue an EXCP macro (SVC 0) to start I/O on each line.		
4	Issue the TIME macro. Test the LCBTSTSW byte for X'80' (successful initial I/O operation) in the LCB for each line. If the initial I/O is not complete, determine whether 28 seconds have elapsed since the EXCP macro was issued. Continue checking for I/O completion until either 28 seconds have elapsed or until LCBTSTSW=X'80' indicating I/O completion. At the end of 28 seconds when I/O completion has not occurred, write a message to the system console to identify the line that has not been successfully opened.	IGG01948	

Chart A3 Executing READY



1 Attach the checkpoint executor subtask

2 Place a checkpoint request element on the enabled ready queue

3 Put all the incident records in the operator control work area

4 Process the records

5 Place appropriate destination queues on the time delay queue

6 Determine if there is enough main storage for TOTE to execute

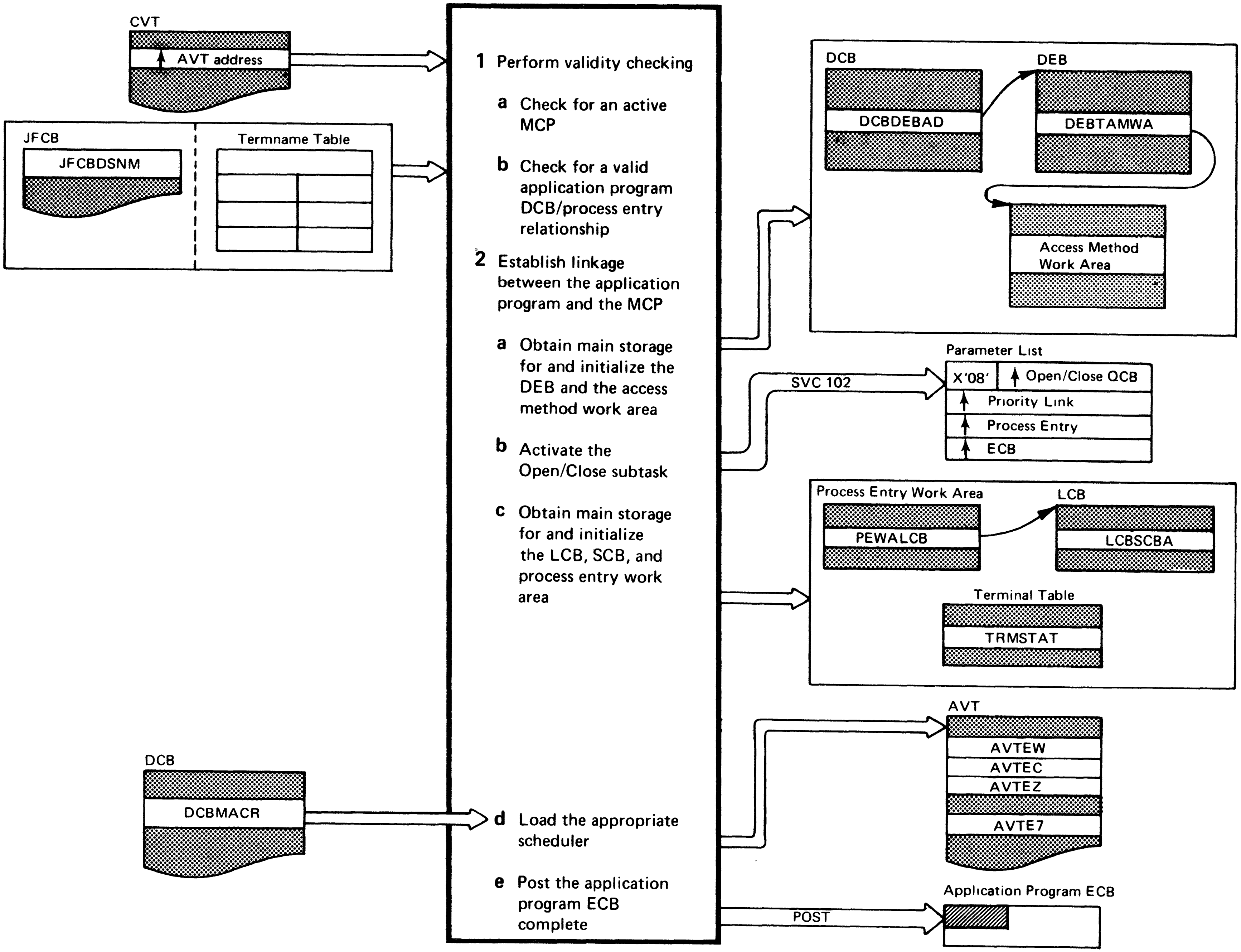
7 Create a parameter list for the TCAM dispatcher

8 Indicate completion of READY execution

Chart A3 Executing READY—Description

	Description	Routine	Register Usage
1	If AVTCKGET (address of the checkpoint work area) is zero, there is an open checkpoint DCB. After all incident records are processed, issue a FREEMAIN macro for the I/O buffer, then issue an ATTACH macro to bring the checkpoint executor into the same system partition as the MCP. Store the address of the checkpoint TCB in AVTCKTCB.	IEDQND	R13 I—AVT address
2	AVTREADY is the enabled ready queue. Place a pointer to the checkpoint request element (AVTCKELE) in the dispatcher save area (AVTSAVE2+24) so that the element is on the ready queue. Place a pointer to AVTCKELE in QCBELCHN. As a result of this, the TCAM dispatcher will take an environment checkpoint when it is activated.		
3	Get the address of the operator control work area from AVTOCGET. Move each incident record, except those for start- or stopline, into the operator control work area (OPCCKERB). (The stop- and start-line incident records are processed during a restart procedure at checkpoint open—see Chart D2.)		
4	Once an incident record is in the operator control work area (starting at OPCCKERB in the operator control AVT), post the operator control ECB complete and wait for IGC0110D to process the post request.		
5	Put any destination QCB that specified a nonzero value for the CLOCK= or the CINTVL= operand on the time delay queue (AVTTIMQ).		
6	If on-line test is specified, determine (by a GETMAIN) whether there is enough main storage available for the test functions to be performed. If there is not enough main storage for the minimum requirements of the test function, the MCP abnormally terminates. If there is enough for minimum requirements, but not as much as requested, issue a warning WTO message (IED094I).		
7	Put the address of AVTSAVE2 into register 1 as the pointer to a parameter list for the TCAM dispatcher.		
8	Turn on the "READY completed" bit (AVTREADN) in the AVT (at AVTBIT1).		

Chart A4 Initializing the Application Program (Part 1 of 2)



1 Perform validity checking

a Check for an active MCP

b Check for a valid application program DCB/process entry relationship

2 Establish linkage between the application program and the MCP

a Obtain main storage for and initialize the DEB and the access method work area

b Activate the Open/Close subtask

c Obtain main storage for and initialize the LCB, SCB, and process entry work area

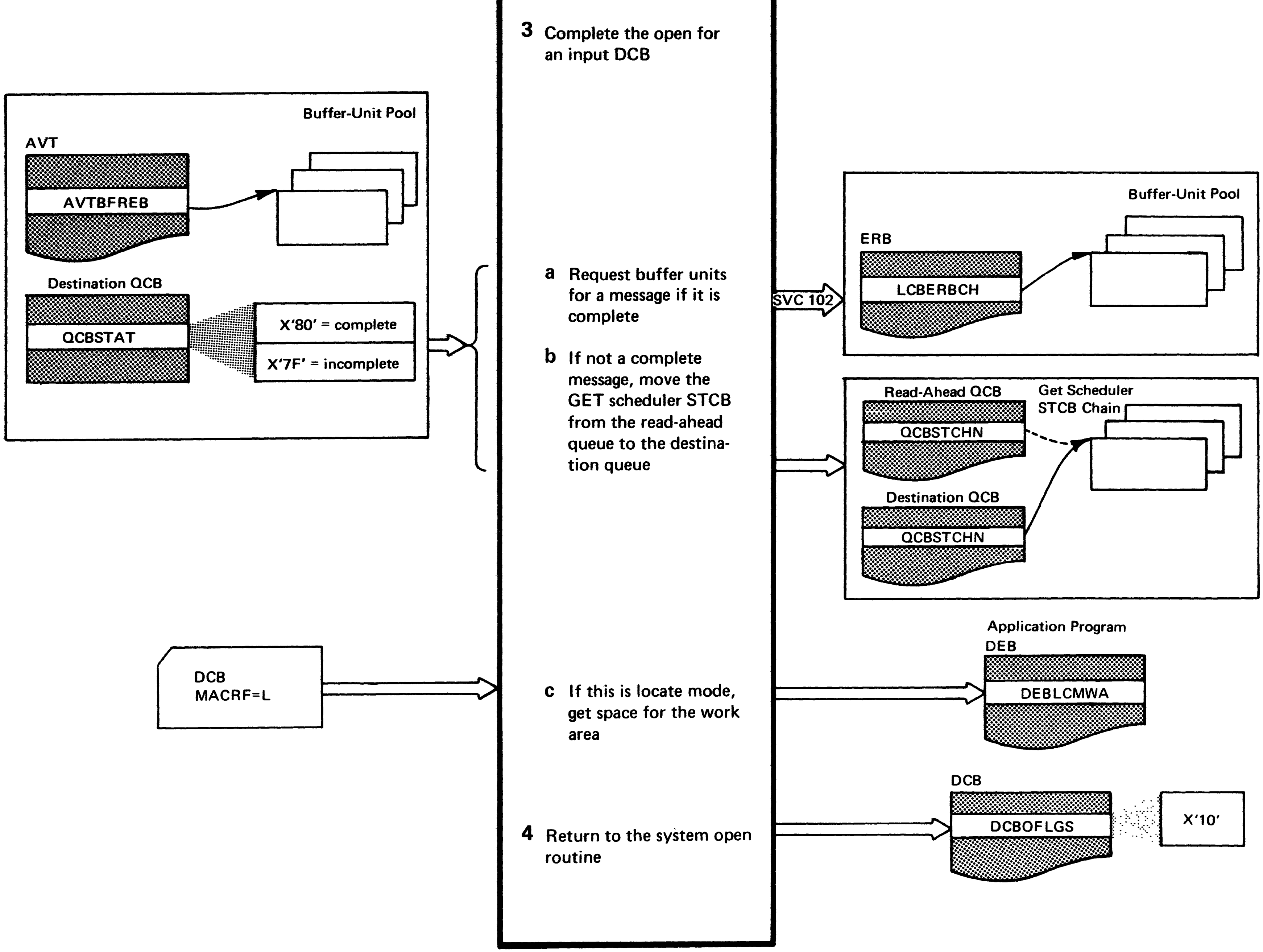
d Load the appropriate scheduler

e Post the application program ECB complete

Chart A4 Initializing the Application Program—Description (1 of 2)

	Description	Routine	Register Usage
1a	A pointer at CVT+240 points to the AVT address. This pointer is a nonzero value when a TCAM MCP is present. If it is zero, set the "unsuccessful open" flag in the DCB and exit to IGG01933.	IGG01946	R5 I—first entry in DCB parameter list R7 I—address of current entry in DCB parameter list
1b	Check the QNAME from the JFCB (JFCBDSNM) against the termname table entries for application programs, by using the Binary Search routine. If the QNAME is invalid or not found, exit to IGG01933 for error processing.	IEDQA1 IGG01933	R15 O—return code
2a	Issue a GETMAIN macro to obtain main storage for the DEB and the access method work area. Put the DEB address into the DCB at DCBDEBAD and put the access method work area address into the DEB at DEBTAMWA. Initialize the access method work area and link it to the DEB. Enqueue the DEB on the application-program TCB DEB chain. Put the address of the DEB into DCBDEBAD.	IGG01946	
2b	Use SVC 102 to tpost a special element to the open/close subtask. Then issue a WAIT macro to allow time for the open/close subtask to execute. The SVC 102 parameter lists are shown on Chart B2.		
2c	Issue a GETMAIN macro to obtain main storage for the LCB, process entry work area, and one or more SCBs. Place the LCB address into the process entry work area at PEWALCB. The address of the process entry work area is located in the TRMSTAT field of the process entry. Store the SCB address in LCBSCBA.	IEDQEU	

Chart A4 Initializing the Application Program (Part 2 of 2)



3 Complete the open for an input DCB

a Request buffer units for a message if it is complete

b If not a complete message, move the GET scheduler STCB from the read-ahead queue to the destination queue

c If this is locate mode, get space for the work area

4 Return to the system open routine

Chart A4 Initializing the Application Program—Description (2 of 2)

Description		Routine	Register Usage
2d	Determine which scheduler to load from the MACRF= field of the DCB, load the scheduler, and link the STCB for that scheduler to the destination QCB for this application program. Set a "good-open" flag in the process entry. AVTEW—address of the GET scheduler AVTEC—address of the PUT scheduler AVTEZ—address of the GET FIFO scheduler AVTE7—address of the Retrieve scheduler		
2e	Issue a POST macro to post the application program ECB complete by turning on bit 1 of the first byte of the ECB.		
3a	If this is a receive operation, inspect the destination QCB for a complete message. If there is a complete message, tpost the ERB to the disk I/O QCB in the MCP.		
3b	If there is not a complete message, move the Get scheduler STCB from the read-ahead QCB to the application program destination QCB.	IGG01947	
3c	If locate mode is specified (MACRF=L), issue a GETMAIN macro to obtain main storage for a work area. Also, store the address of the work area in DEBLCMWA.		
4	The system Open routine sets a "successful open" (DCBOFLGS=X'10') flag in the DCB for this application program.		

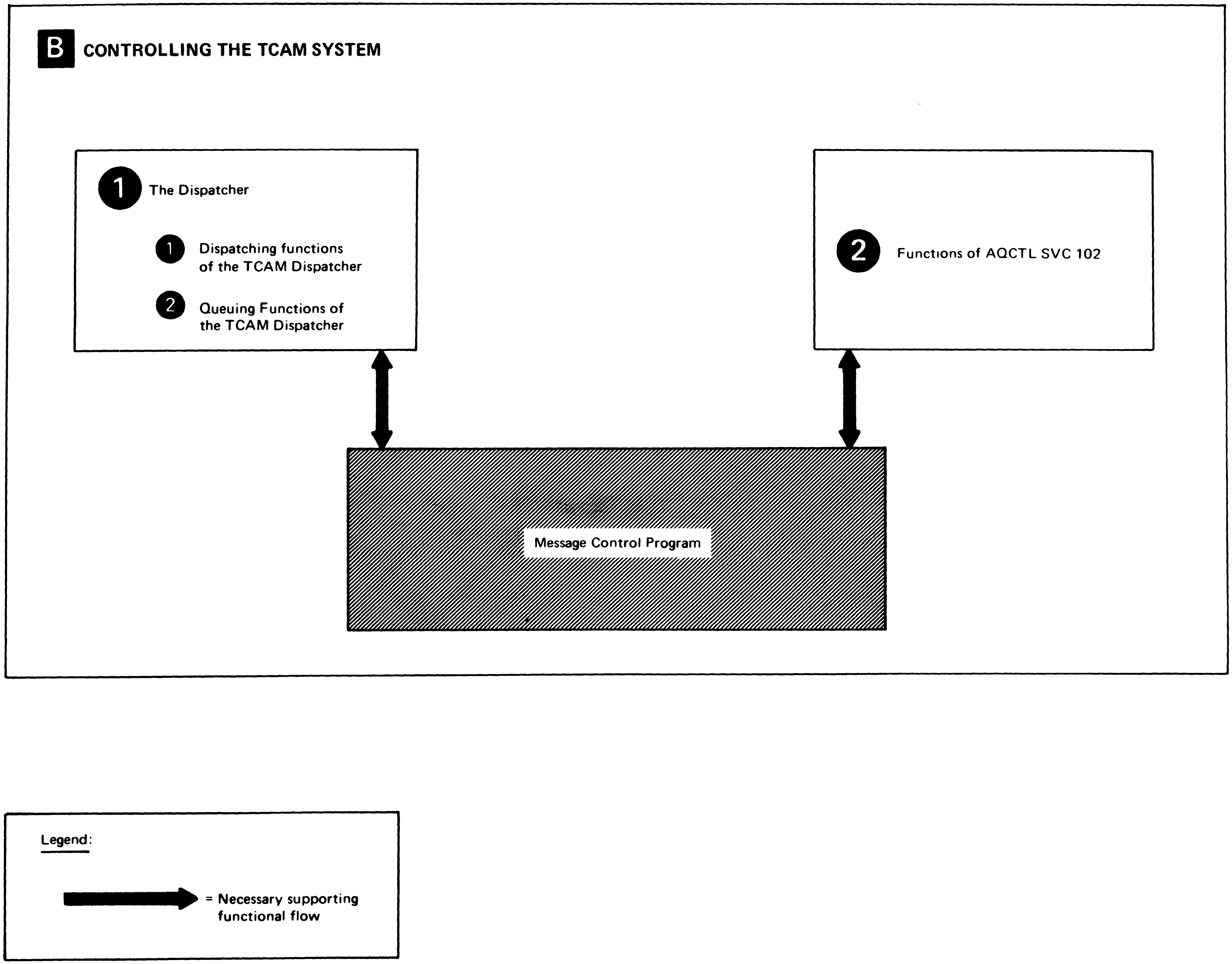
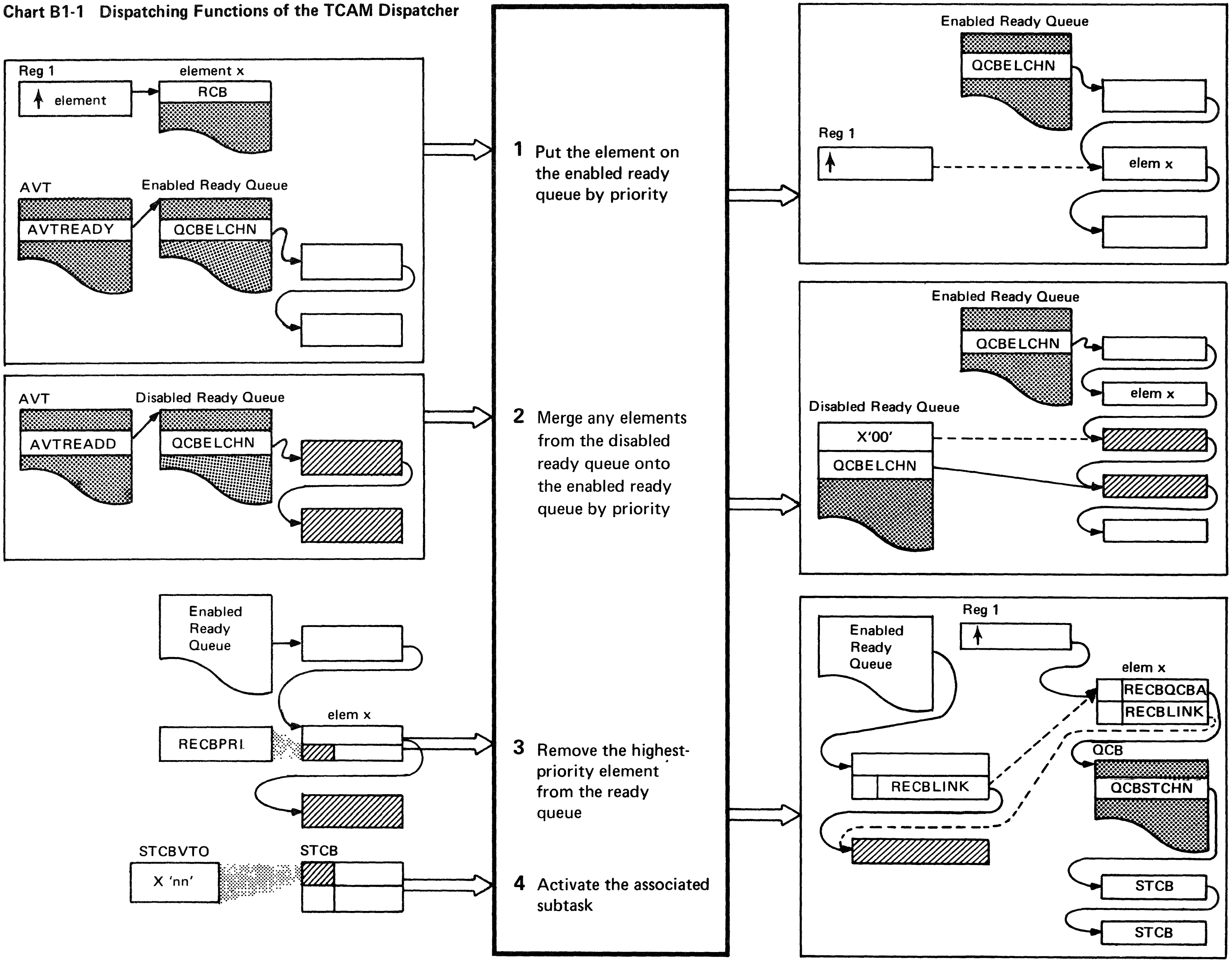

B CONTROLLING THE TCAM SYSTEM
1 The Dispatcher
1 Dispatching functions of the TCAM Dispatcher
2 Queuing Functions of the TCAM Dispatcher
2 Functions of AQCTL SVC 102
Message Control Program
Legend:
= Necessary supporting functional flow

Chart B Controlling the TCAM System

Description	Chart No.
The Dispatcher	B1-1
Dispatching Functions of the TCAM Dispatcher	
Queuing Functions of the TCAM Dispatcher	B1-2
Functions of AQCTL SVC 102	
• Moving data across partition boundaries	
• Posting ECBS in other tasks	
• Tposting elements to the TCAM disabled ready queue	
• Flagging TCBs for application programs as eligible or not eligible for swapping or rollout	

Chart B1-1 Dispatching Functions of the TCAM Dispatcher



1 Put the element on the enabled ready queue by priority

2 Merge any elements from the disabled ready queue onto the enabled ready queue by priority

3 Remove the highest-priority element from the ready queue

4 Activate the associated subtask

Chart B1-1 Dispatching Functions of the TCAM Dispatcher—Description

	Description	Routine	Register Usage
1	Examine the elements on the enabled ready queue. The enabled ready queue is at AVTREADY and points to the first element on the queue. Insert element X (pointed to by register 1) ahead of the first element found that has a lower priority than element X. Chain element X onto the ready queue by moving the link field of the element already on the ready queue to the link field of element X. Then put the contents of register 1 in the link field of the element that was already on the ready queue.	IGG019RB or IGG019RO	R1 I—address of last RCB dispatched
2	Use the procedure described above to merge any elements from the disabled ready queue onto the enabled ready queue, (the disabled ready QCB is at AVTREADD). The only difference is that the first word of the disabled ready queue, rather than register 1, points to the first element to be merged. The link field of the last element on the disabled ready queue contains zero. After the merge, the first word of the disabled ready queue contains zero and the second word contains the address of the last element merged.		
3	Check the RECBPRI field for the highest-priority element. Put the address of the highest-priority element on the enabled ready queue into register 1. After removing the element, put the link field of the element now pointed to by register 1 on the ready queue and then examine the next element on the ready queue. The last element on the ready queue is always at AVTDELM; this is referred to as the "dummy last element."		

Chart B1-1 Dispatching Functions of the TCAM Dispatcher—Description *Continued*

Description	Routine	Register Usage
4 The STCBVTO field (the first byte of an STCB) serves as an index to indicate which subtask gains control. If this field contains X'00', the TCAM dispatcher issues a WAIT macro because there are no elements to process. An STCBVTO value of X'02' indicates that the element to be processed is for an attached task (operator control, on-line test, or FE Common Write). In this case, the TCAM dispatcher links the element to the element chain of the QCB for the attached task and posts the ECB for the task as complete. This allows the attached task to directly compete for system resources when TCAM issues a WAIT macro. When the STCBVTO value is neither X'00' nor X'02', the TCAMdispatcher computes the subtask entry point according to the following STCBVTO values: X'04'—the subtask follows a 2-byte STCB X'06'—the subtask follows a 4-byte STCB X'08'—the subtask follows a 6-byte STCB X'0A'—the subtask follows an 8-byte STCB If the STCBVTO value is greater than X'0A', the TCAM dispatcher uses the STCBVTO value as an index into the list of scheduler addresses at AVTDISP to activate the associated subtask. The following STCBVTO values activate the indicated subtasks: X'0C'—Leased Receive scheduler X'0E'—Send scheduler X'10'—GET scheduler X'12'—PUT scheduler X'14'—GET FIFO scheduler X'16'—Log scheduler X'18'—Dial Receive scheduler X'1A'—Buffered Terminal scheduler X'1C'—Retrieve scheduler X'1E'—Local Receive scheduler X'20'—Concentrator Send Scheduler X'26'—COMMBUF Send Scheduler **Note:** *If a subtask is activated without an element to process, its STCB is tposted to the ready queue with the correct STCBVTO value and the next three bytes containing the address of AVTREADY-8.*		

Chart B1-2 Queuing Functions of the TCAM Dispatcher

READY Macro Expansion

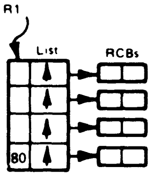
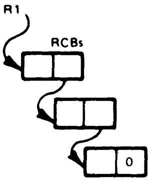
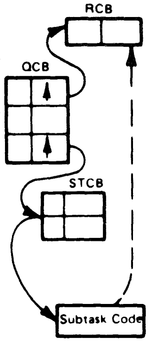
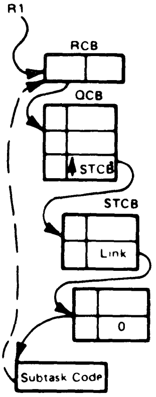
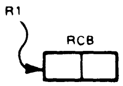
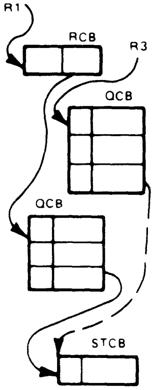
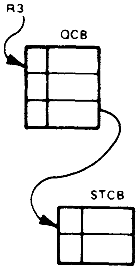
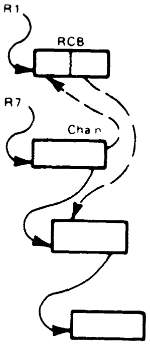
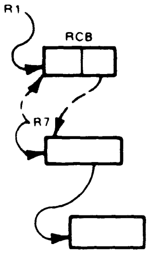
TCAM Dispatcher IGG019RB

Occur only during the first pass through the dispatcher:

- Save the user's registers in AVTSAVE1 of the AVT.
- Retrieve the data in AVTSAVE2 of the AVT and store it for dispatcher use.

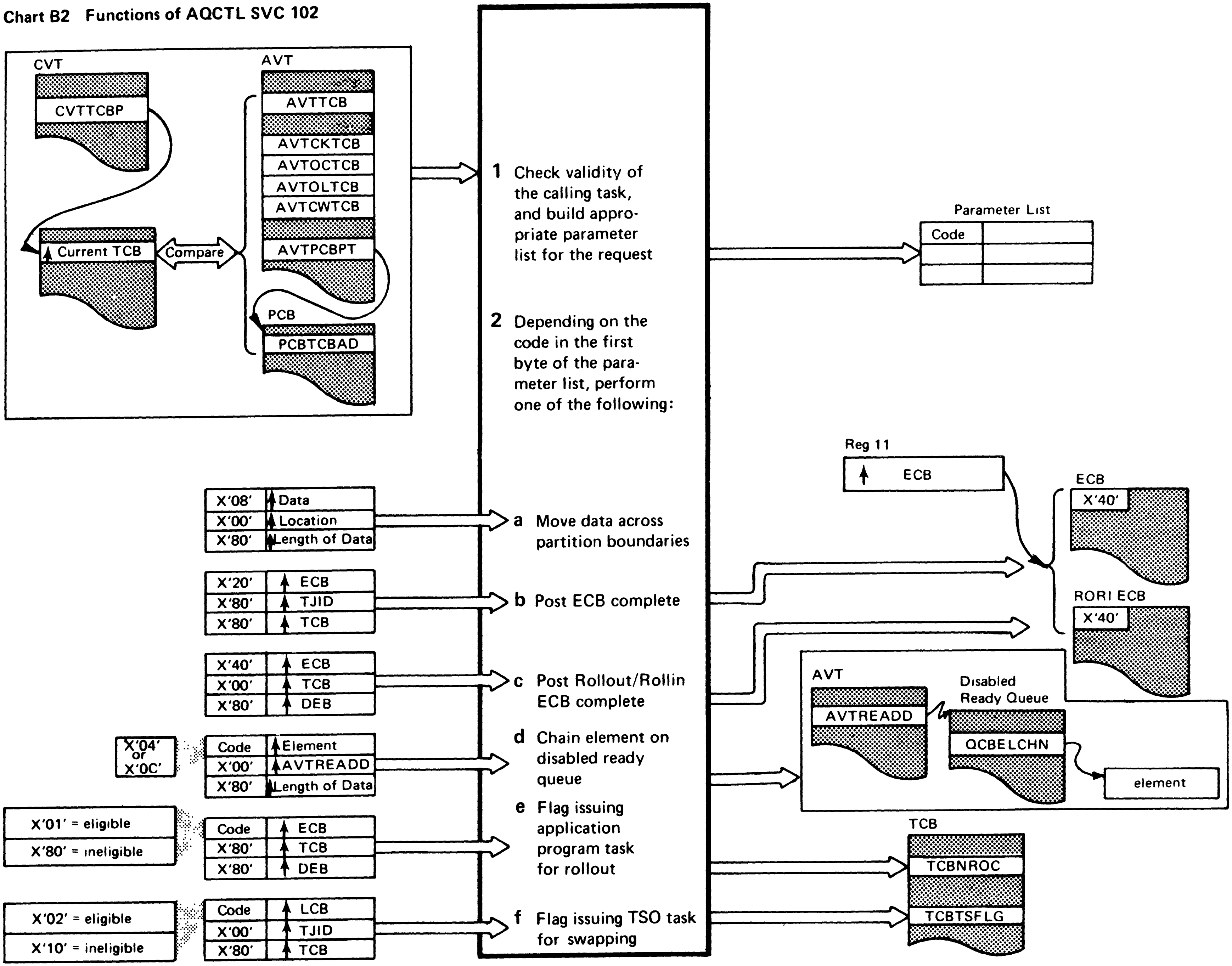
- Perform action according to the entry point designated by the returning subtask:

Entry Point

DSPDISP	DSPLIST	DSPCHAIN	DSPWAIT	DSPBYPAS	DSPDLETE	DSPPOST (DSPPOSTR)	DSPTSTQ (DSPTSTQR)	DSPUNAV (DSPUNAVR)	DSPPRIO (DSPPRIOR)	DSPLIFO (DSPLIFOR)
• The subtask returning to the dispatcher has no elements to add to the ready queue.	• Add the elements whose addresses are in a parameter list pointed to by register 1 to the ready queue. The high-order byte of the last pointer contains X'80' to indicate the end of the chain.  	• Add the elements that are chained together to the ready queue; the first element is pointed to by register 1. The link field of the last item in the chain contains X'XX000000'  	• Process an element from the element chain of the QCB. If no element is present, the subtask twaits for an RCB to be posted to the element chain.  	• Process immediately the next STCB in the STCB chain of the QCB being examined.  R1 RCB QCB STCB STCB Link 0 Subtask Code	• Delete the Start-Up Message routine. • Perform the DSPCHAIN entry point function.	• Tpost one element to the ready queue. Register 1 contains the address of the element (RCB) to be tposted.  	• Determine if the returning subtask's STCB is twaiting in the STCB chain of a QCB pointed to by register 3. If it is not, chain the STCB into that QCB's STCB chain. If it is, continue processing.  	• Remove the returning subtask's STCB from the QCB chain it is in. • Place the removed STCB in the STCB chain of a QCB pointed to by register 3.  	• Place the RCB pointed to by register 1 into a chain pointed to by register 7. Place the RCB in the chain by priority.  	• Place the RCB pointed to by register 1 into the first spot in a chain pointed to by register 7. 

- If the returning subtask did not have an "R" as the last letter of its name, processing continues through the dispatcher, otherwise, control passes to the returning subtask once the queue management functions are complete.

Chart B2 Functions of AQCTL SVC 102



1 Check validity of the calling task, and build appropriate parameter list for the request

2 Depending on the code in the first byte of the parameter list, perform one of the following:

a Move data across partition boundaries

b Post ECB complete

c Post Rollout/Rollin ECB complete

d Chain element on disabled ready queue

e Flag issuing application program task for rollout

f Flag issuing TSO task for swapping

Chart B2 Functions of AQCTL SVC 102—Description

	Description	Routine	Register Usage
1	If SVC 102 is issued when there is not an active MCP in the system (CVT+240 is zero), the requested action is not performed and the AQCTL SVC 102 routine sets an error return code of X'04'. Get the pointer to the current TCB address from CVTTCBP and determine if the current TCB address is equal to the TCB address of one of the following tasks: AVTTCB—TCAM message control program PCBTCBAD—TCAM application program AVTOCTCB—operator control AVTCKTCB—checkpoint/restart AVTOLTCB—TOTE (On-Line Test) AVTCWTCB—COMWRITE (FE Common Write) Any task attached by a valid task Set an error return code of X'08' if the TCB for the calling task is not valid, and return. Build a three-word list of parameters needed to perform a function. X'80' is always the first byte of the third word.	IEDQEB	R1 I—input parameter list R3 I—CVT address R14 I—return address
2a	First byte = X'08': move data across partitions. The first word of the parameter list contains the address of the data to be moved. The second word contains the address of the target field of the move, and the third word contains the address of a halfword that has the length (in bytes) of the data field.		
2b	If the value is X'20', post the TSO or standard task ECB complete. For a TSO task, branch to the time-sharing interface program where the task is flagged either eligible or ineligible for swapping.		

(This page left blank intentionally)

Chart B2 Functions of AQCTL SVC 102–Description *(Continued)*

Description		Routine	Register Usage
2c	If the value is X'40', post the ECB complete for a task that is eligible for rollout. For a task that is currently rolled out, set TCBTRM bit 4 (TCBTCPP)to indicate that a post is pending. The ECB is posted by turning on bit 1 of the first byte. The low-order three bytes of the first word contain the ECB address. The second word contains the TCB address for the task being posted. Word three contains the address of the DEB associated with the ECB being posted. Check the ECB to be posted for validity. Get the Post routine (IEAQSY50) special entry address (IEAOPT01) from the CVT (CVTOPT01) and execute it.		
2d	First byte = X'04' (alone or X'0C'): post the element to the disabled ready queue. AVTREADD is the disabled ready queue. QCBELCHN points to the element chain. Post the MCP ECB complete.		
2e	First byte = X'01' or X'80': flag the application program either eligible or ineligible for rollout, respectively. If X'80', the SVC 102 routine sets TCBNROC to a nonzero hexadecimal digit. If X'01', the SVC 102 routine sets TCBNROC to X'00'.		
2f	First byte = X'02' or X'10': flag the TSO program either eligible or ineligible for swapping, respectively. Turn bit 0 of TCBTSFLG on if eligible for swapping or off if ineligible. **Note:** *If more than one bit in the action code byte is turned on, the AQCTL SVC 102 routine performs the actions specified for each bit. The combinations of the bits used, however, must be compatible, so that the parameter list satisfies all the requirements.*		

C PROCESSING THE MESSAGES

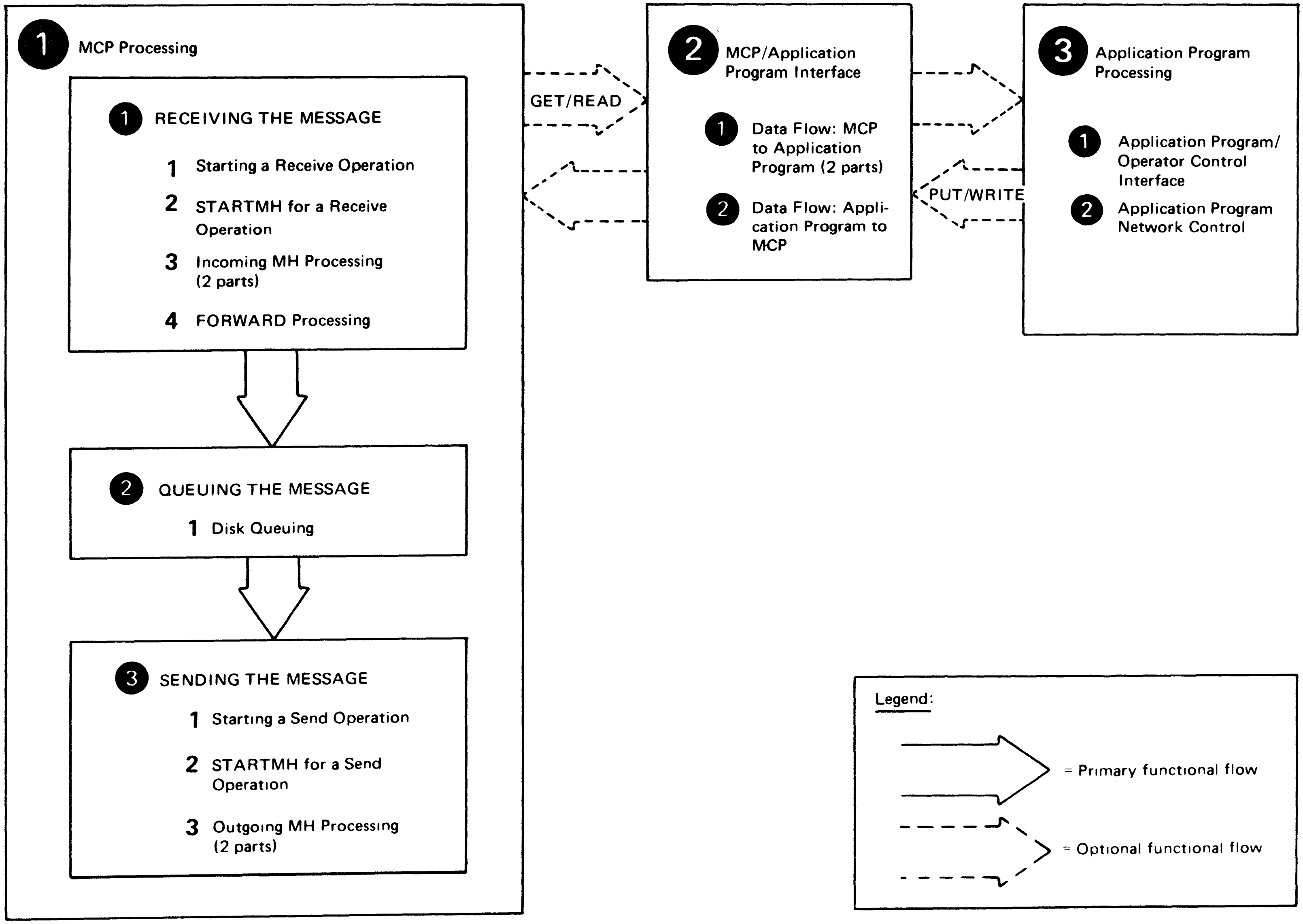
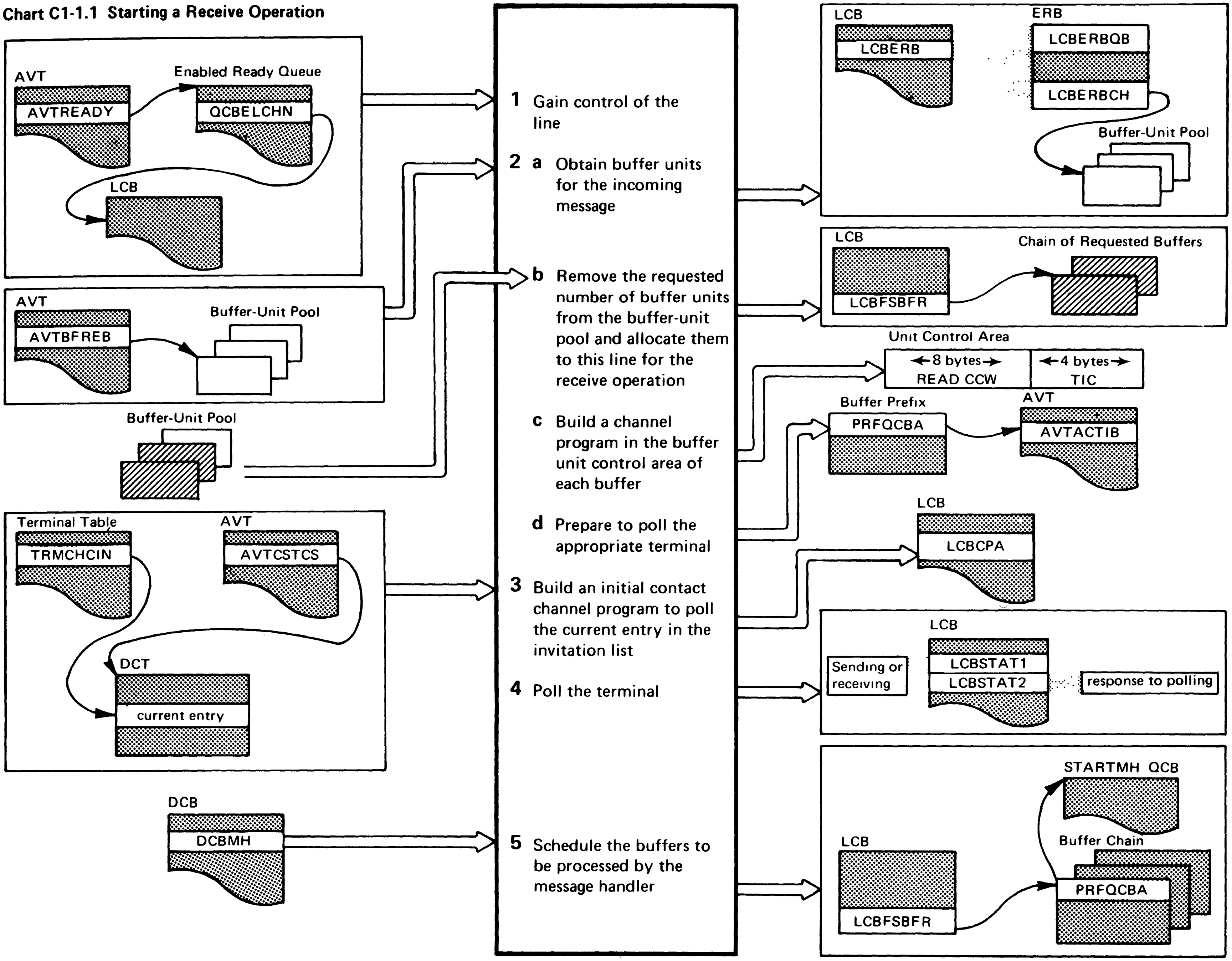

1 MCP Processing
1 RECEIVING THE MESSAGE
1 Starting a Receive Operation
2 STARTMH for a Receive Operation
3 Incoming MH Processing (2 parts)
4 FORWARD Processing
2 QUEUING THE MESSAGE
1 Disk Queuing
3 SENDING THE MESSAGE
1 Starting a Send Operation
2 STARTMH for a Send Operation
3 Outgoing MH Processing (2 parts)
GET/READ
2 MCP/Application Program Interface
1 Data Flow: MCP to Application Program (2 parts)
2 Data Flow: Application Program to MCP
PUT/WRITE
3 Application Program Processing
1 Application Program/ Operator Control Interface
2 Application Program Network Control
Legend:
= Primary functional flow
= Optional functional flow

Chart C Processing the Message

Description	Chart No.
Message Control Processing Receiving the Message: Starting a Receive Operation describes the buffering and polling functions necessary to receive a message.	C1-1.1
STARTMH for a Receive Operation	C1-1.2
Incoming MH Processing	C1-1.3
FORWARD Processing describes placing the message on the destination queue.	C1-1.4
Queuing the Message describes reusable and non-reusable disk queuing.	C1-2
Sending the Message: Starting a Send Operation describes the buffering and addressing functions necessary to send a message.	C1-3.1
STARTMH for a Send Operation	C1-3.2
Outgoing MH Processing	C1-3.3
MCP/Application Program Interface: Data Flow: MCP to Application Program describes the processes that occur when a GET or READ macro is encountered in an application program.	C2-1
Data Flow: Application Program to MCP describes the processes that occur when a PUT or WRITE macro is encountered in an application program.	C2-2
Application Program Message Processing: Application Program/Operator Control Interface describes the processes whereby a user enters operator control commands from his application program, defined as a secondary operator control station.	C3-1
Application Program Network Control describes the functions for dynamically controlling the telecommunications network through macro instructions issued in an application program.	C3-2

Chart C1-1.1 Starting a Receive Operation



1 Gain control of the line

2 a Obtain buffer units for the incoming message

b Remove the requested number of buffer units from the buffer-unit pool and allocate them to this line for the receive operation

c Build a channel program in the buffer unit control area of each buffer

d Prepare to poll the appropriate terminal

3 Build an initial contact channel program to poll the current entry in the invitation list

4 Poll the terminal

5 Schedule the buffers to be processed by the message handler

Chart C1-1.1 Starting a Receive Operation—Description

	Description	Routine	Register Usage
1	The Receive scheduler gains control when an LCB tposted to itself is on top of the ready queue. AVTREADY is the enabled ready queue and points to the LCB.	IGG019R3	R1 I—LCB address
2a	Tpost the ERB to the buffer request QCB (AVTBFREB) to obtain buffer units for the incoming message.		
2b	For initial, application program, operator control, and first-PCI requests, get the requested number of buffer units from the buffer-unit pool and put the address of the first buffer unit into LCBFSBFR. For subsequent PCI requests, chain the ERB by priority into the element chain of the buffer return QCB (AVTBFRTB).	IEDQGA	
2c	Build Read CCWs in the first two words of each unit and TIC CCWs in the third word. Chain the units together into one contiguous channel program.		
2d	Tpost the ERB to the activate-I/O generator QCB (IEDQKA) to poll the terminal. AVTACTIB contains the address of this QCB.		
3	AVTCSTCS points to the beginning of the device characteristics table and TRMCHCIN is an index into the table to the current entry. Build a channel program based on the device characteristics table entry for the device to be polled. Build the channel program in the channel program area (LCBCPA). Issue the SIO command, to initiate polling. Issue an EXCP macro to start the channel program to receive the message on the line.	IEDQKA	R1 I—LCB address R13 I—AVT address
4	If bit 4 in LCBSTAT2 is zero, the response to polling was positive, if LCBSTAT2 is X'08' the response to polling was negative. LCBSTAT1 (X'02') indicates that the line is receiving.		R1 I—buffer address
5	Tpost the buffers to the STARTMH QCB. If PCI=N, the entire message buffer string is tposted to MH. If PCI=A or R, individual buffers are tposted. DCBMH is the address of the Message Handler for this line group. LCBFSBFR points to the chain of buffers to be assigned. PRFQCBA is the QCB address when the buffer is an element.	IGG019R0 or IGG019RN	R1 I—request element address R4 I—DCB address

Chart C1-1.2 STARTMH for a Receive Operation

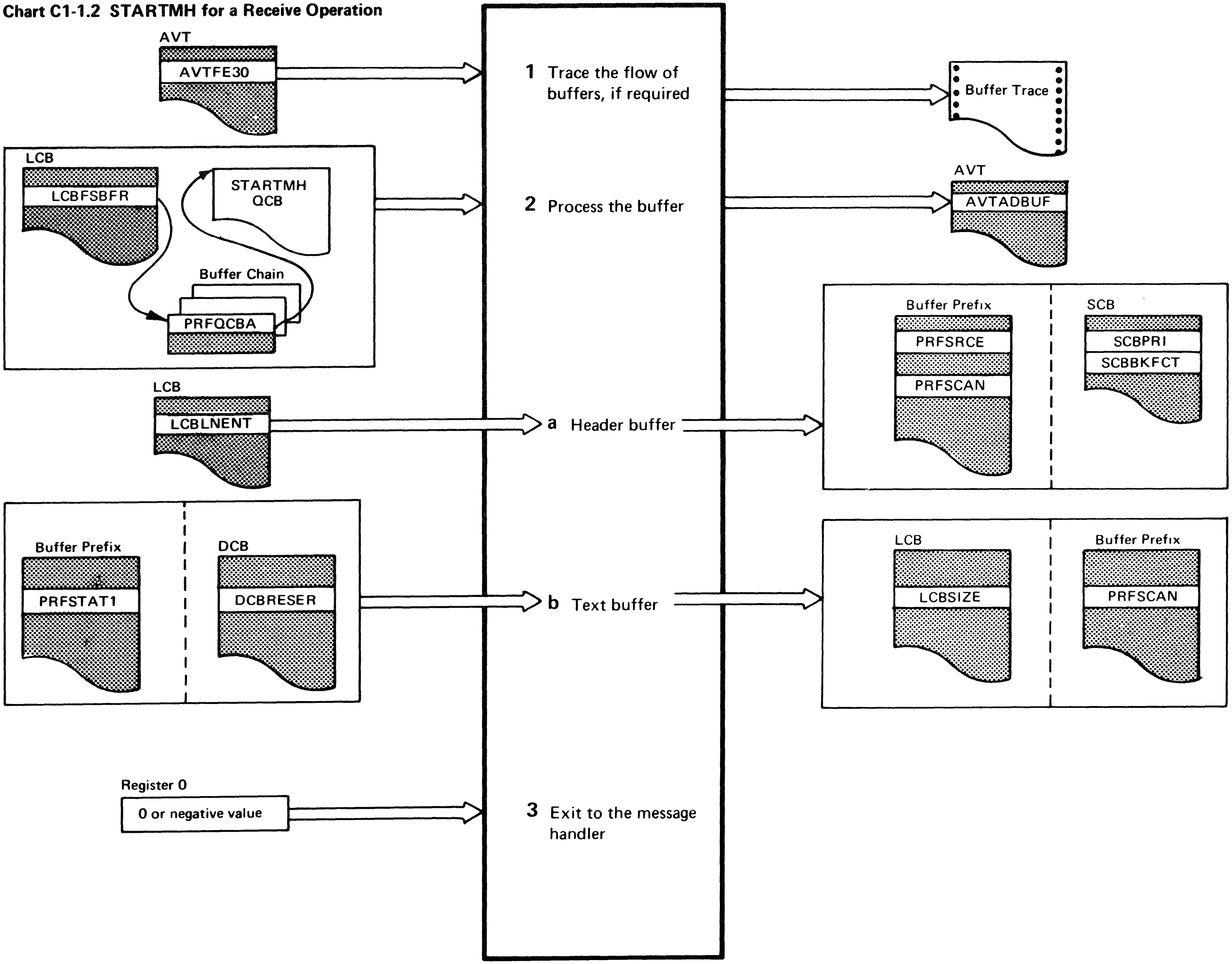
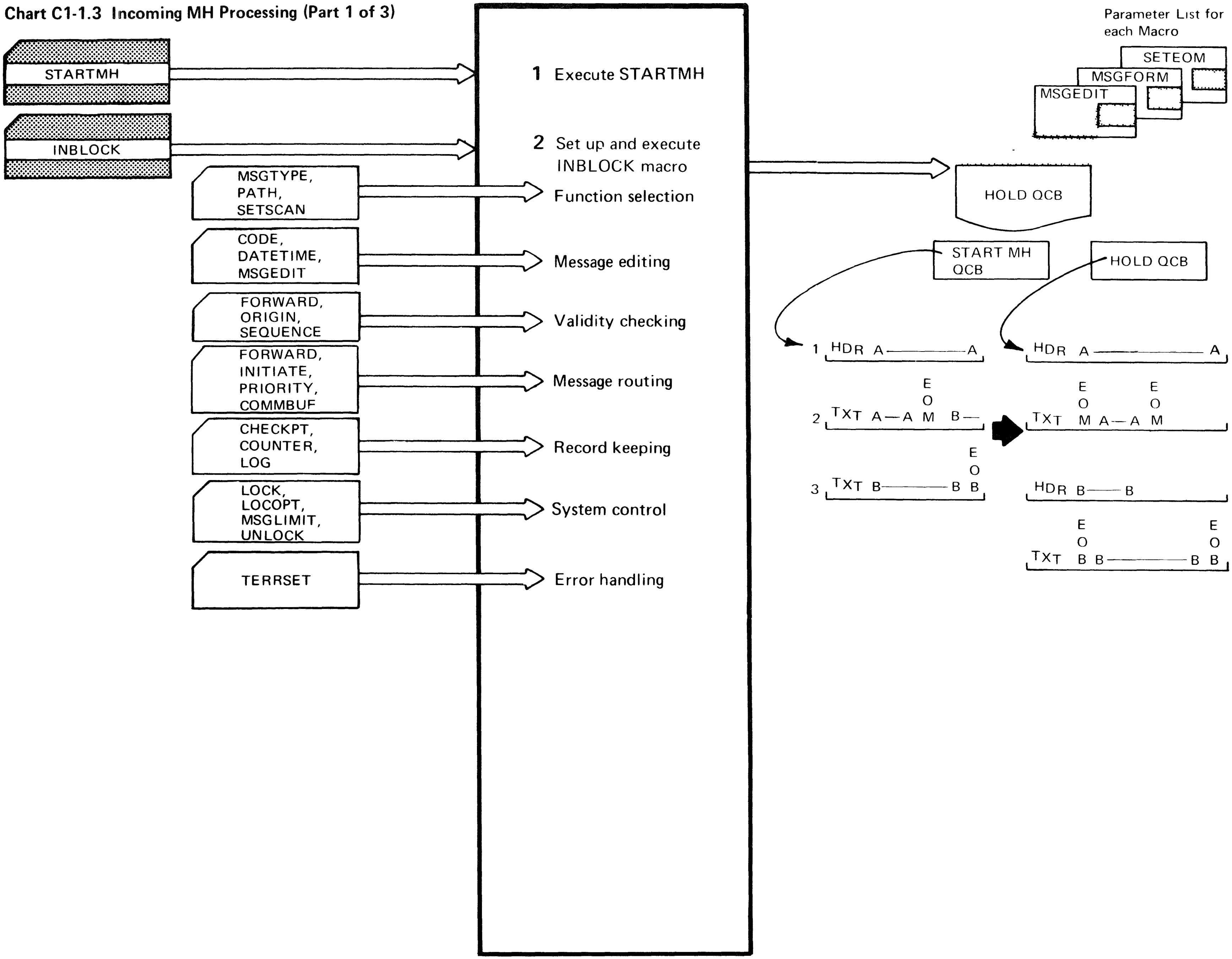


Chart C1-1.2 STARTMH for a Receive Operation—Description

	Description	Routine	Register Usage
	Note: *If EOB/ETB processing is specified on the STARTMH macro, the EOB/ETB handling subtask gains control before the STARTMH subtask.*	IEDQBT	
1	If the AVTFE30 field is not zero, the STARTMH subtask gets the address of the Buffer Trace Dump routine (IEDQFE30) from the AVTFE30 field and links to that routine to trace the flow of buffers.	IEDQAA	R7 I—STARTMH QCB address R15 O—entry point in STARTMH subtask
2a	Place the address of the buffer just tposted to the STARTMH QCB in the AVTADBUF field of the AVT. Initialize the PRFSRCE field from LCBLNENT. Clear the PRFISEQ, SCBPRI, and SCBBKFCT fields to zeros. Initialize the scan pointer (PRFSCAN) to point to the last byte in the prefix or, if reserve characters are used, to the last reserve character.		
2b	If PRFSTAT1 is X'80', indicating a subsequent or text buffer, initialize the prefix origin field (PRFSRCE) from LCBLNENT. Put the number of reserve characters (from DCBRESER) into LCBSIZE. Initialize the scan pointer (PRFSCAN) to point to the last byte of the prefix or, if reserve characters are used, to the last reserve character.		
3	For a normal exit, register 0 will contain zeros; for a multiple-buffer-header condition, register 0 will contain a negative value. Compute the MH entry address and examine register 0. If it contains a negative value, the subtask exits to the MH with a condition code of 4; otherwise, the subtask determines from the LCB whether the line is sending or receiving and exits to the MH with a condition code of 1 or 8, respectively.		

Chart C1-1.3 Incoming MH Processing (Part 1 of 3)



(This page left blank intentionally)

Chart C1-1.3 Incoming MH Processing (Part 2 of 3)

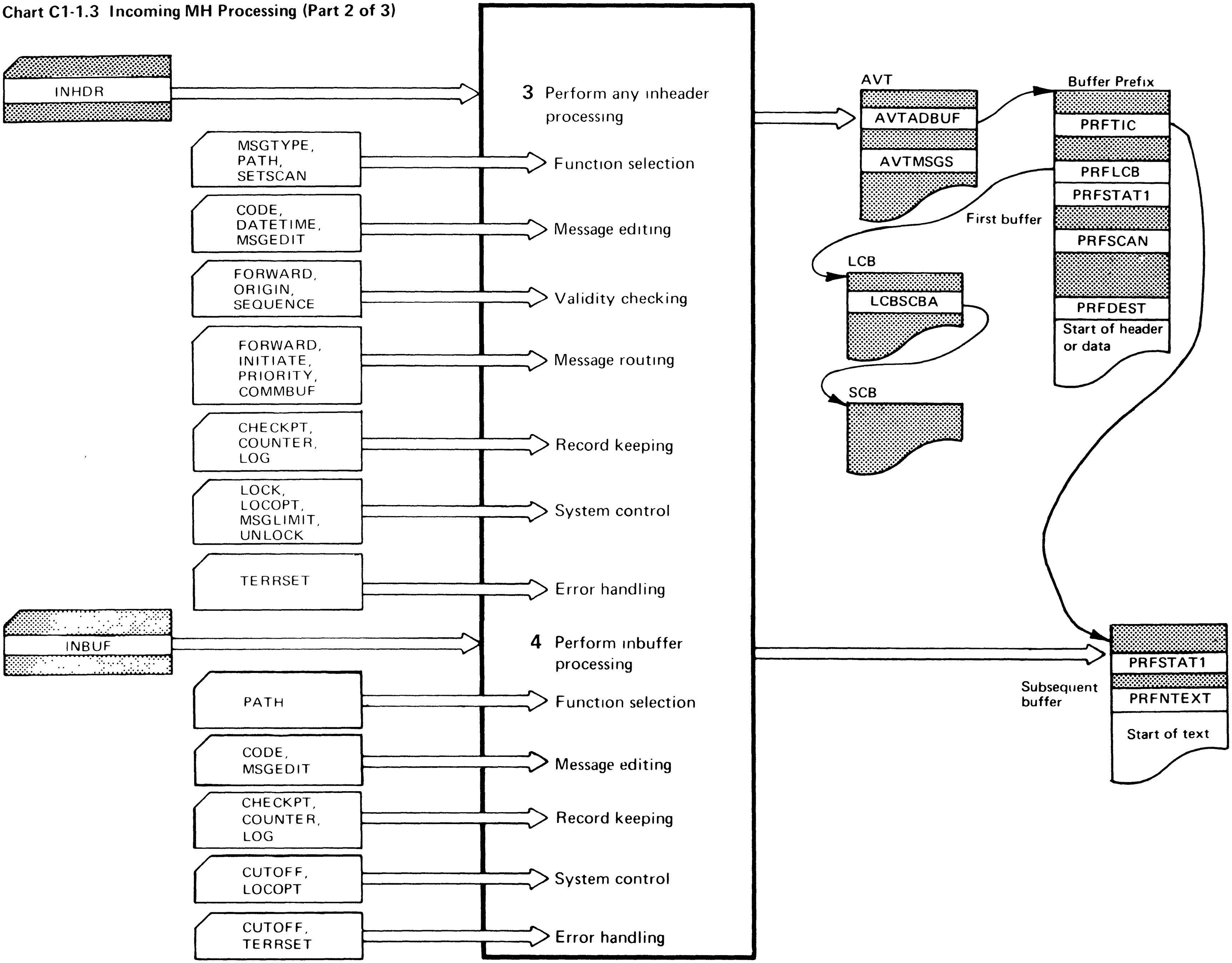
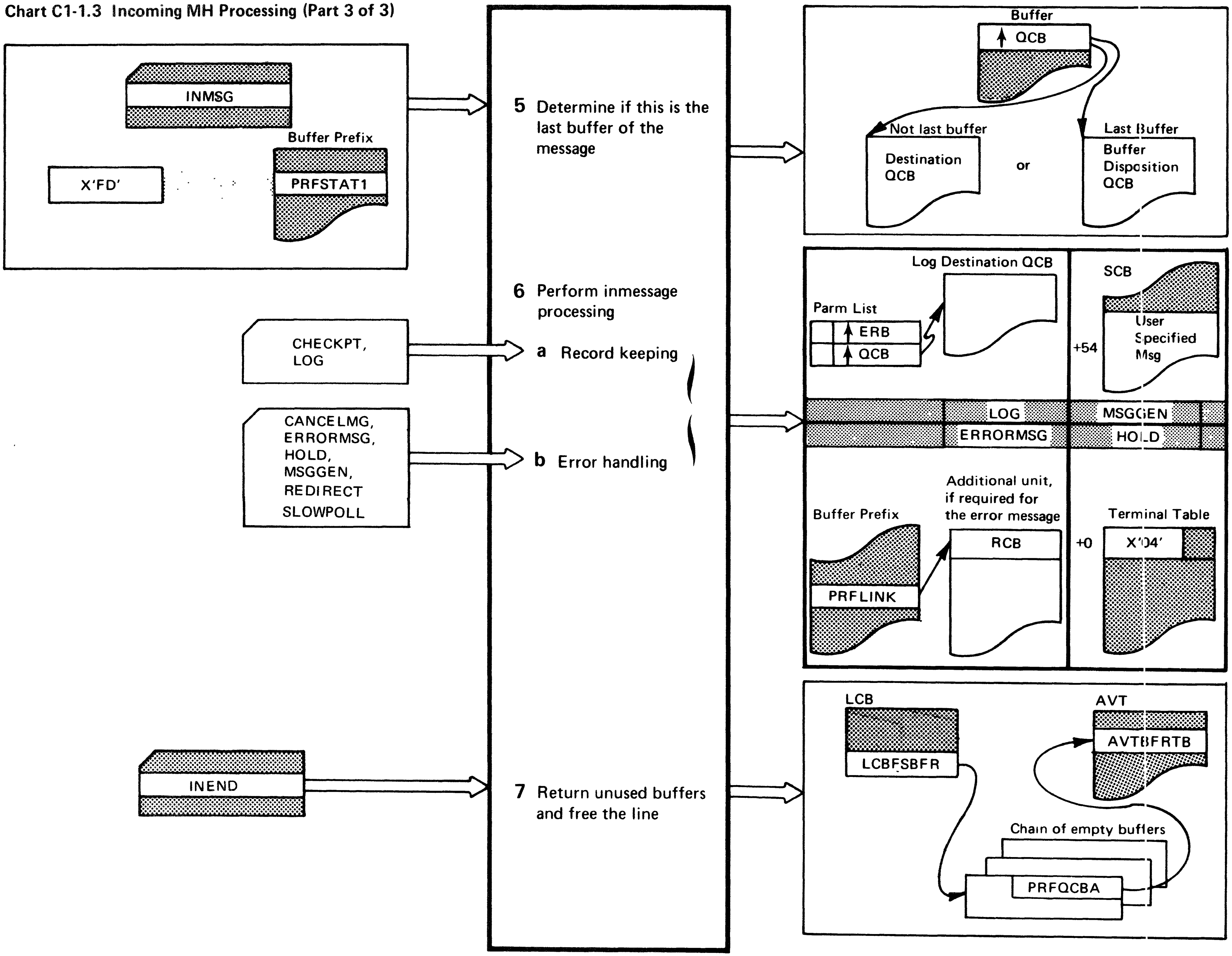


Chart C1-1.3 Incoming MH Processing—Description (1 of 2)

Description		Routine	Register Usage
1	See Chart C1-1.2		
2	The MSGEDIT, MSGFORM, and SETEOM macros in the inblock subgroup generate a hold queue. This queue holds data that cannot be processed with the current buffer. The held data is inserted in the next buffer tposted to this MH from the same transmission. This allows contiguous processing. (All macros coded in INBLOCK, except SETEOM, are handled as explained in #3 below; SETEOM is shown in the *Macro Linkage Charts* in the *Program Organization* section.)		
3-4	MH macro expansions link to functional MH routines through the User Interface routine (IEDQUI). The User Interface routine finds the address of the current buffer in the AVTADBUF field of the AVT, the address of the LCB in the PRFLCB field of the buffer prefix, and the address of the current SCB in the LCBSCBA field of the LCB. PRFDEST is the termname table offset for the destination of the message and PRFSTAT1 is a status byte. The address of the functional MH routine for the macro expansion is found as follows. The AVTMSGS field of the AVT contains the address of the MH VCON table. To this value the User Interface routine adds an index value obtained from the first byte of the input parameter list. The resulting address is placed in register 12. The macros listed, together with their parameter lists and linkages, are shown in the *Program Organization* section.	IEDQUI	R1 I—parameter list address

Chart C1-1.3 Incoming MH Processing (Part 3 of 3)



5 Determine if this is the last buffer of the message

6 Perform inmessage processing

a Record keeping

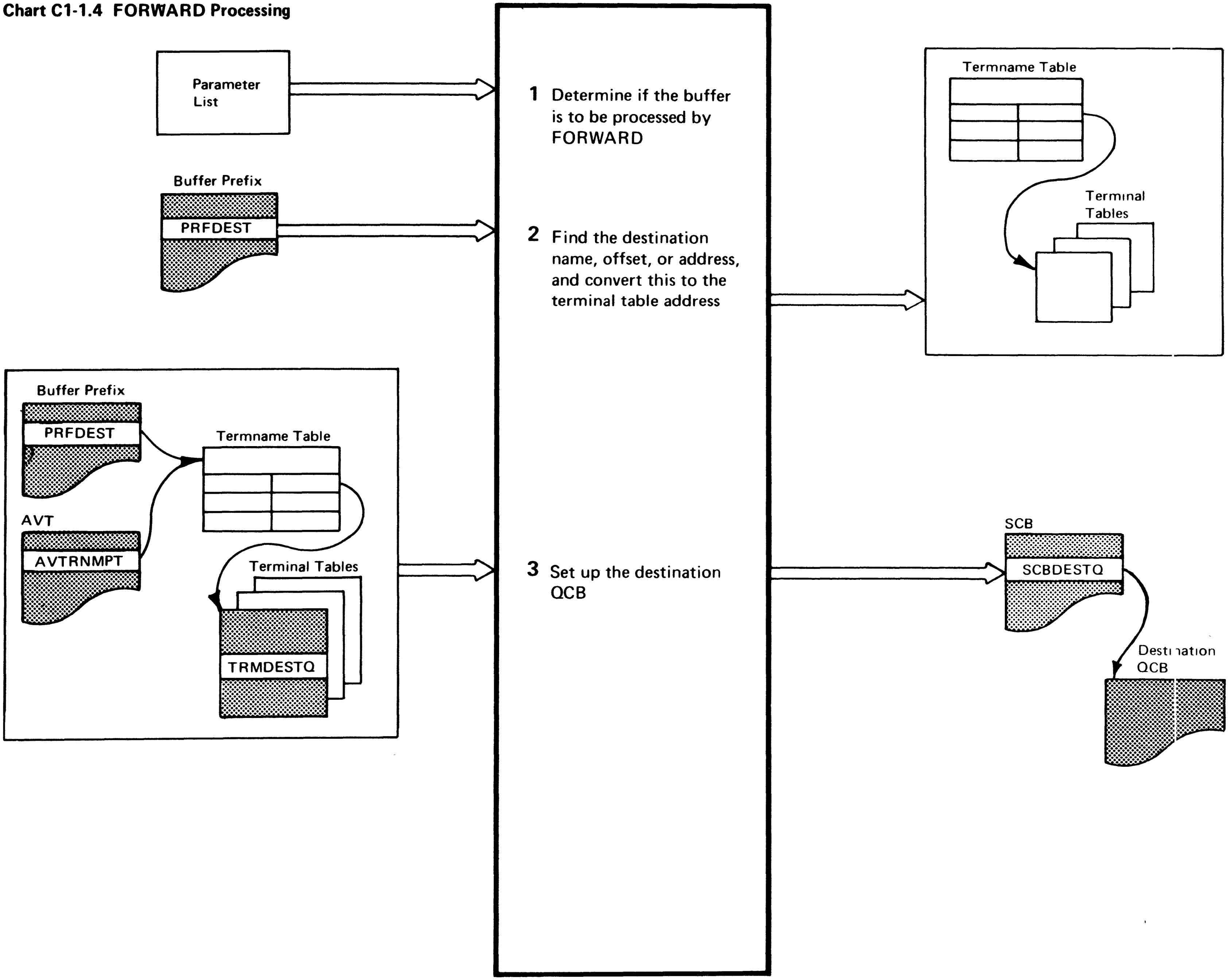
b Error handling

7 Return unused buffers and free the line

Chart C1-1.3 Incoming MH Processing—Description (2 of 2)

	Description	Routine	Register Usage
5	Bit 6 of the buffer prefix status byte (PRFSTAT1) indicates whether this is the last buffer of a message. If this byte contains X'02' (PRFNLSTN), this is not the last buffer. X'FD' (PRFNLSTF) identifies the last buffer of a message. If the buffer is not the final buffer of the message, or if the logical end-of-message indicator is not set in the buffer prefix (PRFITCPN), place the destination QCB address (from SCBDESTQ) in the first word of the buffer. If the buffer is the final buffer, or if the logical end-of-message indicator is set, put the address of the buffer disposition QCB in the first word of the buffer.	IEDQA4	
6	When the last segment of a message has been received and processed by the MH up to the inmessage subgroup, the buffer disposition subtask executes the macro expansions for each macro in the inmessage subgroup. The macros listed, together with their parameter lists and linkages, are shown in the *Program Organization* section.	IEDQBD	
7	When an INEND macro expansion is detected, the buffer disposition subtask checks for distribution list, multiple routing, and checkpoint requests.If any of these functions have been requested, the appropriate subtask receives control through a tpost. The controlling subtask returns unused buffers to the buffer return QCB and frees the line by tposting the LCB to itself. The controlling subtask then tposts the buffer to the destination QCB through the DSPCHAIN entry point in the TCAM dispatcher. This activates the Destination scheduler, which places the Send scheduler STCB in the destination LCB. (For concentrator devices, the Send scheduler STCB goes on the element chain of the concentrator data ready queue. The concentrator Send scheduler STCB is put on the STCB chain of the destination LCB.) The appropriate scheduler (Send, Receive, or GET) is whichever STCB is first in the STCB chain of the LCB.	IEDQBD IGG019RB	

Chart C1-1.4 FORWARD Processing



1 Determine if the buffer is to be processed by FORWARD

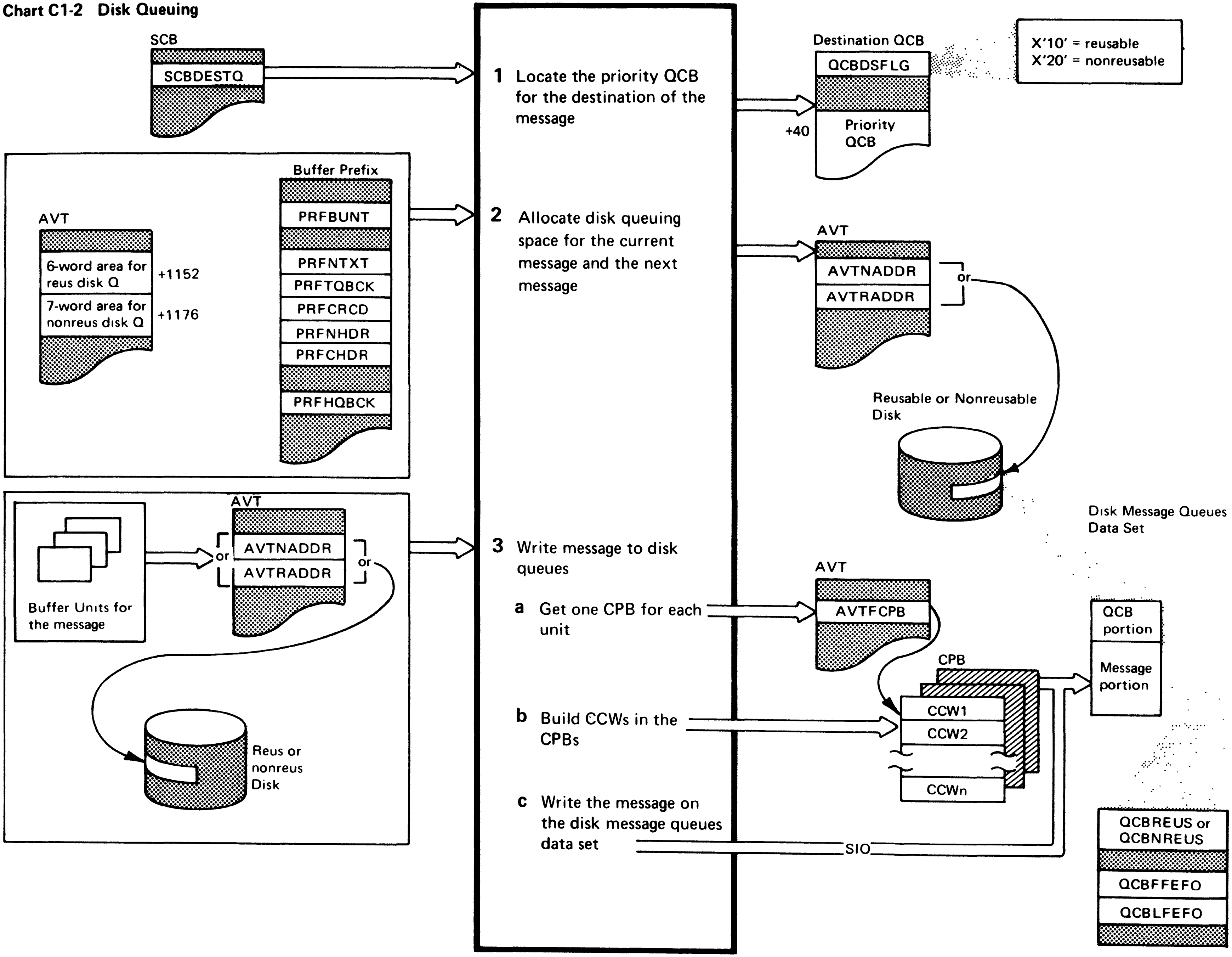
2 Find the destination name, offset, or address, and convert this to the terminal table address

3 Set up the destination QCB

Chart C1-1.4 FORWARD Processing—Description

	Description	Routine	Register Usage
1	If the buffer is zero-length, TSO, or recalled text, or if the line is in extended lock mode, the buffer is not to be processed by the Forward routine. Return to the calling routine. Branch to IEDQAE, IEDQAI, or IEDQA1, depending on the input parameter list. (See the *Macro Linkage Charts* in the *Program Organization* section for details on the input parameter lists for the FORWARD macro.)	IEDQA5	R1 I—parameter list address R6 I—current buffer address
	IEDQAE—Return, in register 15, the destination address from the option field.	IEDQAE	
	IEDQAI—Return, in register 15, a negative 4 if this is a multiple-buffer header or if the EOA string is not found.	IEDQAI	
	IEDQA1—Return, in register 15, the offset to the termname table entry. If the terminal name is not found, return X‘00’.	IEDQA1	
	Place the address of the terminal table entry in register 1.	IEDQTNT	
2	Use PRFDEST (the termname table offset) and AVTRNMPT (address of the termname table) to find the terminal table entry.		
3	Get the address of the QCB (TRMDESTQ) from the terminal table entry and place it into the SCBDESTQ field.	IEDQAV	

Chart C1-2 Disk Queuing



1 Locate the priority QCB for the destination of the message

2 Allocate disk queuing space for the current message and the next message

3 Write message to disk queues

a Get one CPB for each unit

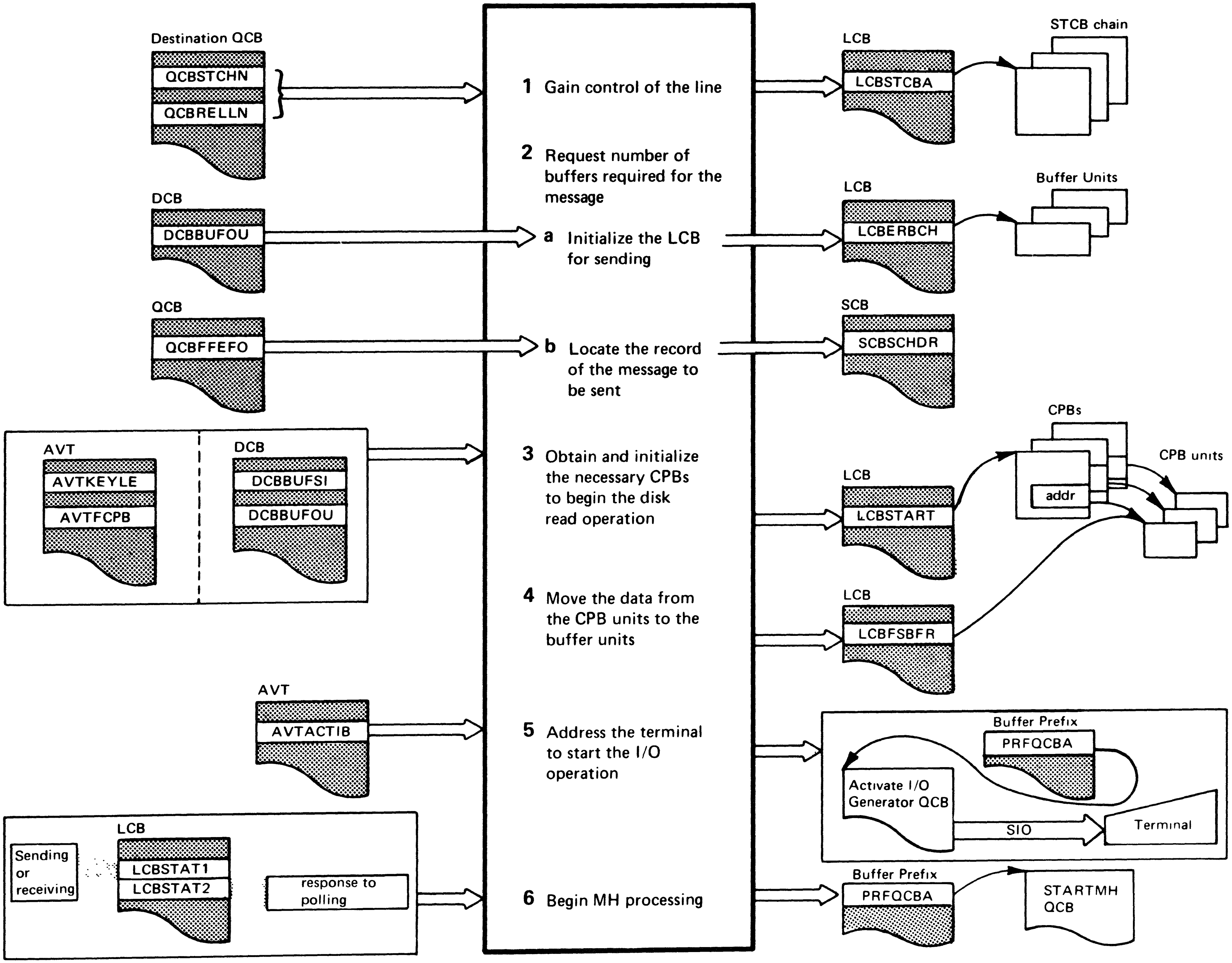
b Build CCWs in the CPBs

c Write the message on the disk message queues data set

Chart C1-2 Disk Queuing—Description

	Description	Routine	Register Usage
1	Locate the priority QCB (begins at QCB+40) for which the buffer is to be written on the message queues data set. This address is at SCBDESTQ. Check QCBDSFLG for X'10'—reusable disk queuing or X'20'—nonreusable disk queuing.	IEDQHM2	R1 I—current buffer address R13 I—AVT address
2	Store the disk address for the first unit of the message header. Indicate the location of the first unit of the next message to be received. Update the index to the disk address to point to the next relative record number (that is, the next message segment). Assign contiguous relative record numbers to the remaining units in the message segment, and update the index in the AVT to keep track of the number of units in the message. Assign the next-buffer location only if this is a multi-buffer message. Update the index to the next available location on the disk.		
3a	Write the message to disk. Issue a GETMAIN macro for the CPB pool and store its address in AVTFCPB. There is one CPB for each buffer unit.		
3b	Build channel programs with the CPBs to write the message on the disk message queues data set. Update the FEFO pointer in the QCB.		
3c	The address in the CCW in the CPB points to the disk buffer unit (CPBXREA) that contains the data. All CCWs may not be present in each CPB; only those necessary to locate the MBBCCHHR for the record are present. Issue an SIO instruction to write the message on the disk message queues data set.	IGG019QE IGG019RC	
	Note: *Figures 11 through 15 illustrate the queuing functions of the Destination scheduler.*		

Chart C1-3.1 Starting a Send Operation



1 Gain control of the line

2 Request number of buffers required for the message

a Initialize the LCB for sending

b Locate the record of the message to be sent

3 Obtain and initialize the necessary CPBs to begin the disk read operation

4 Move the data from the CPB units to the buffer units

5 Address the terminal to start the I/O operation

6 Begin MH processing

Chart C1-3.1 Starting a Send Operation—Description

	Description	Routine	Register Usage
1	An LCB tposted to itself on top of the ready queue indicates that a line is free. A send operation can be initiated when the Send scheduler STCB has top priority in the STCB chain of the LCB. At open time, the Send scheduler STCB is on the STCB chain of the destination QCB to await a full message. Use QCBSTCHN (a pointer to the STCB chain) and QCBRELLN (the relative line number) to find the Send scheduler STCB. When a message is available, move the Send scheduler STCB to the STCB chain of the LCB. LCBSTCBA points to the STCB chain. The Send scheduler STCB remains on the LCB until there is no message to send. At this time, move the Send scheduler STCB to the STCB chain of the destination QCB. For concentrator support: When Concentrator Send Scheduler STCB is on top of the LCB STCB chain, the concentrator Send scheduler gains control to process the STCBs on the element chain of the concentrator data ready queue.	IGG019R4	R1 I—LCB address
2a	Tpost the ERB to the disk I/O QCB to request buffers. Get the number of buffer units assigned for send operations for each line from DCBBUFOU. Put the address of the buffer units into LCBERBCH.		
2b	Use QCBFFEFO to locate the first message to be received and put the message address into SCBSCHDR.	IGG019R4 or IGG019RN	R7 I—QCB address R10 I—DCB address

(This page left blank intentionally)

Chart C1-3.1 Starting a Send Operation—Description *(Continued)*

	Description	Routine	Register Usage
3	Obtain the number of CPBs needed by dividing the size of a buffer (DCBBUFSI) by the size of a unit (AVTKEYLE) and multiplying the result (the number of units per buffer) by the number of buffers (DCBBUFOU). AVTFCPB is the address of the CPB free pool. Build read data CCWs, set sector, and seek and search CCWs in the CPBs, and chain them together. All CPB CCWs may not be present in each CPB; only those necessary to reach the MBBCCHHR of the desired record are present. LCBSTART points to the CPB chain.	IEDQFA IGG019RC	
4	Chain the completed CPBs onto the chain (at AVTDKAPQ) of CPBs to be processed by CPB cleanup . Tpost the CPB cleanup QCB to itself. This notifies IEDQFA that the disk I/O operation is complete. Effectively the data is being moved, but in reality the pointer to the buffer units (LCBFSBFR) is changed to point to the chain of CPB units that contain data, and the empty buffer units are returned to the buffer-unit pool.	IEDQFA IGG019R2	R1 I—IOB address
5	After the required number of buffers are filled, tpost the ERB to the activate-I/O generator, which builds the channel programs to address the terminal and issues the Start I/O instruction. PRFQCBA points to theactivate-I/O generator QCB.	IEDQKA IGG0192E	
6	The LCBSENDN bit on indicates a send operation and the LCBNEGRP bit off indicates a positive response. After a positive response to addressing, tpost the buffer to the STARTMH QCB to begin processing the message through the message handler. (PRFQCBA now points to the STARTMH QCB.) If the response to addressing is negative, tpost a zero-length buffer to the message handler. This indicates an error condition.	IGG019R0	

C1-3.2 STARTMH for a Send Operation

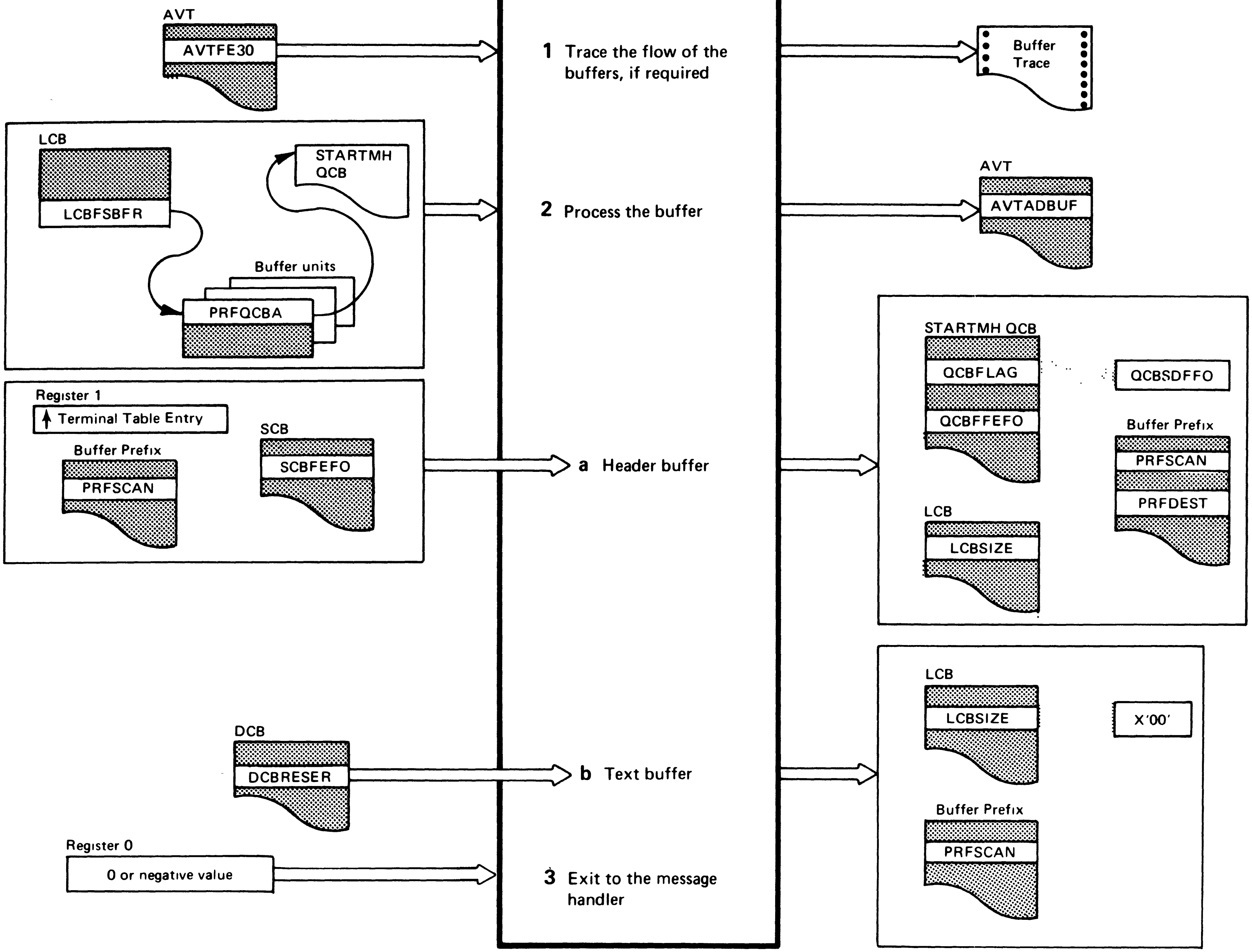


Chart C1-3.2 STARTMH for a Send Operation—Description

	Description	Routine	Register Usage
	Note: *If EOB/ETB processing is specified on the STARTMH macro, the EOB/ETB handling subtask gains control before the STARTMH subtask.*	IEDQBT	
1	If the AVTFE30 field is not zero, the STARTMH subtask gets the address of the Buffer Trace Dump routine (IEDQFE30) from the AVTFE30 field and links to that routine to trace the flow of buffers.	IEDQAA	R1 I—address of the termname table entry R6 I—address of the buffer R13 I—address of the save area in the AVT
2a	LCBFSBFR points to the first buffer on the buffer chain and the first word in the buffer prefix (PRFQCBA) points to the STARTMH QCB. Place the address of the buffer just tposted to the STARTMH QCB in the AVTADBUF field of the AVT. For output header buffers, update the FEFO pointer in the destination QCB (QCBFFEFO) with the FEFO pointer at SCBFEFO, and turn off the "currently sending" flag in the QCB (QCBSDFFO). Put the number of reserve characters in the buffer (from PRFSCAN) into the LCBSIZE field of the LCB. Initialize the scan pointer (PRFSCAN) to point to the last byte of the prefix or, if reserve characters are present, to the last reserve character. Put the termname table entry address into PRFDEST.		
2b	Get the number of reserve characters from DCBRESER. Set the reserve characters count in LCBSIZE to zero, and initialize the scan pointer (PRFSCAN) to point to the last byte in the prefix.		
3	For a normal exit, register 0 will contain zeros; for a multiple-buffer-header condition, register 0 will contain a negative value. Compute the Message Handler entry address and examine register 0. If register 0 contains a negative value, exit to the MH with a condition code of 4; otherwise, determine from the LCB whether the line is sending or receiving, and exit to the MH with a condition code of 1 or 8, respectively.		

Chart C1-3.3 Outgoing MH Processing (Part 1 of 2)

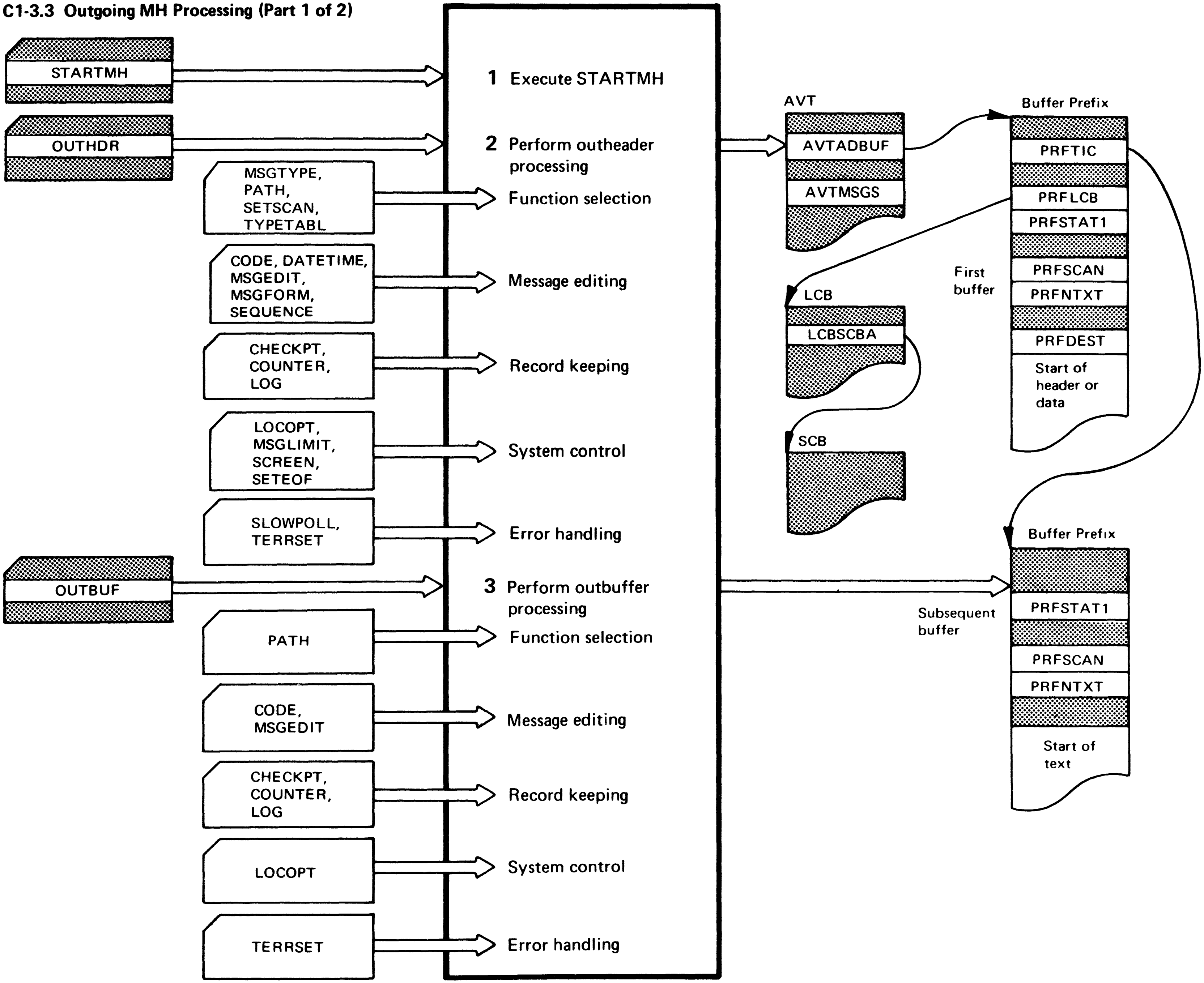
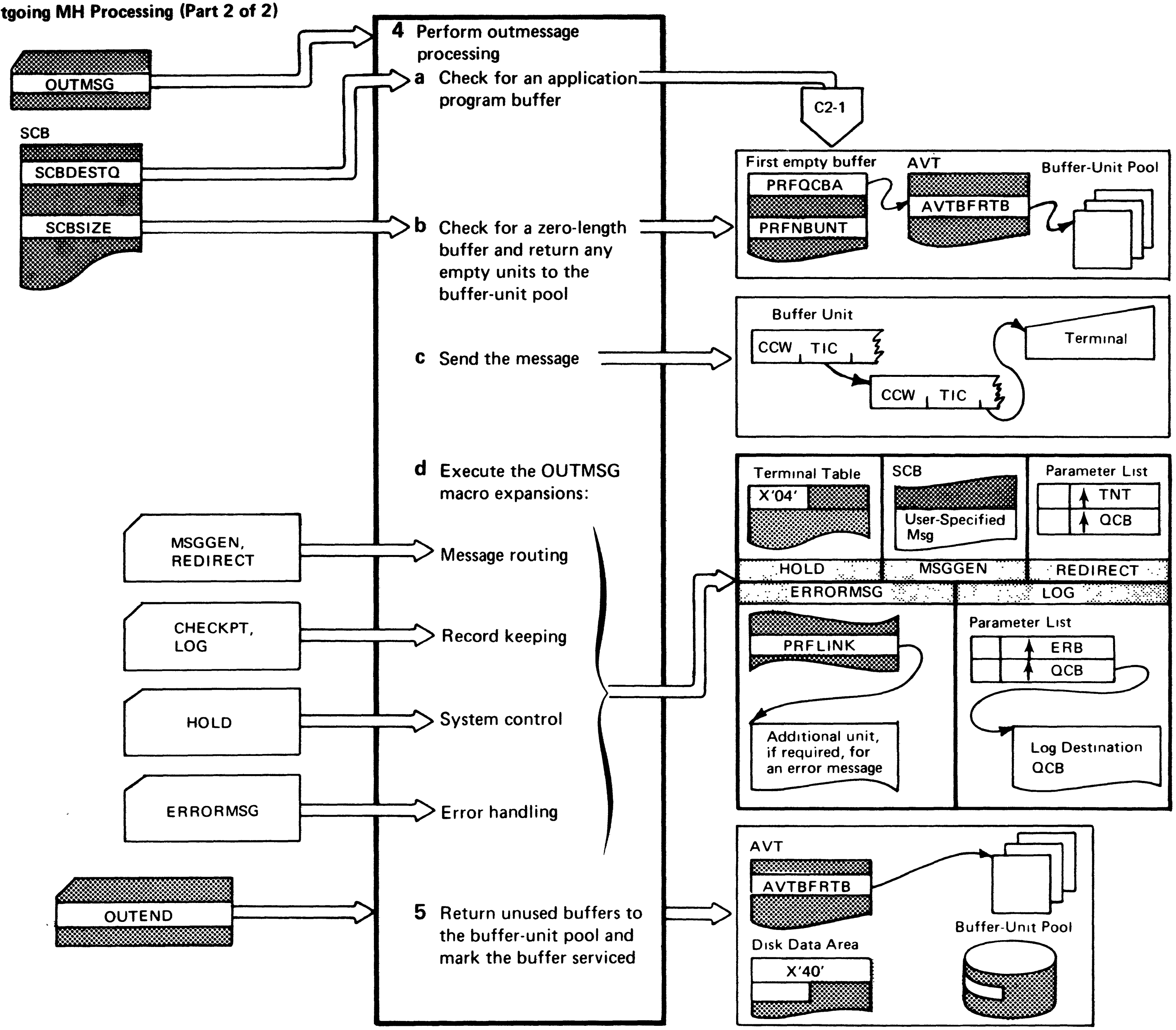


Chart C1-3.3 Outgoing MH Processing—Description (1 of 2)

	Description	Routine	Register Usage
1	See Chart C1-3.2.		
2	MH macro expansions link to functional MH routines through the User Interface routine (IEDQUI). The User Interface routine finds the address of the current buffer in the AVT field AVTADBUF, the address of the LCB in the buffer prefix field PRFLCB, and the address of the current SCB in the LCBSCBA field of the LCB. The PRFTIC field of the buffer prefix points to the next buffer unit of the message. The User Interface routine finds the address of the functional routine for the macro expansion as follows. The AVTMSGS field of the AVT contains the address of the MH VCON table. To this value, the User Interface routine adds an index value obtained from the first byte of the input parameter list. The routine then places the resulting address in register 12. The macros listed, together with their parameter lists and linkages, are shown in the *Macro Linkage Charts* in the *Program Organization* section.	IEDQUI	R1 I—address of parameter list
3	The functions of the macros in the outbuffer subgroup are initiated in the same way as in the outheader subgroup. See the description in item 2, above.		

Chart C1-3.3 Outgoing MH Processing (Part 2 of 2)



4 Perform outmessage processing

a Check for an application program buffer

b Check for a zero-length buffer and return any empty units to the buffer-unit pool

c Send the message

d Execute the OUTMSG macro expansions:

Message routing

Record keeping

System control

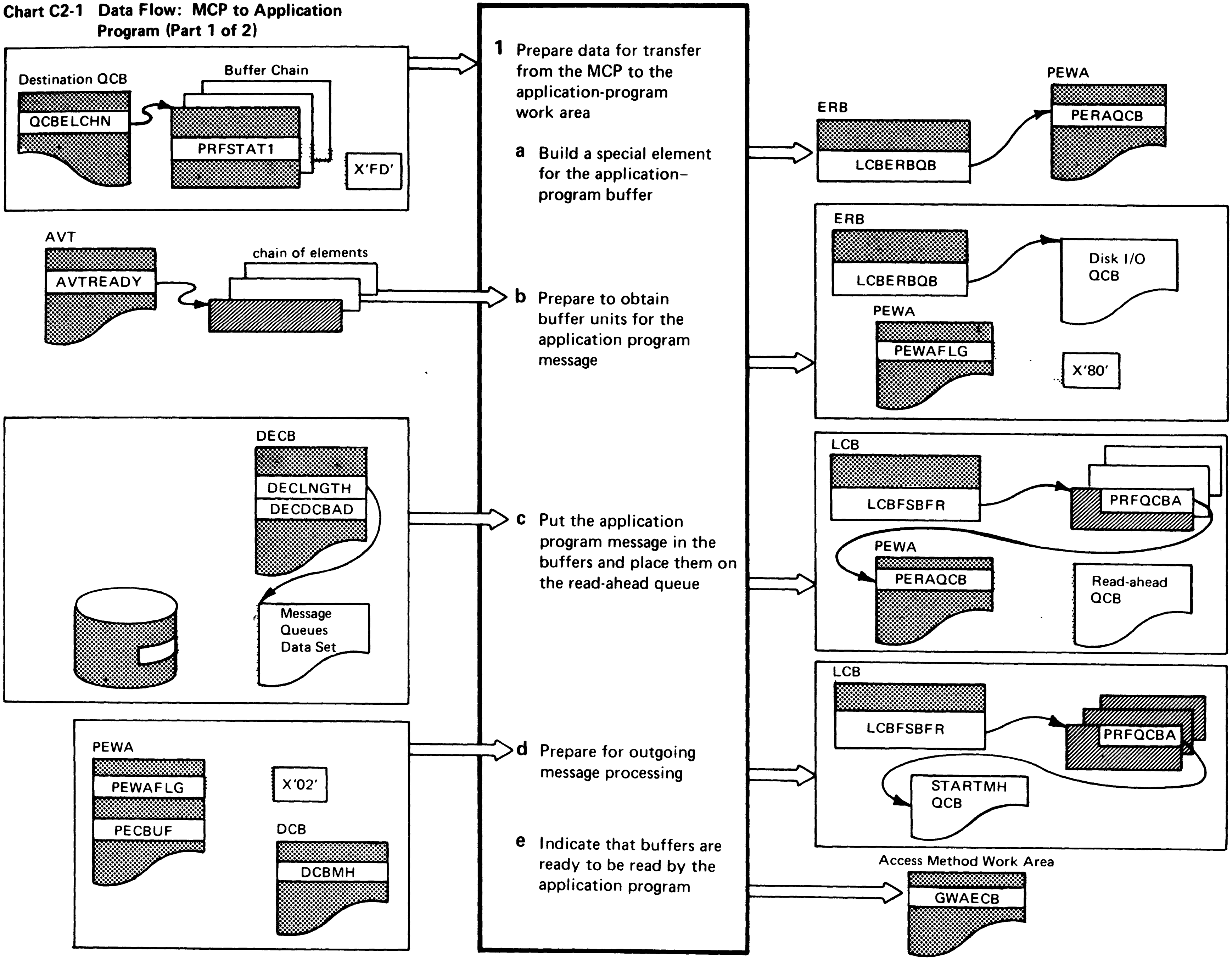
Error handling

5 Return unused buffers to the buffer-unit pool and mark the buffer serviced

Chart C1-3.3 Outgoing MH Processing—Description (2 of 2)

	Description	Routine	Register Usage
4a	Examine the QCBFLAG field of the destination QCB (pointed to by the SCBDESTQ field of the SCB). If QCBFLAG contains a value of X'02', indicating that the QCB is for a process entry, tpost the buffer to the read-ahead QCB. The address of the read-ahead QCB is in the PERAQCB field in the process entry work area.	IEDQA4	
4b	If the buffer has an indicated length of zero, tpost it to the buffer disposition QCB by branching to the DSPPOST entry point in the TCAM dispatcher. If the buffer does not have a length of zero, remove all units that do not contain data from the end of the buffer. When the last empty unit is found, update the PRFNBUNT field of the buffer prefix to indicate only the number of units that contain data. The chain of empty units is now considered a separate buffer. The PRFNBUNT field of the first empty unit contains a count of the number of empty units in the chain. Place the address of the buffer return QCB (AVTBFRTB) in the first word of the first empty unit (PRFQCBA) and tpost the buffer.		
4c	Build Read/Write and TIC CCWs in the first three words of each unit. Include the buffer in the channel program for the line. Issue an I/O interrupt, send the message, and tpost the buffer to the buffer disposition QCB.	IEDQGT	R6 O—address of the buffer
4d	After the last segment of a message has been sent and processed by the MH up to the outmessage subgroup, the buffer disposition subtask executes the macro expansions for each macro in the outmessage subgroup. The macros listed, together with their parameter lists and linkages, are shown in the *Macro Linkage Charts* in the *Program Organization* section.	IEDQBD	
5	When an OUTEND macro is detected, return any unused buffers to the buffer-unit pool by tposting them to the buffer return QCB (AVTBFRTB). Mark as serviced the message that was just sent by making the first six bytes of the unit the data portion of the disk record (disk data record) and putting X'40' in the DATFLAGS field of the disk data area.		

Chart C2-1 Data Flow: MCP to Application Program (Part 1 of 2)



1 Prepare data for transfer from the MCP to the application-program work area

a Build a special element for the application–program buffer

b Prepare to obtain buffer units for the application program message

c Put the application program message in the buffers and place them on the read-ahead queue

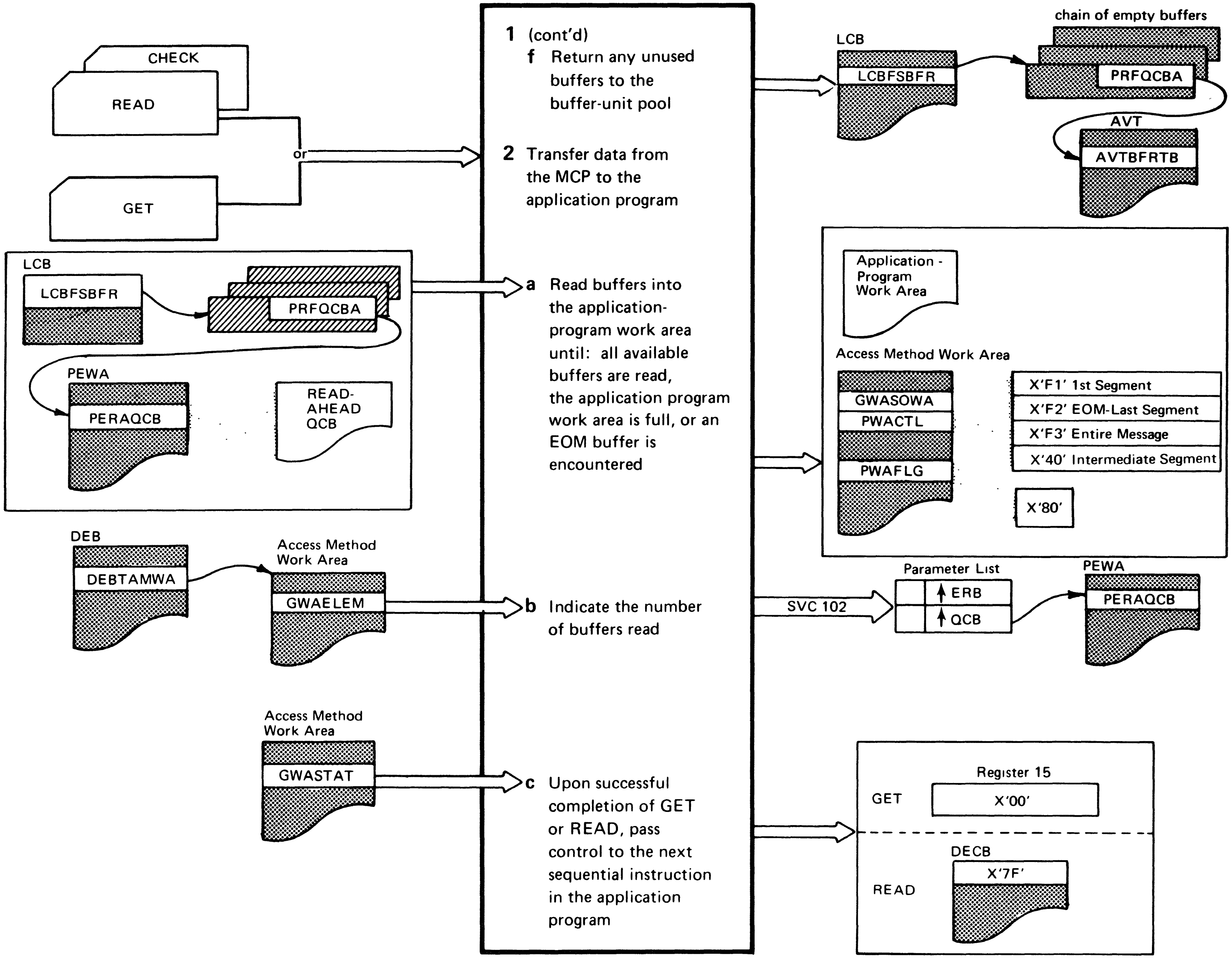
d Prepare for outgoing message processing

e Indicate that buffers are ready to be read by the application program

Chart C2-1 Data Flow: MCP to an Application Program—Description (1 of 2)

	Description	Routine	Register Usage
1	When a buffer of a message is tposted to the destination QCB for an application program, determine (PRFSTAT1=X'FD') if this is the last buffer of the message. If it is not, return control to the TCAM dispatcher.	IEDQHM	
1a	Build a special element for the application-program buffer and tpost it to the read-ahead QCB (PERAQCB) in the process entry work area.	IEDQEW	
1b	AVTREADY points to the first in a chain of elements on the ready queue. When it points to the special element just built, tpost an ERB (with a count of the required buffers for the last message) to the disk I/O QCB. Set the PEWAFLG to X'80' to indicate that the ERB has been tposted to the disk I/O QCB.		
1c	Read the message from the message queues data set into the buffers, and chain the full buffers off the ERB element chain. Tpost the full element chain to the read-ahead QCB.	IEDQFA	
1d	Put the buffers on the pre-MH queue. If the MHOK flag in PEWAFLG is on (X'02'), tpost the first message on the pre-MH queue (PECBUF) to the STARTMH QCB, and turn off the MHOK flag.	IEDQEW	
1e	If a buffer has just been tposted to the STARTMH QCB, post the application-program GET/READ ECB (GWAECB in the access method work area) as complete to indicate that the buffers are ready to be read.This allows the application program to gain control when the MCP enters a wait state.		

Chart C2-1 Data Flow: MCP to Application Program (Part 2 of 2)



1 (cont'd)

f Return any unused buffers to the buffer-unit pool

2 Transfer data from the MCP to the application program

a Read buffers into the application-program work area until: all available buffers are read, the application program work area is full, or an EOM buffer is encountered

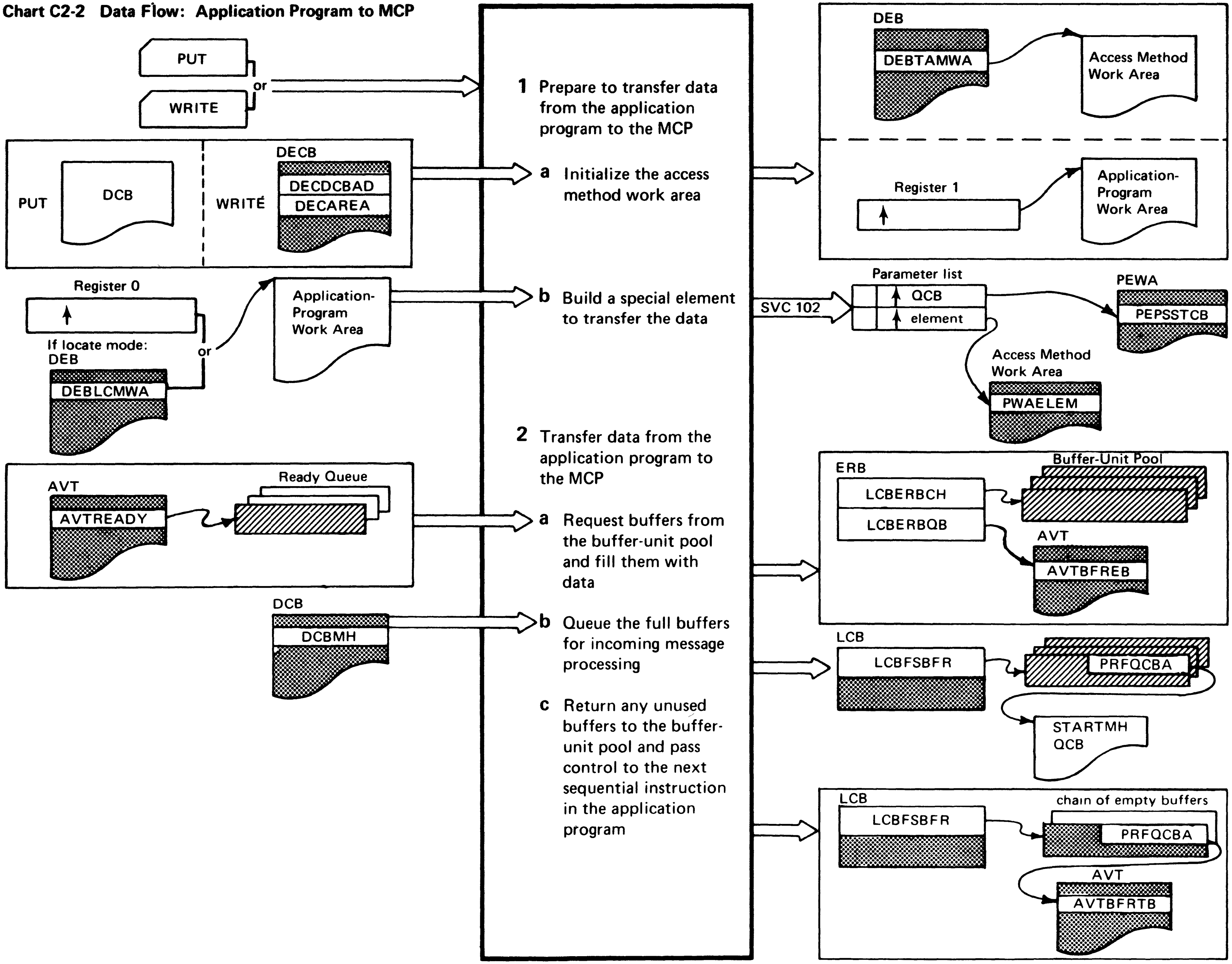
b Indicate the number of buffers read

c Upon successful completion of GET or READ, pass control to the next sequential instruction in the application program

Chart C2-1 Data Flow: MCP to an Application Program—Description (2 of 2)

	Description	Routine	Register Usage
1f	Tpost the empty buffers to the buffer return QCB (AVTBFRTB). If one of the buffers was an EOM, tpost the buffers from the pre-MH queue to the STARTMH QCB, up to an EOM. At EOM, turn on the MHOK flag.		
2a	When a GET or READ macro is issued in an application program, read data from buffers on the element chain of the read-ahead QCB into an application-program work area. The work area contents descriptor byte (PWACTL) contains a value indicating whether the message read into the application-program work area is the first, intermediate, or last segment of the message. The size of the application-program work area is indicated in the GWASOWA field of the access method work area. When an EOM buffer is encountered, set the PWAFLG field of the access method work area to X'80', and turn on the MHOK flag in PEWAFLG to indicate to the GET scheduler that a complete message has been read by the application program. The outgoing MH in the MCP can then begin to process a new message.	IGG019RG	R0 I—address of the application program work area
2b	DEBTAMWA contains the address of the access method work area. Build a buffer return element (GWAELEM) in the access method work area that contains the number of buffers emptied. Using the AQCTL SVC 102 routine, tpost this element to the read-ahead QCB (PERAQCB).	IGG019RG IEDQEB	
2c	When a buffer containing an end-of-file indicator in its prefix is encountered, branch to the user-specified EODAD address. If the SETEOF condition is not present, control is not passed to the next user-coded instruction in the application program until the user request is completely satisfied. The GWASTAT field of the access method work area points to the status indicators for a GET/READ operation. After successful completion of a GET operation, place X'00' into register 15; for a READ operation, place a X'7F' completion code in the DECB.	IGG019RG	

Chart C2-2 Data Flow: Application Program to MCP



1 Prepare to transfer data from the application program to the MCP

a Initialize the access method work area

b Build a special element to transfer the data

2 Transfer data from the application program to the MCP

a Request buffers from the buffer-unit pool and fill them with data

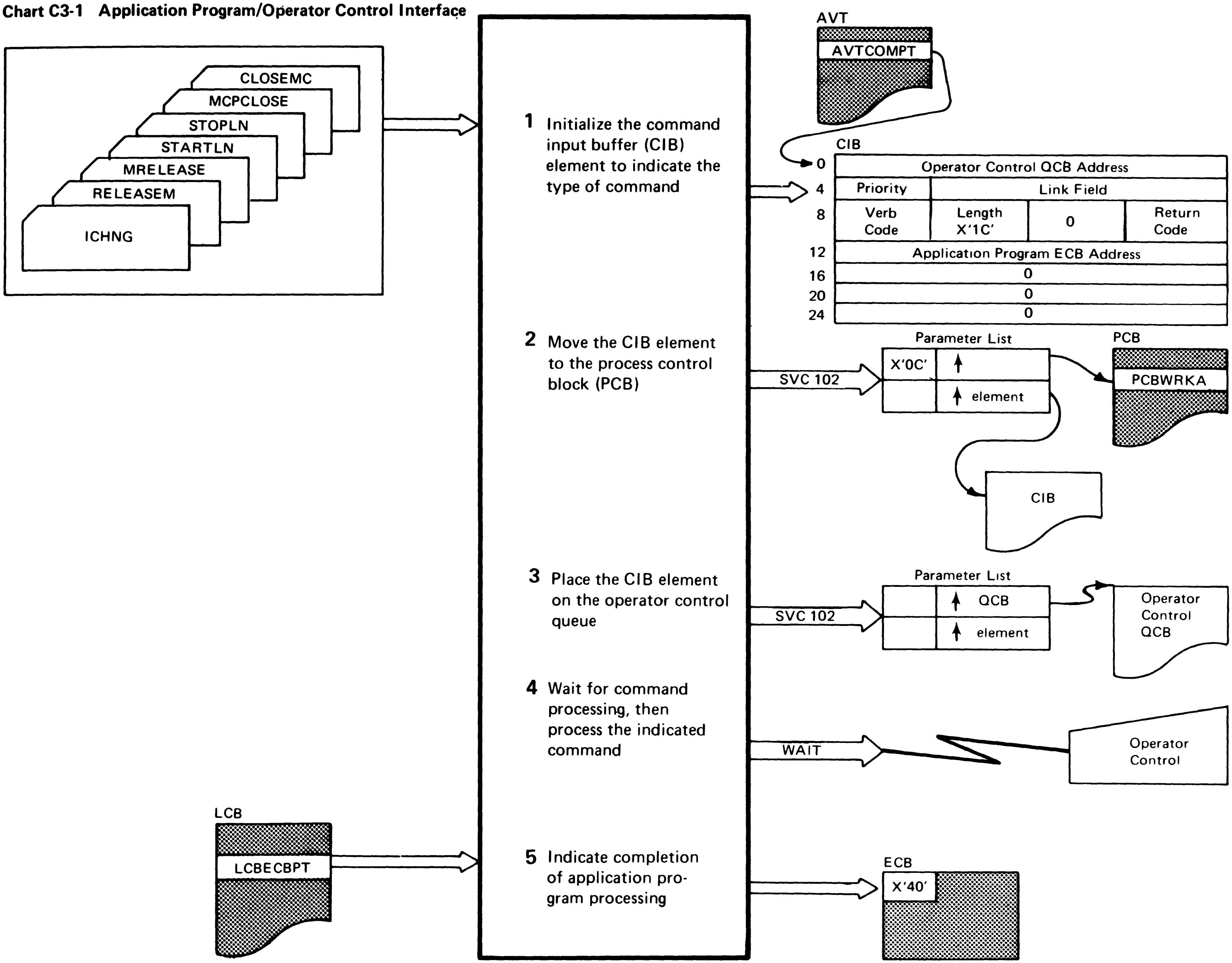
b Queue the full buffers for incoming message processing

c Return any unused buffers to the buffer-unit pool and pass control to the next sequential instruction in the application program

Chart C2-2 Data Flow: Application Program to MCP—Description

	Description	Routine	Register Usage
1a	When a PUT or WRITE macro is issued in a SAM-compatible application program, initialize the access method work area with data from the application-program DCB, DECB, and work area prefix. Put the address of the access method work area into DEBTAMWA.	IGG019RI	R1 I—address of the application program work area
1b	Build a special element (PWAELEM) that contains the address of data in the application-program work area. If locate mode is used, the address of the application-program work area is at DEBLCMWA; otherwise, it is supplied as an operand of the PUT or WRITE macro. Tpost this element, using the AQCTL SVC 102 routine, to the PUT scheduler STCB (PEPSSTCB).	IGG019RI IEDQEB	R0 I—address of element
2a	When the special element reaches the top of the ready queue (AVTREADY), build an ERB to request buffers for the data in the application-program work area. Fill the buffers, one at a time, until the application-program work area is empty.	IEDQEC	R1 I—address of element
2b	Tpost the full buffers to the STARTMH QCB (address in DCBMH) for this application program in order to process the incoming message.	IEDQEC IEDQEB	
2c	Tpost the empty buffers to the buffer return QCB (AVTBFRTB), and use SVC 102 to post the application-program ECB (PWAECB) complete. As a result, the application program can regain control at its next sequential instruction whenever the MCP enters the wait state.	IEDQEC	

Chart C3-1 Application Program/Operator Control Interface



1 Initialize the command input buffer (CIB) element to indicate the type of command

2 Move the CIB element to the process control block (PCB)

3 Place the CIB element on the operator control queue

4 Wait for command processing, then process the indicated command

5 Indicate completion of application program processing

CIB

0	Operator Control QCB Address			
4	Priority	Link Field		
8	Verb Code	Length X'1C'	0	Return Code
12	Application Program ECB Address			
16	0			
20	0			
24	0			

Chart C3-1 Application Program/Operator Control Interface—Description

	Description	Routine	Register Usage
1	AVTCOMPT points to the command input buffer (CIB). The CIB fields are initialized according to the macro specified.	IEDQET	
2	Move the initialized CIB into the PCB work area beginning at PCBWRKA (PCB+60). The parameters are the standard SVC 102 parameters with a X'0C' in the high-order byte of the first word.		
3	Tpost (using AQCTL SVC 102) the CIB in PCBWRKA (PCB+60) to the operator control QCB. Issue a WAIT macro to put the application program into a wait state.	IEDQET IEDQEB	
4	See the *Operator Control Linkage Charts* in the *Program Organization* section for an explanation of the operation performed when a specific command is entered.	IEDQCA	
5	Post the waiting application-program ECB (address in LCBECBPT) complete.	IEDQCA IEDQEB	

Chart C3-2 Application Program Network Control

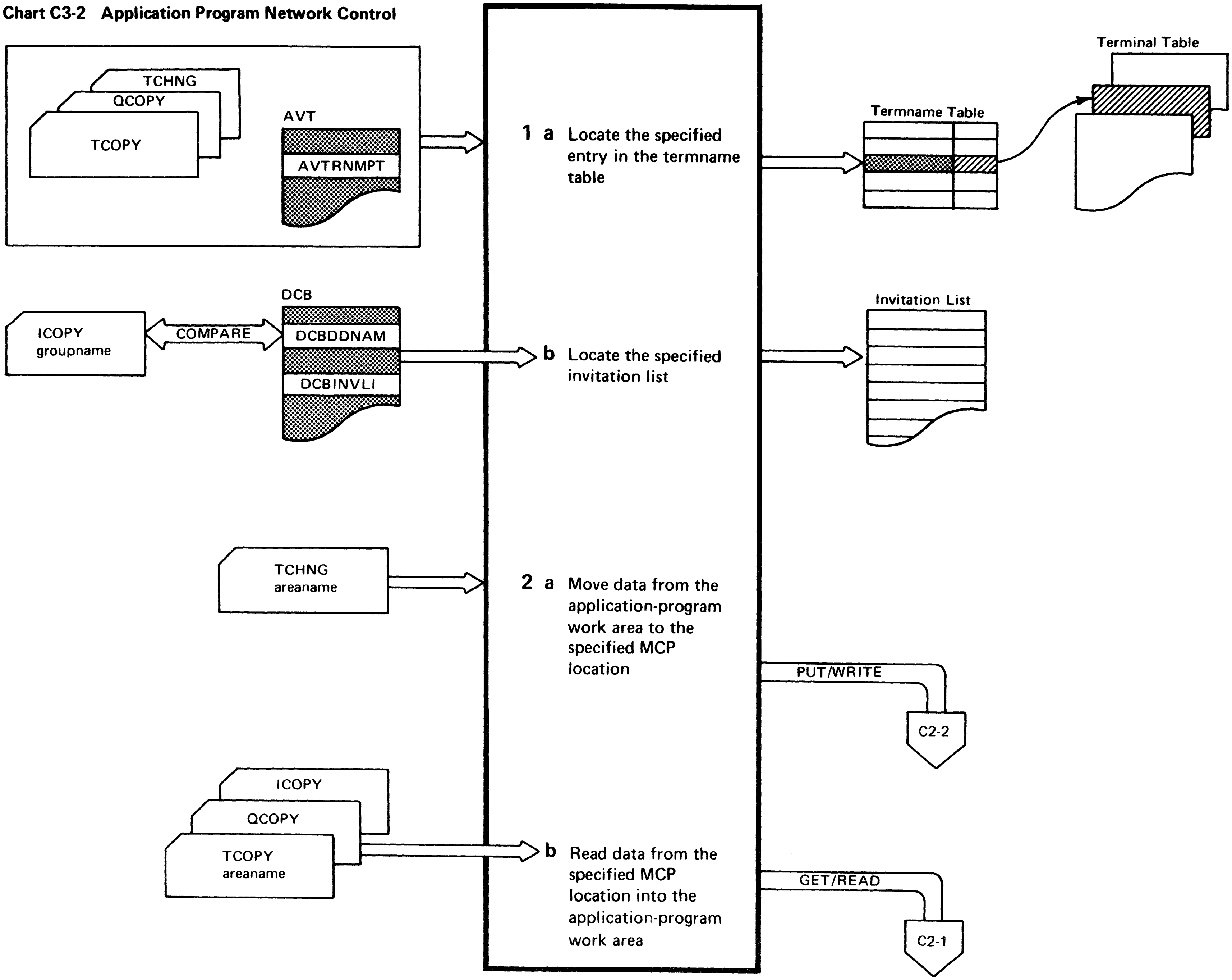
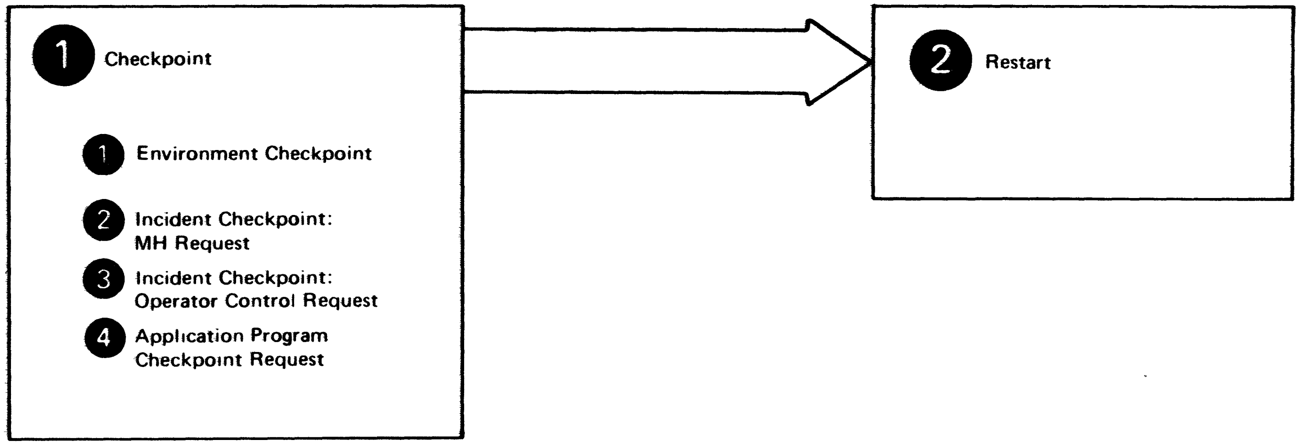

TCHNG
QCOPY
TCOPY
AVT
AVTRNMPT
1 a Locate the specified entry in the termname table
Termname Table
Terminal Table
DCB
ICOPY groupname
COMPARE
DCBDDNAM
DCBINVLI
b Locate the specified invitation list
Invitation List
TCHNG areaname
2 a Move data from the application-program work area to the specified MCP location
PUT/WRITE
C2-2
ICOPY
QCOPY
TCOPY areaname
b Read data from the specified MCP location into the application-program work area
GET/READ
C2-1

Chart C3-2 Application Program Network Control—Description

	Description	Routine	Register Usage
1a	Get the termname table address from AVTRNMPT to locate the termname table entry.	IEDQE1 IEDQE2 IEDQE3	
1b	Compare the *grpname* coded on the ICOPY macro with the *ddname* in each TCAM line group DCB. Use the DCBINVLI field of the matching DCB to locate the specified invitation list.	IEDQE4	
2a	Move the data from the application-program work area into the MCP location coded as *areaname* on the TCHNG macro.	IEDQE3	
2b	Read data from the MCP location (coded as *areaname* on the macro) into the application-program work area.	IEDQE1 IEDQE2 IEDQE4	

D CHECKPOINTING/RESTARTING THE SYSTEM


1 Checkpoint
1 Environment Checkpoint
2 Incident Checkpoint: MH Request
3 Incident Checkpoint: Operator Control Request
4 Application Program Checkpoint Request
2 Restart

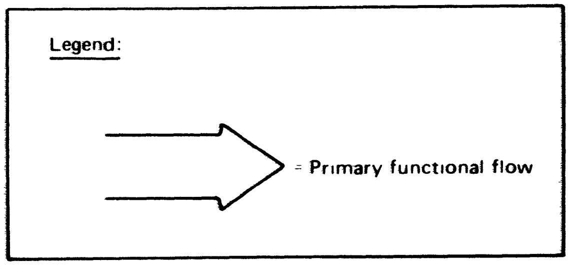
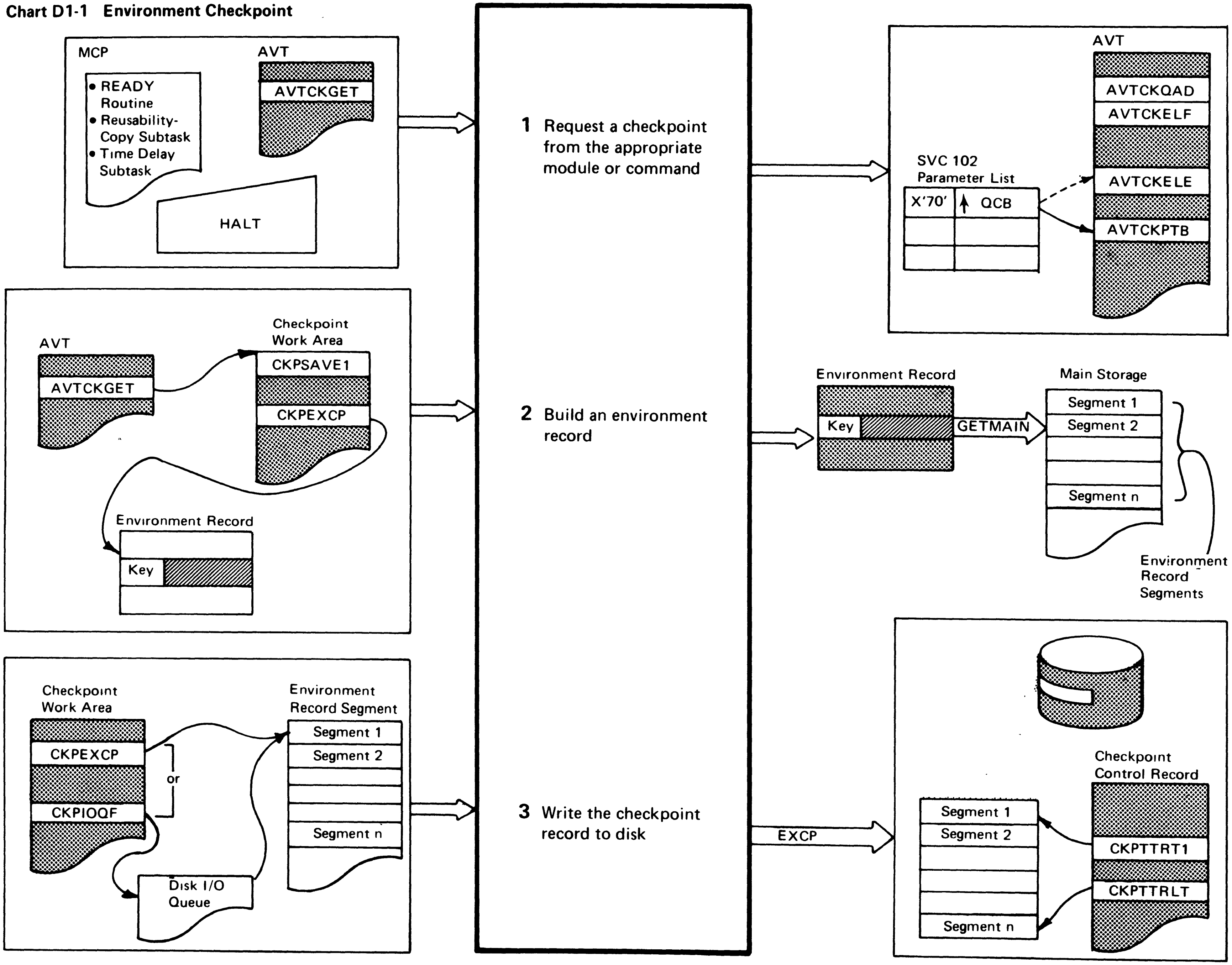


Chart D Checkpointing/Restarting the System

Description	**Chart No.**
Checkpoint	D1-1
describes building a checkpoint record and writing the checkpoint record to disk.	
Environment Checkpoint	
Incident Checkpoint: MH Request	D1-2
Incident Checkpoint: Operator Control Request	D1-3
Application Program Checkpoint Request	D1-4
Restart	D2
describes the reconstruction of the MCP environment and the message queues.	
Note: *This description applies to Diagrams D1-1 through D1-4. For details concerning the checkpoint records, see the* ***Data Area Layouts*** *section of this publication.*	

Chart D1-1 Environment Checkpoint



1 Request a checkpoint from the appropriate module or command

2 Build an environment record

3 Write the checkpoint record to disk

Chart D1-2 Incident Checkpoint: MH Request

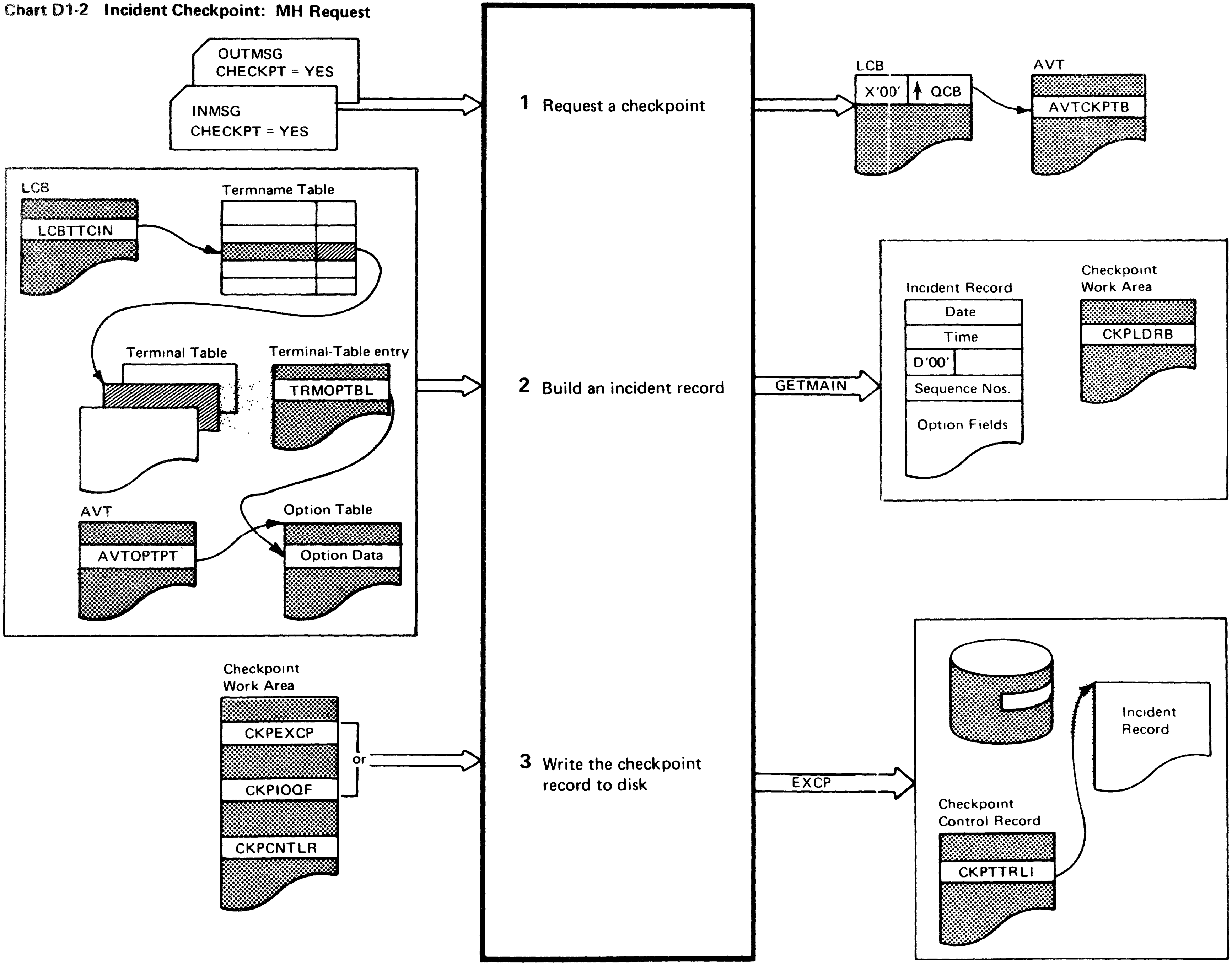
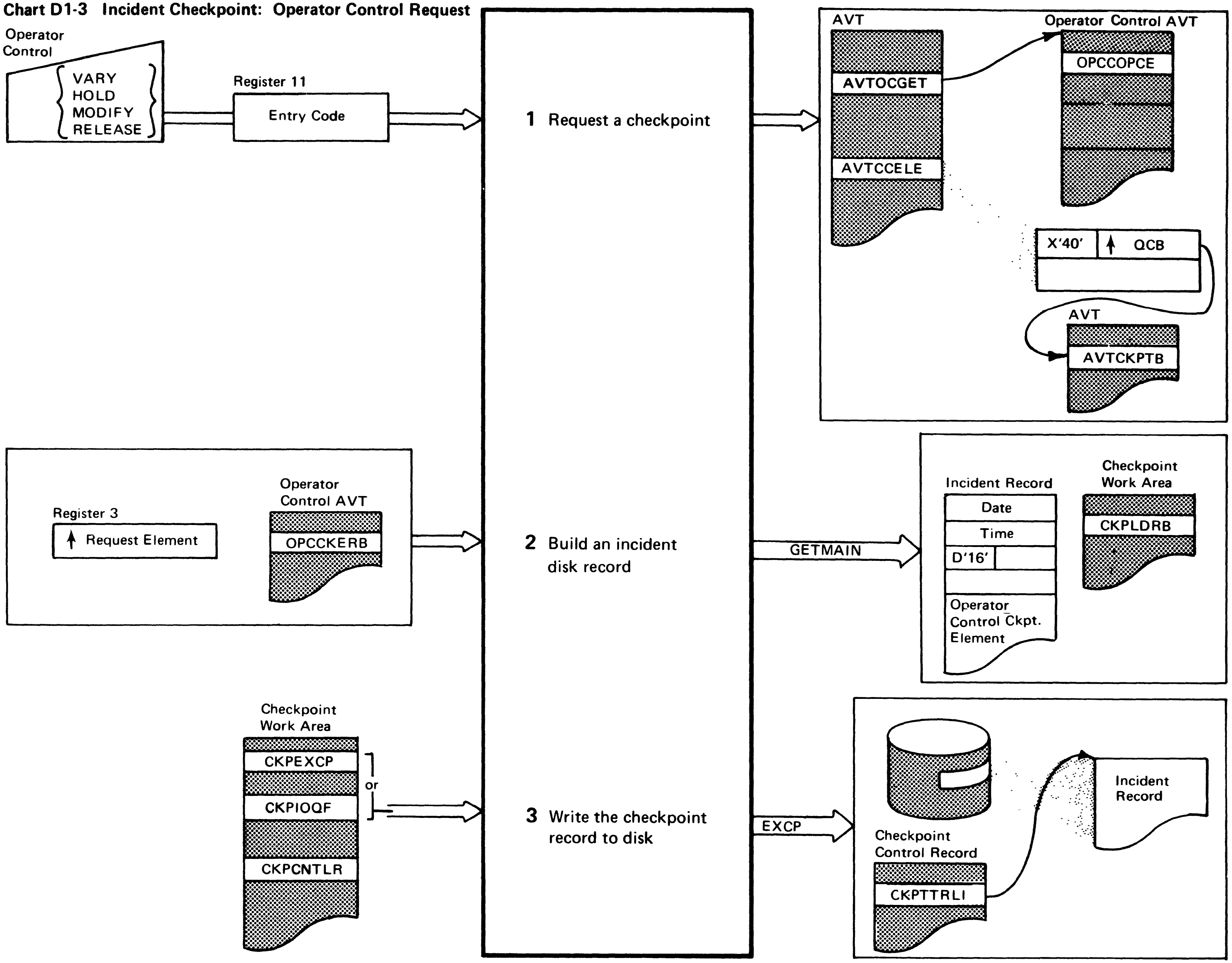


Chart D1-3 Incident Checkpoint: Operator Control Request

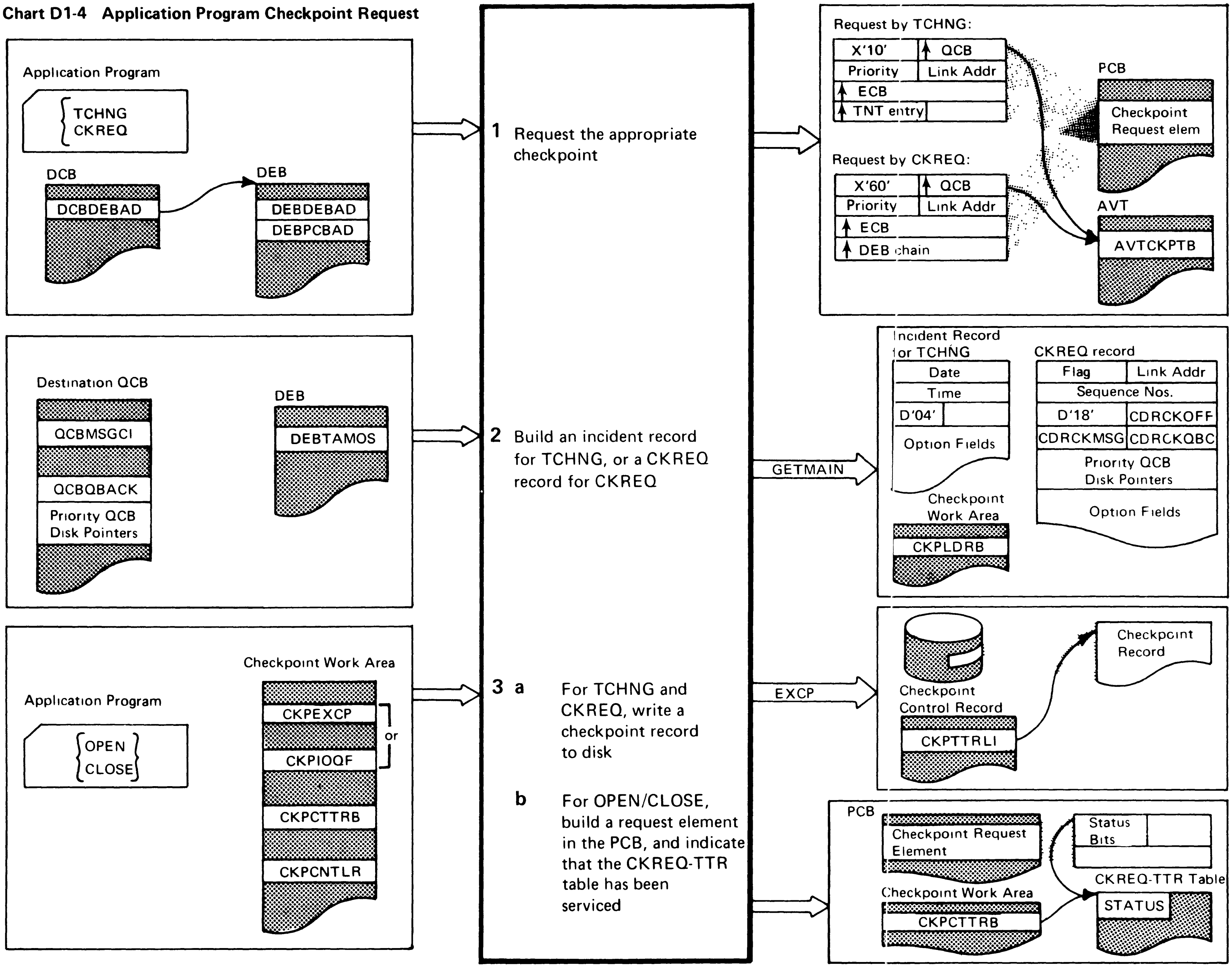


1 Request a checkpoint

2 Build an incident disk record

3 Write the checkpoint record to disk

Chart D1-4 Application Program Checkpoint Request



1 Request the appropriate checkpoint

2 Build an incident record for TCHNG, or a CKREQ record for CKREQ

3 a For TCHNG and CKREQ, write a checkpoint record to disk

b For OPEN/CLOSE, build a request element in the PCB, and indicate that the CKREQ-TTR table has been serviced

(This page left blank intentionally)

Chart D1 Checkpoint—Description (1 of 3)

Description	Routine	Register Usage
1 *Environment Checkpoint Request:* If the READY macro requests a checkpoint, turn on bit 0 of AVTCKELF; put the environment element on the time delay queue (AVTCKELE). If AVTCKGET (the address of the checkpoint work area) is nonzero, process all of the incident records that are more recent than the environment record. Invoke the time delay subtask to remove the environment element from the time delay queue; then tpost the element to AVTCKPTB, the checkpoint QCB, using SVC 102. If a HALT command is issued from a terminal, call the MCP closedown processing routine, which sets the "closedown-in-progress" bit in AVTBIT1 and tposts the checkpoint request element to the checkpoint QCB (using SVC 102).	IGCZ010D IEDQEB IEDQND IGG019RP	
MH Checkpoint Request: Determine that the last segment of a message has been processed by the MH up to the inmessage/outmessage subgroup. Tpost the LCB, which is serving as a checkpoint request element, to the checkpoint QCB (AVTCKPTB).	IEDQBD IEDQNX	
Operator Control Checkpoint Request: The entry code in register 11 should be X'01', indicating a checkpoint. AVTOCGET is the address of the operator control AVT and AVTCCELE is the incident checkpoint request element. Build a checkpoint request element at AVTCCELE. Tpost the element to the checkpoint QCB (AVTCKPTB).	IGC0610D IEDQEB	
Application Program Checkpoint Request: Build a TCHNG or CKREQ request element in the PCB (its address is found at DEBPCBAD). Tpost the element to the checkpoint QCB (AVTCKPTB).	IEDQNB IEDQEB	

Chart D1 Checkpoint—Description (2 of 3)

Description	Routine	Register Usage

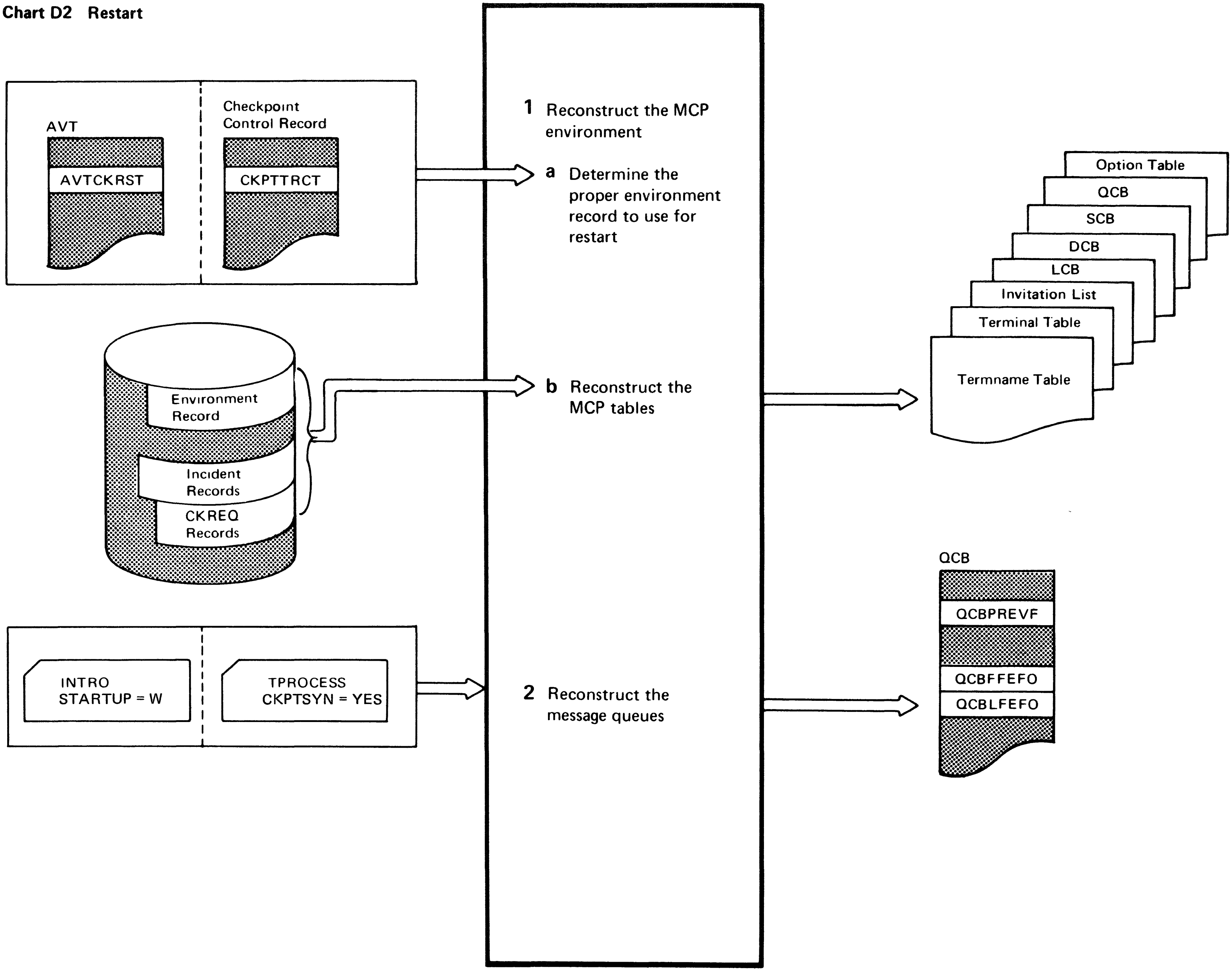
2 *Environment Checkpoint Request:* AVTCKGET points to the checkpoint work area. If the key field of the record pointed to by CKPEXCP is X'20', a subsequent segment of an environment record is to be built. If the key field is X'1C', this is the last segment to be built. For a first (or only) segment, issue a GETMAIN macro for an area into which to build a new segment. Place the address of this area into CKPEXCP. IEDQNK gets registers from CKPSAVE1 and builds the next environment record in the area pointed to by CKPEXCP. If the segment will not contain all the data in the record, place X'20' in the key field and store the register values in CKPSAVE1. For a subsequent record, get the register values from CKPSAVE1 and build the next segment in the work area pointed to by CKPEXCP. Return to IEDQNF with an offset in register 15 indicating the next checkpoint routine to gain control:
Offset = 0: Incident Checkpoint for MH routine (IEDQNG)
Offset = 8: Incident Checkpoint for TCHNG routine (IEDQNH)
Offset = 32: Incident Checkpoint for Operator Control routine
Offset = 40: Environment Checkpoint routine (IEDQNK)
Offset = 48: Build CKREQ Disk Record routine (IEDQNM)
Offset = 64: Checkpoint Queue Manager (IEDQNO)
Offset = 72: Checkpoint Disk I/O routine (IEDQNP)
Offset = 80: Checkpoint Notification and Disposition routine
Offset = 88: No Available Core routine (IEDQNR)
Offset = 96: No Incident Record routine (IEDQNS)
	(IEDQNJ)
	(IEDQNQ)
	IEDQNK
	IEDQNF	

MH Checkpoint Request: Obtain main storage for a disk record by issuing a GETMAIN macro and store the address in CKPLDRB.	IEDQNG SVC 4	
Operator Control Checkpoint Request: Use the request element pointed to by register 3 to build an incident record in a GETMAIN area. Store the address of the area in CKPLDRB.	IEDQNJ	

Chart D1 Checkpoint—Description (3 of 3)

	Description	Routine	Register Usage
	Application Program Checkpoint Request: Build an incident record for TCHNG in a GETMAIN area and store its address in CKPLDRB. Build a CKREQ record in a GETMAIN area. If this is the first record for a request, store the address in CKPLDRB. If this is a subsequent record, store the address in CKPEXCP.	IEDQNB IEDQNH IEDQNM	
3	Locate the next record to be written as follows: If CKPEXCP is nonzero, CKPEXCP contains the address of the record just written, and is the continuation of a checkpoint that requires more than one segment. If CKPEXCP is zero, the first record on the checkpoint disk I/O queue (pointed to by CKPIOQF) is the one just written. Put the address of this record into CKPEXCP. Determine the correct TTR for the record as follows: *Environment Checkpoint, First Segment:* The control record begins at CKPCNTLR in the checkpoint work area and contains the TTR of all first segments (CKPTTRT1) and an index to the current one (CKPTTRCT). *Environment Checkpoint, Subsequent Segments:* The work area contains the TTR of the last segment written (CKPTTRLT). *Incident Record:* The work area contains the TTR of the last incident record written (CDRTTRLI). *Application Program CKREQ Record:* The work area contains a CKREQ-TTR table that associates a terminal name offset with a particular TTR. Issue an EXCP macro to write the record.	IEDQNP SVC 0	
	For OPEN/CLOSE macros in an application program, build a checkpoint request element in the PCB. If an entry in the CKREQ-TTR table is involved, invert the status of the CKREQ-TTR entry.		

Chart D2 Restart



1 Reconstruct the MCP environment

a Determine the proper environment record to use for restart

b Reconstruct the MCP tables

2 Reconstruct the message queues

Chart D2 Restart—Description

	Description	Routine	Register Usage
1a	Subtract the restart number in AVTCKRST from the number of the most current environment record (CKPTTRCT). If the result is positive, use this environment record for the restart. If the result is not positive, add to it the value from CKPCPRCD. This gives the total number of environment records.	IGG01943	
1b	Read the environment record and place the information in the MCP tables (terminal table, QCB, LCB, DCB, termname table, SCB, option table, and invitation list). *Incident Records:* CHECKPT—Update the tables to show the current terminal to receive a message. Operator Control—Update the destination QCB to show the current status of the line as indicated by an incident record for a Startline or Stopline operator control command. All other operator control incident records are processed during READY execution. (See Chart A3.) TCHNG—Update the tables to show the change in status. *CKREQ Records:* Update the MCP tables that pertain to application programs.	IGG01943 IGG01944	
2	Normal restart: Check the messages for logical-read errors. Create the FEFO chain for all complete, unserviced, and uncanceled messages. QCBFFEFO—Disk record number of the first message to be received. Main-storage address of the first record if main-storage-only queuing. QCBLFEFO—Disk record number of the last message to be received. Main-storage address of the last record if main-storage-only queuing. QCBINTLF—Disk record number of the last intercepted message in FEFO order. Update the sequence numbers. STARTUP=WY indicates that no scan of the message queues is to be done. Locate the last message placed on the queue before the checkpoint was taken. Zero the FEFO chain field to any message placed on the queue after that checkpoint, and follow CKPTSYN=YES restart procedures for all queues. CKPTSYN=YES indicates a system synchronized restart. Create the FEFO queue, including all the messages on the FEFO queue after the last checkpoint. If the sequence number of the message on the FIFO queue is greater than the sequence number in the checkpoint record for the application queue, mark the message as unserviced.	IGG01945	

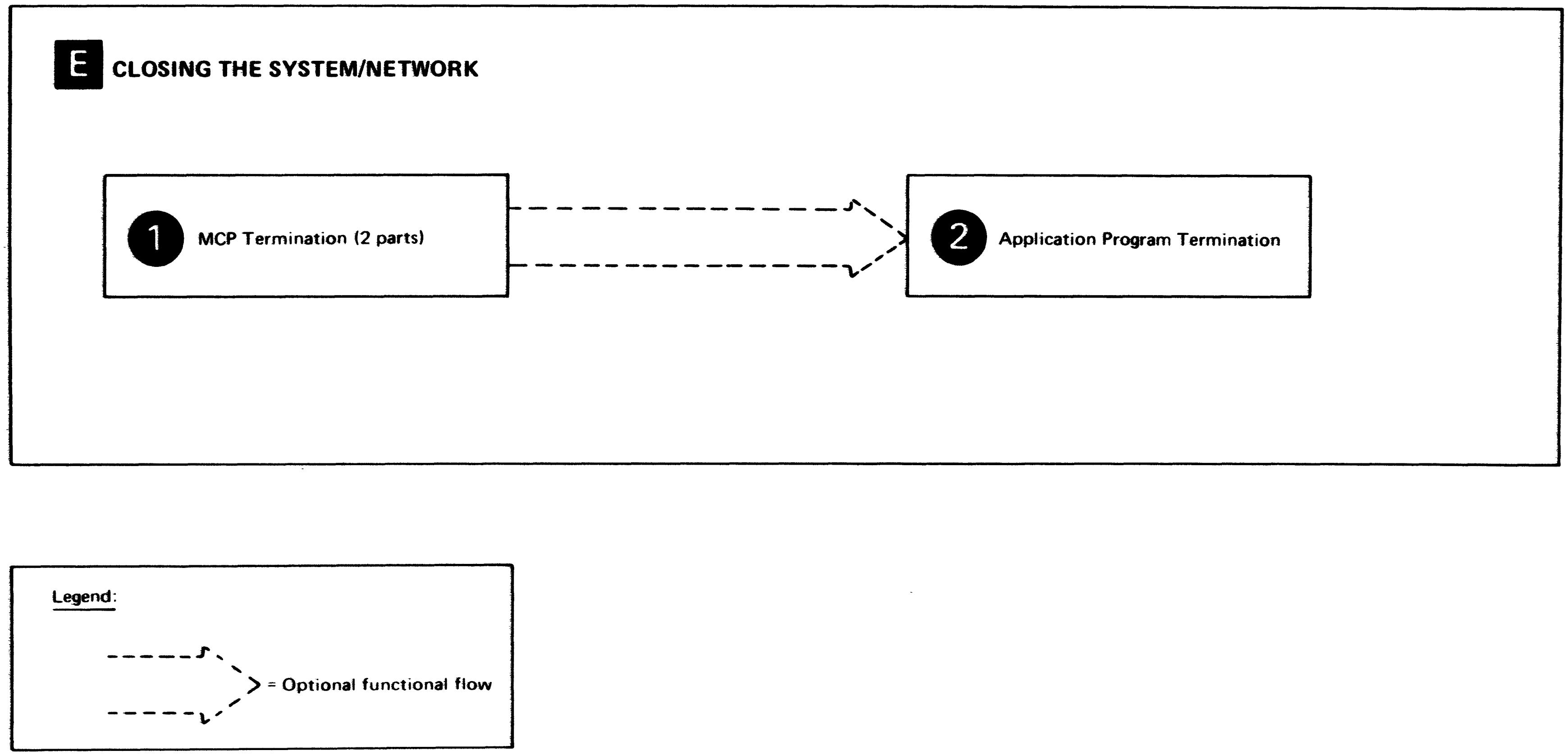
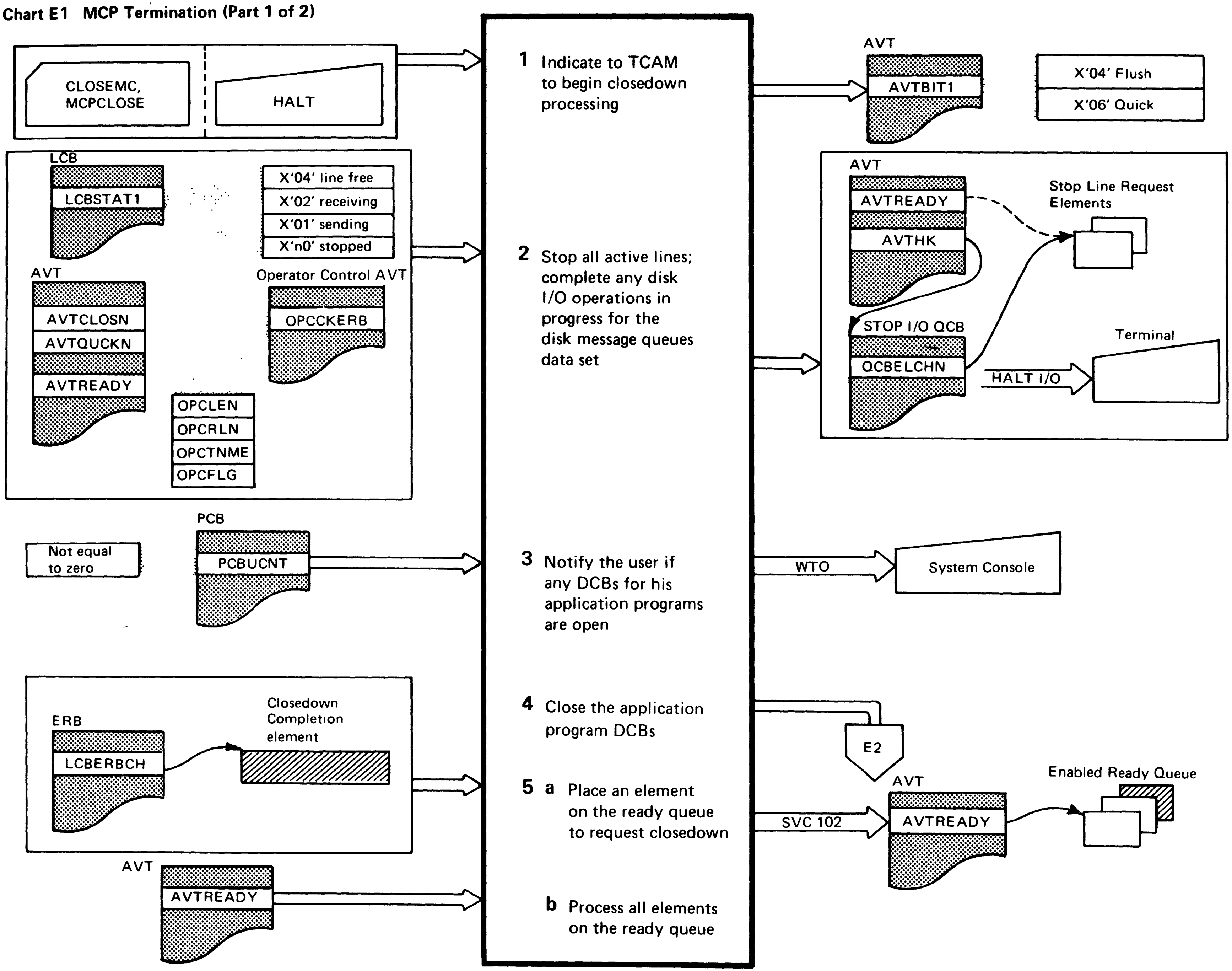

E
CLOSING THE SYSTEM/NETWORK
1 MCP Termination (2 parts)
2 Application Program Termination
Legend:
= Optional functional flow

Chart E Closing the System/Network

Description	**Chart No.**
MCP Termination describes the closing of all opened MCP data sets and the deactivation of the TCAM system	E1
Application Program Termination	E2

Chart E1 MCP Termination (Part 1 of 2)



1 Indicate to TCAM to begin closedown processing

2 Stop all active lines; complete any disk I/O operations in progress for the disk message queues data set

3 Notify the user if any DCBs for his application programs are open

4 Close the application program DCBs

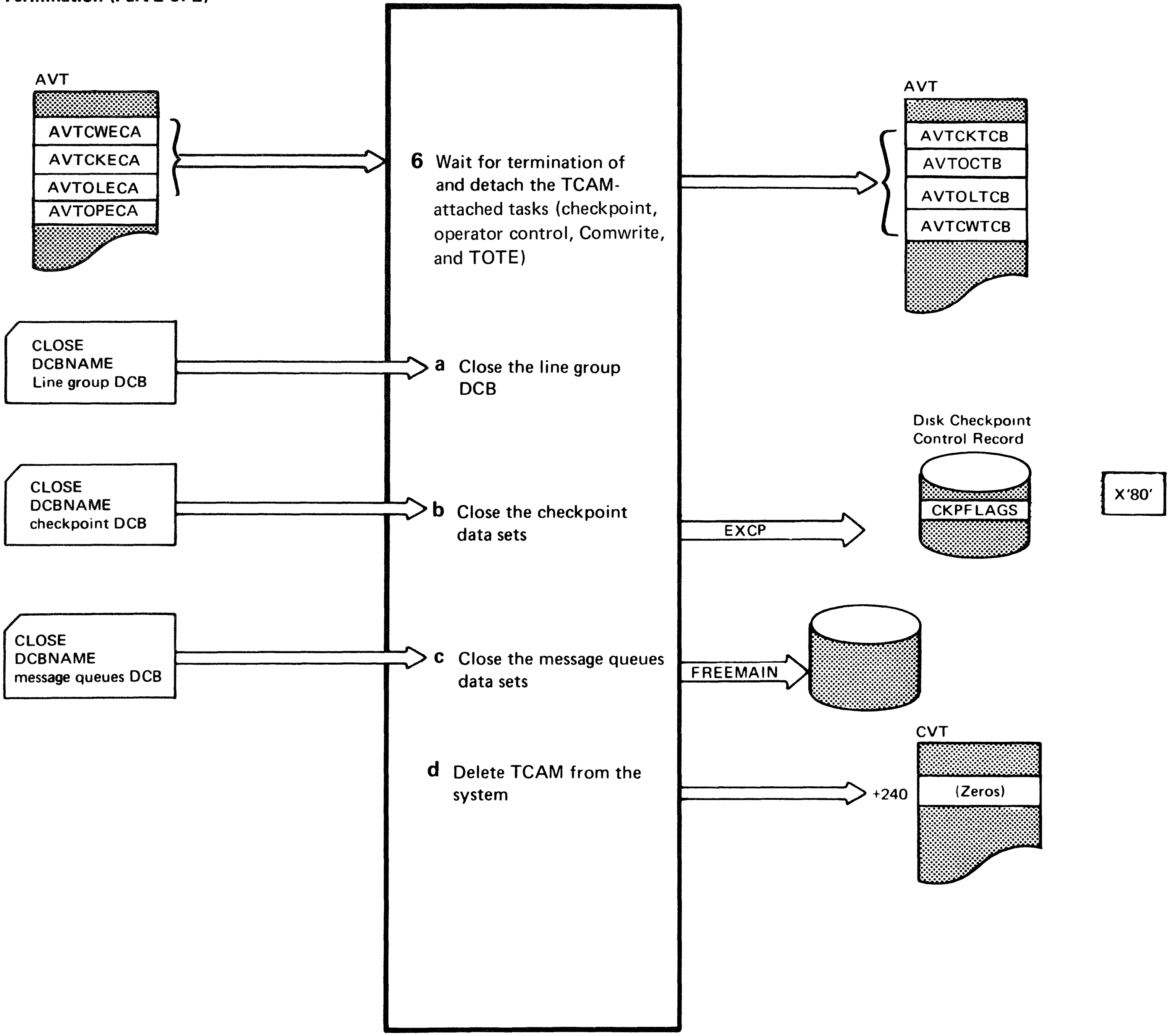
5 a Place an element on the ready queue to request closedown

b Process all elements on the ready queue

Chart E1 MCP Termination—Description (1 of 2)

	Description	Routine	Register Usage
1	Set the closedown switches in the AVTBIT1 field (X'04' for flush closedown or X'06' for quick closedown).	IGCZ010D	
2	For a quick closedown, keep a count of the LCBs until all the lines stop sending. For a flush closedown, wait for all the queues to be serviced.	IGCV310D IEDQHK	
3	Issue the WTO message IED098I DCB OPEN FOR MESSAGE PROCESSING jobname Then wait for the operator control ECB (AVTOPECA) to be posted complete by the closedown routines.	IGCZ010D	
4	See Chart E2.		
5a	Tpost the closedown completion element onto the ready queue (AVTREADY).		
5b	The TCAM dispatcher gives control to each of the elements on the ready queue until it reaches the closedown completion element. In the event that the operator control checkpoint request element is on the ready queue, the checkpoint executor gains control and requests an environment checkpoint. The Environment Checkpoint routine recognizes the source as operator control and activates the Checkpoint Notification and Disposition routine to place the closedown completion element on the ready queue. When the TCAM dispatcher recognizes this element, it continues closedown processing.	IGG019RB or IGG019RO IEDQNF IEDQNK IEDQNQ	

Chart E1 MCP Termination (Part 2 of 2)



6 Wait for termination of and detach the TCAM-attached tasks (checkpoint, operator control, Comwrite, and TOTE)

a Close the line group DCB

b Close the checkpoint data sets

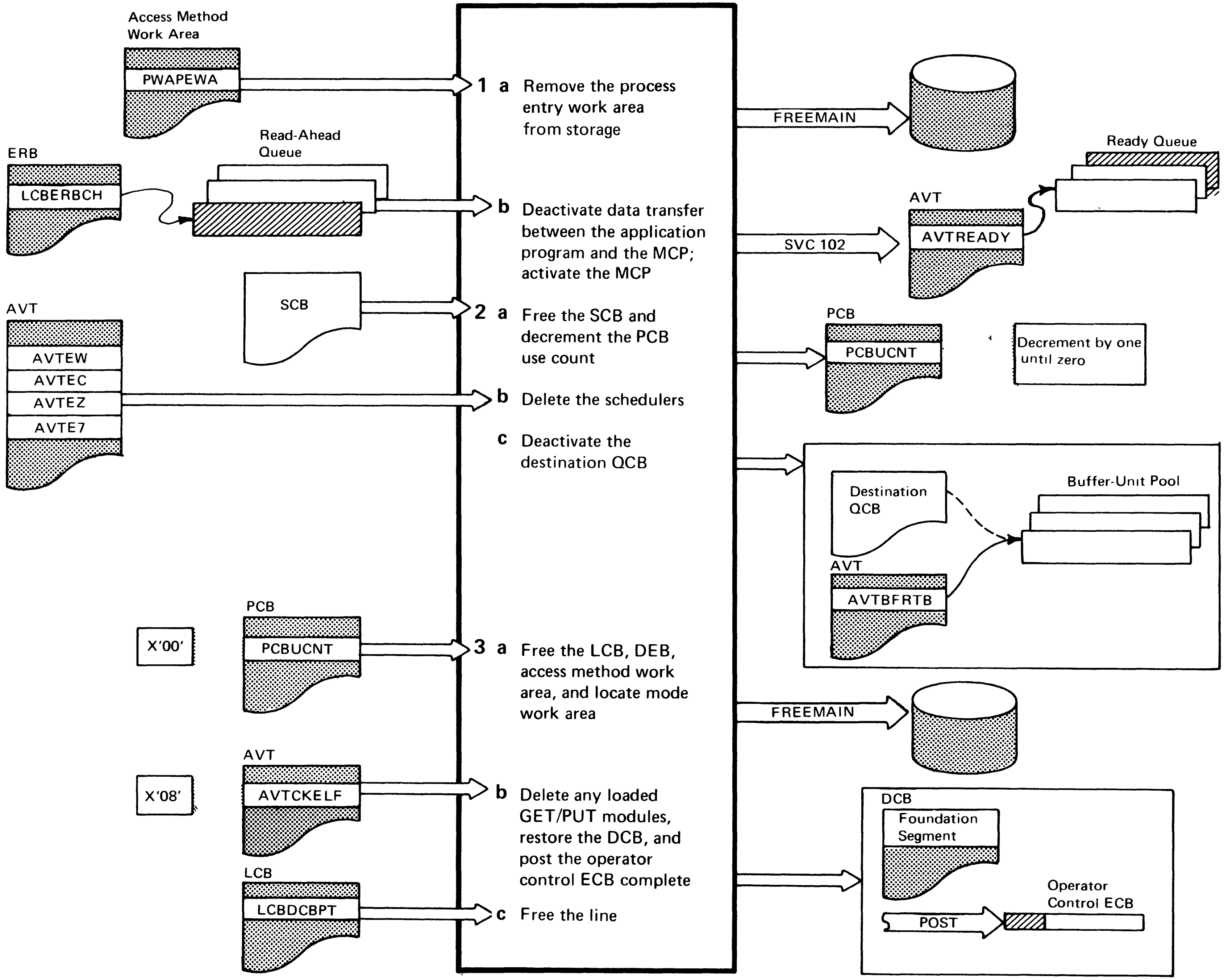
c Close the message queues data sets

d Delete TCAM from the system

Chart E1 MCP Termination—Description (2 of 2)

	Description	Routine	Register Usage
6	When the ECBs for operator control checkpoint, COMWRITE, and TOTE are posted complete, delete the appropriate task by placing zeros into its TCB address field in the AVT. ECB Address: TCB Address: AVTCKECA Checkpoint AVTCKTCB AVTOLECA On-Line Test AVTOLTCB AVTCWECA FE Common Write AVTCWTCB AVTOPECA Operator Control AVTOCTB	IEDQNA	
6a	Close the line group DCB. Perform OBR/SDR error recording on each line. Issue SVC 33 to purge I/O on each line. Issue a DISABLE command to disconnect the line. (This is not done for a Type III Adapter on a 2701 for IBM 2260 remote terminals.) Issue a FREEMAIN macro to free the LCBs. Clear the fields in the cross-reference table. When all the line group DCBs are closed, zero the AVT pointer to the DCB and free the main storage for the cross-reference table.	ERP modules IGG02035	
6b	Close the checkpoint data sets. Issue an EXCP macro to write the closedown checkpoint control record. Issue a FREEMAIN macro to free the work area. Issue a DELETE macro to free the Checkpoint Disk End appendage.	IGG02041	
6c	Close the message queues data sets. Issue a FREEMAIN macro for the DEB, CPBs, IOBs, buffers, and any main-storage data sets. Zero the AVT pointer to the DCB. Zero the AVT address pointer from the CVT.	IGG02030	
6d	Delete TCAM from the system (place zeros in CVT+240).		

Chart E2 Application Program Termination



1 a Remove the process entry work area from storage

b Deactivate data transfer between the application program and the MCP; activate the MCP

2 a Free the SCB and decrement the PCB use count

b Delete the schedulers

c Deactivate the destination QCB

3 a Free the LCB, DEB, access method work area, and locate mode work area

b Delete any loaded GET/PUT modules, restore the DCB, and post the operator control ECB complete

c Free the line

Chart E2 Application Program Termination—Description

	Description	Routine	Register Usage
1a	Issue a FREEMAIN macro to remove the process entry work area from main storage.		
1b	Tpost a special element that contains the address of the DCB process entry in the termname table to the open/close subtask. Issue a WAIT macro to allow the MCP to gain control.	IGG02046	
2a	Free the process entry work area and the SCB. Decrement the PCB use count (PCBUCNT) by one until it is zero.	IEDQEU	
2b	Delete the appropriate schedulers. The scheduler addresses are in the AVT: AVTEW—GET scheduler address AVTEC—PUT scheduler address AVTEZ—GET FIFO scheduler address AVTE7—Retrieve scheduler address		
2c	Turn off the "open" flag in the process entry.		
3a	Free the LCB if the PCB use count (PCBUCNT) field is zero. Free the access method work area (address at DEBTAMWA), the locate mode work area (address at DEBLCMWA), and then free the DEB.	IGG02046	
3b	If AVTCKELF=X'08', issue a DELETE macro for any GET/PUT modules acquired by a LOAD macro. Restore the DCB to its pre-open status. Set the "close" flag in the DCB. Post the operator control ECB in the AVT as complete if AVTCKELF is on. Scan all the TCAM LCBs to determine if any LCBs are locked to this application program.	IGG02046	
3c	Tpost the LCB to itself to free the line.	IGG02047	

(This page left blank intentionally)

Section 4: Program Organization

This section contains six sets of charts of information about the organization of the TCAM system.

1. Executable TCAM Modules Microfiche Directory

   This chart lists each executable TCAM module in alphabetical order. The entry for each module contains a general statement of the function of the module, a list of the entry points to the module, the external routines used by the module, the tables and work areas used by the module, and lists of other modules that activate and receive control from this module. Also, each entry shows which method of operation charts refer to this module and the system library in which this module is stored.

2. Non-executable TCAM Modules Microfiche Directory

   This chart lists each non-executable TCAM module in alphabetical order by DSECT name.

3. Macro Linkage Charts

   This section contains one chart for each TCAM macro. Each chart shows the macro, its input parameter list, the linkage among the TCAM modules that this macro effects, and the function performed by each linkage.

4. Operator Control Command Linkage Chart

   This chart contains an entry for each type of operator control command. Within each entry there is a list of the various command formats, which module a particular format activates, and the function performed by this module.

5. ERP Linkage Charts

   There are two charts in this section, one for start-stop line control and one for BSC. Each chart lists each type of I/O operation with the errors that can occur during that operation. For each error, the chart shows the ERP module activated and the conditions that cause that module to perform certain functions.

6. Flowcharts

   Flowcharts for IEDQFA, IEDQFA1, IEDQFA2, IEDQHM, IEDQHM1, IEDQHM2, IEDQKA, IEDQKB, IEDQKC, IEDQKD, IEDQKE, IGG019R0, IGG019Q2, IGG019Q3, IGG019Q4, and IGG019Q5 are included.

Executable TCAM Modules Microfiche Directory

The modules in the TCAM system have object module names that start with the letters IEDQ. The modules that interface with the operating system have an IGG prefix, the ERP modules have an IGE prefix, the nucleus resident modules have an IGC prefix (with the exception of IEDQATTN), and TCAM-TSO modules have an IEDA prefix.

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDAYA	TSQ Attention Routine	IEDAYA IEDAYA+12	Allows the user to delete lines and/or to interrupt the CPU task	IEDQTNT IGG019RB QTIP	AVT CVT LCB QCB RCB SCB STCB TJB TSB TSID Terminal Table Termname Table TS CVT	IEDAYX IEDAYF	IEDAYM IGG019RB		TELCMLIB
IEDAYB	TSO TIOC 3270 Edit Routine	IEDAYB	Edits output messages contained in TSO buffers and MSGEN messages contained in the SCB	IEDAYS QTIP SVC Routine	AVT LCB QCB TSID TCAM Buffer Prefix TSO Buffer Prefix	IEDAYE	IEDAYO IEDAYM		TELCMLIB
IEDAYC	TSO Carriage Subroutine	IEDAYC	Keeps track of, inserts, or deletes carriage control characters	IEDQTNT	AVT CVT DCB LCB QCB SCB TSB Buffer Prefix Terminal Table Termname Table TS CVT	IEDQUI (CARRIAGE)	IEDQUI (CARRIAGE)		TELCMLIB
IEDAYD	Time Sharing Destination Scheduler	IEDAYD	Performs the same functions as the Destination Scheduler when TSO is in the system.	IEDAYZ	AVT DCB LCB QCB STCB TSID	IGG019RB	IGG019RB		TELCMLIB
IEDAYE	TSO TIOC Edit Routine	IEDAYE	Inspects and edits TSO output and MSGGEN messages, except for messages directed to a 3270.	IEDAVE IEDAYS IEDQTNT QTIP	AVT CVT DCB LCB QCB SCB TSB TSID Buffer Prefix Terminal Table Termname Table TS CVT	IEDAYM IEDAYO	IEDAYM IEDAYO		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDAYF	TSO IOHALT Routine	IEDAYF	Gains control when either a LCB, buffer, or an ERB is tposted from Line End Appendage to effect an IOHALT.	IEDQHG IGG019RB OS IOHALT	AVT DCB DEB ERB LCB SCB TSID	IGG019RB	IGG019RB		TELCMLIB
IEDAYH	TSO Hang-up Routine	IEDAYH IEDAYH+12	Identifies line errors to the terminal user and disconnects the terminal	IEDQTNT IGG019RB QTIP	AVT CVT DCB LCB QCB SCB TJB TSB TIOCRPT Buffer Prefix TS CVT	IEDAYX IGG019RB	IGG019RB		TELCMLIB
IEDAYI	TSINPUT Routine	IEDAYI+2	Moves incoming data from a TCAM buffer into TSO buffers and places the TSO buffers on the TSO input buffer queue	IEDQTNT QTIP	AVT CVT LCB QCB SCB TIOCBUF TIOCRPT TSB TSI Buffer Prefix Terminal Table Termname Table TS CVT	IGG019RB	IGG019RB		TELCMLIB
IEDAYL	TCAM/TSO Logon Routine	IEDAYL	Connects the terminal user to the TSO subsystem for time sharing sessions, on the TCAM subsystem for message-switching applications.	IEDQTNT QTIP	AVT CVT DCB DCT DEB LCB QCB SCB STARTMH QCB TIOCRPT TJB TS CVT TSB UCB Buffer Prefix Terminal Table Termname Table	IEDQUI (LOGON)	LOGON exit address in owning or alternate MH		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDAYM	TSO Message Generation Routine	IEDAYM AYM000	Processes a message.	IEDAYE IEDQAE IEDQTNT	AVT CVT DCB ERB LCB QCB SCB TSB TSID Terminal Table Termname Table TS CVT	IEDAYS IEDQBD IGG019RB	IGG019RB		TELCMLIB
IEDAYO	TSOUTPUT Routine	IEDAYO IEDAYO02	Supervises movement of TSO data from TSO buffers into TCAM buffers.	IEDAYE IGG019RB QTIP	AVT CVT LCB QCB SCB TSB TSID TS CVT	IGG019RB	IGG019RB		TELCMLIB
IEDAYR	STARTMH Subtask for TCAM-TSO Mixed	IEDQAA01	Initializes a buffer before sending it through a TCAM or TSO message handler.	IEDQAL IEDQTNT IEDQUI (IEDQAW)	AVT DCB DEB ERB LCB QCB SCB SCT UCB Buffer Prefix Terminal Table Termname Table	IGG019RB	IGG019RB MH		TELCMLIB
IEDAYS	TSO Simulated Attention Routine	IEDAYS IEDAYS2 IEDAYS3	Handles simulated attention for TSO.	IEDQTNT QTIP	AVT CVT LCB QCB SCB TSB TSID Buffer Prefix Terminal Table Termname Table TS CVT	IEDQUI IEDAYE IGG019RB	IEDAYM IEDQUI IGG019RB		TELCMLIB
IEDAYT	TSO Abend Interface Routine	IEDAYT0 IEDAYT1 IEDAYT2	Informs TSO when TCAM abends, when an attached TCAM task abends, or when, in a mixed TSO/TCAM environment, the EXCP Driver abends.	IEDQEB TCABEND	AVT CVT DCB TSID STAE Work Area	OS STAE	ABEND/ STAE Routine		TELCMLIB
IEDAYX	TSO INMSG/ OUTMSG Linker	IEDAYX	Provides linkage to the TSO Attention and TSO Hang-up routines.	None	AVT DCB LCB	IEDQEB	IEDAYA IEDAYH		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDAYY	TSO Asynchronous Time Delay Removal Routine	IEDAYY	Removes QCBs from the time delay queue when a send or receive operation is to be initiated.	IEDQHG IGG019RB	AVT QCB TSID	TIOC Modules	IGG019RB		TELCMLIB
IEDAYZ	Time Sharing Scheduler	AYZ000 AYZ100 AYZ200 AYZ300 AYZ400 AYZ410 AYZ500 AYZ600 AYZ700	At AYZ000 and AYZ100, determines whether to initiate a read operation. At AYZ200, determines whether another poll operation is desired. At AYZ300, generates a Prepare channel program to free a line connected to a 2741. At AYZ400 and AYZ410, determines whether a Write Break channel command can be issued. At AYZ500, determines whether to initiate a send operation. At AYZ600, to return input buffers to the Buffer Return QCB. At AYZ700, to initiate sending the START MI prompt.	IEDQHG IEDQKA IEDQTNT IGG019RB OS EXCP OS IOHALT	AVT CVT DCB ERB LCB QCB SCB TSB TSID Buffer Prefix Invitation List Terminal Table Termname Table TS CVT	IEDAYD IEDQKA IGG019Q1 IGG019R0 IGG019R1 IGG019R3 IGG019R4	IEDAYD IEDAYM IEDQKA IGG019RB IGG019Q1 IGG019R1 IGG019R3 IGG019R4		TELCMLIB
IEDQAA	STARTMH Subtask	IEDQAA01	Performs the initialization functions necessary for MH to start processing a message.	IEDQAB IEDQFE30 IEDQTNT IEDQUI (IEDQAW)	AVT DCB LCB MH VCON Table QCB SCB Buffer Prefix Terminal Table Termname Table	IGG019RB	IGG019RB STARTMH macro expansion	C1-1.2 C1-3.2	TELCMLIB
IEDQAB	STARTMH Continuation	IEDQAB01	Translates and/or resumes header processing of subsequent buffers.	IEDQUI (IEDQAW) MH	AVT SCB	IEDQUI (first INHDR)	IEDQUI (INHDR or MH)		TELCMLIB
IEDQAC	Date and Time Provision Routine	IEDQAC01	Inserts the current time and/or date in the message header at the current scan pointer location.	IEDQAL IEDQAX OS TIME	AVT CVT Buffer Prefix	IEDQUI (DATETIME)	IEDQUI (DATETIME)		TELCMLIB
IEDQAD	Output Sequence Number Provision Routine	IEDQAD01	Inserts the output sequence number in a buffer of a message.	IEDQUI (IEDQAF)	AVT SCB Buffer Prefix	IEDQUI (SEQUENCE)	IEDQUI (SEQUENCE)		TELCMLIB
IEDQAE	Locate Option Field Address Routine	IEDQAE	Calculates the address of an option field from its index.	IEDQTNT	AVT LCB Option Table Terminal Table Termname Table	IEDQUI (MH macros and routines)	IEDQUI (MH macros and routines)	C1-1.4	TELCMLIB
IEDQAF	Insert Data Routine	IEDQAF01	Inserts data in a buffer or shifts data left within a buffer.	None	AVT LCB SCB Buffer Prefix	IEDQUI (MH macro expansions and routines)	IEDQUI (Calling macro or routine)		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQAG	Message Limit Routine	IEDQAG01	Limits the number of messages to or from a terminal during a single transmission sequence.	None	AVT DCB LCB SCB Buffer Prefix	MSGLIMIT macro expansion	MSGLIMIT macro expansion		TELCMLIB
IEDQAH	Input Sequence Number Insertion Routine	IEDQAH01	Verifies and updates an input sequence number specified by the user in a message	IEDQTNT	AVT LCB SCB Buffer Prefix Terminal Table Termname Table	IEDQUI (SEQUENCE)	IEDQUI (SEQUENCE)		TELCMLIB
IEDQAI	Skip Forward and Scan Routine	IEDQAI01	Moves the scan pointer forward in the message header a specified number of bytes, or finds and returns to the caller the next field beyond the scan pointer.	None	AVT SCB Buffer Prefix	IEDQUI (MH macro or routine)	IEDQUI (calling macro or routine)	C1-1.4	TELCMLIB
IEDQAJ	Skip to Character Set Routine	IEDQAJ01	Advances the scan pointer to the end of a specified character string in a message header	IEDQAX	AVT SCB Buffer Prefix	IEDQUI (FORWARD, SETSCAN, IEDQAN, or IEDQAP)	IEDQUI (calling routine)		TELCMLIB
IEDQAK	Line Control Insertion Routine	IEDQAK01	Checks line control characters and inserts them in a message that is ready to be sent.	IEDQAL IEDQTNT IEDQUI (IEDQAF, IEDQAO)	AVT DCB LCB SCB SCT Buffer Prefix	IEDQUI (OUTMSG, OUTEND)	IEDQA4 (next macro)		TELCMLIB
IEDQAL	Compare at Offset Routine	IEDQAL01	Finds and compares the next field in a buffer to a character string.	None	AVT Buffer Prefix	MH routine	Calling routine		TELCMLIB
IEDQAM	Origin Routine	IEDQAM01	Verifies or initializes the origin of a message.	IEDQTNT	AVT LCB QCB Buffer Prefix Terminal Table Termname Table	IEDQUI (ORIGIN)	IEDQUI (ORIGIN)		TELCMLIB
IEDQAN	Multiple Insert/ Remove Routine	IEDQAN01	Inserts, deletes, and replaces data at locations specified by character strings in the buffer.	IEDQAL IEDQAX IEDQTNT IEDQUI (IEDQAF, IEDQAO)	AVT LCB SCB Buffer Prefix Translation Table	IEDQUI (MSGEDIT)	IEDQUI (MSGEDIT)		TELCMLIB
IEDQAO	Unit Request Interface Routine	IEDQAO01	Obtains a buffer unit requested by one of the insert routines and adds the unit to the buffer currently being processed.	None	AVT LCB SCB Buffer Prefix	IEDQUI (IEDQAN, IEDQAP, IEDQA2, IEDQA8, IEDQBO)	IEDQUI (Calling routine)		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQAP	Remove at Offset Routine	IEDQAP01	Removes data from a single specified location in a buffer and, optionally, replaces that data with new data.	IEDQAX IEDQUI (IEDQAF, IEDQAO)	AVT LCB SCB Buffer Prefix	IEDQUI (MSGEDIT)	IEDQUI (MSGEDIT)		TELCMLIB
IEDQAQ	Operator Control Interface Routine	IEDQAQ01	Tests for operator control characters and conditionally tposts the buffer to Operator Control for processing.	IEDQTNT	AVT LCB SCB Buffer Prefix Terminal Table Termname Table	IEDQUI (CODE)	IEDQUI (CODE) IGG019RB		TELCMLIB
IEDQAR	Cancel Message Routine		IEDQAR Cancels a message by setting a flag in the buffer prefix	IEDQTNT	AVT LCB SCB Buffer Prefix Terminal Table Termname Table	IEDQBD	IGG019RB		TELCMLIB
IEDQAS	Hold/Release Terminal Routine	IEDQAS IEDQAS01 GETCPB	Holds a terminal. Releases a terminal that was being held.	IEDQHG IEDQHM IEDQTNT IGG019RB	AVT CPB DCB DEB DRQ QCB QCB Extension SCB STCB UCB Buffer Prefix Terminal Table Termname Table	IEDQBD IEDQCQ IEDQUI (HOLD) IGG019RB	IEDQFC IGG019RB		TELCMLIB
IEDQAT	Create an Error Message Routine	IEDQAT01 STCBAT+2	Builds an error message in a buffer and tposts that buffer to its destination.	IEDQUI (IEDQAF, IEDQAO)	AVT LCB SCB Buffer Prefix	IEDQAZ IGG019RB	IGG019RB		TELCMLIB
IEDQATTN	Attention Routine	IEDQATTN	Determines whether TCAM is running in the system when an attention interrupt is presented by a 2848 or a 3270 control unit. Activates IGG019R5 when TCAM is running in the system.	None	AVT	IOS	IGG019R5 IOS		NUCLEUS
IEDQAU	Cutoff Message Transmission Routine and Subtask	IEDQAU CUTFFQCB +12	Tests the cutoff count. Cuts off the transmission of a message being received after the receipt of a user-specified number of bytes or on detection of identical characters in the buffer.	OS EXCP	AVT DCB DEB LCB QCB SCB UCB Buffer Prefix	IEDQUI (CUTOFF) IGG019RB	IEDQUI (CUTOFF) IGG019RB		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQAV	Look-up Terminal Entry Routine	IEDQAV01	Assigns a buffer to its destination.	IEDQTNT	AVT LCB SCB Buffer Prefix Terminal Table Termname Table	IEDQUI (FORWARD, IEDQAZ, IEDQA5)	IEDQUI (FORWARD, IEDQAZ, IEDQA5)	C1-1.4	TELCMLIB
IEDQAW	Translate Buffer Routine	IEDQAW01	Translates the data in a buffer.	IEDQA3	AVT DCB LCB SCB Buffer Prefix Translation Table	IEDQUI (CODE, IEDQAA, IEDQAB)	IEDQUI (CODE, IEDQAA, IEDQAB)		TELCMLIB
IEDQAX	Buffer Scan Routine	IEDQAX01	Scans the buffer for a specified character string.	None	AVT Buffer Prefix	IEDQAC IEDQAI IEDQAJ IEDQA4	IEDQAC IEDQAI IEDQAJ IEDQA4		TELCMLIB
IEDQAY	Screen Routine	IEDQAY01	Initializes for a screen command modification operation on the message destination.	IEDQTNT	AVT DEB LCB SCB UCB Buffer Prefix Terminal Table	IEDQUI (SCREEN)	IEDQUI (SCREEN) IEDQAT		TELCMLIB
IEDQAZ	Redirect a Message Routine	IEDQAZ01	Redirects a message to the destination specified by the user.	IEDQAV IEDQUI (IEDQAE, IEDQA1)	AVT SCB Buffer Prefix	IEDQBD	IGG019RB		TELCMLIB
IEDQA0	Skip Backward Routine	IEDQA001	Moves the scan pointer backward a specified number of bytes in the header of a message.	None	AVT Buffer Prefix	IEDQUI (SETSCAN)	IEDQUI (SETSCAN)		TELCMLIB
IEDQA1	Binary Search Routine	IEDQA101	Searches a table that is arranged in collating sequence.	None	AVT Termname Table	IEDQUI (Any TCAM routine)	IEDQUI (Calling routine)	A4 C1-1.4	TELCMLIB
IEDQA2	Insert at Offset Routine	IEDQA201	Inserts data in a message buffer at a specific location.	IEDQUI (IEDQAF, IEDQAO)	AVT LCB SCB Buffer Prefix	IEDQUI (MSGEDIT)	IEDQUI (MSGEDIT)		TELCMLIB
IEDQA3	Dynamic Translation Routine	IEDQA3	Determines which translation table to use for all input and output translation for a specific terminal. Retrieves a translation table address from the appropriate option field.	IEDQUI (IEDQAE)	AVT DCB LCB SCB	IEDQAW	IEDQAW		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQA4	Incoming/Outgoing Message Delimiter Routine	IEDQA401	During incoming message processing, tposts the buffer to the proper QCB. During outgoing processing, passes the buffer to either the Buffer Association routine in the Buffer Management module or to the Transparent CCW Building routine	IEDQAX IEDQTNT IGG019RB	AVT DCB DEB LCB QCB SCB SCT UCB Buffer Prefix Process Entry Work Area Terminal Table Termname Table	IEDQUI (INEND, INMSG, OUTEND, or OUTMSG)	IEDQGD IEDQGT IGG019RB	C1-1 3 C1-3.3	TELCMLIB
IEDQA5	Forward Routine	IEDQA501	Determines the destination to which a message is to be sent.	IEDQAV IEDQB3 IEDQUI (IEDQAE, IEDQAI, IEDQA1)	AVT SCB Buffer Prefix	IEDQUI (FORWARD, IEDQBA)	IEDQUI (FORWARD, IEDQBA)	C1-1.4	TELCMLIB
IEDQA6	Line Control Initialization Routine	IEDQA601	Initializes fields in the SCB to indicate the intervals between the line control characters to be inserted.	IEDQTNT	AVT SCB Buffer Prefix Terminal Table Termname Table	IEDQUI (MSGFORM)	IEDQUI (MSGFORM)		TELCMLIB
IEDQA7	Counter Routine	IEDQA701	Counts either the complete messages or message segments that are processed by the MH subgroup in which a COUNTER macro appears.	IEDQUI (IEDQAE)	AVT Buffer Prefix Option Table	COUNTER macro expansion	COUNTER macro expansion		TELCMLIB
IEDQA8	Multiple Insert at Offset Routine	IEDQA801	Inserts a character string at specified intervals in a message.	IEDQUI (IEDQAE, IEDQAF, IEDQAO)	AVT LCB SCB Buffer Prefix Option Table	IEDQUI (MSGEDIT)	IEDQUI (MSGEDIT)		TELCMLIB
IEDQA9	Redial Routine	IEDQA9	Causes the CPU to try again to initiate contact with a switched station. This routine goes to the Time Delay routine to place a QCB on the time delay queue for an interval of time specified by the user in the RETRY macro.	IEDQTNT IEDQHG01 IGG019RB	AVT LCB QCB Terminal Table	IEDQBD	IGG019RB		TELCMLIB
IEDQBA	Multiple Routing Subtask	IEDQBA01	Queues a message for additional destinations.	IEDQUI (IEDQA5)	AVT LCB SCB Buffer Prefix	IGG019RB	IGG019RB		TELCMLIB
IEDQBB	Checkpoint Request Routine	IEDQBB	Indicates that a checkpoint request has been made.	None	AVT LCB SCB	IEDQBD IEDQUI (CHECKPT)	IGG019RB IEDQUI (CHECKPT)		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQBC	Distribution List Subtask	IEDQBC	Directs a message to each of the destinations specified in the distribution list to which the message was routed.	IEDQTNT	AVT LCB QCB SCB Buffer Prefix Terminal Table Termname Table	IGG019RB	IGG019RB		TELCMLIB
IEDQBD	Buffer Disposition Subtask	IEDQBD01 IEDQBD02	Controls MH processing when the last segment of a message has been sent or received.	IEDQTNT	AVT LCB QCB SCB Buffer Prefix Terminal Table Termname Table	IGG019RB	IEDQNX IGG019RB	C1-1.3 C1-3.3 D1-2	TELCMLIB
IEDQBE	Lock Routine	IEDQBE	Locks the connection between the currently connected terminal and its process entry destination.	IEDQTNT	AVT LCB QCB SCB Terminal Table Termname Table	IEDQUI (LOCK)	IEDQUI (LOCK)		TELCMLIB
IEDQBF	Unlock Routine	IEDQBF	Unlocks the currently connected terminal.	None	AVT SCB	IEDQUI (UNLOCK)	IEDQUI (UNLOCK)		TELCMLIB
IEDQBG	Cascade List Subtask	IEDQBG	Directs a message to the appropriate entry in the cascade list to which the message was routed.	IEDQTNT	AVT DCB LCB QCB SCB Buffer Prefix Terminal Table Termname Table	IGG019RB	IGG019RB		TELCMLIB
IEDQBH	Concentrator Buffer Disposition Subtask	IEDQBH IEDQBH01 IEDQBH02	Locates the proper QCB-SCB and interfaces with IEDQBD to control the execution of OUTMSG for a concentrator MH.	IGG019RB	AVT DRQ LCB QCB QCB Extension SCB Terminal Table	IGG019RB	IGG019RB	C1-3.3	TELCMLIB
IEDQBL	Message Generation Routine	IEDQBL	Directs a user-provided message to a specified destination.	IEDQTNT	AVT DCB LCB SCB Terminal Table Termname Table	IEDQBD	IGG019RB		TELCMLIB
IEDQBM	Origin Routine for System with Concentrated Message Handling Support	IEDQBM01	Performs source determination on messages entered by both terminals not defined in a concentrator network and terminals defined as attached to a concentrator	IEDQTNT IEDQA1 IEDQUI (IEDQAI)	AVT LCB QCB QCB Extension SCB Buffer Prefix Device ID Table Terminal Table	IEDQUI (ORIGIN)	IEDQUI		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQBN	Data Attach Routine	IEDQBN01	Combines data that could not be processed from the previous buffer with the current buffer.	IEDQUI (IEDQAF, IEDQAO)	AVT LCB Buffer Prefix	IEDQUI (MSGEDIT, MSGFORM, SETEOM)	IEDQUI		TELCMLIB
IEDQBO	SETEOM Routine	IEDQBO01	Blocks or deblocks physical transmissions into logical messages.	IEDQAX IEDQUI (IEDQAE, IEDQAF, IEDQAO, IEDQBN, IEDQBR)	AVT LCB SCB Buffer Prefix	IEDQUI (SETEOM)	IEDQUI (MH) IGG019RB		TELCMLIB
IEDQBP	TGOTO Routine	IEDQBP01	Passes control from one MH to a second level MH for the handling of LMD messages entered by terminals attached to a concentrator.	IEDQTNT IEDQUI (IEDQAE)	AVT LCB Line SCB LMD SCB Buffer Prefix Terminal Table	IEDQUI (TGOTO)	IGG019RB IEDQUI		TELCMLIB
IEDQBQ	QACTION Routine	IEDQBQ IEDQBQ02	Forces INMSG execution. Forces OUTMSG execution. Performs a temporary hold and release. Sets the user error bit.	IEDQA1 IEDQBD IEDQTNT IEDQUI User Routine IGG019Q9 IGG019RB	AVT DCB DEB DRQ LCB QCB QCB Extension SCB DVC ID Table Buffer Prefix Terminal Table User Built Parm List	IEDQUI (QACTION) IEDQBD IGG019RB	IEDQUI IGG019RB		TELCMLIB
IEDQBT	EOB/ETB Handling Subtask	IEDQBT	Performs EOT/ETB handling on a buffer.	IEDQUI (IEDQAE)	AVT LCB SCB Buffer Prefix	IGG019RB	IGG019RB	C1-1.2 C1-3.2	TELCMLIB
IEDQBU	CANCELBK Routine	IEDQBU	Performs mid-batch recovery when the LEVEL=BLK operand is coded on a CANCELMG macro.	IEDQTNT	AVT LCB SCB Buffer Prefix	IEDQBD	IGG019RB		TELCMLIB
IEDQBV	COMMBUF Routine	IEDQBV	Moves the current buffer to the next available data area and inserts COMMBUF STCB into STCB chain of the appropriate LCBs.	IEDQTNT	AVT CMB Common Buffer Data, Area Prefix DCB LCB QCB STCB Terminal Table	IEDQUI (COMMBUF)	IGG019RB IEDQUI		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQBX	Log Segment Routine	IEDQBX	Writes (logs) a message segment onto the logging medium specified by the user in a BSAM DCB.	OS BSAM CHECK OS BSAM WRITE OS GETMAIN	AVT DCB	IEDQUI (LOG)	IEDQUI (LOG)		TELCMLIB
IEDQBY	Log Message Routine	IEDQBY	Directs a message header to the destination specified as the log for messages.	None	SCB Buffer Prefix	IEDQBD	IGG019RB		TELCMLIB
IEDQBZ	Log Scheduler	IEDQBZ	Schedules the logging of messages.	IGG019RB OS BSAM CHECK OS BSAM WRITE OS GETMAIN	AVT DCB LCB QCB	IGG019RB	IGG019RB		TELCMLIB
IEDQB1	MCOUNT Routine	IEDQB1	Returns the number of complete messages on an application program queue.	None	AVT CVT DCB DEB QCB Process Entry Work Area Terminal Table	MCOUNT	MCOUNT		TELCMLIB
IEDQB2	TPDATE Routine	IEDQB2	Indicates that record delimiters are to be deleted from data going to an application program.	None	CVT DCB DEB Access Method Work Area	TPDATE macro expansion	TPDATE macro expansion		TELCMLIB
IEDQB3	DATETIME Insertion Routine for Processing Programs	IEDQB3	Inserts the date and/or time in a buffer for an application program.	None	AVT LCB SCB Buffer Prefix	IEDQA5	IEDQA5		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQB4	Slow Poll Routine	IEDQB4	Suspends further polling on a line when an error occurs.	IEDQBD IGG019RB IEDQHG	AVT LCB SCB	IEDQBD	IGG019RB		TELCMLIB
IEDQCA	Resident Operator Control Module	IEDQCA TRMOFLOC DCBLOCAT ALLOCBUF LCBSETUP OSTCB	Defines the operator control AVT, sets up a wait list and issues SVC 104. Contains subroutines linked to by various operator control transient modules. OSTCB is entered directly by the TCAM dispatcher when the summy LCB is posted to itself to be freed.	IGC0010D OS WAIT Calling Routine	AVT Op Ctl AVT OPCE	OS Task Management Transient Operator Control Routines TCAM dispatcher OS Task Management Transient Operator Control Routines		C3-1	LINKLIB
IEDQEB	AQCTL SVC 102 Routine	IGC102	Moves data across partition boundaries. Posts ECBs in other tasks. Tposts elements to the TCAM disabled ready queue. Flags TCBs for application programs as either eligible or not eligible for swapping or rollout.	IKJTSI00 OS POST OS STATUS TESTDSP	AVT CVT TS CVT	SVC 102 from any TCAM routine·	The calling routine	B2 C2-1 C2-2 C3-1 D1-1 D1-3 D1-4	NUCLEUS
IEDQEC	Put Scheduler	IEDQEC	Removes data from an application program work area and places it in MCP buffers.	IEDQEB IGG019RB	AVT DCB DEB LCB PCB QCB SCB Access Method Work Area Process Entry Work Area Terminal Table Termname Table	IGG019RB	IGG019RB	C2-2	LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQES	Retrieve Service Routine	IEDQES	Provides TCAM support for message retrieval from a QTAM application program.	IEDQEB IEDQUI (IEDQA1) OS WAIT	AVT CVT DEB LCB PCB QCB SCB TCB Access Method Work Area Buffer Prefix Terminal Table Termname Table	RETRIEVE macro expansion	RETRIEVE macro expansion		TELCMLIB
IEDQET	Operator Control/ Application Program Interface Routine	IEDQET	Provides the interface through which operator control commands can be issued from an application program.	IEDQEB IEDQE6 OS WAIT	AVT CIB CVT PCB QCB	An Operator Control command in an application program	Application Program	C3-1	LINKLIB
IEDQEU	Open/Close Subtask	IEDQEU	If an open procedure, allocates main storage for an application program in the MCP and loads the appropriate schedulers. If a close procedure, deallocates the main storage for the application program in the MCP and deletes the appropriate schedulers.	IEDQEB OS GETMAIN OS DELETE FREEMAIN OS LOAD	PCB QCB SCB Process Entry Work Area Termname Table	IGG019RB	IGG019RB	A4 E2	TELCMLIB
IEDQEW	Get Scheduler	IEDQEW	Reads messages from the message queues data set in anticipation of a GET command from an application program.	IEDQEB	AVT DCB ECB QCB Process Entry Work Area Termname Table	IGG019RB	IGG019RB	C2-1	LINKLIB
IEDQEZ	Get Scheduler FIFO Routine	IEDQEZ	Recognizes a retrieve element and tposts it to the Destination QCB for an application program.	None	QCB	IGG019RB	IGG019RB		LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQE1	TCOPY Service Routine	IEDQE1	Copies a terminal entry into an application program work area.	IEDQUI (IEDQA1)	AVT CVT DCB TCB Access Method Work Area Application Program Work Area Terminal Table Termname Table	TCOPY macro expansion	TCOPY macro expansion	C3-2	LINKLIB
IEDQE2	QCOPY Service Routine	IEDQE2	Copies a QCB into an application program work area.	IEDQUI (IEDQA1)	AVT CVT DEB QCB TCB Application Program Work Area Terminal Table Termname Table	QCOPY macro expansion	QCOPY macro expansion	C3-2	LINKLIB
IEDQE3	TCHNG Service Routine	IEDQE3	Updates the contents of a terminal entry by copying an altered entry from an application program work area into the terminal table.	IEDQEB IEDQE6 IEDQNB IEDQUI (IEDQA1)	AVT CVT DEB TCB Access Method Work Area Application Program Work Area Terminal Table Termname Table	TCHNG macro expansion	TCHNG macro expansion	C3-2	LINKLIB
IEDQE4	ICOPY Service Routine	IEDQE4	Copies the invitation list for a line group into an application program work area.	None	AVT CVT DCB DEB TCB TIOT Application Program Work Area Invitation List	ICOPY macro expansion	ICOPY macro expansion	C3-2	LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQE6	Password Scramble Routine	IEDQE6	Scrambles the characters of an input password so that it can be compared to an already scrambled password in the AVT.	None	None	IEDQET IEDQE3	IEDQET IEDQE3	A1	LINKLIB
IEDQE7	Retrieve Scheduler	IEDQE7	Retrieves a buffer from a message queues data set for a QTAM-compatible application program.	IEDQEB IGG019RB FREEMAIN OS GETMAIN	AVT LCB PCB QCB SCB Terminal Table	IGG019RB	IGG019RB		LINKLIB
IEDQE8	Binary Search Routine for Processing Programs	IEDQE801	Searches a table that is arranged in collating sequence.	None	AVT Termname Table	Application Program routines	Calling routines	A4	TELCMLIB
IEDQFA	CPB Initialization	IEDQFA IEDQFQ	Initializes CPBs to read or write buffer units to or from disk. Frees serviced messages from the main-storage message queues data set. Handles CPBs after completion of the disk operation.	IEDQHM IEDQTNT IGG019RB	AVT CPB DCB LCB QCB SCB Buffer Prefix Disk Data Area Terminal Table Termname Table	IGG019RB	IGG019RB IGG019RC IGG019RP	C1-2 C1-3.1 C2-1	TELCMLIB
IEDQFA1	CPB Initialization Main-Storage-Only Queuing	IEDQFA1 IEDQFQ	Obtains full buffers from the main-storage message queues data set. Flags messages serviced in the main-storage message queues data set. Builds a buffer for the main-storage message queues data set.	IEDQHM1 IGG019RB	AVT CPB DCB LCB QCB SCB Buffer Prefix Disk Data Area Terminal Table	IGG019RB	IGG019RB IGG019RC IGG019RP	C2-1	TELCMLIB
IEDQFA2	CPB Initialization Disk-Only Queuing	IEDQFA2 IEDQFQ	Initializes CPBs to read or write buffer units from or to disk. Handles CPBs after the disk operation completes.	IEDQHM2 IEDQTNT IGG019RB	AVT CPB DCB LCB QCB SCB Buffer Prefix Disk Data Area Terminal Table Termname Table	IGG019RB	IGG019RB IGG019RC IGG019RP	C1-2 C1-3.1 C2-1	TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQGA	Buffer Management Module	IEDQGA IEDQGB IEDQGD	Handles buffer requests by either assigning the requested buffers or queuing the request to be satisfied later. Returns buffers to the buffer unit pool. Builds CCWs for data transfer in each unit of a buffer. For non-concentrators, links buffers into the idles loop.	IGG019RB OS EXCP	AVT DCB LCB MH VCON Table QCB Buffer Prefix	IGG019RB IEDQA4	IGG019RB IEDQGH	C1-1.1 C1-3.3	TELCMLIB
IEDQGH	CTBFORM Routine	IEDQGH IEDQGH01 IEDQGH01+4	Inserts DVCIDs, CTB ending characters, and option field data. Determines CTB end for byte count terminals. Marks messages serviced for queues specifying multi messages from the queue. Determines concentrated end of message. Links buffers into the idles loop. Tposts the ERB to the Concentrator Send Scheduler to continue servicing the DRQ.	IEDQUI (IEDQAF, IEDQAO, IEDQAE) IGG019RB	AVT DCB DRQ DVCID LCB QCB QCB Extension SCB SCT Buffer Prefix Terminal Table	IGG019RB IEDQGD IEDQUI (CTBFORM) IEDQUI (OUTMSG)	IEDQUI IGG019RB	C1-3.3	TELCMLIB
IEDQGP	MHGET/MHPUT Routine	IEDQGP	Moves data between current buffer and user work area in main storage.	None	AVT DCB LCB PRF	MHGET MHPUT	MHGET MHPUT		TELCMLIB
IEDQGQ	Queue Reset Executor	IEDQGQ	Recalls message header from a disk message queue and schedules the rewriting of DATFEFO to reflect a message not serviced.	IEDQEB IEDQFA IEDQGB	AVT Buffer prefix LCB QCB SCB Terminal Table	IGG019RB	IGG019RB		TELCMLIB
IEDQGR	GRESET Service	IEDQGR	Checks validity of user input for resetting a FEFO pointer and builds the parameter list for IEDQGQ.	IEDQEB IEDQE8 IEDQTNT OS WAIT	AVT Access method WORK AREA	QRESET Macro expansion	QRESET Macro expans on		LINKLIB
IEDQGT	Transparent Transmission CCW Building Routine	IEDQGT	Builds in each buffer unit the CCWs that are necessary to send transparent data.	None	AVT CCW DCB LCB SCB Buffer Prefix	IEDQA4	IGG019RB	C1-3.3	TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQHG	Time Delay Subtask	IEDQHG IEDQHG01 IEDQHG02 IEDQHG03 TIMEEXIT	Places a time delay request element on the time delay QCB Places a time delay request element on the time delay queue. Removes a time delay request element from the time delay queue.	IEDQEB IGG019RB OS STIMER OS TIME	AVT QCB	Attached task TCAM subtask IGG019RB OS Interrupt Routine	Attached task TCAM subtask IGG019RB OS Interrupt Routine	D1-1	TELCMLIB
IEDQHI	System Delay Subtask	IEDQHI	Causes the system to cease line activity for the number of seconds specified on the INTVAL= operand of the INTRO macro.	IEDQHG IGG019RB OS IOHALT OS WTO	AVT DCB DEB IOB LCB QCB	IGG019RB	IGG019RB	A-1	LINKLIB
IEDQHK	Stop Line I/O Subtask	IEDQHK	Stops line activity for a line or line group. Issues a Prepare HIO command sequence to check for an attention signal from a non-TSO 2741 or 1050 terminal.	OS EXCP OS IOHALT	AVT CCW DCB DEB IOB LCB	IGG019RB IGG019R0 IEDQCV	IGG019RB IGG019R0	E1	TELCMLIB
IEDQHM	Destination Scheduler	IEDQHM IEDQHM02 IEDQHM03	Assigns a buffer or a unit to a location in the message queues data set	IEDQTNT IGG019RB Scheduler Subroutine	AVT DCB LCB QCB SCB Buffer Prefix Disk Data Area Terminal Table Termname Table	IEDQAS IEDQFA IGG019RB IGG019RP IGG019R6	IGG016RB IGG019RP	C1-2 C2-1	TELCMLIB
IEDQHM1	Destination Scheduler-Main-Storage-Only Queuing	IEDQHM1 IEDQHM02 IEDQHM03	Assigns a buffer or a unit to a location in the main-storage message queues data set.	IEDQTNT Scheduler Subroutine	AVT DCB LCB QCB SCB Buffer Prefix Disk Data Area Terminal Table Termname Table	IEDQAS IEDQFA1 IGG019RB IGG019RP IGG019R6	IGG019RB IGG019RP	C2-1	TELCMLIB
IEDQHM2	Destination Scheduler-Disk-Only Queuing	IEDQHM2 IEDQHM02 IEDQHM03	Assigns a buffer or a unit to a location in the disk message queues data set.	IEDQTNT IGG019RB Scheduler Subroutine	AVT DCB LCB QCB SCB Buffer Prefix Disk Data Area Terminal Table Termname Table	IEDQAS IEDQFA2 IGG019RB IGG019RP IGG019R6	IGG019RB IGG019RP	C1-2 C2-1	TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQKA	Activate-I/O Generator Subtask	IEDQKA IEDQKA02	Builds channel programs for initial contact, continue, and reset sequences.	IEDQTNT OS EXCP	AVT CCW DCB LCB SCB Buffer Prefix Terminal Table Termname Table	IGG019RB IGG019R0	IGG019RB IGG019R0	C1-1.1 C1-3.1	TELCMLIB
IEDQKB	Activate-I/O Generator for BSC Lines	IEDQKB IEDQKA02	Builds channel programs for initial contact, continue, and reset sequences on BSC lines only.	IEDQTNT OS EXCP	AVT CCW DCB LCB SCB Buffer Prefix Terminal Table Termname Table	IGG019RB IGG019R0	IGG019RB IGG019R0		TELCMLIB
IEDQKC	Activate-I/O Generator for Start/Stop Lines	IEDQKC IEDQKA02	Builds channel programs for initial contact, continue, and reset sequences on start/stop lines only.	IEDQTNT OS EXCP	AVT CCW DCB LCB SCB Buffer Prefix Terminal Table	IGG019RB IGG019R0	IGG019RB IGG019R0		TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQKD	Activate-I/O Generator Subtask for Leased and Start/Stop Lines and No TSO	IEDQKD IEDQKA02	Builds channel programs for initial contacts, continue, and reset sequences for leased and start/stop lines when there is no TSO interface necessary.	IEDQTNT OS EXCP	AVT CCW DCB LCB SCB Buffer Prefix Terminal Table	IGG019RB IGG019R0	IGG019RB IGG019R0		TELCMLIB
IEDQKE	Activate-I/O Generator Subtask for a QTAM-Compatible System	IEDQKE IEDQKA02	Builds channel programs for initial contact, continue, and reset sequences for only those devices that QTAM supports.	IEDQTNT OS EXCP	AVT CCW DCB LCB SCB Buffer Prefix Terminal Table	IGG019RB IGG019R0	IGG019RB IGG019R0		TELCMLIB
IEDQNA	Resident Closedown Completion Routine	IEDQNA IEDQNA3	Activates the nonresident closedown completion routine. Determines whether a TCAM attached task has terminated abnormally.	IEDQNA2	AVT TCB	IGG019RB OS Termination Routine	User code following READY OS Termination Routine	E1	TELCMLIB
IEDQNA2	Nonresident Closedown Completion Routine	IEDQNA2	Closes down the MCP and any TCAM attached tasks.	IGG019RB OS DETACH OS POST OS WAIT OS WTO	AVT DCB DEB IOB TCB	IEDQNA	IEDQNA		LINKLIB
IEDQNB	Application Program/ Checkpoint Interface Routine	IEDQNB IEDQNB02 IEDQNB05	Builds a checkpoint request element and tposts it to the ready queue in the MCP when an application program issues a TCAM macro that changes the MCP environment.	IEDQEB IEDQTNT OS WAIT	AVT DCB DEB LCB PCB Access Method Work Area Checkpoint Work Area Terminal Table Termname Table	CKREQ macro expansion CLOSE macro expansion IEDQE3 OPEN macro expansion TCHNG macro expansion	CKREQ macro expansion CLOSE macro expansion IEDQE3 OPEN macro expansion TCHNG macro expansion	D1-2 D1-3 D1-4	LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQND	Ready Routine	IEDQND	If indicated, reads and processes all incident checkpoint records that are more recent than the environment record. Attaches the Checkpoint Executor. Loads IEDQNX and/or IEDQHI, if necessary. Attaches On-Line Test, if indicated.	IECPCNVT IECOSCR1 IEDQTNT OS ATTACH OS EXCP FREEMAIN OS GETMAIN OS LOAD OS POST OS WAIT OS WTO	AVT Operator Control AVT Terminal Table Termname Table	READY macro expansion	READY macro expansion	A3 D1-1	LINKLIB
IEDQNF	Checkpoint Executor	IEDQNF	Determines the action required by the checkpoint task and which module is required to do the work.	IEDQNG IEDQNH IEDQNJ IEDQNK IEDQNM IEDQNO IEDQNP IEDQNQ IEDQNR IEDQNS OS DELETE OS LOAD OS WAIT	AVT CVT Checkpoint Work Area	OS Task Management	OS Task Management	D1-1 E1	LINKLIB
IEDQNG	Build Incident Record for MH Routine	IEDQNG	Builds an incident disk record when the checkpoint request element is an LCB from an MH macro.	IEDQTNT OS GETMAIN	AVT Checkpoint Work Area Option Table Terminal Table Termname Table	IEDQNF	IEDQNF	D1	LINKLIB
IEDQNH	Build Incident Record for TCHNG Routine	IEDQNH	Builds an incident checkpoint disk record when the checkpoint request element is from a TCHNG macro in an application program.	IEDQTNT OS GETMAIN	AVT Checkpoint Work Area Option Table Terminal Table Termname Table	IEDQNF	IEDQNF	D1-4	LINKLIB
IEDQNJ	Incident Checkpoint for Operator Control Routine	IEDQNJ	Builds an incident checkpoint disk record when the checkpoint request element is from an operator control command.	OS GETMAIN	AVT Checkpoint Work Area Operator Control AVT	IEDQNF	IEDQNF	D1-3	LINKLIB
IEDQNK	Environment Checkpoint Routine	IEDQNK	Builds environment checkpoint records for disk.	OS GETMAIN	AVT QCB Checkpoint Work Area Invitation List Option Table Terminal Table Termname Table	IEDQNF	IEDQNF	D1-1 E1	LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQNM	Build CKREQ Disk Record Routine	IEDQNM	Builds a CKREQ checkpoint disk record for each open destination QCB that is associated with the application program that has issued the CKREQ macro.	IEDQTNT OS GETMAIN	AVT DEB QCB Checkpoint Work Area Option Table Terminal Table Termname Table	IEDQNF	IEDQNF	D1-4	LINKLIB
IEDQNO	Checkpoint Queue Manager	IEDQNO	Puts disk records on the checkpoint I/O queue and updates the last request element for which a disk record was built.	FREEMAIN	AVT Checkpoint Work Area	IEDQNF	IEDQNF		LINKLIB
IEDQNP	Checkpoint Disk I/O Routine	IEDQNP	Locates the next disk record to be written, determines the proper TTR, and issues an EXCP.	IECPCNVT IECOSCR1 OS EXCP OS TIME OS WTO	AVT CVT DCB DEB Checkpoint Work Area Termname Table	IEDQNF	IEDQNF	D1-1	LINKLIB
IEDQNQ	Checkpoint Notification and Disposition Routine	IEDQNQ	Removes checkpoint request elements and frees a checkpoint record in main storage after the record is written on disk.	IEDQEB FREEMAIN OS WTO	AVT Checkpoint Work Area	IEDQNF	IEDQNF	E1	LINKLIB
IEDQNR	Checkpoint- No Available Core Routine	IEDQNR	Handles the situation in which a conditional GETMAIN for a checkpoint record cannot be satisfied.	OS WTO	AVT Checkpoint Work Area Termname Table	IEDQNF	IEDQNF		LINKLIB
IEDQNS	Checkpoint- No Incident Records Routine	IEDQNS	Causes an environment checkpoint to be taken when all the incident records on the checkpoint data set have been used.	IEDQEB IEDQHG	AVT Checkpoint Work Area	IEDQNF	IEDQNF		LINKLIB
IEDQNX	Operator Awareness Message Router	IEDQNX	Directs messages to the primary operator control terminal when that terminal is not the system console.	IEDQTNT	AVT LCB QCB SCB Buffer Prefix Terminal Table	IEDQBD	IGG019RB	A1 D1-2	LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQOA	GETMAIN, Termname Table Sort, and Attach Routine	IEDQOA	Controls the transient routine IEDQOB, the WTOR Interpreter. Obtains main storage for line buffers, a main-storage message queues data set, CPBs, trace tables, and cross-reference tables Sorts the termname table entries and the concentrator device ID table entries. Attaches the operator control, on-line test, and FE common write tasks. Loads IEDQHI and/or IEDQNX, if indicated in the AVT. Scrambles the input password.	OS ATTACH OS CHAP OS EXTRACT FREEMAIN OS GETMAIN OS LINK OS LOAD OS QEDIT OS WTO	AVT Terminal Table Termname Table	INTRO macro expansion	INTRO macro expansion	A1	LINKLIB
IEDQOB	WTOR Interpreter Routine	IEDQOB	Allows the system operator to redefine certain system values that were specified on the INTRO macro at assembly time.	OS WAIT OS WTO OS WTOR	AVT CVT TCB	OS LINK from IEDQOA	OS LINK to IEDQOA	A1	LINKLIB
IEDQOT	TCAM Abnormal Close Routine	IEDQOT01	Resets any error or attention flags that TCAM has modified in the UCBs.	None	CVT UCB	System Abend	System Abend		SVCLIB
IEDQTNT	Termname Table Code	IEDQTNT	Converts the two-byte ordinal index of a termname table entry to the actual address of that entry in the terminal table.	None	Termname Table	Any TCAM routine	Calling routine	C1-1.4	Stored in the Termname Table
IEDQUI	User Interface Routine	IEDQUI01	Handles the linkage between MH macro expansions and the functional MH routines. Handles the linkage among some of the functional MH routines and the lower-level MH routines.	None	AVT CVT LCB MH VCON Table SCB Buffer Prefix	MH macro expansions and routines	MH routines	C1-1.3 C1-1.2 C1-3.3	TELCMLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQWA	TOTE Resident Module	IEDQWA	Calls in and establishes the functions required to execute an on-line test.	IEDQWB IEDQWK IEDQWN (IEDQW35) IEDQWP (IEDQW39) IEDQWQ (IEDQW37) IEDQWR (IEDQW28) IEDQWS (IEDQW36) IEDQWV (IEDQW21 IEDQWAC) IEDQWX (IEDQW41) IEDQWY (IEDQW16) IEDQW47 IEDQWAB IEDQWAI IEDQW42 (IEDQW43) IEDQW44 IEDQ24	AVT LCB OLTCB TCB		IEDQ05		LINKLIB
IEDQWAB	TIME Service Module	IEDQWAB	Returns time of day, in packed decimal, to the unit test in Register 1.	None	None	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQWAJ	CU Test Service Module	IEDQWAJ	Verifies that all requested channel addresses are offline or issues a command to assign them to TOTE.	None	GETMAIN area OLTCB Unit Test	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQWB	Resource Management Module	IEDQWB	Services requests from IEDQWA.	None	AVT CVT OLTCB	IEDQWA	IEDQWA IEDQWC		LINKLIB
IEDQWB1	Test Request Message Analysis Buffer Analyzer	IEDQWB1	Obtains the TRM from the TCAM buffer and returns the buffer to TCAM.	None	AVT CVT LCB OLTCB SCB	IEDQWB	IEDQWC XCTL to IEDQWH if TRM from numeric terminal		LINKLIB
IEDQWC	Test Request Message Analysis Module 1	IEDQWC	Analyzes TRMs and turns over control to the appropriate routine for further processing.	IEDQWK AVTUI IEDQCV	ACB AVT CVT DCB DEB IEDQWC work area LCB OLTCB SCB TCB Terminal Table Termname Table TTE	IEDQWB	IEDQWC1 IEDQWJ IEDQWI		LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQWC1	Test Request Message Analysis Module 2	IEDQWC1	Analyzes the test device field of the TRM.	IEDQWK AVTVI	AVT CVT DCB DEB IEDQWC1 work area LCB OLTCB QCB SCB TCB Terminal Table Termname Table UCB	IEDQWC	IEDQWC2 IEDQWE		LINKLIB
IEDQWC2	Test Request Message Analysis Module 3	IEDQWC2	Verifies the test and option fields of the TRM.	IEDQWK	IEDQWCZ Work area OLTCB VCB	IEDQWC1	IEDQWD IEDQWE		LINKLIB
IEDQWD	TOTE Dispatcher Module	IEDQWD	Sets NCP flag if no control print option was specified in TRM; performs data protection checks; protects other OLTs from using the same devices/lines as required; stops test devices to prevent their use by TCAM; and builds TOTPRENT if test is for terminals on switched lines.		AVT BEB CDS CVT DCB LCB OLTCB TTE Termname Table UCB	IEDQWC	IEDQWE		LINKLIB
IEDQWE	TOTE Test Control Module	IEDQWE	Schedules the OLTs requested in the TRM and cleans up after their execution.	IEDQWK IEDQCU IEDQCV	OLTCB VCB		IEDQWF IEDQWF		LINKLIB
IEDQWF	OLT Test Control Module 2	IEDQWF	Frees the main storage required by IEDQWE during OLT execution passes control to the OLT Root Module, and receives control back from it.	Device Tests (OLTs)	OLTCB	IEDQWE	IEDQWE1		LINKLIB
IEDQWH	Numeric Test Request Message Handler	IEDQWH	Processes a TRM from a numeric entry terminal.	IEDQWO	AVT CVT DCB OLTCB Termname Table	IEDQWC	IEDQWC IEDQWE		LINKLIB
IEDQWI alias IEDQWI1	TOTE Configurator Scheduler	IEDQWI IEDQWI1	Clears the CDS work and input areas, sets up the output area for CDS members and prompts the user for the type of configuration function.	IEDQWIA IEDQWID IEDQWIE IEDQWK IEDQWE	CDSWORK	IEDQWC IEDQWIA IEDQWID IEDQWIE	IEDQWE		LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQWIA	Configurator and Scheduler	IEDQWIA	Handles the addition of configuration data to the local and remote configuration data sets.	IEDQWID IEDQWIU IEDQWI7 IEDQWI8 IEDQWI9 IEDQWK	CDSWORK	IEDQWI IEDQWID	IEDQWI1 Error Recovery		LINKLIB
IEDQWID	Configurator Change/Delete Scheduler	IEAQWID	Deletes old CDS entries. Gets CDS records for CHANGE function.	IEDQWI5U IEDQWI8 IEDQWIA IEDQWK	OLTCB UCB CDS work area	IEDQWI IEDQWIA	IEDQWI1 IEDQWIA IEDQWI5U IEDQWI8		LINKLIB
IEDQWIE	Configuration Exhibit Module	IEDQWIE	Exhibit TP configuration data set numbers.	IEDQWK	CDS Work area	IEDQWI	IEDQWI		LINKIB
IEDQWI5U	Configurator Submodule	IEDQWI5U	Determines whether the TP line address is for communication or graphic devices and obtains the TCU adapter type.	IEDQWK	CDS Work area UCB	IEDQWIA IEDQWID IEDQWI9	IEDQWIA IEDQWID		LINKLIB
IEDQWI7	Configurator Submodule 3	IEDQWI7	Determines line type and translation code.	IEDQWK CECOM Service Module	CDS Work area UCB	IEDQWIA	IEDQWIA IEDQWID		LINKLIB
IEDQWI8	Configurator Submodule	IEDQWI8	Requests from C. E. and verifies terminal name. Obtains polling and addressing characters from Terminal Table.	AVTUI IEDQWK	CDS Work area Terminal Table Termname Table UCB	IEDQWIA IEDQWID IEDQWIE	IEDQWIA IEDQWID		LINKLIB
IEDQWI9	Configurator Submodule	IEDQWI9	Asks the C. E. for the terminal type and, if the line is bisync, determines the translation code of the terminal.	IEDQWK	CDS Work area OLTCB	IEDQWIA	IEDQWI IEDQWIA		LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQWJ	Test Request Message Prompter Module 1	IEDQWJ	Analyzes OLTCB flag bytes to determine why it was called.	IEDQWK AVTUI	AVT CVT DCB DEB IEDQWI Work area LCB OLTCB QCB SCB TCB Terminal Table Termname Table	IEDQWC IEDQWC1 IEDQWC2	IEDQWJ1 IEDQWE		LINKLIB
IEDQWJ1	Test Request Message Prompter Module 2	IEDQWJ1	Prompts the C. E. for the test and option fields of the TRM.	IEDQWK	IEDQWJ1 Work area OLTCB SCT	IEDQWJ	IEDQWJ2 IEDQWA		LINKLIB
IEDQWJ2	Test Request Message Prompter Module 3	IEDQWJ2	Prompts the C E. for the alternate printer	IEDQWK AVTUI	AVT CVT DCB DEB IEDQWJ2 Work area OLTCB QCB Terminal Table Termname Table TTE UCB	IEDQWJ1	IEDQWD IEDQWE		LINKLIB
IEDQWK	TOTE Message Module	IEDQWK	Provides two-way communication between TOTE and the operator.	IEDQWL IEDQWL1 IEDQWL2 IEDQWL3 IEDQWO	OLTCB TCAM Buffers Message Parameter List	Any TOTE modules requiring message service except TOTE service modules	Any TOTE module requiring message service except TOTE service modules		LINKLIB
IEDQWL	TOTE Message Submodule 1	IEDQWL	Moves requested output message to the output buffer in the OLTCB.	None	OLTCB Message Parameter List	IEDQWK	IEDQWK		LINKLIB
IEDQWL1	TOTE Message Submodule 2	IEDQWL1	Moves the requested output message to the output buffer in the OLTCB.	None	OLTCB Message Parameter List	IEDQWK	IEDQWK		LINKLIB
IEDQWL2	TOTE Message Submodule 3	IEDQWL2	Moves the requested output message to the output buffer in the OLTCB.	None	OLTCB Message Parameter List	IEDQWK	IEDQWK		LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQWL3	TOTE Message Submodule 4	IEDQWL3	Moves requested output message to output buffer in the OLTCB.	None	OLTCB Message Parameter List	IEDQWK	IEDQWK		LINKLIB
IEDQWM2	Trace Function Module	IEDQWM2	Provide C. E. with a limited trace facility for OLT execution and permits evaluation of service module return codes.	IEDQWO	IEDQWM2 Work area OLTCB SCT	All service modules	All service modules		LINKLIB
IEDQWN alias IEDQW35	EXIO Service Module	IEDQWN	Initiates I/O operations.	None	DCB ECB IOB IOBLOCKS OLTCB TECB	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQWO	Access Manager	IEDQWO	Determines the destination output device and communicates with the on-line test operator.	IEDQGA IEDQBD IEDQHM	AVT LCB OLTCB TRM	IEDQWK IEDQWP IEDQWP1 IEDQWP2 IEDQWQ	IEDQWK IEDQWP IEDQWP1 IEDQWP2 IEDQWQ		LINKLIB
IEDQWP alias IEDQW39	DPRINT Service Module	IEDQWP	Services the DPRINT macro by formatting the output messages.	IEDQWO	OLTCB Section Preface	IEDQWA	IEDQWP1 IEDQWM2		LINKLIB
IEDQWP1	DPRINT Service Module	IEDQWP1	Continues servicing the DPRINT macro.	IEDQWO	OLTCB	IEDQWP	IEDQWP2 IEDQWM2		LINKLIB
IEDQWP2	DPRINT Service Module 2	IEDQWP2	Continues servicing the DPRINT macro.	IEDQWO	OLTCB	IEDQWP1	IEDQWA IEDQWM2		LINKLIB
IEDQWQ alias IEDQW37	CECOM Service Module	IEDQWQ	Service requests for communication with the control terminal.	IEDQWO	OLTCB	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQWR alias IEDQW28	PLINK Service Module	IEDQWR IEDQW28	Loads and deletes modules.	None	IEDQWR Work area OLTCB	IEDQWA	IEDQWA IEDQWM2		LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IEDQWS alias IEDQW36	Wait I/O Service Routine	IEDQWS	Causes the on-line test routine to wait until the initiated I/O event has been completed.	None	IOBLOCKS OLTCB TECB VCB	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQWV alias IEDQWAC, IEDQW21	TOTE GRAB Service Module & $LETGO Service Module	IEDQWV	Assigns a secondary device to the unit test, or removes such an assignment.	None	OLTCB	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQWX alias IEDQW41	TOTE Convert Service Module	IEDQWX	Converts data from hexadecimal to EBCDIC or vice versa as specified by the macro parameter list.	None	None	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQWY alias IEDQW16	GETCONFG Service Module	IEDQWY	Reads the CDS data set for configuration data about a TCV or terminal.	IEDQWK	DCB DECB OLTCB				LINKLIB
IEDQW24	READD Service Module	IEDQW24	Reads data from a sequential data set.	IEDQWK	DCB DECB IOB				LINKLIB
IEDQW42 alias IEDQW43	MORECORE Service Module and FREECORE Service Module	IEDQW42	Obtains additional main storage for the unit test by a GETMAIN macro. Frees the main storage obtained by the MORECORE module.	None	OLTCB	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQW44	DIO Service Module	IEDQW44	Issues a Halt IO to a device.	None	DCB ECB IOB OLTCB TECB	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQW47	Routine Service Module	IEDQW47	Handles the selection and running of routines within an OLT section.	None	IEDQW47 Work area SCT	IEDQWA	IEDQWA IEDQWM2		LINKLIB
IEDQXA	Disk Message Queue Initializer	IEDQXA	Builds a formatted disk data set.	OS BSAM OS EXCP OS FEOV OS GETMAIN OS RDJFCB OS WAIT OS WTO	None	OS Task Management	OS Task Management		LINKLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGC0010D	Operator Control Input Handler	IGC0010D	Checks validity of command format. Processes commands from application program, TOTE, and the system console. Checks for freed resources and acquires the element for command processing. Dequeues Processed input from the input queue.	SVC 102 (AQCTL) OS QEDIT OS XCTL	AVT CIBTBL MPPTBL OP Ctl AVT XCTLTBL	IEDQCA IGC0710D	IEDQCA IGCM010D IGCZ010D IGCD010D IGC0310D IGCR010D IGC0110D IGCV010D IGC1010D IGCH010D		SVCLIB
IGC0110D	Terminal Input Scanner	IGC0110D	Processes operator control commands from a terminal. Checks for a canceled control command. Verifies JOBNAME or PROCNAME on MODIFY commands.	OS XCTL	AVT OP Ctl AVT Verb Table XCTLTBL	IGC0010D	IGC0710D IGCD010D IGCH010D IGCM010D IGCR010D IGCZ010D IGC0310D		SVCLIB
IGC0310D	Operator Control Error Message Generator 1	IGC0310D	Generates an error message and transfers control to the output writer. When the message requested is not generated by this module, control is transferred to Message Generator 2.	OS XCTL	AVT OPCE OP Ctl AVT	All operator control command processing routines.	IGC0410D IGC0710D		SVCLIB
IGC0410D	Operator Control Error Message Generator 2	IGC0410D	Generates an error message and transfers control to the output writer. When the message requested is not generated by this module, control is transferred to Message Generator 3.	OS XCTL	AVT OPCE OP Ctl AVT	IGC0310D	IGC0510D IGC0710D		SVCLIB
IGC0510D	Operator Control Error Message Generator 3	IGC0510D	Generates an error message and transfers control to the output writer. If the message requested is not generated by this module, control is transferred to Message Generator 4.	OS XCTL	AVT OP Ctl AVT TCB TIOT	IGC0410D	IGC0710D IGC0810D		SVCLIB
IGC1303D	TCAM Command Scheduler - SVC 34	IGC1303D	Builds a CIB for any operator control command entered from the system console.	FREEMAIN OS GETMAIN OS QEDIT	AVT CIB CVT	SVC 34	IGC0503D IGC2103D Address in register 14		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGC0610D	Operator Control Incident Checkpoint Interface Routine	IGC0610D	Posts a request to checkpoint (if checkpoint is active) to write an operator control incident record for the command	OS XCTL	AVT Ckpt work area OPCE Op Ctl AVT	IGC0410D IGCM210D IGCM510D IGCM610D IGCM710D IGCM810D IGCH010D IGCR010D IGCI010D IGCI110D IGCV110D IGCV210D IGCV310D IGCV410D	The module identified by OCWTG field in the element pointed to by OPCCOPCE, or to the address in OPCRSAVE if restart is in progress		SCVLIB
IGC0710D	Operator Control Output Message Writer	IGC0710D	Sends a message to a terminal or to the console. Passes a return code to TOTE or an application program. Frees any buffer units associated with the command	AQCTL OPCGETBUF OPCLCB OS XCTL WTO	AVT IEAQFX Op Ctl AVT OPCE PCB	IGC0310D IGC0410D IGC0510D IGC0810D IGC0910D IGCM110D IGCMA10D IGCD110D IGCD210D IGCD310D IGCD410D IGCD510D IGCD610D IGCD710D IGCD810D IGCD910D	IGC0010D		SVCLIB
IGC0810D	Operator Control Error Message Generator 4	IGC0810D	Generates an error message and transfers control to the output writer	OS XCTL	AVT ERRORTAB OPCE OP Ctl AVT	IGC0510D	IGC0710D		SVCLIB
IGC0910D	Operator Control VARY, HOLD, RELEASE Message Module	IGC0910D	Generates replies requested by VARY, HOLD, and RELEASE operator control modules.	OS XCTL	AVT Op Ctl AVT	IGCH010D IGCR010D IGCV110D IGCV210D IGCV310D IGCV410D	IGC0710D		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGCD010D	DISPLAY Scan/ Map/Dispatch Routine	IGCD010D	Performs validity checking of display commands. Maps command data into element. Determines the request and transfers control to the proper display module.	OS XCTL OPCDCBLK OPCTPFLK	AVT OPCE Op Ctl AVT Op Ctl work area	IGC0110D	IGC0310D IGCD110D IGCD210D IGCD310D IGCD410D IGCD510D IGCD610D IGCD710D IGCD810D IGCD910D		SVCLIB
IGCD110D	DISPLAY Control Terminal	IGCD110D	Processes operator control commands requesting display of primary or secondary operator control terminals.	OS XCTL	AVT OPCE Op Ctl AVT Terminal Entry Termname Table	IGCD010D	IGC0310D IGC0710D		SVCLIB
IGCD210D	DISPLAY Queue Status Routine	IGCD210D	Processes operator control commands requesting display of QCB fields.	OS XCTL	AVT OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGCD010D	IGC0310D IGC0710D		SVCLIB
IGCD310D	DISPLAY Invitation List Entries	IGCD310D	Processes operator control commands requesting display of active or inactive terminals.	OS XCTL	AVT DCB DEB OPCE Op Ctl AVT Termname Table	IGCD010D	IGC0310D IGC0710D		SVCLIB
IGCD410D	DISPLAY Intercepted Terminals	IGCD410D	Processes operator control commands requesting display of the list of terminals being held.	OS XCTL	AVT OPCE Op Ctl AVT Op Ctl work area Terminal Entry Termname Table	IGCD010D	IGCD0310D IGCD0710D		SVCLIB
IGCD510D	DISPLAY Terminal Information Routine	IGCD510D	Processes operator control commands requesting display of specified terminal entry fields.	OS XCTL	AVT OPCE Op Ctl AVT Sense Byte Conversion Table Status Byte Table Terminal Entry Termname Table	IGCD010D	IGC0310D IGC0710D		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGCD610D	DISPLAY Line Address Routine	IGCD610D	Processes operator control commands requesting display of the line address and relative line number for a specified terminal.	OS XCTL	AVT DCB DEB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGCD010D	IGC0310D IGC0710D		SVCLIB
IGCD710D	DISPLAY Invitation List Routine	IGCD710D	Processes operator control commands requesting display of the status field of invitation lists.	OS XCTL	AVT DCB DEB OPCE Op Ctl AVT Op Ctl work area Status Conversion Table	IGCD010D	IGC0310D IGC0710D		SVCLIB
IGCD810D	DISPLAY Option Field Routine	IGCD810D	Processes operator control commands requesting display of the terminal option fields	OS XCTL	AVT OPCE Op Ctl AVT Terminal Entry Termname Table Translate Tables	IGCD010D	IGC0310D IGC0710D		SVCLIB
IGCD910D	DISPLAY Line Information Routine	IGCD910D	Processes operator control commands requesting display of the LCB fields for a specified line.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB SCB Status Conversion Table	IGCD010D	IGC0310D IGC0710D		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGCH010D	HOLD Terminal Transmission Routine	IGCH010D	Processes a request to prevent terminal from accepting messages	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGC0010D IGC0110D IGCV210D	IGC0610D IGC0310D		SVCLIB
IGCI010D	Deactivate Invitation List Entry Routine	IGCI01D	Deactivates a specified invitation list entry.	OS XCTL	AVT DCB DEB LCB OPCE OP Ctl AVT QCB Terminal Entry Termname Table	IGC0010D	IGCI110D IGC0610D IGC0710D		SVCLIB
IGCI110D	Activate or Move Invitation List Entry Routine	IGCI110D	Activates a specified invitation list entry or moves a new invitation list.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGCI010D	IGC0610D IGC0710D IGCV110D		SVCLIB
IGCMA10D	MODIFY Scan/ Map/Dispatch Module II	IGCMA10D	Continues scan on MODIFY operator commands. Checks validity and dispatches control to proper module.	OS XCTL	AVT OPCE Op Ctl AVT Op Ctl work area	IGCM010D	IGC0310D IGC0710D IGCM410D IGCM510D IGCM710D IGCM810D IGCM910D		SVCLIB
IGCM010D	MODIFY Scan/ Map/Dispatch Module	IGCM010D	Scans MODIFY commands, checks validity, maps into element, and dispatches control to module or to scan 2 for further scan operations.	OS XCTL OPCDCBLK OPCTOFLK	AVT OPCE Op Ctl AVT Op Ctl work	IGC0110D IGC0010D	IGC0310D IGCM210D IGCM710D IGCMA10D		SVCLIB
IGCM110D	MODIFY Function Message Module	IGCM110D	Formats message when operator control modify function is successful.	OS XCTL OPCDBLK OPCTOFLK	AVT OPCE Op Ctl AVT Op Ctl work area	IGC0610D IGCM910D	IGC0710D		SVCLIB
IGCM210D	MODIFY Poll Routine	IGCM210D	Processes operator control commands requesting that Auto Poll be started or stopped.	OS XCTL OPCDCBLK	AVT DCB LCB OPCE Op Ctl AVT	IGCM010D	IGC0310D IGC0610D		SVCLIB
IGCM410D	MODIFY Interval Routine	IGCM410D	Processes operator control commands requesting activation of the system or poll delay interval.	OS XCTL OPCDCBLK OPCTOFLK	AVT OPCE Op Ctl AVT Op Ctl work area	IGCMA10D	IGC0310D IGC0610D IGC0710D		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGCM510D	MODIFY Intense Routine	IGCM510D	Processes operator control commands requesting modification of sense information for intensive recording.	OS XCTL OPCDCBLK OPCTOFLK	AVT DCB LCB OPCE Op Ctl AVT Terminal Entry Termname Table	IGCMA10D	IGC0310D IGC0610D		SVCLIB
IGCM610D	MODIFY Trace Status Routine	IGCM610D	Processes operator control commands requesting modification of trace status for a specified line.	OS XCTL OPCDCBLK	AVT DCB LCB OPCE Op Ctl AVT	IGCMA10D	IGC03´0D IGC06´0D		SVCLIB
IGCM710D	MODIFY Control Terminal Routine	IGCM710D	Processes operator control commands requesting that the primary operator control terminal be changed to the terminal specified in the command.	OS XCTL OPCDCBLK OPCTOFLK	AVT OPCE Op Ctl AVT Op Ctl work area	IGCMA10D	IGC03 I0D IGC06 I0D		SVCLIB
IGCM810D	MODIFY Options Routine	IGCM810D	Processes operator control commands requesting modification of terminal option fields.	OS XCTL OPCDCBLK OPCTOFLK	AVT DCB LCB OPCE Op Ctl AVT Option Characteristics Table Option Table Terminal Entry Termname Table	IGCMA10D	IGC03I0D IGC0610D		SVCLIB
IGCM910D	DEBUG Service Aid Routine	IGCM910D	Processes operator control commands requesting the loading or deleting of the service aid routines.	OS XCTL FE Service Aid Routine FREEMAIN GETMAIN OS BLDL OS DELETE OS LOAD	AVT CVT OPCE Op Ctl AVT TCB	IGCMA10D	IGCM´10D IGC0310D IGC0610D		SVCLIB
IGCR010D	Resume Terminal Transmission	IGCR010D	Processes requests to release a specified intercepted terminal.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGC0010D IGC0110D IGCV410D	IGC0310D IGC0610D		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGCV010D	VARY Scan/ Map/Dispatch Module	IGCV010D	Scans VARY operator control commands, checks validity, maps into element, and dispatches control to proper module.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGC0010D IGC0110D	IGC0310D IGCV110D IGCV210D IGCV310D IGCV410D		SVCLIB
IGCV110D	Stop Line Routine	IGCV110D	Processes operator control commands requesting that line activity be stopped immediately or upon completion of the current operation.	AQCTL OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Stopline Request Element	IGC0010D IGC0710D IGCV010D IGCV210D IGCV410D IGCI010D	IGCV210D IGCV410D IGCI010D IGC0310D IGC0610D		SVCLIB
IGCV210D	Stop Terminal Routine	IGCV210D	Processes operator control commands requesting that a specified terminal be deactivated for entering, or deactivated for both entering and accepting.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGCV010D IGCV110D IGC0010D	IGCV110D IGC0310D IGC0610D		SVCLIB
IGCV310D	Start Line Routine	IGCV310D	Processes operator control commands requesting that activity be started on a line or line group.	OS XCTL EXCP	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Stopline Request Element Terminal Entry Termname Table	IGCV010D IGC0010D IGC0710D	IGC0310D IGC0610D	E1	SCVLIB
IGCV410D	Start Terminal Routine	IGCV410D	Processes operator control commands requesting that a specified terminal be activated for entering, or activated for both entering and accepting.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGCV010D IGCV110D IGC0010D	IGCV110D IGC0310D IGC0610D		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGCV510D	Stop General Poll Routine	IGCV510D	Processes operator control commands requesting that general polling be stopped.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGCV210D	IGC0610D IGC0910D		SVCLIB
IGCV610D	Start General Poll Routine	IGCV610D	Processes operator control commands requesting that general polling be started.	OS XCTL	AVT DCB DEB LCB OPCE Op Ctl AVT QCB Terminal Entry Termname Table	IGCV410D	IGC0610D IGC0910D		SVCLIB
IGCZ010D	MCPCLOSE Scan/ Map/Dispatch Routine	IGCZ010D	Scans MCPCLOSE commands, checks validity, maps and dispatches control to the MCPCLOSE module.	OS XCTL	AVT CVT OPCE Op Ctl AVT Op Ctl work area TCB	IGC0010D IGC0110D IGCV110D	IGCZ110D IGC0010D IGC0310D	D1 E1	SVCLIB
IGCZ110D	MCP Closedown Processing Routine	IGCZ110D	Processes MCPCLOSE from an application program or a HALT command from a terminal or a console	AQCTL OS XCTL	AVT CVT DCB DEB LCB OPCE TCB	IGCV110D	IGC0010D IGCV110D		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGE0004G	Start/Stop ERP Control Module	IGE0004G	Transfers control to the appropriate ERP module to process a specific error condition.	IEDQTNT OS ERREXCP	AVT CCW DCB LCB SCB Terminal Table Termname Table	OS I/O Supervisor	IGE0104G IGE0204G IGE0304G IGE0404G IGE0504G IGE0604G IGE0804G IGE0904G IGG019R0 IGE0025F		SVCLIB
IGE0004H	BSC ERP Control Module	IGE0004H	Transfers control to the appropriate BSC ERP module to process a specific error condition.	IEDQTNT OS ERREXCP	AVT CCW DCB LCB Terminal Table Termname Table	OS I/O Supervisor	IGE0104H IGE0204G IGE0204H IGE0304G IGE0404G IGE0404H IGE0504H IGE0804H IGE0904H IGG019R0		SVCLIB
IGE0104G	READ/WRITE Unit Check ERP Module	IGE0104G	Processes read/write unit check (except time-out) error conditions that occur on start-stop lines.	OS ERREXCP	AVT CCW IOB LCB SCB UCB	IGE0004G	IGE0504G IGG019R0		SVCLIB
IGE0104H	BSC Read/Write Equipment Check, Lost Data, Intervention Required, and Unit Exception ERP Module	IGE0104H	Processes read/write, unit check, and unit exception error conditions that occur on BSC lines.	OS ERREXCP	AVT CCW IOB LCB SCB UCB	IGE0004H	IGE0504H IGG019R0		SVCLIB
IGE0204G	Non-operational Control Unit, Unit Exception, and Unit Check with Time-Out ERP Module	IGE0204G	Informs the operator that a specific control unit is not operational. Processes unit exception and unit check with time-out error conditions.	OS ERREXCP OS WTO	AVT CCW LCB SCB UCB	IGE0004G IGE0004H	IGG019R0 IGE0504G		SVCLIB
IGE0204H	BSC Read/Write Data Check, Overrun, and Command Reject ERP Module	IGE0204H	Processes data check command reject, or overrun errors on a Read or Write CCW	OS ERREXCP	AVT CCW IOB LCB SCB UCB	IGE0004H	IGE0404H IGE0504H		SVCLIB
IGE0304G	Unit Check for Non-read, Non-write, and Non-poll CCWs ERP Module	IGE0304G	Processes unit checks for non-read, non-write or non-poll CCWs.	OS ERREXCP	CCW IOB LCB SCB	IGE0004G	IGE0504G		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGE0404G	Auto Poll and Read Response to Poll Unit Check and Unit Exception ERP Module	IGE0404G	Processes unit checks and unit exceptions for poll CCWs and read response to poll CCWs.	OS ERREXCP	CCW LCB	IGE0004G IGE0004H	IGE0504G		SVCLIB
IGE0404H	BSC Second Level CCW Return Module	IGE0404H	Retry channel programs initiated by an ERP module	OS ERREXCP	AVT CCW LCB SCB Terminal Table	IGE0004H IGE0204H	IGE0504H IGG019R0		SVCLIB
IGE0504G	Error Post and Second Level CCW Return Module	IGE0504G	Attempts to retry channel programs and handles permanent error situations.	IEDQTNT ERREXCP	AVT CCW LCB SCB Terminal Table Termname Table	IGE0004G IGE0104G IGE0204G IGE0304G IGE0404G IGE0604G	IGE0025F IGG019R0 OS Message Writer		SVCLIB
IGE0504H	BSC Error Post Module	IGE0504H	Handles permanent error situations on BSC lines	IEDQTNT	AVT CCW LCB SCB Terminal Table Termname Table	IGE0004H IGE0104H IGE0204H IGE0404H IGE0804H	IGE0025F		SVCLIB
IGE0604G	Unit Check and Unit Exception on Read/Write CCWs for Audio and 2260 Local Devices ERP Module	IGE0604G	Adjusts the retry count and retries the failing CCW sequence when IOS detects an error on an audio or local device.	OS ERREXCP	CCW LCB SCB	IGE0004G	IGE0504G		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGE0804G	Start/Stop Channel Check ERP Module	IGE0804G	Processes channel ending status errors.	OS ERREXCP	CCW LCB ERPIB	IGE0004G	IGE0504G		SVCLIB
IGE0804H	BSC Channel Check ERP Module	IGE0804H	Processes channel ending status errors on BSC lines.	OS ERREXCP	CCW ERPIB LCB	IGE0004H	IGE0504H		SVCLIB
IGE0904G	Closedown Terminal Statistics Recording Module	IGE0904G	Provides for terminal statistics recording when end-of-day recording is specified.	IEDQTNT OS ERREXCP	LCB Terminal Table Termname Table	IGE0004G	IGG019R0		SVCLIB
IGE0904H	TPER Recorder Module	IGE0904H	Interface with IGE0625F for TPER recording of SOH % E records.	IEDQTNT IGE0625F GETMAIN FREEMAIN	AVT CCW CVT DCB DCT LCB RQE SCB SCT Buffer Prefix	IGE0004H	IOS		SVCLIB
IGG019AO	TOTE Start I/O Appendage	IGG019AO	Turns on UCBNALOC bit in UCB for graphic devices.	None	UCB	IOS	IOS		SVCLIB
IGG019AP	TOTE Channel End and Abnormal End Appendage	IGG019AP	Provides for separate channel and device end.	None	UCB IOB	IOS	IOS		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG019Q0	Line I/O Interrupt Trace Routine	IGG019Q0	Makes an entry in the line I/O interrupt trace table each time that it is activated.	None	Line I/O Interrupt Trace Table	IGG019R0	IGG019R0 User Trace Exit Routine		SVCLIB
IGG019Q1	Local Receive Scheduler	IGG019Q1	Schedules receive operations for 2260 and 3270 local lines.	None	AVT DCB LCB	IGG019RB	IGG019RB		SVCLIB
IGG019Q2	Line End Appendage for BSC Lines	IGG019Q2 SCAN	Handles I/O interrupts that occur with device or channel ending status on BSC lines. Schedules ERP, when necessary. Scans for BSC line control characters.	IEDQKB IEDQTNT IGG019Q0 OS POST TESTDSP	AVT CCW DCB LCB QCB RCB SCB Buffer Prefix Terminal Table	IOS ERP routine IGG019RN	IOS		SVCLIB
IGG019Q3	Line End Appendage for Start/Stop Lines	IGG019Q3	Handles I/O interrupts that occur with device or channel ending status on start/stop lines. Schedules ERP, when necessary.	IEDQHK IEDQKC IEDQTNT IGG019Q0 OS POST TESTDSP	AVT CCW DCB LCB QCB RCB SCB Buffer Prefix Terminal Table	IOS ERP routine	IOS		SVCLIB
IGG019Q4	Line End Appendage for Leased and Start/Stop Lines and No TSO	IGG019Q4	Handles I/O interrupts that occur with device or channel ending status on leased or start/stop lines. Schedules ERP, when necessary.	IEDQKD IEDQTNT IGG019Q0 OS POST TESTDSP	AVT CCW DCB LCB QCB RCB SCB Buffer Prefix Terminal Table	IOS ERP routine	IOS		SVCLIB
IGG019Q5	Line End Appendage for a QTAM-Compatible System	IGG019Q5	Handles I/O interrupts that occur with device or channel ending status only for those devices supported by QTAM.	IEDQHK IEDQKE IEDQTNT IGG019Q0 OS POST TESTDSP	AVT CCW DCB LCB QCB RCB SCB Buffer Prefix Terminal Table	IOS ERP routine	IOS		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG019Q6	Send Scheduler for Leased Lines and No TSO	IGG019Q6 LCBSCAN	Schedules send operations for leased lines only with no TSO interface logic. Calculates the LCB address of a destination.	IGG019RB OS IOHALT	AVT DCB LCB QCB RCB STCB Terminal Table	IGG019RB IEDQHM IEDQAS	IGG019RB IEDQHM IEDQAS		SVCLIB
IGG019Q7	Send Scheduler with No TSO	IGG019Q7 LCBSCAN	Schedules send operations in a TCAM system that contains no TSO interface. Calculates the LCB address of a destination.	IEDQTNT IGG019RB OS IOHALT	AVT DCB LCB QCB RCB STCB Terminal Table	IGG019RB IEDQHM IEDQAS	IGG019RB IEDQHM IEDQAS		SVCLIB
IGG019Q8	Checkpoint Continuation Restart Subroutine	IGG019Q8 IGG019Q8+4 IGG019Q8+8 IGG019Q8 +12 IGG019Q8 +16 IGG019Q8 +20 IGG019Q8 +28	Checks terminal entries to determine whether a message queues scan should be performed. Executes disk I/O on the message queues data set. Updates the message sequence number in a terminal entry Examines and, if necessary, updates the queuing indices in the AVT. Initializes registers with values for IGG01945.	IECPCNVT IECOSCR1 IGG019RC OS GETMAIN FREEMAIN OS WAIT	AVT CPB QCB Checkpoint Work Area Terminal Table	IGG01943 IGG01945	IGG01943 IGG01945		SVCLIB
IGG019Q9	Concentrator Send Scheduler	IGG019Q9 DESTENT	Schedules a sending operation and effects reading from multiple QCBs for concentrator output.	IGG019RB OS POST	AVT DCB DEB DRQ LCB QCB QCB Extension SCB Terminal Table	IGG019RB IEDQHM IEDQBQ	IGG019RB IEDQHM IEDQBQ	C1-3.1	SVCLIB
IGG019RA	Checkpoint Disk End Appendage	IGG019RA	Writes the checkpoint control record after the last segment of an environment record is written on disk.	None	DEB Checkpoint Work Area	OS I/O Supervisor	OS I/O Supervisor		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG019RB	TCAM Dispatcher	IGG019RB DSPBYPAS DSPCHAIN DSPDLETE DSPDISP DSPLIFO DSPLIFOR DSPLIST DSPPOST DSPPOSTR DSPPRIO DAPPRIOR DSPTSTQ DSPTSTR DSPUNAV DSPUNAVR DSPWAIT	Allocates and schedules the system resources by processing the elements on the ready queue. Acts as a queue manager according to the label that returning routines branch to in the DSECT table RETTBL	OS DELETE OS POST OS WAIT	AVT QCB RCB STCB	Any TCAM subtask	Any TCAM subtask	A2-3 B1-1 B1-2 C1-1.3	SVCLIB
IGG019RC	EXCP Driver	IGG019RC	Completes building the CCWs in the CPBs that were started by IEDQFA Starts disk I/O and handles disk errors	IECOSCR1 OS EXCP OS WTO	AVT CPB DEB IOB	IEDQFA IGG019Q8	IGG019RB IGG019Q8	C1-2 C1-3.1	SVCLIB
IGG019RD	Buffered Terminal Scheduler	IGG019RD BTSTDQCB TAG	Schedules receive and send operations for buffered terminals. Calculates the LCB address of a destination	IEDQTNT IGG019RB IEDQHG OS POST	AVT DCB DEB LCB QCB SCB Invitation List Terminal Table	IGG019RB IEDQHM	IGG019RB		SVCLIB
OGG019RE	COMMBUF Send Scheduler	IGG019RE	Schedules a broadcast send operation from a common buffer data area		AVT CMB Common Buffer Data area Prefix LCB QCB SCB STCB	IGG019RB	IGG019RB		SVCLIB
IGG019RF	EXCP Driver for a Single CPB	IGG019RF	Completes building the CCWs in the CPB that was begun by IEDQFA Starts disk I/O and handles disk errors	IECOSCR1 OS EXCP OS WTO	AVT CPB DEB IOB	IEDQFA	IGG019RB		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG019RG	GET/READ Routine	IGG019RG	Reads data from the MCP into a work area in the application program.	IEDQEB OS WAIT User Checkpoint Exit Routine	AVT CVT DCB DEB DECB PCB QCB Access Method Work Area Application Program Work Area Process Entry Work Area Termname Table	GET/READ	GET/READ	C2-1	SVCLIB
IGG019RH	GET Compatible Routine	IGG019RH	Moves data from the MCP to an application program when the application program is written in Compatible QTAM.	IEDQEB OS WAIT	AVT CVT DCB DEB PCB QCB Access Method Work Area Buffer Prefix Process Entry Work Area Terminal Table Termname Table	GET macro expansion	GET macro expansion		SVCLIB
IGG019RI	PUT/WRITE Routine	IGG019RI	Prepares data in the application program work area for transfer into buffers in the MCP.	IEDQEB IEDQUI (IEDQE8) OS WAIT User Checkpoint Exit Routine	AVT CVT DCB DEB DECB PCB QCB Access Method Work Area Process Entry Work Area Termname Table	PUT/WRITE	PUT/WRITE	C2-2	SVCLIB
IGG019RJ	PUT Compatible Routine	IGG019RJ	Prepares data in an application program work area for transfer into buffers in the MCP.	IEDQEB IEDQUI (IEDQE8) OS WAIT	AVT CVT DCB DEB PCB QCB Access Method Work Area Application Program Work Area Process Entry Work Area Termname Table	PUT macro expansion	PUT macro expansion		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG019RK	Disk End Appendage for a Single CPB	IGG019RK	Removes the single CPB from the IOB and makes it available for CPB cleanup. Detects disk errors Reactivates the TCAM task.	OS POST	AVT CPB IOB QCB	IOS	IOS		SVCLIB
IGG019RL	Check Routine	IGG019RL	Provides a check function by testing for the completion of a READ or WRITE request and testing for errors that may have occurred during that request	IGG019RG OS WAIT	DCB DEB DECB Access Method Work Area	CHECK macro expansion	CHECK macro expansion		SVCLIB
IGG019RM	Point Routine	IGG019RM	Builds a message retrieval control block to be used to retrieve a specified message.	IEDQUI (IEDQE8)	AVT CVT DCB DEB QCB Access Method Work Area Terminal Table Termname Table	POINT macro expansion	POINT macro expansion		SVCLIB
IGG019RN	PCI Appendage	IGG019RN	Handles program-controlled channel interruptions Frees buffers from the line operation just completed and, if necessary, obtains additional buffers.	IEDQTNT IGG019RO OS POST TESTDSP	AVT CCW DCB LCB Buffer Prefix Terminal Table Termname Table	IOS	IOS	A2-3 C1-1.1 C1-3.1	SVCLIB
IGG019RO	TCAM Dispatcher with Subtask Trace	IGG019RO DSPBYPAS DSPCHAIN DSPDLETE DSPDISP DSPLIFO DSPLIFOR DSPLIST DSPPOST DSPPOSTR DSPPRIO DSPPRIOR DSPTSTQ DSPTSTR DSPUNAV DSPUNAVR DSPWAIT	Allocates and schedules the system resources by processing the elements on the ready queue. Acts as a queue manager according to the label that returning routines branch to in the DSECT table RETTBL. Makes an entry in the subtask trace table each time that a subtask is activated.	IEDQFE10 OS DELETE OS POST OS WAIT	AVT QCB RCB STCB Subtask Trace Table	Any TCAM subtask	Any TCAM subtask	A2-3 B1-1 B1-2 C1-1.3	SVCLIB
IGG019RP	Reusability-Copy Subtask	IGG019RP REUSQCB READONE UNITQCB COPY WRITQCB	Makes the disk message queues data set available for reuse. Copies an entire message from one message queue to another.	IEDQHG IEDQHM IEDQTNT OS POST	AVT CPB LCB QCB Buffer Prefix Terminal Table Termname Table	IEDQFA IGG019RB	IEDQFA IGG019RB	D1-1	SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG019RQ	Post Pending Routine	IGG019RQ	Posts complete the ECB for a task that has an OS POST pending when that task is being rolled in	OS POST	AVT CVT DEB PCB TCB Process Entry Work Area	IEAQRORI	IEAQRORI		SVCLIB
IGG019R0	Line End Appendage	IGG019R0 SCAN	Handles I/O interrupts that occur with device or channel ending status. Schedules ERP, when necessary Scans for BSC line control characters.	IEDQHK IEDQKA IEDQTNT IGG019Q0 OS POST TESTDSP	AVT CCW DCB LCB QCB RCB SCB Buffer Prefix Terminal Table	IOS ERP routine IGG019RN	IOS	C1-1.1 C1-3 1	SVCLIB
IGG019R1	Dial Receive Scheduler	IGG019R1	Initiates receive operations for a dial line and prepares for send operations upon completion of the input	IEDAYZ IEDQHG IEDQTNT IGG019RB OS EXCP OS TIME	AVT DCB DEB LCB QCB RCB STCB TS QCB Terminal Table Termname Table	IGG019RB	IGG019RB		SVCLIB
IGG019R2	Disk End Appendage	IGG019R2	Removes CPBs from the IOB and makes them available for CPB cleanup Detects disk errors. Reactivates the TCAM task	OS POST	AVT CPB IOB QCB	IOS	IOS	C1-3 1	SVCLIB
IGG019R3	Leased Receive Scheduler	IGG019R3 QEVENT	Services receive operations on leased lines	IEDAYZ IGG019RB OS POST	AVT DCB LCB QCB RCB STCB TS QCB Terminal Table	IGG019RB	IGG019RB	C1-1.1	SVCLIB
IGG019R4	Send Scheduler	IGG019R4 LCBSCAN	Schedules send operations. Calculates the LCB address of a destination.	IEDAYZ IEDQTNT IGG019RB OS IOHALT	AVT DCB LCB QCB RCB STCB TS QCB Terminal Table	IGG019RB IEDQHM IEDQAS	IGG019RB IEDQHM IEDQAS	C1-3 1	SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG019R5	Attention Handler	IGG019R5	Schedules a receive operation for a device that has entered an attention interrupt.	OS POST TESTDSP	AVT DCB DEB LCB	IEDQATTN	IOS		SVCLIB
IGG019R6	Start-up Message Routine	IGG019R6	Obtains and queues any messages that the user has to send to a terminal at start-up time.	IEDQHM02 IEDQTNT IGG019RC OS WAIT User routines	AVT CPB DCB LCB QCB SCB Option Table Buffer Prefix Termname Table Terminal Table	IGG019RB	IGG019RB		SVCLIB
IGG01930	Disk Message Queues Open Routine-Load 1	IGG01930	Obtains main storage for and initializes a DEB for a message queues DCB.	OS GETMAIN	AVT DCB UCB	OS XCTL	OS XCTL (IGG01931 or IGG01933)	A2-1	SVCLIB
IGG01931	Disk Message Queues Open Routine-Load 2	IGG01931	Completes initialization of the DEB extents. Builds and initializes all IOBs required for disk operation.	OS GETMAIN	AVT DEB IOB	OS XCTL (IGG01930)	OS XCTL (IGG01934 or IGG01933)	A2-1	SVCLIB
IGG01933	Open Error Handler	IGG01933	Handles all serious errors detected during the opening of an application program DCB, a message queues DCB, or a line group DCB.	OS SYNCH OS WTO	AVT DCB	OS XCTL from any TCAM open executor	OS ABEND Any TCAM open executor	A4	SVCLIB
IGG01934	Disk Message Queues Open Routine-Load 3	IGG01934	Performs all the disabled initialization functions, loads the TCAM Dispatcher, EXCP Driver, Disk End Appendage, and the Reusability-Copy subtask.	FREEMAIN OS LOAD OS GETMAIN	AVT DCB DEB	OS XCTL (IGG01931)	OS XCTL (IGG01934, IGG0190S, or IGG01941)	A2-1	SVCLIB
IGG01935	Line Group Open Routine-Load 1	IGG01935	Obtains main storage for and initializes a line DEB	OS GETMAIN	DEB DCT LCB UCB	OS XCTL	OS XCTL (IGG01936 or IGG01933)	A2-3	SVCLIB
IGG01936	Line Group Open Routine-Load 2	IGG01936	Determines the size of the channel programs for all devices in the line group being opened. Obtains main storage for an LCB for each line.	OS GETMAIN	DCT LCB QCB STCB UCB	OS XCTL (IGG01935)	OS XCTL (IGG01937 or IGG01933)	A2-3	SVCLIB
IGG01937	Line Group Open Routine-Load 3	IGG01937	Builds and initializes all LCBs for this line DCB open	None	LCB STCB	OS XCTL (IGG01936)	OS XCTL (IGG01938)	A2-3	SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG01938	Line Group Open Routine-Load 4	IGG01938	Builds channel programs in the LCBs for the lines of the line group being opened	None	LCB	OS XCTL (IGG01937)	OS XCTL (IGG01939)		SVCLIB
IGG01939	Line Group Open Routine-Load 5	IGG01939	Loads some of the modules required for line operation. the PCI Appendage and the Line End Appendage Loads the device-dependent special characters required for initial I/O operations	FREEMAIN OS GETMAIN OS LOAD	AVT CVT DCB DEB TCB	OS XCTL (IGG01938)	OS XCTL (IGG01940)	A2-3	SVCLIB
IGG0194B	Application Program Open Error Interface Routine	IGG0194B	Cleans up partially open DCBs that exist as a result of an open error that occurred for other than the first DCB in a multiple-open macro	None	AVT CVT DCB DEB TCB Process Entry Work Area Terminal Table	OS XCTL (IGG01946 or IGG01947)	OS XCTL (IGG01933)		SVCLIB
IGG01940	Line Group Open Routine-Load 6	IGG01940	Completes loading the modules required for line operation: the Send Scheduler, the TCAM Dispatcher, the appropriate receive schedulers, and the Start-up Message routine Starts I/O on each line in the line group	OS EXCP OS LOAD	AVT CVT DCB DCT DEB LCB TIOT Special Characters Table	OS XCTL (IGG01939)	OS XCTL (IGG01948)	A2-3	SVCLIB
IGG01941	Checkpoint Open Routine	IGG01941	Obtains main storage for and initializes a checkpoint work area in a MCP	OS EXCP OS GETMAIN OS WTO	AVT CVT JFCB Checkpoint DCB Checkpoint DEB Checkpoint Work Area	OS XCTL (IGG01934)	OS XCTL (IGG01942, IGG01943, or IGG01949)	A2-2	SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG01942	Checkpoint Disk Initialization Routine	IGG01942	Initializes the disk checkpoint data set into specific areas for a control record, environment checkpoint records, CKREQ records, and incident records.	IECPCNVT IECOSCR1 OS EXCP OS WAIT OS WTO	AVT CVT Checkpoint DCB Checkpoint DEB Checkpoint QCB Checkpoint Work Area	OS XCTL (IGG01949, IGG01941, or IGG01943)	OS XCTL (IGG01944 or IGG019OS)	A2-2	SVCLIB
IGG01943	Checkpoint/Restart from Environment Record Routine	IGG01943	Reconstructs the MCP environment from the environment record segments in the checkpoint data set.	IGG019Q8 OS LOAD	AVT CVT TCB Checkpoint DCB Checkpoint DEB Checkpoint QCB Checkpoint Work Area Invitation List Terminal Table Termname Table	OS XCTL (IGG01941)	OS XCTL (IGG01942 or IGG01944)	D2	SVCLIB
IGG01944	Checkpoint/Restart from Incident and CKREQ Records Routine	IGG01944	Updates the MCP environment with the incident checkpoint record for stop line or start line and with the CKREQ records.	IEDQTNT IGG019Q8 OS WTO	AVT CVT DCB QCB Checkpoint Work Area Option Table Terminal Table Termname Table	OS XCTL (IGG01942 or IGG01943)	OS XCTL (IGG01945 or IGG019OS)	D2	SVCLIB
IGG01945	Checkpoint Continuation Restart Routine	IGG01945	Performs any required processing of the message queues data set at restart time.	IEDQTNT IGG019Q8 OS DELETE	AVT CPB QCB Buffer Prefix Checkpoint Work Area Disk Data Area Termname Table Terminal Table	OS XCTL (IGG01944)	OS XCTL	D2	SVCLIB
IGG01946	GET/PUT and READ/WRITE Open Executor - Load 1	IGG01946	Opens input and output DCBs in an application program.	IEDQEB IEDQUI (IEDQA1) OS GETMAIN OS LOAD OS WAIT	AVT CVT DCB DEB JFCB TCB Access Method Work Area Process Entry Work Area Termname Table	OS XCTL	OS XCTL (IGG01947) IGG01933	A4	SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG01947	GET/PUT and READ/WRITE Open Executor-Load 2	IGG01947	Completes opening an input DCB in an application program	IEDQEB IEDQNB OS GETMAIN	AVT CVT DCB DEB JFCB TCB Access Method Work Area Process Entry Work Area Termname Table	OS XCTL (IGG01946)	OS XCTL	A4	SVCLIB
IGG01948	Line Group Open Routine-Load 7	IGG01948	Places line-specific information in the cross-reference table Examines each line for completion of the initial I/O operations	OS TIME OS WTO	LCB QCB UCB Cross-Reference Table	OS XCTL (IGG01940)	OS XCTL (IGG0190S)	A2-3	SVCLIB
IGG01949	Checkpoint Disk Allocation Routine	IGG01949	Determines the size of the various checkpoint records for the checkpoint data sets. Initializes the checkpoint work area with the number of tracks in the checkpoint data set, the size of each disk record, and the number of records per track	OS WTO	AVT CVT DCB DEB QCB OS I/O DCT Checkpoint Work Area Invitation List Option Table Terminal Table Termname Table	OS XCTL (IGG01941)	OS XCTL (IGG01942)	A2-2	SVCLIB
IGG02030	Disk Message Queues Close Routine	IGG02030	Closes a message queues DCB in a TCAM MCP	FREEMAIN	DCB DEB TCB Cross-Reference Table	OS XCTL	OS XCTL	E1	SVCLIB
IGG02035	Line Group Close Routine-Load 1	IGG02035	Issues an EXCP on the line to perform error recording. Abends any application programs, if necessary.	OS ABEND OS EXCP OS WAIT OS WTO	DCB DEB TCB	OS XCTL	OS XCTL (IGG02036)	E1	SVCLIB
IGG02036	Line Group Close Routine-Load 2	IGG02036	Purges all I/O on the lines associated with this DCB. Disables the lines and frees the associated LCBs. Clears any associated entries in the cross-reference table.	FREEMAIN OS PURGE OS WAIT OS WTO	CVT DCB DEB LCB TCB Cross-Reference Table	OS XCTL (IGG02035)	OS XCTL		SVCLIB

Module Name	Generic Name	Entry Points	Functions	External Routines	Tables/ Work Areas	Entered From	Exits To	Method of Operation Chart	Library
IGG02041	Checkpoint Close Routine	IGG02041	Closes the checkpoint DCB in an MCP.	OS DELETE OS EXCP FREEMAIN OS WAIT OS WTO	AVT CVT Checkpoint Work Area	OS XCTL (IGG02030)	OS XCTL	E1	SVCLIB
IGG02046	GET/PUT and READ/WRITE Close Executor-Load 1	IGG02046	Closes a GET/PUT or a READ/WRITE DCB in an application program by deactivating the data communication link between the application program and the MCP.	IEDQEB IEDQNB IEDQTNT OS DELETE FREEMAIN OS WAIT	AVT CVT DCB DEB LCB QCB TCB Access Method Work Area Process Entry Work Area Terminal Table	OS XCTL (OS CLOSE)	OS XCTL (IGG02047)	E2	SVCLIB
IGG02047	GET/PUT and READ/WRITE Close Executor-Load 2	IGG02047	Completes closing a GET/PUT or a READ/WRITE DCB in an application program by unlocking any TCAM LCBs that are locked to the application program DCB.	IEDQEB IEDQTNT	AVT CVT DCB LCB Termname Table	OS XCTL (IGG02046)	OS XCTL	E2	SVCLIB

Non-Executable TCAM Modules Microfiche Directory

DSECT	Generic Name	Macro Name
IEDCBDA	Common Buffer Data Area Prefix	IEDCBDA
IEDCMB	Common Buffer Master QCB	IEDCMB
IEDQAVTD	Address Vector Table	TAVTD
IEDQCCW	Channel Command Word	TCCWD
IEDQCDRD	Incident or Environment Checkpoint Disk Record	
IEDQCIBD	Command Input Block	CIB
IEDQCKPD	Checkpoint Work Area	TCKPD
IEDQCPB	Channel Program Block	TCPBD
IEDQCRED	Checkpoint Request Element—Incident or CKREQ	
IEDQC5	Operator Control Work Area	
IEDQDATA	Disk Data Record Area	TDATAD
IEDQDEB	Data Extent Block for TCAM Application Programs	TDEBAPD
IEDQDEB	Data Extent Block	TDEBD
IEDQDISP	TCAM Dispatcher DSECT	TDISPD
IEDQDRQ	Concentrator Data Ready Queue	TDRQD
IEDQDVCT	Concentrator Device ID Table	TDVCIDTD
IEDQIOB	Input/Output Block	TIOBD
IEDQLCB	Line Control Block	TLCBD
IEDQOPCD	Operator Control AVT	TOPCAVTD
IEDQOPCE	Operator Control Element	TOPCED
IEDQPCB	Process Control Block	TPCBD
IEDQPEWA	Process Entry Work Area	TPEWAD
IEDQPRF	Buffer Prefix	TPRFD
IEDQQCB	Queue Control Block	TQCBD
IEDQQCBE	Queue Control Block Extension	TQCBED
IEDQRECB	Resource Control Block	TRECBD
IEDQSCB	Station Control Block	TSCBD
IEDQSECT	Work Area Macro	FORECORE

DSECT	Generic Name	Macro Name
IEDQSTCB	Subtask Control Block	TSTCBD
IEDQTCB	Task Control Block	TTCBD
IEDQTNTD	Termname Table	TTNTD
IEDQTRM	Terminal Table Entry	TTRMD
IEDQTSI	Time Sharing Queue Control Block	TTSID
IEDQWRKA	Access Method Work Area	TACSMD
IEDQXSA	Extended Save Area Macro	IEEXSA
IEDQ10	IBM 1030 Translate Table	
IEDQ11	IBM 1050 Translate Table	
IEDQ12	IBM 1050 Folded Translate Table	
IEDQ13	IBM 1060 Translate Table	
IEDQ14	IBM 2260 Translate Table	
IEDQ15	Alias for IEDQ14	
IEDQ16	IBM 2740 Translate Table	
IEDQ17	IBM 2740 Folded Translate Table	
IEDQ18	World Trade Teletype Adapter (WTTA), ITA2 Translate Table	
IEDQ19	World Trade Teletype Adapter (WTTA), ZSC3 Translate Table	
IEDQ20	AT&T 115A or Western Union 83B3 Translate Table	
IEDQ21	AT&T TWX, with Parity Translate Table	
IEDQ22	AT&T TWX, without Parity Translate Table	
IEDQ23	IBM 2780, 6-bit Code Translate Table	
IEDQ24	USASCII Code Translate Table	
IEDQ25	Dummy Table (EBCDIC to EBCDIC)	
IEDQ26	IBM 2741, BCD Code Translate Table	
IEDQ27	IBM 2741, EBCD Code Translate Table	
IEDQ28	IBM 2741, Correspondence Code Translate Table	
IGG019RR	IBM 1030, 1050, 1060, 2740, 2741 Special Characters Table	
IGG019RS	IBM 2260 Remote Special Characters Table	
IGG019RT	AT&T 115A or Western Union 83B3 Special Characters Table	

DSECT	Generic Name	Macro Name
IGG019RU	ATT&T TWX, with Odd Parity Special Characters Table	
IGG019RV	IBM 2260 Local Special Characters Table	
IGG019RW	World Trade Teletype Adapter (WTTA) Special Characters Table	
IGG019RX	AT&T TWX, with Even Parity Special Characters Table	
IGG019RY	Audio Special Characters Table	
IGG019R7	BSC EBCDIC Code Special Characters Table	
IGG019R8	BSC USASCII Code Special Characters Table	
IGG019R9	BSC 6-bit Code Special Characters Table	

Macro Linkage Charts

The macro linkage charts show the functional results that occur when TCAM macros are issued. The macro and any information about where it can appear is located on the extreme left of each chart. If the MCP assembly generates a parameter list from a macro, this list is shown under the heading *Parameter List*. The *Linkage* column shows which TCAM modules gain control when the specified macro is coded. Next to each module name there is a brief statement of the functions of that module.

The macros are arranged in alphabetical order.

The following symbols show the linkage among the modules:

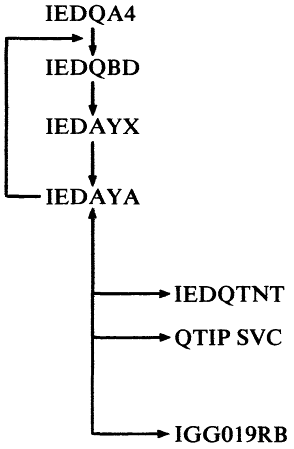
——→	Branch
←——→	Branch and link
←---→	Transfer control (XCTL)
←(xx)→	Issue SVC xx

Macro **Parameter List** **Linkage** **Function**

ATTEN
(INMSG/OUTMSG)
(TSO)

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td></td><td>Index to IEDAYX</td><td>Parameter List Length</td><td>X'01' Indicates Mask Present</td><td>Mask</td></tr>
<tr><td>4</td><td colspan="4">Mask
X00 X'12' X00 X00</td></tr>
<tr><td>8</td><td>Unused</td><td colspan="3">Address of Attention Routine</td></tr>
</table>

Bit 1



Linkage	Function
IEDQA4	
IEDQBD	Return any unused buffers.
IEDAYX	Provide linkage to the TSO Attention or Hangup routines.
IEDAYA	Provide the user the ability to effect line deletion or CPU task interruption.
IEDQTNT	Get the terminal entry address.
QTIP SVC	Clear the input and output queues and swap the user into main storage.
IGG019RB	Tpost the ERBs.

Macro **Parameter List** **Linkage** **Function**

CANCELMG with LEVEL=BLK

	0		1	2	3
	Index to IEDQBU	Bits	Parameter List Length and Logical	X'00'	Mask
4	Mask				

Bit 6 Recall is necessary
7 Indicates an unconditional mask

Note: With LEVEL=BLK, the logical bit is always off, indicating an *OR* function.

Linkage	Function
IEDQA4	
↕ (Tpost) ↓	
IEDQBD	Return any unused buffers, execute the INMSG/OUTMSG macro expansions, and check the parameter list.
↕	
IEDQBU	Recall the last good block of data from disk and tpost it back, with an adjusted PRFSIZE, to the destination QCB.
↕	
IEDQTNT	Get the terminal entry address.

CANCELMG with LEVEL=MSG (INMSG/OUTMSG)

	0	1	2	3
	Index to IEDQAR and Bits	Parameter List Length and Logical	X'00'	Mask
4	Mask			

Bit 6 Recall is necessary
7 Indicates an unconditional mask

Logical 7 AND the mask

Linkage	Function
IEDQA4	
↕	
IEDQBD	Return any unused buffers.
↕	
IEDQAR	Notify the Destination Schedular to cancel the message currently being received.
↕	
IEDQTNT	Get the terminal entry address.

<table>
<tr><th>Macro</th><th>Parameter List</th><th>Linkage</th><th>Function</th></tr>
<tr><td>CARRIAGE
(TSO)</td><td></td><td>IEDQUI</td><td>Activate an MH routine.</td></tr>
<tr><td></td><td></td><td>↕
IEDAYC</td><td>Maintain the position count for carriages of keyboard devices.</td></tr>
<tr><td></td><td></td><td>↕
IEDQTNT</td><td>Get the terminal entry address.</td></tr>
<tr><td>CHECK</td><td></td><td>IGG019RL</td><td>Test for completion or errors in the execution of READ or WRITE macros.</td></tr>
<tr><td>CHECKPT</td><td>(INHDR/OUTHDR, INBUF/OUTBUF)<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>Index to IEDQBB</td><td>Parameter List Length - X'04'</td><td>X'00'</td><td>X'00'</td></tr>
</table></td><td>IEDQUI</td><td>Activate an MH routine.</td></tr>
<tr><td></td><td></td><td>↕
IEDQBB</td><td>Set the "checkpoint request" flag.</td></tr>
<tr><td></td><td></td><td><i>or</i></td><td></td></tr>
<tr><td></td><td>(INMSG/OUTMSG)</td><td>IEDQA4</td><td></td></tr>
<tr><td></td><td></td><td>↕
IEDQBD</td><td>Return any unused buffers.</td></tr>
<tr><td></td><td></td><td>↕
IEDQBB</td><td>Tpost the ERB to the buffer disposition QCB.</td></tr>
</table>

Macro	Parameter List	Linkage	Function
CKREQ		IEDQNB	Build a "checkpoint request" element and tpost it to the checkpoint QCB. The format of this element is:

Offset		1
0	Key X'60'	Address of Checkpoint QCB
4	Priority	Link Address
8	Address of Application Program ECB	
12	Address of Application Program DEB Chain	

Macro	Parameter List	Linkage	Function
		IEDQTNT	Get the terminal entry address.
		IEDQEB (SVC 102)	Tpost the "checkpoint request" element to the checkpoint QCB in the MCP.

Macro	Parameter List	Linkage	Function
CLOSE (Application Program)		SVC 20	
		IGG02046	Deactivate the data transfer communication link between an application program and the MCP.
		IGG02047	
		IEDQEU	Post the application program ECB as complete and return to the TCAM Dispatcher.
CLOSE (Checkpoint)		SVC 20	
		IGG02030	Remove the DEB for the DCB from the DEB chain in the TCB. Free all IOBs associated with the DCB.
		IGG02041	Close the checkpoint DCB.
CLOSE (Line Group)		SVC 20	
		IGG02035	Perform error recording. If the close is the result of an MCP ABEND, terminate the application program.
		IGG02036	Close the line group DCB.
CLOSE (Message Queues)		SVC 20	
		IGG02030	Remove the DEB for the DCB from the DEB chain in the TCB. Free all IOBs associated with the DCB.

Macro	Parameter List	Linkage	Function
CLOSEMC (Compatible QTAM)		IEDQET ↓	Perform the TCAM operator control functions from an application program without using PUT.
		IEDQEB ↓	Move data across partition boundaries.
		IEDQE6	Scramble the password.
CODE		IEDQUI ↓	Activate an MH routine.
		IEDQAW	Translate the data in the buffer.

	0	1	2	3
	Index to IEDQAW	Parameter List Length	X'00'	Status
4	Address of the Translation Table			

Status X'80' Use the Translation Table address in the DCB
X'40' Use a nonstandard Translation Table
X'20' Entry is from INBUF or OUTBUF
X'10' Entry is from INMSG or OUTMSG
X'00' Use a standard Translation Table

	0	1	2	3
	Index to IEDQAI	Parameter List Length	Register 15 Offset	Variable Length Indicator
4	Blank Character	Address of Characters		

Macro	Parameter List	Linkage	Function

COMMBUF

	0	1	2	3
	Index to IEDQBV	Parameter List Length	MAXDEEP	
4	Address of TLIST			

Linkage	Function
IEDQUI	Link to functional MH routine.
IEDQBV →	Insert COMMBUF STCBs into the STCB chains of the appropriate LCBs.

COUNTER

0	1	2	3
Index to IEDQA7	Parameter List Length - X'04'	Option Field Offset	Register 15 Offset

Linkage	Function
IEDQA7	Count either complete messages or message segments.
IEDQUI	Link to IEDQAE.
IEDQAE	Get the option field address.

CTBFORM (OUTBUF)

	0	1	2	3
	Index to IEDQGH	Parameter List Length	CTBFORM Options	Reserved
4	Index to IEDQAF	Index to IEDQAO	Index to IEDQAE	Option Field Offset

CTBFORM Options:

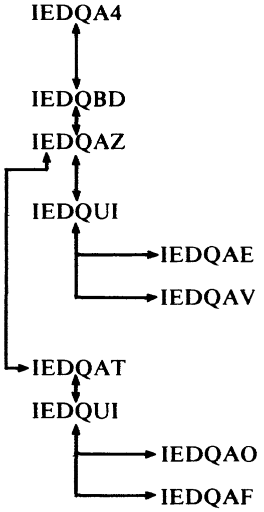
X'01'	Insert the option field data
X'02'	Insert the device identification
X'04'	Insert a CTB ending character

Linkage	Function
IEDQUI	Activate an MH routine.
IEDQGH	Insert device identifications, CTB ending characters, and option field data.
→IEDQTNT	Get the terminal entry address.
IEDQUI	Activate an MH routine.
→ IEDQAE	Locate an option field address.
→ IEDQAO	Get a buffer unit.
→ IEDQAF	Insert data in the buffer.

<table>
<tr><th>Macro</th><th>Parameter List</th><th>Linkage</th><th>Function</th></tr>
<tr><td>CUTOFF</td><td><table><tr><td>0</td><td>1</td><td>2</td></tr><tr><td>Index to IEDQAU</td><td>Parameter List Length - X'04'</td><td>Requested Cutoff Length</td></tr></table></td><td>IEDQUI
↕
IEDQAU</td><td>Activate an MH routine.
Cut off the transmission of a message being received after receipt of a user-specified number of bytes, or on detection of identical characters in the buffer.</td></tr>
<tr><td>DATETIME</td><td><table><tr><td>0</td><td>1</td></tr><tr><td>Index to IEDQAF and Bit</td><td>Data Type Flag</td></tr></table>Bit 6 ON Requests the expand buffer function</td><td>IEDQUI
↕
IEDQAF</td><td>Activate an MH routine.
Insert data and return.
Insert data, adjust the prefix, adjust the offset by the length of the data inserted, and return.
Shift data across several units and return.
Expand the buffer by shifting data left into the reserve characters area.</td></tr>
<tr><td></td><td><i>or</i></td><td></td><td></td></tr>
<tr><td></td><td><table><tr><td>0</td><td>1</td><td>2</td></tr><tr><td>Index to IEDQAC</td><td>Parameter List Length</td><td>Count of Bytes to be Inserted</td></tr></table></td><td>IEDQUI
↕
IEDQAC</td><td>Activate an MH routine.
Insert the current date and/or time of day into the message header.</td></tr>
<tr><td>DCB</td><td>This macro generates no executable code.</td><td></td><td></td></tr>
</table>

Macro	Parameter List	Linkage	Function
ERRORMSG			

	0	1		3
	Index to IEDQAZ and Bits	Parameter List Length and Logical	Status	Mask
	Mask			
8	Destination Status	Variable Data		
12	Index to IEDQAT	Parameter List Length	Index to IEDQAF	Index to IEDQAO
16	Count	Address of the Message		
20	Address of the Exit Routine			



Linkage	Function
IEDQA4	
IEDQBD	Return any unused buffers.
IEDQAZ	Redirect a message, as specified by the user.
IEDQUI	Activate an MH routine.
IEDQAE	Get the option field address.
IEDQAV	Get the terminal entry for a specified destination.
IEDQAT	Generate an error message.
IEDQUI	Activate an MH routine.
IEDQAO	Get a buffer unit.
IEDQAF	Insert the error message in the buffer.

Bit	6	Recall is necessary
	7	Indicates an unconditional mask
Logical	7	AND the mask
Status	X'01' indicates that the IEDQAT parameter list follows	

Destination Status and Variable Data
C'S' + AL3(0) - send to the source
C'D' + AL3(0) - send to the destination
C'N' + AL3(destination name) - send to the named destination
C'O' + I(AE) + AL1(opfld) + AL1(16) - send to the destination named in the option field

Macro	Parameter List	Linkage	Function
FORWARD		IEDQUI	Activate an MH routine.
		IEDQA5	Determine if this is a buffer to be executed by FORWARD.
		IEDQAE or IEDQAI	Find the destination address.
		IEDQA1	Find the terminal entry offset.
		IEDQAV	Set up the destination QCB.
		IEDQTNT	Get the terminal entry address.

	0	1	2	3
	Index to IEDQA5	Parameter List Length	Status	Index to IEDQBA
4	EOA String Length	Address of EOA String		
8	Address of the Exit Routine			
12	Variable Data			
16	Index to IEDQA1	Parameter List Length	X'00'	Length
20	Address of the Character String			

Status	0	Destination=name
	1	Destination=option field
	2	Destination is in the buffer
	3	An exit is specified
	4	EOA is specified
	5	THRESH is specified
	6	DEST=ORIGIN is specified
	7	Destination is specified in register

Variable Data

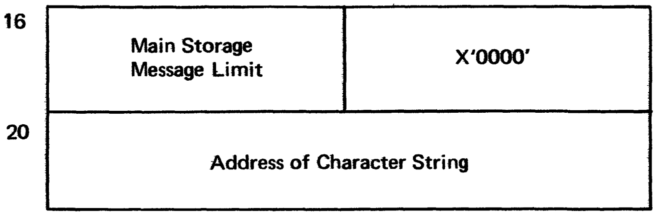
Index to IEDQAE,length,option,X'16' - if the destination is in the option field

Index to IEDQAI,length,X'16',address field length - if the destination is in the buffer

Macro	Parameter List	Linkage	Function

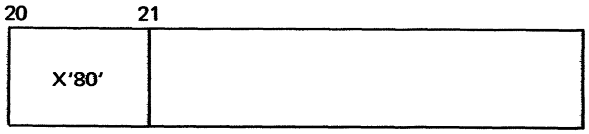
FORWARD with the THRESH operand

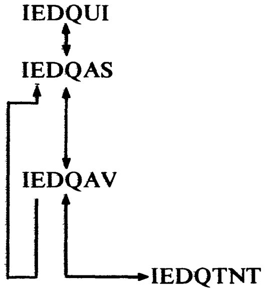
Bytes 0-15 are the same as above



FORWARD with DEST=destname or DEST=ORGIN (after the first message has been sent to that destination)

Bytes 0-19 are the same as above

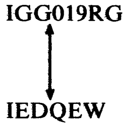




IEDQUI: Determine if this is a buffer to be executed by FORWARD.

IEDQAV: Convert the destination offset to a destination address and set up the destination QCB.

GET



IGG019RG: Read data from full buffers on the element chain of the read-ahead QCB.

IEDQEW: Read data from the message queues data set in anticipation of a GET command from an application program.

Macro	Parameter List	Linkage	Function
GET (Compatible QTAM)		IGG019RH ↕	Compatible to a QTAM GET. Move data from buffers on the read-ahead QCB into the application program work area.
		IEDQEW	Read data from the message queues data set in anticipation of a GET command from an application program.
HANGUP (TSO)		IEDAYX ↓	Activate a TSO module.
		IEDAYH ↑	Identify line errors and disconnect the terminal.
		└→ IEDQTNT	Get the terminal entry address.

	0	1	2	3
0	Index to IEDAYX	Parameter List Length	Unused	
4	Unused	Address of Hangup Routine		

<table>
<tr><th>Macro</th><th>Parameter List</th><th>Linkage</th><th>Function</th></tr>
<tr><td>HOLD (INMSG/OUTMSG)</td><td>
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td></td><td>Index to IEDQAS and Bits</td><td>Parameter List Length and Logical</td><td>Type</td><td>Mask</td></tr>
<tr><td>4</td><td colspan="4">Mask</td></tr>
<tr><td>8</td><td colspan="2">Interval</td><td>X'00'</td><td>X'00'</td></tr>
</table>
<table>
<tr><td>Bit</td><td>6</td><td>Recall is necessary</td></tr>
<tr><td>Logical</td><td>7</td><td>AND the mask</td></tr>
<tr><td>Type</td><td>X'02'</td><td>LEVEL=BLK is specified</td></tr>
<tr><td></td><td>X'00'</td><td>LEVEL=MSG is specified</td></tr>
</table>
</td><td>IEDQA4
↓
IEDQBD
↑
└→ IEDQAS</td><td>Return any unused buffers.
Hold messages or a message block from being sent to a specified terminal.</td></tr>
<tr><td>ICHNG</td><td></td><td>IEDQET</td><td>Perform TCAM operator control functions from an application program without using PUT.</td></tr>
<tr><td>ICOPY</td><td></td><td>IEDQE4</td><td>Copy the invitation list for a line group into a work area in an application program.</td></tr>
<tr><td>INBUF</td><td>
<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>Index to IEDQAE</td><td>Parameter List Length - X'04'</td><td>Option Field Offset</td><td>Register 15 Offset</td></tr>
</table>
</td><td>IEDQUI
↕
IEDQAE</td><td>Activate an MH routine.
Calculate the address of an option field from its index.</td></tr>
</table>

Macro	Parameter List	Linkage	Function

INEND

0	1
Index to IEDQA4	Parameter List Length - X'02'

Linkage: IEDQA4

If INMSG is not coded, the INEND macro generates the INMSG parameter list.

If INMSG is coded, the INEND macro generates X'0100', which indicates the end of the INMSG subgroup.

INHDR

	0	1	2	3
	Index to IEDQAE	Parameter List Length - X'04'	Option Field Offset	Register 15 Offset
4	Index to IEDQAB	Length X'04'	X'00'	X'00'

Linkage	Function
IEDQUI	Activate an MH routine.
→ IEDQAE	Calculate the address of an option field from its index.
IEDQAB	Check for translation needed or for a multiple-buffer header.
IEDQUI	Activate an MH routine.
→ IEDQAW	Translate the buffer.
→ MH macros	Continue header processing.

INITIATE

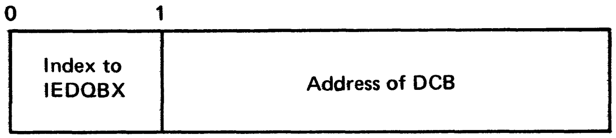
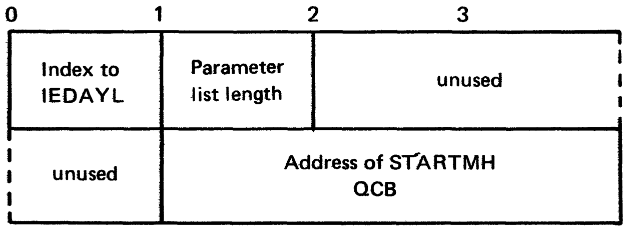
	0		1	2	3
	Index to IEDQAI	Bit	Parameter List Length - X'08'	Register 15 Offset	Length
4	Blank Character		Address of Characters		

Bit 7 No blank character is specified

Linkage	Function
IEDQUI	Activate an MH routine.
IEDQAI	Move the scan pointer as requested, or find and return to the user the next field beyond the scan pointer.

Macro	Parameter List	Linkage	Function
INMSG		IEDQUI	Activate an MH routine.
	0: Index to IEDQAE 1: Parameter List Length - X'04' 2: Option Field Offset 3: Register 15 Offset	→ IEDQAE	Calculate the address of an option field from its index.
		→ IEDQAK	Insert line-control characters in a message to be sent.
	0: Index to IEDQAK 1: Parameter List Length - X'04' 2: Index to IEDQAF 3: Index to IEDQAO	IEDQUI	
		→ IEDQAO	Get a buffer unit.
		→ IEDQAF	Insert data in the buffer.
		→ IEDQA4	
	0: Index to IEDQA4 1: Parameter List Length - X'02'		
INTRO		IEDQOA	Sort the termname table entries, scramble the password, attach any specified tasks, obtain main storage for the TCAM work areas.
		↕ IEDQOB	Allow the user to alter certain TCAM system parameters through the system console.
INVLIST	No executable code is generated.		

Macro	Parameter List	Linkage	Function
LOCK	0 Index to IEDQBE 1 Bit Bit 7 Message lock	IEDQUI	Activate an MH routine.
		↕ IEDQBE	Lock the connection between the terminal and an application program.
		↕ IEDQTNT	Get the terminal entry address.
LOCOPT	0 Index to IEDQAE 1 Parameter List Length - X'04' 2 Option Field Offset 3 Register Offset	IEDQUI	Activate an MH routine.
		↕ IEDQAE	Calculate an option field address.
		↕ IEDQTNT	Get the terminal entry address.
LOG (INMSG/OUTMSG)	0 Index to IEDQBY and Bits 1 Parameter List Length - X'08' 2 X'00' 3 X'00' 4 Address of LOGTYPE Entry	IEDQA4	
		↕ (Tpost) IEDQBD	Return any unused buffers, execute the INMSG/OUTMSG macro expansions, and check the parameter list.
		↔ IEDQTNT	Get the terminal entry address.
		↕ IEDQBY	Tpost a recalled header to the destination QCB and activate the Log Scheduler.

Macro	Parameter List	Linkage	Function
LOG	(INHDR/OUTHDR, INBUF/OUTBUF) 	IEDQUI	Activate an MH routine.
		IEDQBX	Log a message segment.
		OS BSAM WRITE	Write the units of the buffers.
		OS BSAM CHECK	Check the WRITE operation.
		OS GETMAIN	Get main storage.
LOGON (TSO/TCAM)	 0 1 2 3 Index to IEDAYL Parameter list length unused unused Address of STARTMH QCB	IEDQUI	Activate an MH routine.
		IEDAYL	Inform TSO of a successful user log on and route the messages.
		IEDQTNT	Get the terminal entry address.
		QTIP SVC	Connect to a TSO or TCAM MH.
		BR14	Exit to LOGON exit address in STARTMH QCB of connected MH.
LOGTYPE	No executable code is generated.		
MCOUNT		IEDQB1	Supply the count of complete messages for an application program.

<table>
<tr><th>Macro</th><th>Parameter List</th><th>Linkage</th><th>Function</th></tr>
<tr><td>MCPCLOSE</td><td></td><td>IEDQET</td><td>Perform a subset of the TCAM operator control functions without issuing a PUT macro instruction.</td></tr>
<tr><td></td><td></td><td>(SVC 102)
→ IEDQEB</td><td>Move data across partition boundaries and post ECBs complete.</td></tr>
<tr><td></td><td></td><td>→ OS WAIT</td><td>Put the application program into the wait state.</td></tr>
<tr><td></td><td></td><td>→ IEDQE6</td><td>Scramble the password.</td></tr>
<tr><td>MHGET</td><td>
<table>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td></td><td>MHGET ID X'00'</td><td>Parameter List Length</td><td>Work Area Register</td><td>Answer Register</td></tr>
<tr><td>4</td><td colspan="4">Address of Work Area (If Specified)</td></tr>
</table>
</td><td>IEDQGP</td><td>Make data in buffer available to user routine.</td></tr>
<tr><td>MHPUT</td><td>
<table>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td></td><td>MHPUT ID X'01'</td><td>Parameter List Length</td><td>Work Area Register</td><td>User Reserve Count</td></tr>
<tr><td>4</td><td colspan="4">Address of Work Area (If Specified)</td></tr>
</table>
</td><td>IEDQGP</td><td>Move data from user work area to buffer.</td></tr>
</table>

Macro	Parameter List	Linkage	Function
MRELEASE		IEDQET (SVC 102)	Perform a subset of the TCAM operator control functions without issuing a PUT macro instruction.
		→ IEDQEB	Move data across partition boundaries and post ECBs complete.
		→ OS WAIT	Put the application program into the wait state.
		→ IEDQE6	Scramble the password.

Macro	Parameter List	Linkage	Function
MSGEDIT		IEDQUI	Activate an MH routine.

	0	1	2	3
	Index to IEDQAN	Parameter List Length	Index to IEDQAF	Index to IEDQAO
4	Index to IEDQAJ	Blank Character	Number of Entries	Reserved
8	Reserved	Address of the Characters Table		
12	Key	Status	Data Description	
16	"FROM" Delimiter Description		"TO" Delimiter Description	
20	A Total of 31 Entries			

Status
0 Data=characters
1 Data=idles (reserve characters)
2 Data=CONTRACT
3 TO=character string
4 TO=offset or SCAN
5 TO=count
6 Inclusive FROM
7 Inclusive TO

Linkage		Function
IEDQUI ↕ IEDQAN		Translate and test all the data in a buffer.
IEDQAN →	IEDQAF	Insert or shift data in a buffer.
IEDQAN →	IEDQAX	Scan for a *TO* delimiter character string.
IEDQAN →	IEDQAO	Get an additional buffer for the insert function.
IEDQAN →	IEDQAL	Find the address of the character string.
IEDQAN →	IEDQTNT	Get the terminal entry address.

Macro	Parameter List	Linkage	Function
MSGEDIT	Data=REPLACE,TO=character string:	IEDQUI	Activate an MH routine.
		IEDQAP	Remove and optionally replace data from a single specified location in a buffer.
		→ IEDQAF	Shift the logically empty area to the end of the buffer.
		→ IEDQAX	Scan for the *TO* delimiter character string.
		→ IEDQAO	Insert replacement data in the buffer.

	0	1	2	3
0	Index to IEDQAP and Bit	Status	Index to IEDQAF	Index to IEDQAO
4	Insert Data			
8	Index to IEDQAJ	Parameter List Length	X'00'	Register 15 Offset
12	Blank Character (optional)	Character String Address		
16	"AT" Offset (optional)			

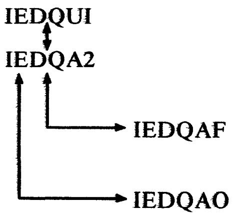
,Bit 7 ON - remove at the scan pointer
OFF - remove at the specified offset

Status See the MSGEDIT parameter list for IEDQAN

Insert Data
Characters: length of the character string (1 byte),address of the character string (3 bytes)
Idles: number of idles (1 byte),idle character (1 byte),X'0000'

Macro	Parameter List	Linkage	Function
MSGEDIT	Data=REPLACE,TO=extent or offset:	IEDQUI	Activate an MH routine.
		IEDQA2	Insert data in the message buffer.
		IEDQAF	Shift the logically empty space to the end of the buffer.
		IEDQAO	Get a buffer unit and insert the data in the buffer.

	0	1	2	3
0	Index to IEDQA2 and Bit	Status	Index to IEDQAF	Index to IEDQAO
4	Insert Data			
8	"TO" Extent or Offset		"AT" Offset (optional)	



Bit See the first MSGEDIT parameter list for IEDQAP

Status See the MSGEDIT parameter list for IEDQAN

Insert Data: See the first MSGEDIT parameter list for IEDQAP

Macro	Parameter List	Linkage	Function

MSGEDIT Data=CONTRACT,TO=character string:

	0	1	2	3
0	Index to IEDQAP and Bit	Status	Index to IEDQAF	Index to IEDQAO
4	Index to IEDQAJ	X'08'	X'00'	Register 15 Offset
8	Blank Character (optional)	Character String Address		
12	"AT" Offset (optional)			

Bit See the first MSGEDIT parameter list for IEDQAP

Status See the MSGEDIT parameter list for IEDQAN

MSGEDIT Data=CONTRACT,TO=extent or offset:

	0	1	2	3
0	Index to IEDQAP and Bit	Status	Index to IEDQAF	Index to IEDQAO
4	"TO" Extent or Offset		"AT" Offset (optional)	

Bit See the first parameter list for IEDQAP

Status See the parameter list for IEDQAN

Macro	Parameter List	Linkage	Function

MSGEDIT coded in INBLOCK:

	0	1	2	3
	Index to IEDQBN	Parameter List Length	Index to IEDQAF	Index to IEDQAO
4	QCB			

A second parameter list will follow depending on the MSGEDIT function asked for.

Linkage	Function
IEDQUI	Activate an MH routine.
↕ IEDQBN	
→ IEDQAF	Shift data in the buffer.
IEDQUI	
→ IEDQAO	Get a buffer unit and insert the data.

MSGEDIT Offset specified:

	0	1	2	3
	Index to IEDQA2 and Bit	Parameter List Length - X'0A'	Index to IEDQAF	Index to IEDQAO
4	Insert Data			
8	"AT" Offset			

Bit 7 ON - Data=idles
OFF - Data=characters

Insert Data: See the first MSGEDIT parameter list for IEDQAP

Linkage	Function
IEDQUI	Activate an MH routine.
↕ IEDQA2	Insert a data string in the message.
→ IEDQAE	Find the option field address.
→ IEDQAF	Shift left in the buffer.
→ IEDQAO	Get a buffer unit and insert the character string.

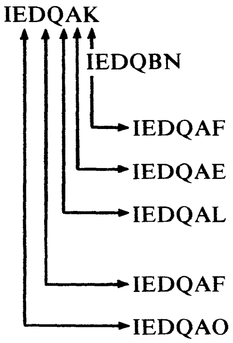
Macro **Parameter List** **Linkage** **Function**

MSGFORM (INBLOCK)

	0	1	2	3	
	Index to IEDQAK and Bit	Parameter List Length	Block Size		
4	Index to IEDQBN	X'00'	Index to IEDQAF	Index to IEDQAO	
8	QCB				
12	Character String Length	Character String Address			*
	Index to IEDQAE	Parameter List Length	Option Offset	Register	**

* This format is used if characters are specified.
** This format is used if an option field is specified.

Bit 6 ON - An option field is specified

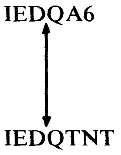


Linkage	Function
IEDQAK	Check and delete characters.
IEDQBN	Scan the queue for data to be inserted.
IEDQAF	Insert data in the buffer.
IEDQAE	Locate the option field.
IEDQAL	Scan the buffer at the specified offset.
IEDQAF	Shift data in the buffer.
IEDQAO	Get a buffer unit.

MSGFORM SUBBLOCK, BLOCK, or no operands:
(OUTHDR)

	0	1	2
	Index to IEDQA6	Parameter List Length	Block*
4	SUBBLOCK*		

*Optional



Linkage	Function
IEDQA6	Initialize fields in the SCB to indicate intervals between line-control characters.
IEDQTNT	Get the terminal entry address.

Macro	Parameter List	Linkage	Function

MSGFORM ENDCHAR and COUNT operands:

0	1	2	3
Index to IEDQA6	Parameter List Length	Character Field	Count Field

Linkage	Function
IEDQA6	Initialize fields in the SCB to indicate intervals between the line-control characters.
IEDQTNT	Get the terminal entry address.

MSGFORM issued in the OUTHDR subgroup:

0	1	2	3
Index to IEDQAK	Parameter List Length	Index to IEDQAF	Index to IEDQAO

Linkage	Function
IEDQAK	Insert line-control characters.
IEDQUI	Activate an MH routine.
IEDQAO	Get a buffer unit.
IEDQAF	Insert data in the buffer unit.

MSGGEN (INMSG/OUTMSG)

	0	1	2	3
	Index to IEDQBL	Parameter List Length	Status	Mask
4	Mask			
8	Address of Message			
12	Address of Table			

Status X'80' Use the table in the DCB
X'00' Use the specified table

Linkage	Function
IEDQA4	
IEDQAL	Get the scan pointer address.
IEDQAX	Scan for a specified character.
IGG019RB	Tpost empty units to the buffer return QCB.
IEDQTNT	Get the terminal entry address.
IEDQBD	Return any unused buffers, execute the INMSG/OUTMSG macro expansions, and check the parameter list.
IEDQTNT	Get the terminal entry address.
IEDQBL	Find a message, move it to the SCB, and translate it to the appropriate line code.

Macro	Parameter List	Linkage	Function
MSGGEN (TSO)		IEDAYM	Process a message.
		IEDQTNT	Get the terminal entry address.
		IEDQAE	Locate the translation table option field.
		IEDAYE	Edit the message.
MSGLIMIT		IEDQUI	Activate an MH routine.
		IEDQAE	Calculate an option field address.
		IEDQTNT	Get a terminal entry address.
		IEDQA6	Limit the number of messages to or from a terminal.

0	1	2	3
Index to IEDQAE	Parameter List Length	Option Field Offset	Register 15 Offset

0	1
Reserved	Limit

Macro	Parameter List	Linkage	Function
MSGTYPE where conchars is specified		IEDQUI	Activate an MH routine.
		IEDQAI	Move the scan pointer forward.

	0	1	2	3
	Index to IEDQAI and Bit	Parameter List Length	Register 15 Offset	Length
4	Blank Character	Address of Characters		

Bit 7 No blank character is specified

Macro	Parameter List	Linkage	Function

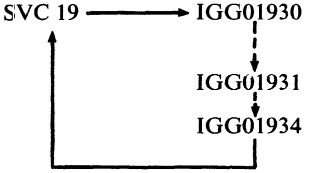
MSGTYPE where TABLE=EXIT is specified

0	Address of TABLE			
4	Address of EXIT			
8	Index to IEDQAI and Bit	Parameter List Length	Register 15 Offset	Length of Scan (always 1)

Macro	Parameter List	Linkage	Function
OPEN (Application Program)			
		SVC 19 → IGG01946	Establish a data transfer communication link.
		IGG01947	
		→ IEDQUI (SVC 102)	Activate an MH routine.
		→ IEDQEB	Tpost a special element to the ready queue in the MCP.
		→ IEDQNB05	Take an MCP checkpoint.
		→ GETMAIN	Get main storage for the DEB.
		→ OS LOAD	Load the appropriate module.
		→ OS WAIT	Wait for the Open/Close subtask to complete.
		IEDQEU	Allocate main storage for the application program.
		→ IEDQEB	Post the application program ECB.
		→ GETMAIN	Get main storage for an LCB and the process entry work area.
		→ FREEMAIN	Free main storage for an LCB and the process entry work area.
		→ OS LOAD	Load the appropriate scheduler.

Macro	Parameter List	Linkage	Function
OPEN (Checkpoint - Cold Restart)			
		SVC 19 → IGG01930	Determine the device type used for message queues.
		IGG01931	Build and initialize I/O blocks.
		IGG01934	Perform the disabled initialization functions.
		IGG01941	Open a checkpoint data set.
		IGG01949	Determine the size of records for the checkpoint data set.
		IGG01942	Initialize the disk checkpoint data set into specific areas.
		GETMAIN	Get main storage for DEBs and IOBs.
		FREEMAIN	Release main storage for all DECBs.
		OS LOAD	Load TCAM modules.

Macro	Parameter List	Linkage	Function
OPEN (Checkpoint - Warm or Continuation Restart)		SVC 19 → IGG01930	Determine the device type used for message queues.
		IGG01931	Build and initialize I/O blocks.
		IGG01934	Perform disabled initialization functions.
		IGG01941	Open a checkpoint data set.
		IGG01943	Reconstruct the MCP environment.
		IGG01944	Update the MCP environment from the incident records.
		IGG01945	Process the message queues data set at restart time.
		IGG019Q8	(An extension of IGG01945)
		IGG019RC	Set the absolute disk address and start disk I/O on the line.
OPEN (Line Group)		SVC 19 → IGG01935	Build and initialize a line DEB.
		IGG01936	Determine the size of channel programs.
		IGG01937	Build and initialize the LCBs.
		IGG01938	Build channel programs in the LCBs.
		IGG01939	Load the required modules.
		IGG01940	Load the required modules.
		IGG01948	Place line-specified information in the cross-reference table.

Macro	Parameter List	Linkage	Function
OPEN (Message Queues)		 SVC 19 → IGG01930	Determine the device type used for the message queues.
		IGG01931	Build and initialize I/O blocks.
		IGG01934	Perform disabled initialization functions.
OPTION	This macro generates no executable code.		

Macro	Parameter List	Linkage	Function

ORIGIN

	0	1	2	3
	Index to IEDQAM	Parameter List Length-X'06'	Index to IEDQAI and Bit	Parameter List Length-X'04'
4	Blank Character	Length of Field		

Bit 7 No blank character is specified

Linkage	Function
IEDQUI	Activate an MH routine.
IEDQAM	Verify origin or initialization.
IEDQTNT	Get the terminal entry address.
IEDQUI	Activate an MH routine.
IEDQAI	Move the scan pointer forward.
IEDQA1	Search the termname table for a match.

If a Concentrator is specified:

	0	1	2	3
	Index to IEDQBM and Bits	Parameter List Length-X'06'	Index to IEDQAI	Parameter List Length-X'04'
4	Zero	Length of Field		

BIT 6 FORM=ID is specified
7 FORM=NAME is specified

Linkage	Function
IEDQUI	Activate an MH routine.
IEDQBM	Verify origin or initialization. If FORM=ID is specified, search the device ID table for a match.
IEDQTNT	Get the terminal entry address.
IEDQUI	Activate an MH routine.
IEDQAI	Move the scan pointer forward.
IEDQA1	Search the termanme table for a match. (Entered only if the FORM=NAME or no FORM parameter is specified.)

OUTBUF

0	1	2	3
Index to IEDQAE	Parameter List Length - X'04'	Option Field Offset	Register 15 Offset

Linkage	Function
IEDQUI	Activate an MH routine.
IEDQAE	Calculate the option field address.
IEDQTNT	Get the terminal entry address.

Macro **Parameter List** **Linkage** **Function**

OUTEND

0	1	2	3
Index to IEDQAK	Parameter List Length - X'04'	Index to IEDQAF	Index to IEDQAO

If OUTMSG is not coded, OUTEND also generates the OUTMSG parameter list.

IF OUTMSG is coded, OUTEND generates a X'0100', which indicates the end of the OUTMSG subgroup.

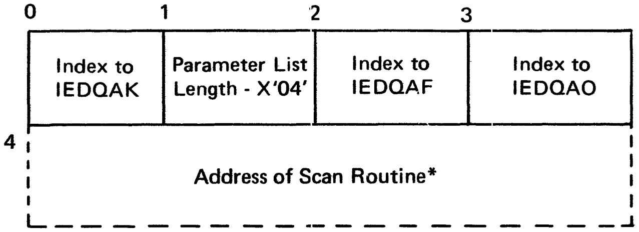
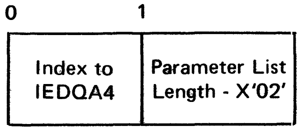
0	1
Index to IEDQA4	Parameter List Length - X'02'

OUTHDR

0	1	2	3
Index to IEDQAE	Parameter List Length - X'04'	Option Field Offset	Register 15 Offset

Linkage	Function
IEDQUI	Activate an MH routine.
IEDQAE	Calculate the option field address.
IEDQTNT	Get the terminal entry address.

Macro	Parameter List	Linkage	Function
OUTMSG	0 Index to IEDQAE 1 Parameter List Length - X'04' 2 Option Field Offset 3 Register 15 Offset	IEDQUI	Activate an MH routine.
		IEDQAE	Calculate the option field address.
		IEDQTNT	Get the terminal entry address.
	0 Index to IEDQGH and Flags 1 Parameter List Length	IEDQUI	Activate an MH routine.
	Parameter List Length: X'04' - No MSGFORM is specified X'08' - MSGFORM without ENDCHAR and COUNT is specified X'C0' - MSGFORM with ENDCHAR and COUNT is specified	IEDQGH	Insert the CTB end characters, determine the CTB end and concentrator end-of-message. Link the buffers in the chain.
		IEDQTNT	Get the terminal entry address.
	The parameter list length is the sum of: 1. The IEDQGH parameter list 2. The IEDQAK parameter list 3. The IEDQA4 parameter list	IEDQUI	Activate an MH routine.
	Flag: X'01' Entry is from OUTMSG	IEDQAO	Get a buffer unit.
		IEDQAF	Insert the CTB ending character.
		IGG019RB	Return the excess buffers and tpost the ERB to the concentrator scheduler. Mark the message serviced.
		IEDQUI	Activate an MH routine.
		IEDQA4	
		IGG019RB	Exit

Macro	Parameter List	Linkage	Function
OUTMSG	 0 Index to IEDQAK 1 Parameter List Length - X'04' 2 Index to IEDQAF 3 Index to IEDQAO 4 Address of Scan Routine*	IEDQUI	Activate an MH routine.
		IEDQAK	Check and insert line-control characters.
		IEDQAF	Insert data in the buffer.
		IEDQAO	Insert line-control characters in the buffer.
	*Present if ENDCHAR and COUNT are specified on MSGFORM in OUTHDR.	IEDQTNT	Get the terminal entry address.
		IEDQAL	Get the data byte address.
	 0 Index to IEDQA4 1 Parameter List Length - X'02'	IEDQUI	Activate an MH routine.
		IEDQA4	
		IEDQAL	Get the scan pointer address.
		IEDQAX	Scan for the specified character.
		IGG019RB	Tpost empty units to the buffer.
		IEDQTNT	Get the terminal entry address.
		IEDQGD* or IEDQGT	Build CCWs in the buffer and link the buffers in the idles loop for transparent output to a BSC terminal.
			*For concentrator support, IEDQGD builds the CCWs and returns to IEDQGH.

<table>
<tr><th>Macro</th><th>Parameter List</th><th>Linkage</th><th>Function</th></tr>
<tr><td>PATH</td><td><table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td>Index to IEDQAI and Bit</td><td>Parameter List Length - X'08'</td><td>Register 15 Offset</td><td>Variable Length</td></tr>
<tr><td>4
Blank Character</td><td colspan="3">Address of Characters</td></tr>
</table>
Bit 7 No blank character is specified</td><td>IEDQUI
IEDQAE
IEDQTNT</td><td>Activate an MH routine.
Calculate the option field address.
Get the terminal entry address.</td></tr>
<tr><td></td><td><table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td>Index to IEDQAE</td><td>Parameter List Length - X'04'</td><td>Option Field Offset</td><td>Register 15 Offset</td></tr>
</table></td><td>IEDQUI
IEDQA1</td><td>Activate an MH routine.
Search a table that is arranged in collating sequence.</td></tr>
<tr><td>PCB</td><td colspan="3">No executable code is generated. This is an application program work area.</td></tr>
<tr><td>POINT</td><td></td><td>IGG019RM
IEDQUI
IEDQAI</td><td>Build a message retrieval control block.
Activate an MH routine.
Scan the termname table for the specified terminal name.</td></tr>
</table>

Macro	Parameter List	Linkage	Function
PRIORITY		IEDQUI	Activate an MH routine.
		↕ IEDQAI	Search a table that is arranged in collating sequence.
PUT		IGG019RI	Prepare data for transfer into buffers in the MCP.
		→ IEDQE8	Scan the termname table.
		→ OS WAIT	Allow the Put Scheduler in the MCP to empty the application program PUT/WRITE work area.
		→ IEDQEB	Tpost a special element to the Put Scheduler in the MCP.
		→	User checkpoint exit routine.
		↕ IEDQEC	Move data from the application program into MCP buffers.
		→ IEDQEB	OS POST the application program ECB as complete.

	0	1	2	3
	Index to IEDQAI and Bit	Parameter List Length - X'08'	Register 15 Offset	Length
4	Blank Character	Address of Characters		

Bit 7 No blank character is specified

Macro	Parameter List	Linkage	Function
PUT (Compatible QTAM)		IGG019RJ	Prepare data for transfer into MCP buffers.
		IEDQE8	Scan the termname table for the specified terminal name.
		OS WAIT	Allow the Put Scheduler in the MCP to empty the application program PUT/WRITE work area.
		IEDQEB	Tpost a special element to the Put Scheduler in the MCP.
		IEDQEC	Move data from the application program to MCP buffers.
		IEDQEB	OS POST the application program ECB as complete.
QACTION (INHDR)		IEDQUI	Activate an MH routine.
		IEDQBO	
		IEDQUI	Activate an MH routine.
		IEDQA1	Get the termname table offset.
		IEDQTNT	Get the terminal entry address.
		IGG019RB	Tpost to IEDQBD to execute the OUTMSG macro. Exit and tpost the buffer to the queue.
		IGG019Q9	Put the attached terminal on the data ready queue.
QCOPY		IEDQE2	Copy a queue control block into a work area.
		IEDQE8	Scan the termname table for the specified terminal name.

	0	1	2
	Index to IEDQBQ	Parameter List Length	Reserved
4	Reserved	Address of User-Written Status Analysis Exit	

Macro	Parameter List	Linkage	Function
QSTART	No executable code is generated.		
READ		IGG019RG	Read data from a full buffer into the program area.
		→OS WAIT	Wait for the data to arrive.
		→IEDQEB	Tpost a special element to the Get Scheduler.
		→	User checkpoint exit routine.
		IEDQEW	Read from the message queues data set.
		→IEDQEB	OS POST the application program ECB as complete to activate the waiting application program.

Macro	Parameter List	Linkage	Function
READY		IEDQND	Read and process checkpoint records or update the TRMSTATE and option fields. Or move data into the operator control work area.
		OS ATTACH	Attach the Checkpoint Executor and on-line test.
		GETMAIN	Get main storage needed by on-line test.
		FREEMAIN	Free the main storage acquired by the GETMAIN SVC.
		IEDQTNT	Obtain the terminal entry address.
		OS EXCP	Start an I/O operation.
		OS LOAD	Load a TCAM module.
		OS POST	Post an ECB.
		OS WTO	Send an operator message.
		OS WAIT	Allow time to complete the event.
		IECPCNVT	Convert the TTR to an MBBCCHHR address.

Macro	Parameter List	Linkage	Function
REDIRECT	(INMSG/OUTMSG)		

	0	1	2	3
0	Index to IEDQAZ and Bits	Parameter List Length and Logical	Status	Mask
4	Mask			
8	Destination Status	Variable Data		

Bit 6 Recall is necessary
7 An unconditional mask is specified

Bit 6 Recall is necessary

Logical AND the mask

Status X'01' indicates that the IEDQAT parameter list follows

Destination Status and Variable Data:
C'S'+AL3(0) - send to the source
C'D'+AL3(0) - send to the destination
C'N'+AL3(destination name) - send to the named destination
C'O'+index to IEDQAE+AL1(optional offset) - send to the destination named in the option field

Linkage		Function
IEDQA4		
(Tpost)		
IEDQBD		Return any unused buffers, execute the INMSG/OUTMSG macro expansions, and check the parameter list.
	→ IEDQTNT	Get the terminal entry address.
IEDQAZ		Redirect a message to its destination.
IEDQUI		Activate an MH routine.
	→ IEDQAE	Get an option field address.
	→ IEDQAI	Get the destination key for the message.
	→ IEDQAV	Get the terminal entry for the destination.

Macro	Parameter List	Linkage	Function
RELEASEM		IEDQET	Perform a subset of the TCAM operator control functions without issuing a PUT.
		IEDQEB	Move data across partition boundaries and post ECBs as complete.
		OS WAIT	Put the into application program into a wait state.
		IEDQE6	Scramble the password.

0	Operator Control QCB Address			
4	Priority	Link Field		
8	Verb Code	Parameter List Length-X'1C'	X'00'	Return Code
12	ECB Address for the Application Program			
16	0			
20	0			
24	0			

Macro	Parameter List	Linkage	Function
RETRIEVE		IEDQES	Provide TCAM support for message retrieval.
		IEDQUI	Activate an MH routine.
		→ IEDQA1	Scan the termname table for the specified terminal name.
		→ OS WAIT	Wait for the requested buffer to be retrieved.
		→ IEDQEB	Tpost the special element to the retrieve scheduler QCB.
		IEDQE7	Retrieve a buffer from a disk message queue.
		→ IGG019RB	Tpost elements.
		→ IEDQEB	OS POST the Retrieve Service routine ECB as complete.
		→ GETMAIN	Get main storage for an LCB and an SCB.
		→ FREEMAIN	Release main storage for the LCB and SCB.
RETRY	0 Index to IEDQA9 1 Parameter List Length-X'04' 2 3 Interval	IEDQA4 (Tpost) IEDQBD IEDQA9	The RETRY macro causes the CPU to try again to initiate contact with a switched station at the expiration of a specified interval of time.

Macro	Parameter List	Linkage	Function
SCREEN		IEDQUI ↕	Activate an MH routine.
		IEDQAI	Move the scan pointer forward.

	0	1	2	3
	Index to IEDQAI and Bit	Parameter List Length - X'08'	Register 15 Offset	Variable Length
4	Blank Character	Address of Characters		

Bit 7 No blank character is specified

0	1	2
Index	Request Code	Flags

Linkage	Function
IEDQUI ↕	Activate an MH routine.
IEDQAY	Check the unit control block and initialize.
└→ IEDQTNT	Get the terminal entry address.

Index

Bits 0-6Index to IEDQAY

Bit 7 ON - indicates that the user specified one of the following: WRE, WLA, WOC, XRE, XLA, XDC, EAU. OFF - indicates that none of the above were specified.

Request Code

X'00'	WDC or no operand
X'01'	WLA
X'02'	WRE
X'03'	EAU
X'10'	XDC
X'11'	XLA
X'12'	XRE

Flag Byte

Bit 0 ON - RETRIEVE=YES
OFF - RETRIEVE=NO

Bits 1-7 reserved

Macro	Parameter List	Linkage	Function
SEQUENCE	0 Index to IEDQAD 1 Index to IEDQAF	IEDQUI	Activate an MH routine.
		IEDQAD	Insert the output sequence number.
		IEDQUI	Activate an MH routine.
		IEDQAF	Expand the buffer.
	Bit 7 No blank character is specified		
	0 Index to IEDQAH 1 Parameter List Length - X'02'	IEDQUI	Activate an MH routine.
		IEDQAH	Verify and update an input sequence number.
		IEDQTNT	Get a terminal entry address.
	0 Index to IEDQAI and Bit 1 Parameter List Length - X'08' 2 X'00' 3 Variable Length Indicator 4 Blank Character Address of Characters	IEDQUI	Activate an MH routine.
		IEDQAI	Move the scan pointer forward.

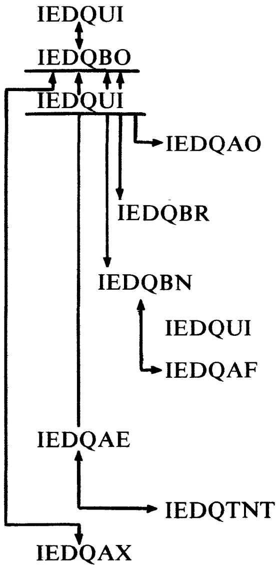
Macro	Parameter List	Linkage	Function
SETEOF		IEDQUI ↕	Activate an MH routine.
		IEDQAI	Move the scan pointer forward.

	0	1	2	3
	Index to IEDQAI and Bit	Parameter List Length - X'08'	Register 15 Offset	Variable Length
4	Blank Character	Address of Characters		

Bit 7 No blank character is specified

Macro	Parameter List	Linkage	Function
SETEOM			

	0	1	2	3
0	Index to IEDQBO	Parameter List Length	Status Byte	Index to IEDQAE
4	Index to IEDQBN	Index to IEDQBR	Index to IEDQAF	Index to IEDQAO
8	Hold QCB			
12*	Integer (LENGTH)		Opfld$_2$ of LENGTH	X'00'
16**	Length of ENDCHAR String	Address of ENDCHAR String		
20				
24	SETEOM QCB - expanded only if PROCESS=YES is specified			
28				



Linkage	Function
IEDQUI	Activate an MH routine.
IEDQBO	SETEOM control module.
IEDQUI	Activate an MH routine.
IEDQAO	Get and attach an additional buffer.
IEDQBR	Determine the logical message, if on the COUNT macro.
IEDQBN	Combine data.
IEDQUI	Activate an MH routine.
IEDQAF	Attach a buffer unit or shift data.
IEDQAE	Calculate the option field address.
IEDQTNT	Get the terminal entry address.
IEDQAX	Scan for an EOM string.

Status Byte:

X‘01’ - ENDCHAR is specified
X‘02’ - ENDCHAR is in the option field
X‘04’ - LENGTH is specified
X‘08’ - LENGTH is in the option field
X‘10’ - PROCESS=YES
X‘20’ - REMOVE=YES
X‘40’ - EOM = ETB
X‘80’ - reserved

Macro	Parameter List	Linkage	Function

SETEOM *If both LENGTH operands are in the option fields:

12	X'00'	Opfld$_1$ of LENGTH	Opfld$_2$ of LENGTH	X'00'

or X'00000000' if LENGTH is not specified

**If ENDCHAR is in the option field:

16	Opfld of ENDCHAR	X'000000'

or X'00000000' if ENDCHAR is not specified

SETSCAN

	0	1	2	3
	Index to IEDQAJ and Bit	Parameter List Length - X'08'	Status	Register Offset
4	Blank Character	Address of Characters		

Bit 7 No blank character is specified

Status X'00' - return the scan pointer
X'FF' - update the scan pointer

Linkage	Function
IEDQUI ↕	Activate an MH routine.
IEDQAJ	Move the scan pointer to the end of the character string.
└──► IEDQAX	Scan for a character string.

Macro	Parameter List	Linkage	Function
SETSCAN		IEDQUI ↕ IEDQAI	Activate an MH routine. Move the scan pointer forward.

0	1	2
Index to IEDQAI and Bit	Parameter List Length - X'06'	Skip Count
4		
Blank Character	X'00'	

Bit 7 No blank character is specified

0	1	2
Index to IEDQA0	Parameter List Length - X'04'	Skip Count

Macro	Parameter List	Linkage	Function
		IEDQUI ↕ IEDQA0	Activate an MH routine. Move the scan pointer backward.
SIMATTN (TSO)		IEDQUI ↕ IEDAYS	Activate an MH routine. Handle simulated attention for TSO.
		└→ IEDQTNT	Get the terminal entry address.
		(SVC 101) └→ QTIP	Turn the "QCB tposted" flag off and determine if TPUT is requested.

Macro	Parameter List	Linkage	Function
SLOWPOLL		IEDQAV	Assign buffer to destination
		IEDQBD	Return any unused buffers, execute INMSG/OUTMSG macro expansions, and check the parameter list.
		IEDQTNT	Obtain the terminal table address of an entry.
		IEDQB4	Suspend polling on error.
		IEDQHG	Insert LCB in time delay queue.
		IGG019RB	Exit.

	0	1	2	3
0	Index to IEDQB4	Parameter list length and logical		mask
4	mask			
8	Seconds Time Delay			

Bit 7 AND the mask.

	0	1	2	3
0	Index to IEDQB4 and Bits	Parameter list length		
4	Seconds Time Delay			

Macro	Parameter List	Linkage	Function
STARTLN (Compatible QTAM)		IEDQET	Perform a subset of the TCAM operator control functions without issuing a PUT macro.
		IEDQEB	Move data across partition boundaries and post ECBs complete.
		OS WAIT	Put the application program into a wait state.
		IEDQE6	Scramble the password.

STARTMH No executable code is generated.

Macro	Parameter List	Linkage	Function
STOPLN (Compatible QTAM)		IEDQET	Perform a subset of the TCAM operator control functions without issuing a PUT macro.
		IEDQEB	Move data across partition boundaries and post ECBs as complete.
		OS WAIT	Put the application program into a wait state.
		IEDQE6	Scramble the password.
TCHNG		IEDQE3	Update the contents of a terminal entry.
		IEDQE8	Scan the termname table for the specified terminal name.
		IEDQEB	Move data from the work area to the terminal entry.
		IEDQE6	Scramble the password.
		IEDQNB	Take a checkpoint of the MCP.
TCOPY		IEDQE1	Copy a terminal entry into a work area.
		IEDQE8	Scan the termname table.
TERMINAL	No executable code is generated.		
TERRSET	No executable code is generated.		

Macro	Parameter List	Linkage	Function
TGOTO	(INHDR/INBUF)	IEDQUI	Activate an MH routine.
		IEDQBP	Pass the buffer to the second MH, if necessary.
		IEDQTNT	Get the terminal entry address.
		IEDQUI	Activate an MH routine.
		IEDQAE	Get the option field address, if necessary.
		IGG019RB	Bypass to the STARTMH subtask for the second MH.

0	1	2	3
Index to IEDQBP	Parameter List Length-X'08'	Status	Reserved

Status X'00' - address of the MH is in the parameter list:

4	5
Reserved	Address of MH of STARTMH QCB

Status X'80' - address of the MH is in the option field:

4	5	6	7
Index to IEDQAE	Parameter List Length-X'04'	Option Field Offset	Register 15 Offset
8 Address of IEDAYR if mixed environment otherwise Address of IEDQAA			

TLIST	No executable code is generated.		
TPROCESS	No executable code is generated.		
TRANLIST	No executable code is generated.		
TSINPUT	No executable code is generated.		

Macro	Parameter List	Linkage	Function
TTABLE	IEDQTNT code is generated and placed at the beginning of the termname table.		
TYPETABL	No executable code is generated.		
UNLOCK	0 Index to IEDQBF 1 X'00'	IEDQUI	Activate an MH routine.
		IEDQBF	Unlock the currently connected terminal.
WRITE		IGG019RI	Prepare data for transfer into buffers in the MCP.
		IEDQE8	Scan the termname table for the specified terminal name.
		OS WAIT	Wait for the PUT/WRITE work area to empty.
		IEDQEB	Tpost a special element to the Put Scheduler in the MCP.
			User checkpoint exit routine.
		IEDQEC	Move data from the application program to MCP buffers.
		IEDQEB	Post the application program ECB as complete after the data in the application program work area has been transfered to the MCP buffers.

Operator Control Command Linkage Chart

Command	Command Format	Module	Function
DISPLAY	D TP,ACT,lineaddress	IGCD310D	If the indicated line is open and active, displays the names of all the active stations on the line.
	D TP,INACT,lineaddress	IGCD310D	If the indicated line is open and active, displays the names of all the inactive stations on the line.
	D TP,LINE,lineaddress	IGCD910D	If the indicated line is open and active, displays the contents of the status field (LCBSTATE) and of the message error record (SCBERRST) for the line.
	D TP,LIST,lineaddress	IGCD710D	If the indicated line is open and active, displays whether the invitation list for the line may be polled and whether the Auto Poll feature is being used to poll the list.
	D TP,PRITERM	IGCD110D	Displays the name of the current primary operator control station for the system.
	D TP,SECTERM	IGCD110D	Displays the names of all secondary operator control stations for the system (as defined by the SECTERM =YES operand of the TERMINAL and TPROCESS macro instructions).
	D TP,INTER	IGCD410D	Displays the names of all the stations in the system that are intercepted (that is, stations that can enter messages, but to which transmission of messages is suspended).
	D TP,ADDR,statname	IGCD610D	If the indicated station is part of the system, displays the name of the line group of which the station is a part, the relative line number within the line group on which the station is located, and the machine address of the line.
	D TP,QUEUE,statname	IGCD210D	If the indicated station is part of the system, displays the number of messages currently queued for the station, the type of queuing being used,and all permissible priority levels for messages to be sent to this station.
	D TP,TERM,statname	IGCD510D	If the indicated station is part of the system, displays the status of the station (TRMSTATE), the input and output sequence numbers, and the current intensive mode recording status.
	D TP,OPTION,statname,opfldname,format	IGCD810D	If the indicated station is part of the system and the indicated option field exists for the station, displays the contents of the option field in hexadecimal (X), character (C), or decimal (D) format.

Command	Command Format	Module	Function
HALT	Z TP,QUICK	IGCZ110D	Stops message traffic on each line in the system as soon as transmission of any message currently being sent or received,on the line is completed. Messages remaining in the system are sent to the appropriate destinationsafter TCAM is restarted. Messages for a status-analysis or a byte-count terminal attached to a concentrator are sent up to the first CTB; subsequent CTBs should be saved for restart.
	Z TP,FLUSH	IGCZ110D	Stops message transmission from each station in the system as soon as transmission of any message currently being sent is completed. All messagesare then sent before the system is halted. Intercepted messages that cannot be sent are transmitted to the appropriate destination after TCAM is restarted. Messages for a status-analysis or a byte-count terminal attached to a concentrator are sent up to the first CTB; subsequent CTBs should be savedfor restart.
HOLD	H TP=statname	IGCH010D	Suspends transmission of messages to the indicated station if the station is part of the system. The station is intercepted, but it can enter messages.
MODIFY	F identifier,OLT=message	IGC0710D	Begins the on-line test requested by the message in the indicated procedure or job.
	F identifier,DEBUG=L,routine	IGCM910D	In the indicated procedure or job, starts the TCAM service aid routine that writes the Dispatcher subtask trace table (IEDQFE10), the line I/Ointerrupt trace table (IEDQFE20), or the buffer trace table (IEDQFE30).
	F identifier,OPERATOR=statname	IGCM710D	In the indicated procedure or job, changes the station named from a secondary operator control station to the primary operator control station for the system.
	F identifier,INTERVAL=SYSTEM,value	IGCM410D	In the indicated procedure or job, changes the duration of the system interval to the specified decimal number of seconds (not to exceed 65,535).
	F identifier,INTERVAL=SYSTEM	IGCM410D	In the indicated procedure or job, causes the system to enter a delay for the duration of the currently-defined system interval.
	F identifier,INTERVAL=POLL,statname,value	IGCM410D	In the indicated procedure or job, changes the polling interval of the line group associated with the named station to the specified decimal number of seconds (not to exceed 255).

Command	Command Format	Module	Function
MODIFY	F identifier,INTENSE=LINE,lineaddress, sense[,sensecount]	IGCM510D	In the indicated procedure or job, records recoverable I/O errors of the type specified on the line named. The number recorded is that specified by *sensecount*.
	F identifier,AUTOPOLL=lineaddress,ON	IGCM210D	In the indicated procedure or job, changes the specified line from programmed poll to Auto Poll if the automatic polling bit is on in the UCB.
	F identifier,AUTOPOLL=lineaddress,OFF	IGCM210D	In the indicated procedure or job, changes the specified line from Auto Poll to programmed poll.
	F identifier,TRACE=lineaddress,ON	IGCM610D	In the indicated procedure or job, starts the line I/O interrupt trace facility for the specified line.
	F identifier,TRACE=lineaddress,OFF	IGCM610D	In the indicated procedure or job, stops the line I/O interrupt trace facility for the specified line.
	F identifier,INTENSE=TERM,statname, sense[,sensecount]	IGCM510D	In the indicated procedure or job, records recoverable I/O errors of the type specified for the station named. The number recorded is that specified by *sensecount*.
	F identifier,OPT=statname,opfldname,data	IGCM810D	In the indicated procedure or job, changes the contents of the specified option field for the station named to the specified data.
RELEASE	A TP=statname	IGCR010D	If the indicated station is a part of the system, releases that station from the intercepted state. Messages intended for the station, but not yet sent, can then be transmitted.

Command	Command Format	Module	Function
VARY	V statname,ONTP,E	IGCV410D	If the indicated station is part of the system and is nonswitched, activates the station for only entering messages.
	V statname,OFFTP,E	IGCV210D	If the indicated station is part of the system and is nonswitched, prevents the station from entering messages.
	V statname,ONTP,B	IGCV410D, IGCR010D	If the indicated station is part of the system and is nonswitched, activates the station for both accepting and entering messages.
	V statname,OFFTP,B	IGCV210D IGCH010D	If the indicated station is part of the system and is nonswitched, prevents the station from both accepting and entering messages.
	V lineaddress,ONTP	IGCV310D	If the indicated line or line group is open, begins or resumes message transmission on that line or line group.
	V lineaddress,OFFTP,C	IGCV110D	If the indicated line or line group is open and active, stops transmission of messages on that line or line group after the current message.
	V lineaddress,OFFTP,I	IGCV110D	If the indicated line or line group is open and active, immediately stops transmission of messages on that line or line group.
	V gpstatname,OFFTD,$\left\{\begin{matrix}E\\B\end{matrix}\right\}$	IGCV510D	Changes general poll invitation list entry from active to inactive status.
	V gpstatname,ONTP, $\left\{\begin{matrix}E\\B\end{matrix}\right\}$	IGCV610D	Changes general poll invitation list entry from inactive to active status.

ERP Linkage Charts

START-STOP ERP

Operation	Error	Module	Condition	Action
Any	Control unit not operational	IGE0204G		Issue WTO message IED064I. Set the permanent error flag. Exit to the Line End Appendage.
	Channel check	IGE0804G		Retry, if possible. Exit to IGE0504G.
READ or WRITE	Unit check	IGE0104G	Equipment check, lost data, busout, or intervention required	Retry, if possible. Exit to IGE0504G.
		IGE0204G	Time-out	If other than READ CCWs for other than non-TSO 2741 terminals, retryif possible or exit to IGE0504G for permanent error recording. If READ TEXT CCWs for non-TSO 2741 terminals, issue a WRITE BREAKand exit to IGE0504G for permanent error recording. For non-text READ CCWs for non-TSO 2741 terminals, issue WRITE BREAK under two conditions: 1. If the terminal is not in lock mode. 2. If there is output queued for the terminal. Exit to IGE0504G to clear the error status. If neither of the above conditions is met, restart I/O on the failing READ.
		IGE0604G	Audio or Local device	Retry, if possible.
	Unit exception	IGE0204G	Teletype adapter and non-TSO 2741 terminals	Issue a WRITE BREAK.
			2701 Control Unit	Issue a READ SKIP.
			Other	Retry, if possible. If the retry limit is reached, for TSO exit to IGE0504G. Otherwise, issue a READ SKIP and exit to IGE0504G.

Operation	Error	Module	Condition	Action
READ or WRITE	Unit exception	IGE0604G	Audio or Local device	Retry, if possible.
POLL	Unit check	IGE0404G	Time-out, busout, data check, or intervention required	Retry, if possible.
			Other	Exit to IGE0504G.
	Unit exception	IGE0404G		Retry, if possible. Exit to IGE0504G.
READ RESPONSE to Auto Poll	Unit check	IGE0404G	Time-out, data check, overrun, or lost data	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
	Unit exception	IGE0404G		Exit to Line End Appendage.
Closedown		IGE0904G		Put end-of-day statistics in the OBR/SDR file. Exit to Line End Appendage.
DIAL	Unit check	IGE0304G	Lost data, busout, time-out, orintervention required	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
PREPARE	Unit check	IGE0304G	Time-out or intervention required.	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
DISABLE or ENABLE	Unit check	IGE0304G	Time-out	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
Other	Unit check	IGE0304G		Exit to IGE0504G.
All	Permanent error	IGE0504G	System console is the primary operator control station	Exit to the OS Message Writer to indicate a permanent error condition.
			OBR/SDR recording is required	Record the error in the OBR/SDR file. Exit to the Line End Appendage.

BSC ERP

Operation	Error	Module	Condition	Action
Any	Control unit not operational	IGE0204G		Issue the WTO message IED064I. Set the permanent error flag. Exit to Line End Appendage.
	Channel check	IGE0804H		Retry, if possible. Exit to IGE0504H.
READ or WRITE	Unit exception	IGE0104H	Write ENQ	Restart I/O at the Read Response CCW.
			Other	Execute a READ SKIP channel program.
	Unit check	IGE0104H	Equipment check	Exit to IGE0504H.
			Intervention required	Exit to IGE0504H.
			Busout on command	Retry the channel program.
			Busout on data	Execute a READ RESPONSE channel program.
			Lost data on WRITE	Exit to IGE0504H.
			Lost data on READ RESPONSE to ENQ	Execute a WRITE ENQ channel program.
			Lost data on READ ENQ	Retry the channel program.
			Lost data on READ RESPONSE to text	Execute a WRITE ENQ, READ RESPONSE channel program.
			Lost data on READ TEXT	Execute a READ SKIP channel program.
	Command Reject	IGE0204H IGE0404H		Retry the channel program until the retry count is exhausted. Exit to IGE0504H.
	Data check or overrun	IGE0204H		Retry, if possible. Exit to IGE0504H.
	Other	IGE0204H		Exit to IGE0504H.
POLL	Unit check	IGE0404G	Time-out, busout, data check, or intervention required	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
	Unit exception	IGE0404G		Retry, if possible. Exit to IGE0504G.

Operation	Error	Module	Condition	Action
READ RESPONSE to POLL	Unit check	IGE0404G	Time-out, overrun, data check, or lost data	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
	Unit exception	IGE0404G		Exit to Line End Appendage.
Closedown		IGE0904G		Put end-of-day statistics in the OBR/SDR file. Exit to Line End Appendage.
DIAL	Unit check	IGE0304G	Lost data, busout, time-out, orintervention required	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
PREPARE	Unit check	IGE0304G	Time-out or intervention required	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
DISABLE or ENABLE	Unit check	IGE0304G	Time-out	Retry, if possible. Exit to IGE0504G.
			Other	Exit to IGE0504G.
Other	Unit check	IGE0304G		Exit to IGE0504G.
All	Permanent error	IGE0504G, IGE0504H	System console is the primary operator control station	Exit to the OS Message Writer to indicate a permanent error condition.
			OBR/SDR recording is required	Record a permanent error in the OBR/SDR file. Exit to Line End Appendage.

(This page left blank intentionally)

Chart FA-1 (FA1) CPB INITIALIZATION

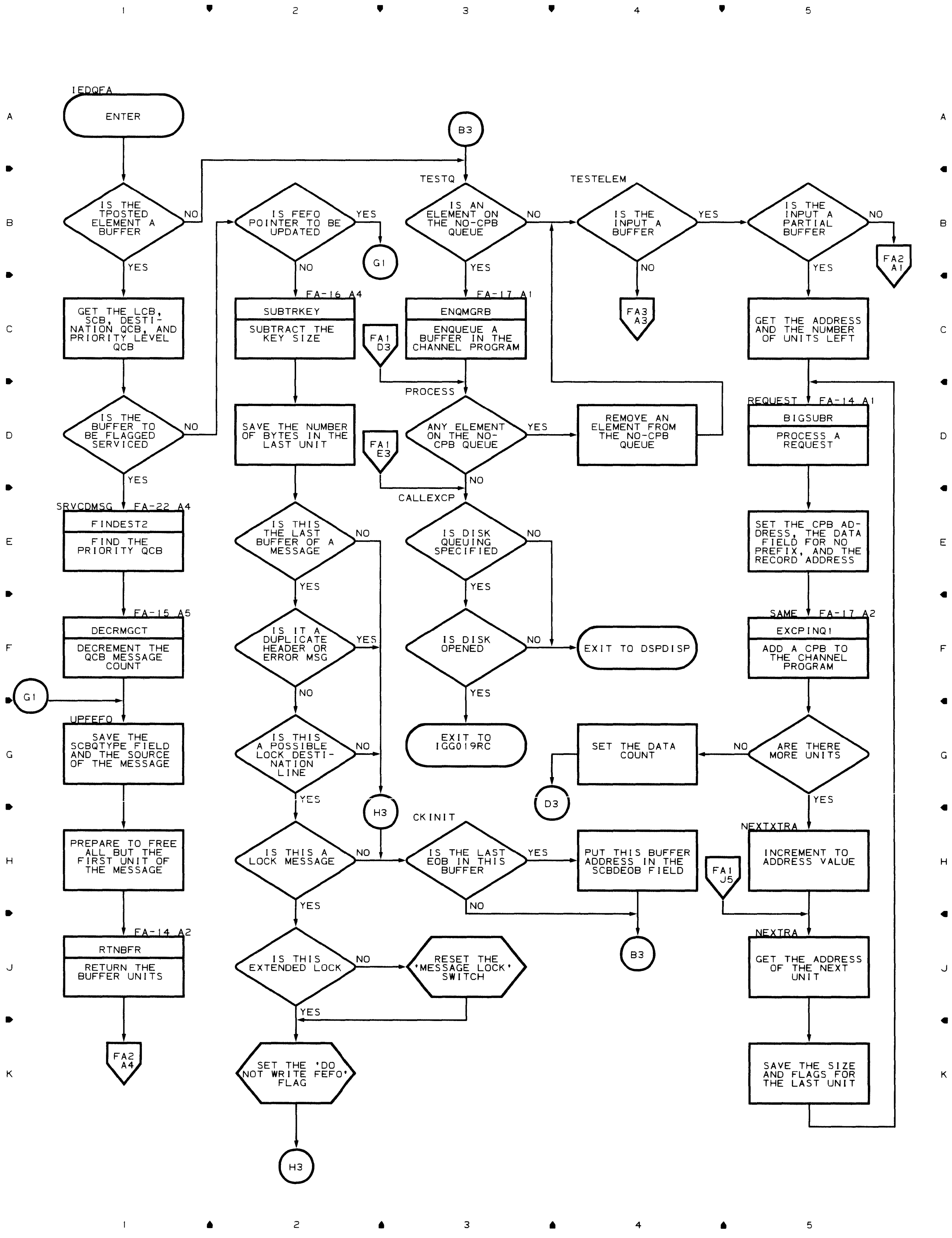
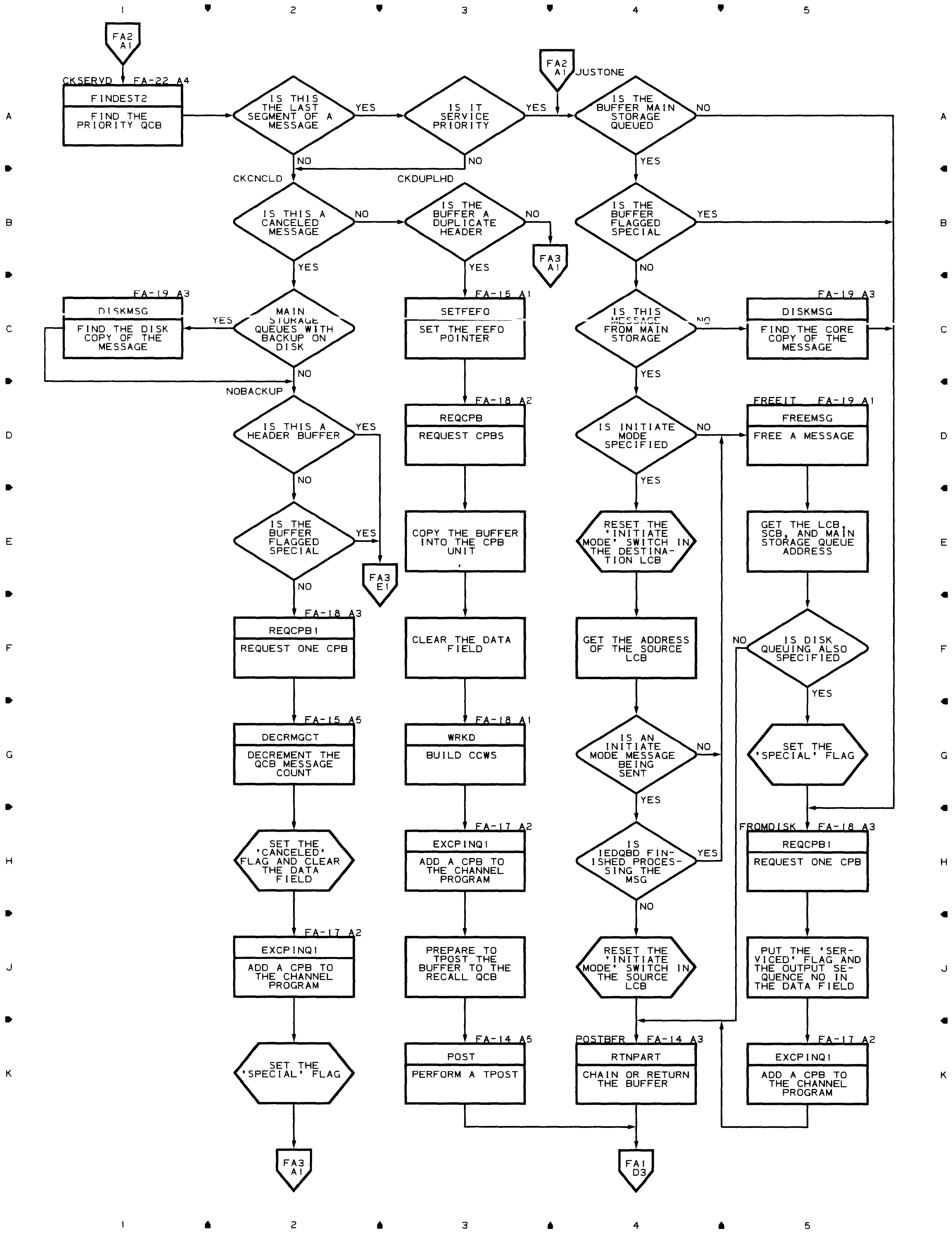

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
IEDQFA
ENTER
B3
TESTQ
TESTELEM
IS THE TPOSTED ELEMENT A BUFFER
NO
YES
IS FEFO POINTER TO BE UPDATED
YES
NO
G1
IS AN ELEMENT ON THE NO-CPB QUEUE
NO
YES
IS THE INPUT A BUFFER
YES
NO
IS THE INPUT A PARTIAL BUFFER
NO
YES
FA2 A1
GET THE LCB, SCB, DESTINATION QCB, AND PRIORITY LEVEL QCB
FA-16 A4
SUBTRKEY
SUBTRACT THE KEY SIZE
FA1 D3
FA-17 A1
ENQMGRB
ENQUEUE A BUFFER IN THE CHANNEL PROGRAM
FA3 A3
GET THE ADDRESS AND THE NUMBER OF UNITS LEFT
PROCESS
IS THE BUFFER TO BE FLAGGED SERVICED
NO
YES
SAVE THE NUMBER OF BYTES IN THE LAST UNIT
FA1 E3
ANY ELEMENT ON THE NO-CPB QUEUE
YES
NO
REMOVE AN ELEMENT FROM THE NO-CPB QUEUE
REQUEST
FA-14 A1
BIGSUBR
PROCESS A REQUEST
SRVCDMSG
FA-22 A4
FINDEST2
FIND THE PRIORITY QCB
CALLEXCP
IS THIS THE LAST BUFFER OF A MESSAGE
NO
YES
IS DISK QUEUING SPECIFIED
NO
YES
SET THE CPB ADDRESS, THE DATA FIELD FOR NO PREFIX, AND THE RECORD ADDRESS
FA-15 A5
DECRMGCT
DECREMENT THE QCB MESSAGE COUNT
IS IT A DUPLICATE HEADER OR ERROR MSG
YES
NO
IS DISK OPENED
NO
YES
EXIT TO DSPDISP
SAME
FA-17 A2
EXCPINQ1
ADD A CPB TO THE CHANNEL PROGRAM
G1
UPFEFO
SAVE THE SCBQTYPE FIELD AND THE SOURCE OF THE MESSAGE
IS THIS A POSSIBLE LOCK DESTINATION LINE
NO
YES
EXIT TO IGG019RC
SET THE DATA COUNT
NO
ARE THERE MORE UNITS
YES
D3
H3
CKINIT
NEXTXTRA
PREPARE TO FREE ALL BUT THE FIRST UNIT OF THE MESSAGE
IS THIS A LOCK MESSAGE
NO
YES
IS THE LAST EOB IN THIS BUFFER
YES
NO
PUT THIS BUFFER ADDRESS IN THE SCBDEOB FIELD
FA1 J5
INCREMENT TO ADDRESS VALUE
FA-14 A2
RTNBFR
RETURN THE BUFFER UNITS
IS THIS EXTENDED LOCK
NO
YES
RESET THE 'MESSAGE LOCK' SWITCH
B3
NEXTRA
GET THE ADDRESS OF THE NEXT UNIT
FA2 A4
SET THE 'DO NOT WRITE FEFO' FLAG
SAVE THE SIZE AND FLAGS FOR THE LAST UNIT
H3

Chart FA-2 (FA2) CPB INITIALIZATION



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FA2 A1
CKSERVD FA-22 A4
FINDEST2
FIND THE PRIORITY QCB
IS THIS THE LAST SEGMENT OF A MESSAGE
YES
NO
IS IT SERVICE PRIORITY
YES
NO
FA2 A1
JUSTONE
IS THE BUFFER MAIN STORAGE QUEUED
NO
YES
CKCNCLD
CKDUPLHD
IS THIS A CANCELED MESSAGE
NO
YES
IS THE BUFFER A DUPLICATE HEADER
NO
YES
FA3 A1
IS THE BUFFER FLAGGED SPECIAL
YES
NO
FA-19 A3
DISKMSG
FIND THE DISK COPY OF THE MESSAGE
YES
MAIN STORAGE QUEUES WITH BACKUP ON DISK
NO
NOBACKUP
FA-15 A1
SETFEFO
SET THE FEFO POINTER
IS THIS MESSAGE FROM MAIN STORAGE
NO
YES
FA-19 A3
DISKMSG
FIND THE CORE COPY OF THE MESSAGE
IS THIS A HEADER BUFFER
YES
NO
FA-18 A2
REQCPB
REQUEST CPBS
IS INITIATE MODE SPECIFIED
NO
YES
FREEIT FA-19 A1
FREEMSG
FREE A MESSAGE
IS THE BUFFER FLAGGED SPECIAL
YES
NO
FA3 E1
COPY THE BUFFER INTO THE CPB UNIT
RESET THE 'INITIATE MODE' SWITCH IN THE DESTINATION LCB
GET THE LCB, SCB, AND MAIN STORAGE QUEUE ADDRESS
FA-18 A3
REQCPB1
REQUEST ONE CPB
CLEAR THE DATA FIELD
GET THE ADDRESS OF THE SOURCE LCB
NO
IS DISK QUEUING ALSO SPECIFIED
YES
FA-15 A5
DECRMGCT
DECREMENT THE QCB MESSAGE COUNT
FA-18 A1
WRKD
BUILD CCWS
IS AN INITIATE MODE MESSAGE BEING SENT
NO
YES
SET THE 'SPECIAL' FLAG
FA-17 A2
EXCPINQ1
ADD A CPB TO THE CHANNEL PROGRAM
IS IEDQBD FINISHED PROCESSING THE MSG
YES
NO
FROMDISK FA-18 A3
REQCPB1
REQUEST ONE CPB
SET THE 'CANCELED' FLAG AND CLEAR THE DATA FIELD
FA-17 A2
EXCPINQ1
ADD A CPB TO THE CHANNEL PROGRAM
PREPARE TO TPOST THE BUFFER TO THE RECALL QCB
RESET THE 'INITIATE MODE' SWITCH IN THE SOURCE LCB
PUT THE 'SERVICED' FLAG AND THE OUTPUT SEQUENCE NO IN THE DATA FIELD
FA-14 A5
POST
PERFORM A TPOST
POSTBFR FA-14 A3
RTNPART
CHAIN OR RETURN THE BUFFER
FA-17 A2
EXCPINQ1
ADD A CPB TO THE CHANNEL PROGRAM
SET THE 'SPECIAL' FLAG
FA3 A1
FA1 D3

Chart FA-3 (FA3) CPB INITIALIZATION

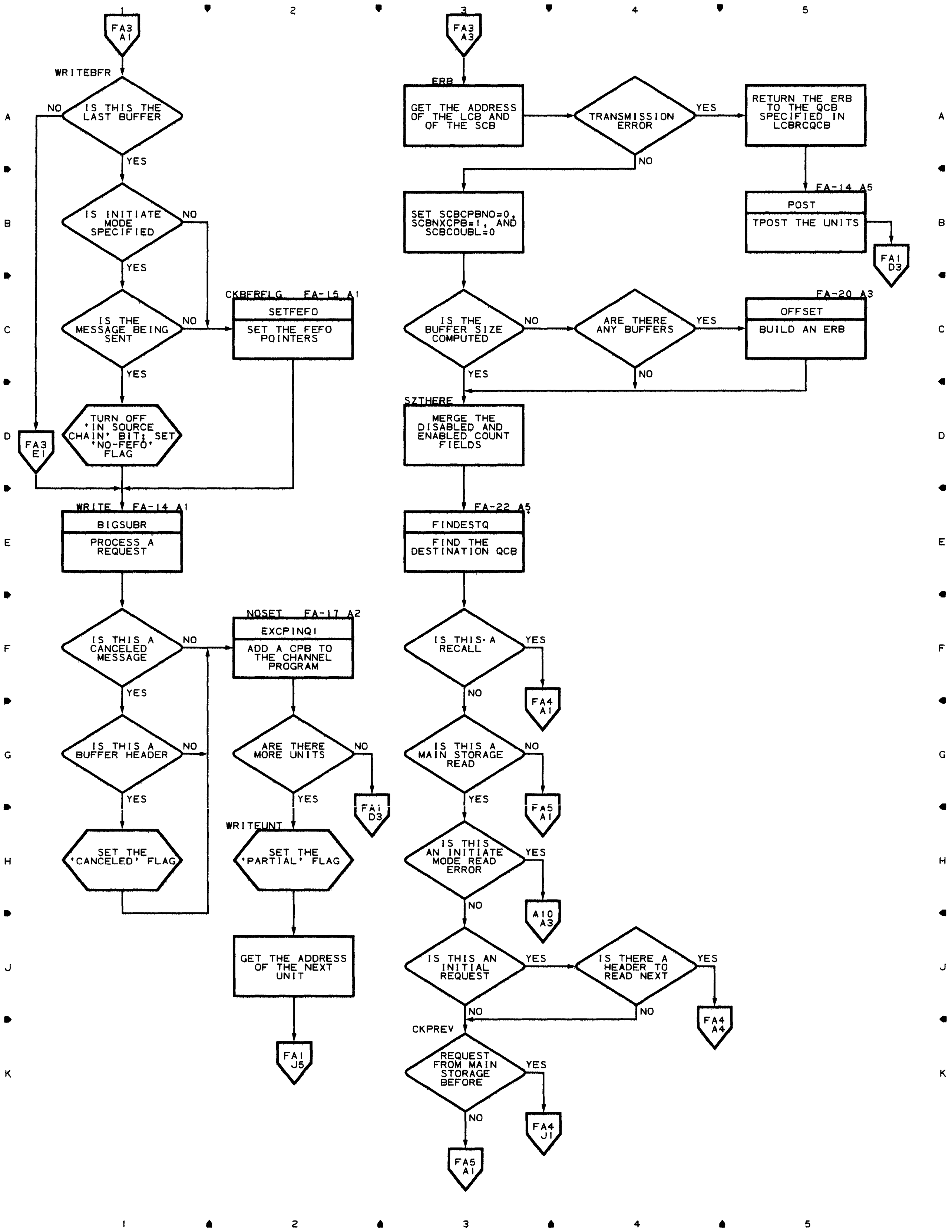
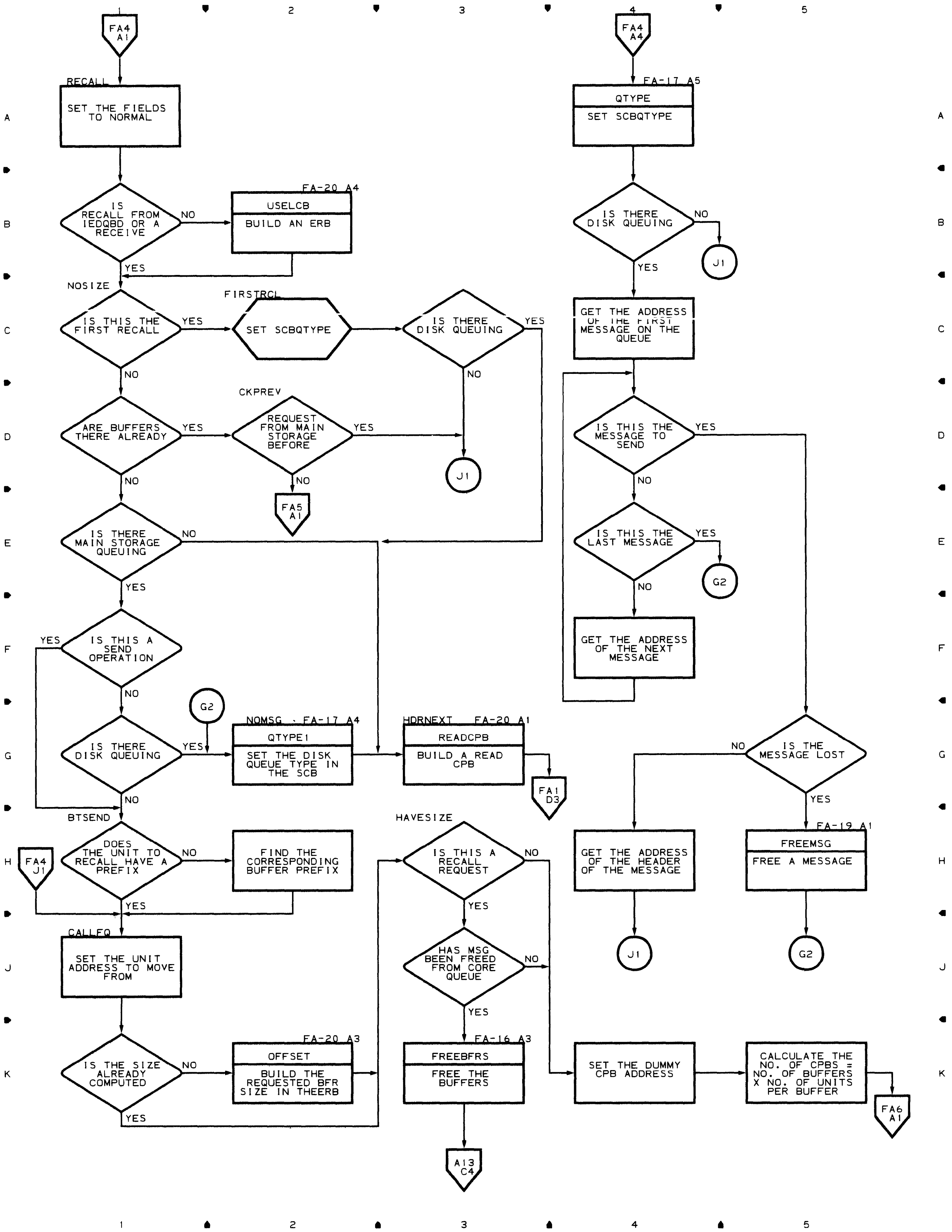


Chart FA-4 (FA4) CPB INITIALIZATION



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FA4 A1
RECALL
SET THE FIELDS TO NORMAL
IS RECALL FROM IEDQBD OR A RECEIVE
NO
FA-20 A4
USELCB
BUILD AN ERB
YES
NOSIZE
IS THIS THE FIRST RECALL
YES
FIRSTRCL
SET SCBQTYPE
IS THERE DISK QUEUING
YES
NO
ARE BUFFERS THERE ALREADY
YES
CKPREV
REQUEST FROM MAIN STORAGE BEFORE
YES
NO
J1
FA5 A1
IS THERE MAIN STORAGE QUEUING
NO
YES
IS THIS A SEND OPERATION
YES
NO
G2
IS THERE DISK QUEUING
YES
NO
NOMSG · FA-17 A4
QTYPE1
SET THE DISK QUEUE TYPE IN THE SCB
HDRNEXT FA-20 A1
READCPB
BUILD A READ CPB
FA1 D3
BTSEND
DOES THE UNIT TO RECALL HAVE A PREFIX
NO
FIND THE CORRESPONDING BUFFER PREFIX
FA4 J1
YES
CALLFQ
SET THE UNIT ADDRESS TO MOVE FROM
IS THE SIZE ALREADY COMPUTED
NO
FA-20 A3
OFFSET
BUILD THE REQUESTED BFR SIZE IN THEERB
YES
HAVESIZE
IS THIS A RECALL REQUEST
NO
YES
HAS MSG BEEN FREED FROM CORE QUEUE
NO
YES
FA-16 A3
FREEBFRS
FREE THE BUFFERS
A13 C4
FA4 A4
FA-17 A5
QTYPE
SET SCBQTYPE
IS THERE DISK QUEUING
NO
J1
YES
GET THE ADDRESS OF THE FIRST MESSAGE ON THE QUEUE
IS THIS THE MESSAGE TO SEND
YES
NO
IS THIS THE LAST MESSAGE
YES
G2
NO
GET THE ADDRESS OF THE NEXT MESSAGE
NO
IS THE MESSAGE LOST
YES
FA-19 A1
FREEMSG
FREE A MESSAGE
GET THE ADDRESS OF THE HEADER OF THE MESSAGE
J1
G2
SET THE DUMMY CPB ADDRESS
CALCULATE THE NO. OF CPBS = NO. OF BUFFERS X NO. OF UNITS PER BUFFER
FA6 A1

Chart FA-5 (FA5) CPB INITIALIZATION

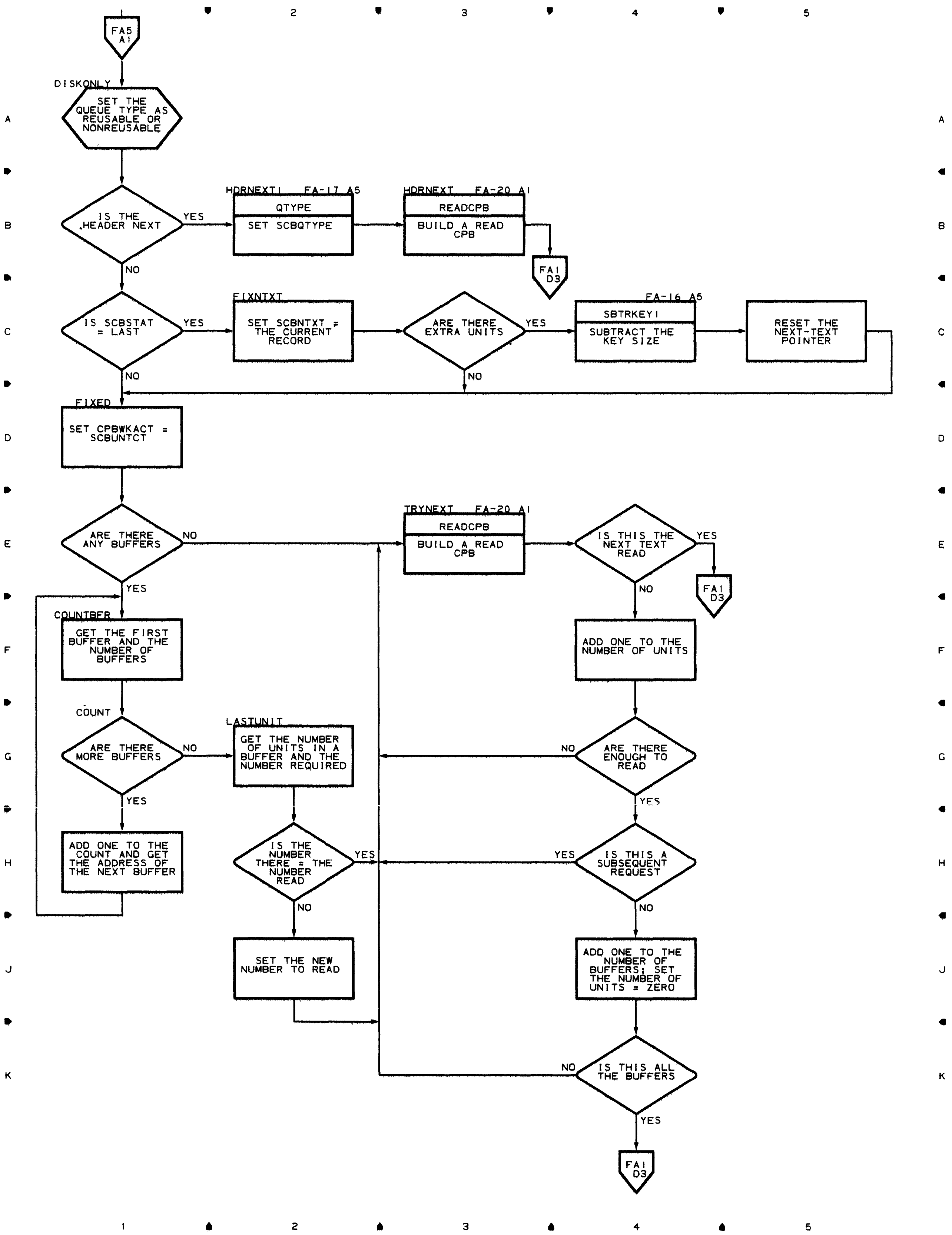

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FA5 A1
DISKONLY
SET THE QUEUE TYPE AS REUSABLE OR NONREUSABLE
IS THE HEADER NEXT
YES
NO
HDRNEXT1 FA-17 A5
QTYPE
SET SCBQTYPE
HDRNEXT FA-20 A1
READCPB
BUILD A READ CPB
FA1 D3
IS SCBSTAT = LAST
YES
NO
FIXNTXT
SET SCBNTXT = THE CURRENT RECORD
ARE THERE EXTRA UNITS
YES
NO
FA-16 A5
SBTRKEY1
SUBTRACT THE KEY SIZE
RESET THE NEXT-TEXT POINTER
FIXED
SET CPBWKACT = SCBUNTCT
ARE THERE ANY BUFFERS
NO
YES
TRYNEXT FA-20 A1
READCPB
BUILD A READ CPB
IS THIS THE NEXT TEXT READ
YES
NO
FA1 D3
COUNTBFR
GET THE FIRST BUFFER AND THE NUMBER OF BUFFERS
ADD ONE TO THE NUMBER OF UNITS
COUNT
ARE THERE MORE BUFFERS
NO
YES
LASTUNIT
GET THE NUMBER OF UNITS IN A BUFFER AND THE NUMBER REQUIRED
ARE THERE ENOUGH TO READ
NO
YES
ADD ONE TO THE COUNT AND GET THE ADDRESS OF THE NEXT BUFFER
IS THE NUMBER THERE = THE NUMBER READ
YES
NO
IS THIS A SUBSEQUENT REQUEST
YES
NO
SET THE NEW NUMBER TO READ
ADD ONE TO THE NUMBER OF BUFFERS; SET THE NUMBER OF UNITS = ZERO
IS THIS ALL THE BUFFERS
NO
YES
FA1 D3

Chart FA-6 (FA6) CPB INITIALIZATION

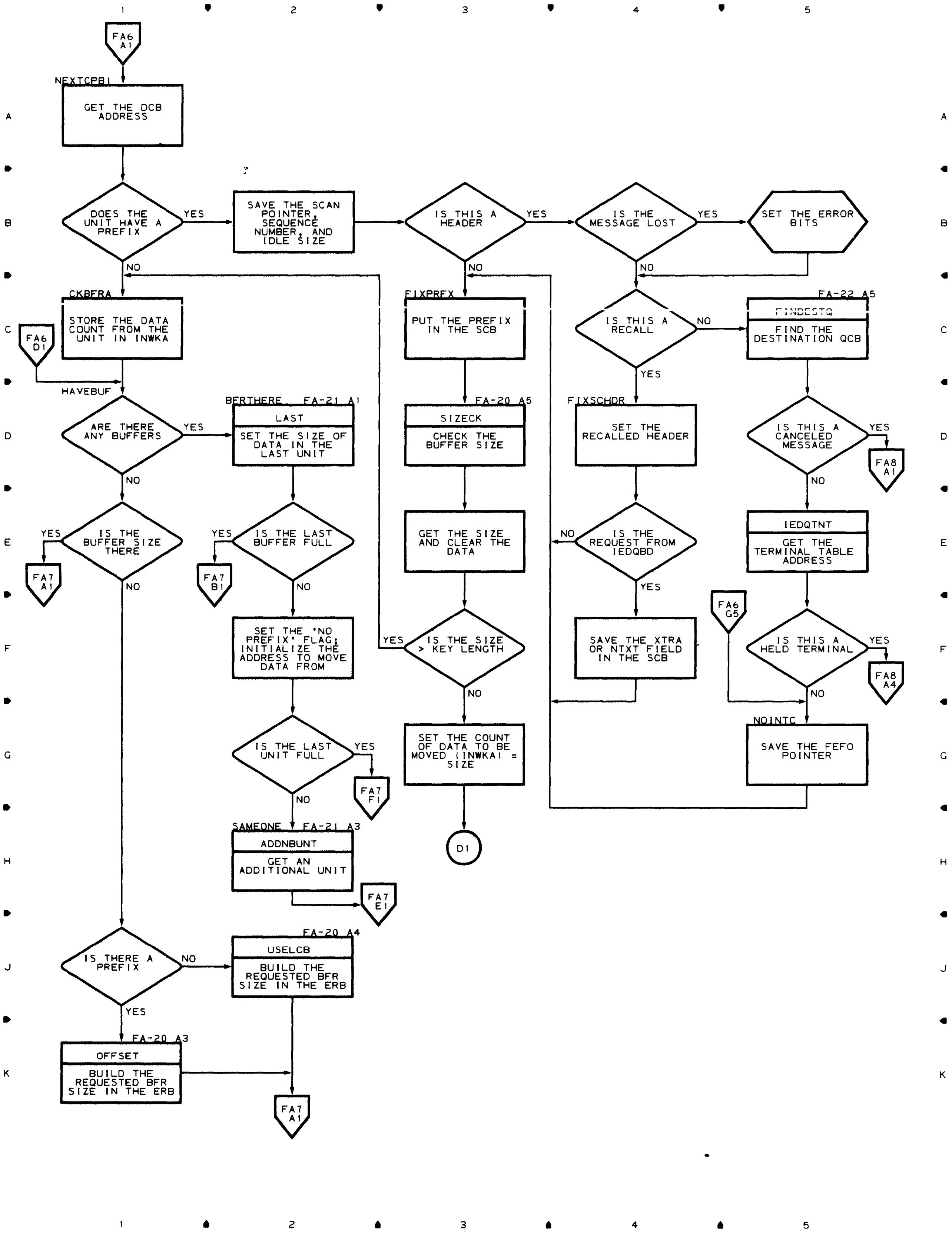


Chart FA-7 (FA7) CPB INITIALIZATION

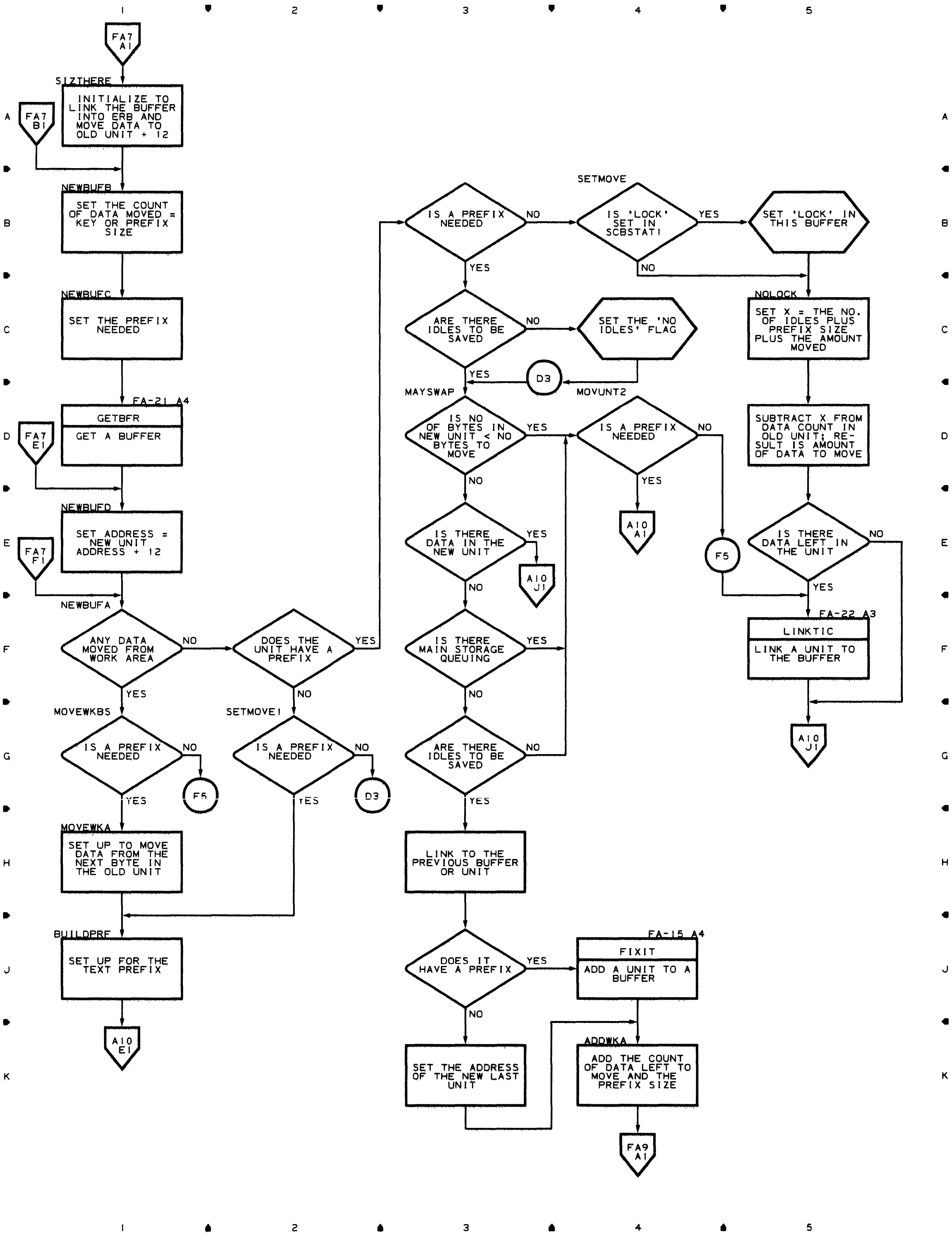

FA7 A1
SIZTHERE
INITIALIZE TO LINK THE BUFFER INTO ERB AND MOVE DATA TO OLD UNIT + 12
FA7 B1
NEWBUFB
SET THE COUNT OF DATA MOVED = KEY OR PREFIX SIZE
NEWBUFC
SET THE PREFIX NEEDED
FA-21 A4
GETBFR
GET A BUFFER
FA7 E1
NEWBUFD
SET ADDRESS = NEW UNIT ADDRESS + 12
FA7 F1
NEWBUFA
ANY DATA MOVED FROM WORK AREA
NO
YES
MOVEWKBS
IS A PREFIX NEEDED
NO
F5
YES
MOVEWKA
SET UP TO MOVE DATA FROM THE NEXT BYTE IN THE OLD UNIT
BUILDPRF
SET UP FOR THE TEXT PREFIX
A10 E1
DOES THE UNIT HAVE A PREFIX
YES
NO
SETMOVE1
IS A PREFIX NEEDED
NO
D3
YES
IS A PREFIX NEEDED
NO
YES
ARE THERE IDLES TO BE SAVED
NO
YES
MAYSWAP
IS NO OF BYTES IN NEW UNIT &lt; NO BYTES TO MOVE
YES
NO
IS THERE DATA IN THE NEW UNIT
YES
A10 J1
NO
IS THERE MAIN STORAGE QUEUING
YES
NO
ARE THERE IDLES TO BE SAVED
NO
YES
LINK TO THE PREVIOUS BUFFER OR UNIT
DOES IT HAVE A PREFIX
YES
NO
SET THE ADDRESS OF THE NEW LAST UNIT
SETMOVE
IS 'LOCK' SET IN SCBSTAT1
YES
NO
SET THE 'NO IDLES' FLAG
D3
MOVUNT2
IS A PREFIX NEEDED
NO
YES
A10 A1
F5
FA-15 A4
FIXIT
ADD A UNIT TO A BUFFER
ADDWKA
ADD THE COUNT OF DATA LEFT TO MOVE AND THE PREFIX SIZE
FA9 A1
SET 'LOCK' IN THIS BUFFER
NOLOCK
SET X = THE NO. OF IDLES PLUS PREFIX SIZE PLUS THE AMOUNT MOVED
SUBTRACT X FROM DATA COUNT IN OLD UNIT; RESULT IS AMOUNT OF DATA TO MOVE
IS THERE DATA LEFT IN THE UNIT
NO
YES
FA-22 A3
LINKTIC
LINK A UNIT TO THE BUFFER
A10 J1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA-8 (FA8) CPB INITIALIZATION

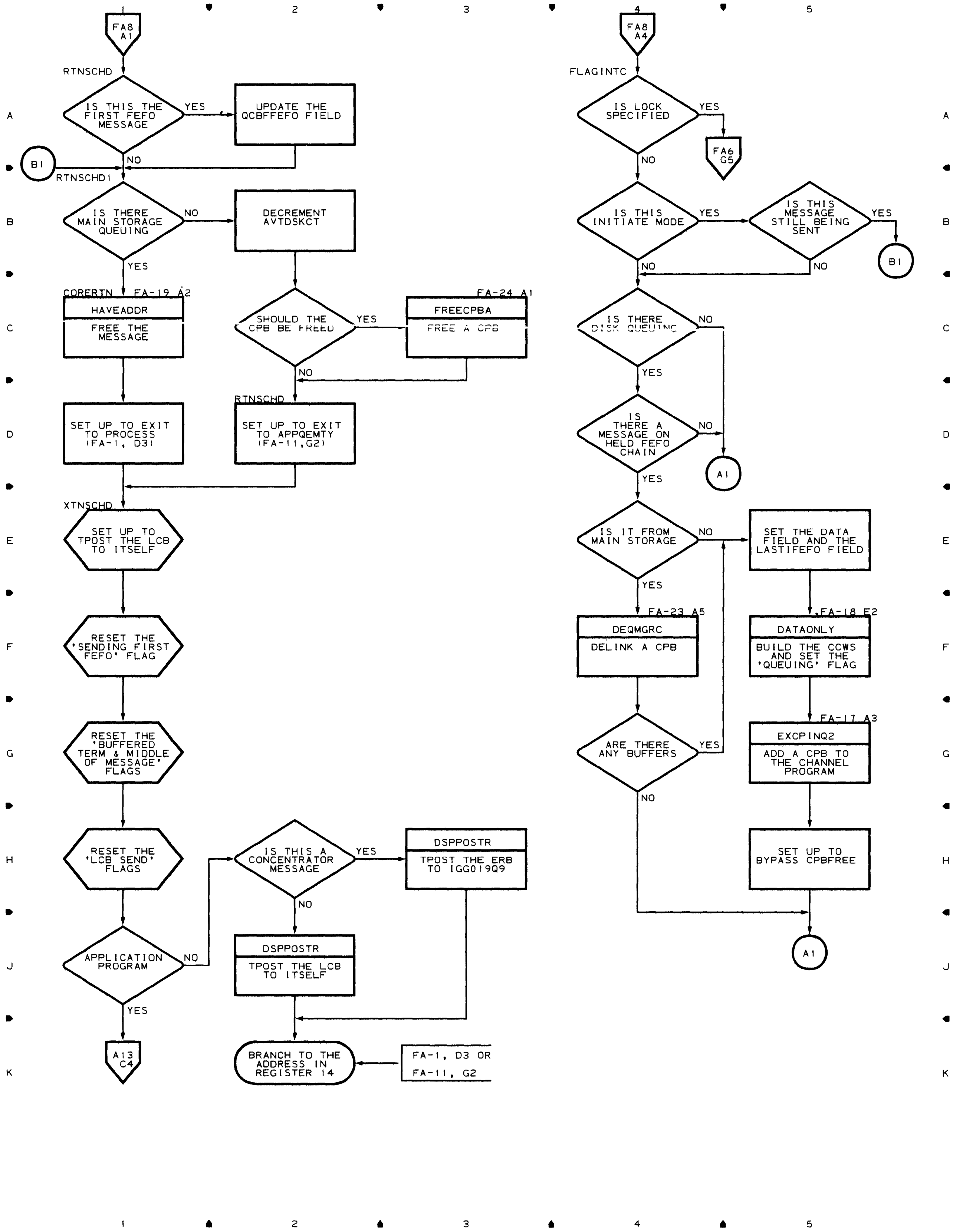

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FA8 A1
RTNSCHD
IS THIS THE FIRST FEFO MESSAGE
YES
UPDATE THE QCBFFEFO FIELD
NO
B1
RTNSCHD1
IS THERE MAIN STORAGE QUEUING
NO
DECREMENT AVTDSKCT
YES
CORERTN FA-19 A2
HAVEADDR
FREE THE MESSAGE
SHOULD THE CPB BE FREED
YES
FA-24 A1
FREECPBA
FREE A CPB
NO
RTNSCHD
SET UP TO EXIT TO PROCESS (FA-1, D3)
SET UP TO EXIT TO APPQEMTY (FA-11,G2)
XTNSCHD
SET UP TO TPOST THE LCB TO ITSELF
RESET THE 'SENDING FIRST FEFO' FLAG
RESET THE 'BUFFERED TERM & MIDDLE OF MESSAGE' FLAGS
RESET THE 'LCB SEND' FLAGS
IS THIS A CONCENTRATOR MESSAGE
YES
DSPPOSTR
TPOST THE ERB TO IGG019Q9
NO
APPLICATION PROGRAM
NO
DSPPOSTR
TPOST THE LCB TO ITSELF
YES
A13 C4
BRANCH TO THE ADDRESS IN REGISTER 14
FA-1, D3 OR
FA-11, G2
FA8 A4
FLAGINTC
IS LOCK SPECIFIED
YES
FA6 G5
NO
IS THIS INITIATE MODE
YES
IS THIS MESSAGE STILL BEING SENT
YES
B1
NO
NO
IS THERE DISK QUEUING
NO
YES
IS THERE A MESSAGE ON HELD FEFO CHAIN
NO
A1
YES
IS IT FROM MAIN STORAGE
NO
SET THE DATA FIELD AND THE LASTIFEFO FIELD
YES
FA-23 A5
DEQMGRC
DELINK A CPB
FA-18 E2
DATAONLY
BUILD THE CCWS AND SET THE 'QUEUING' FLAG
ARE THERE ANY BUFFERS
YES
FA-17 A3
EXCPINQ2
ADD A CPB TO THE CHANNEL PROGRAM
NO
SET UP TO BYPASS CPBFREE
A1

Chart FA-9 (FA9) CPB INITIALIZATION

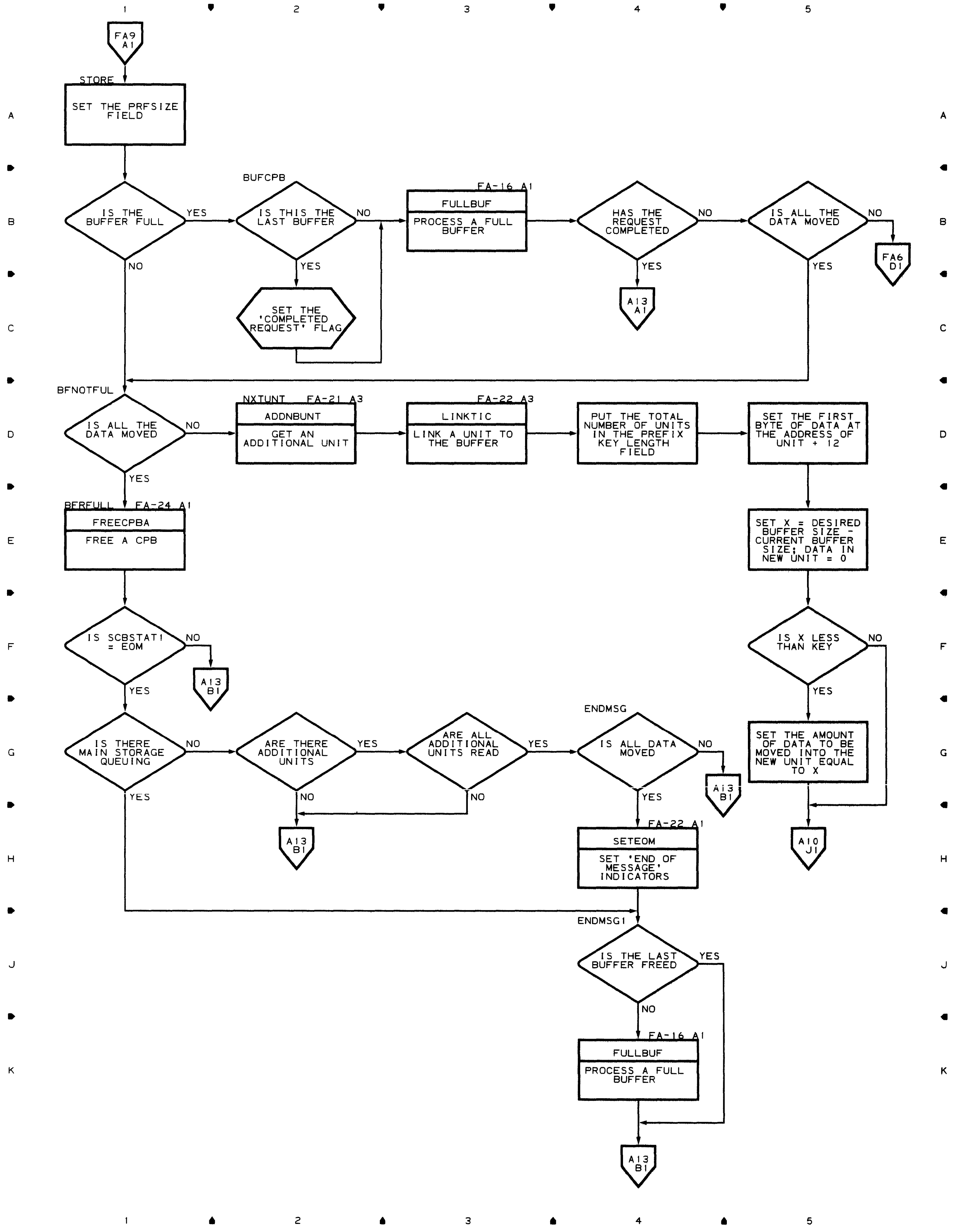
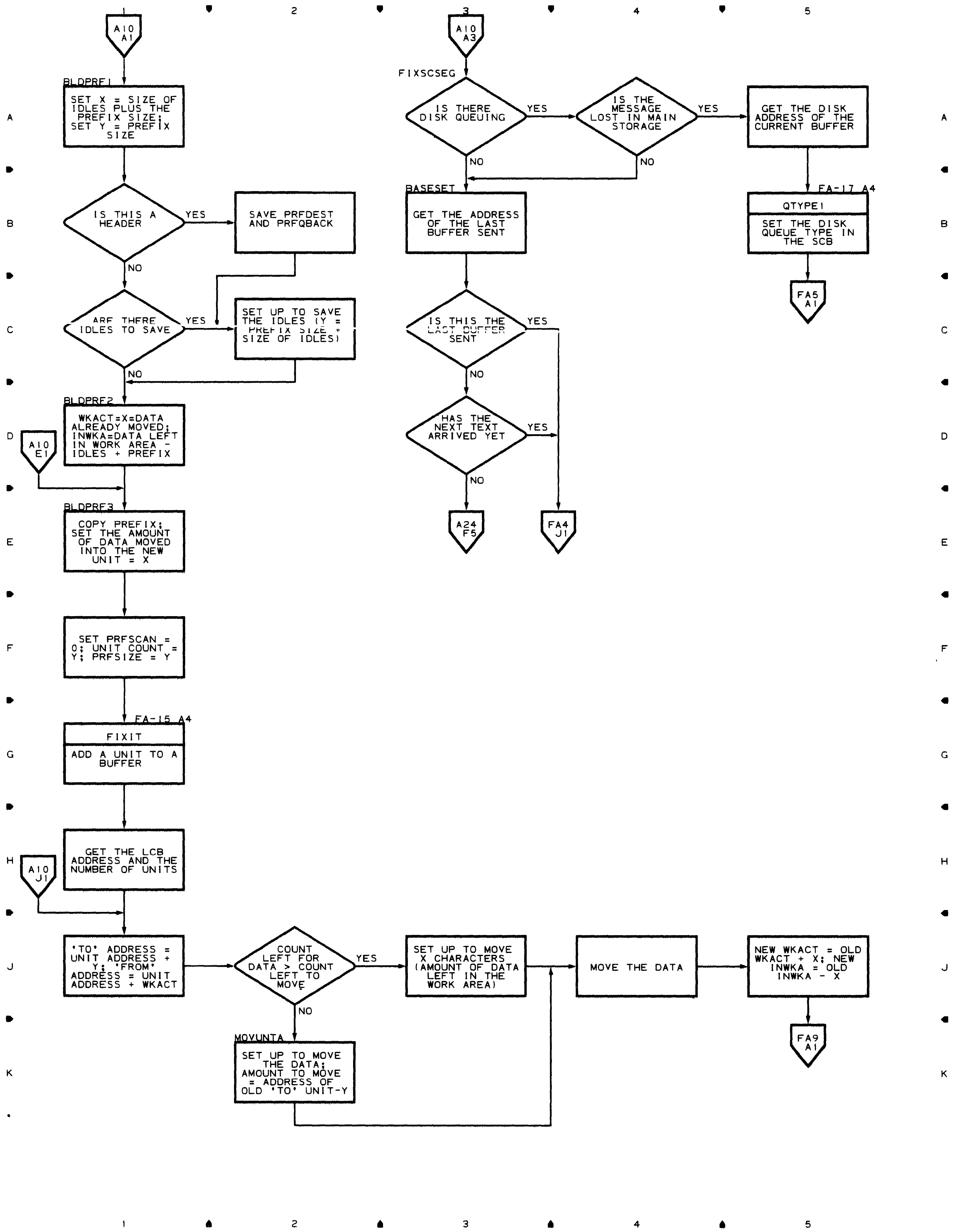


Chart FA-10 (A10) CPB INITIALIZATION



A10 A1
BLDPREF1
SET X = SIZE OF IDLES PLUS THE PREFIX SIZE; SET Y = PREFIX SIZE
IS THIS A HEADER
YES
SAVE PRFDEST AND PRFQBACK
NO
ARE THERE IDLES TO SAVE
YES
SET UP TO SAVE THE IDLES (Y = PREFIX SIZE + SIZE OF IDLES)
NO
BLDPREF2
WKACT=X=DATA ALREADY MOVED; INWKA=DATA LEFT IN WORK AREA - IDLES + PREFIX
A10 E1
BLDPREF3
COPY PREFIX; SET THE AMOUNT OF DATA MOVED INTO THE NEW UNIT = X
SET PRFSCAN = 0; UNIT COUNT = Y; PRFSIZE = Y
FA-15 A4
FIXIT
ADD A UNIT TO A BUFFER
GET THE LCB ADDRESS AND THE NUMBER OF UNITS
A10 J1
'TO' ADDRESS = UNIT ADDRESS + Y; 'FROM' ADDRESS = UNIT ADDRESS + WKACT
COUNT LEFT FOR DATA > COUNT LEFT TO MOVE
YES
NO
MOVUNTA
SET UP TO MOVE THE DATA; AMOUNT TO MOVE = ADDRESS OF OLD 'TO' UNIT-Y
SET UP TO MOVE X CHARACTERS (AMOUNT OF DATA LEFT IN THE WORK AREA)
MOVE THE DATA
NEW WKACT = OLD WKACT + X; NEW INWKA = OLD INWKA - X
FA9 A1
A10 A3
FIXSCSEG
IS THERE DISK QUEUING
YES
IS THE MESSAGE LOST IN MAIN STORAGE
YES
GET THE DISK ADDRESS OF THE CURRENT BUFFER
NO
NO
BASESET
GET THE ADDRESS OF THE LAST BUFFER SENT
FA-17 A4
QTYPE1
SET THE DISK QUEUE TYPE IN THE SCB
FA5 A1
IS THIS THE LAST BUFFER SENT
YES
NO
HAS THE NEXT TEXT ARRIVED YET
YES
NO
A24 F5
FA4 J1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

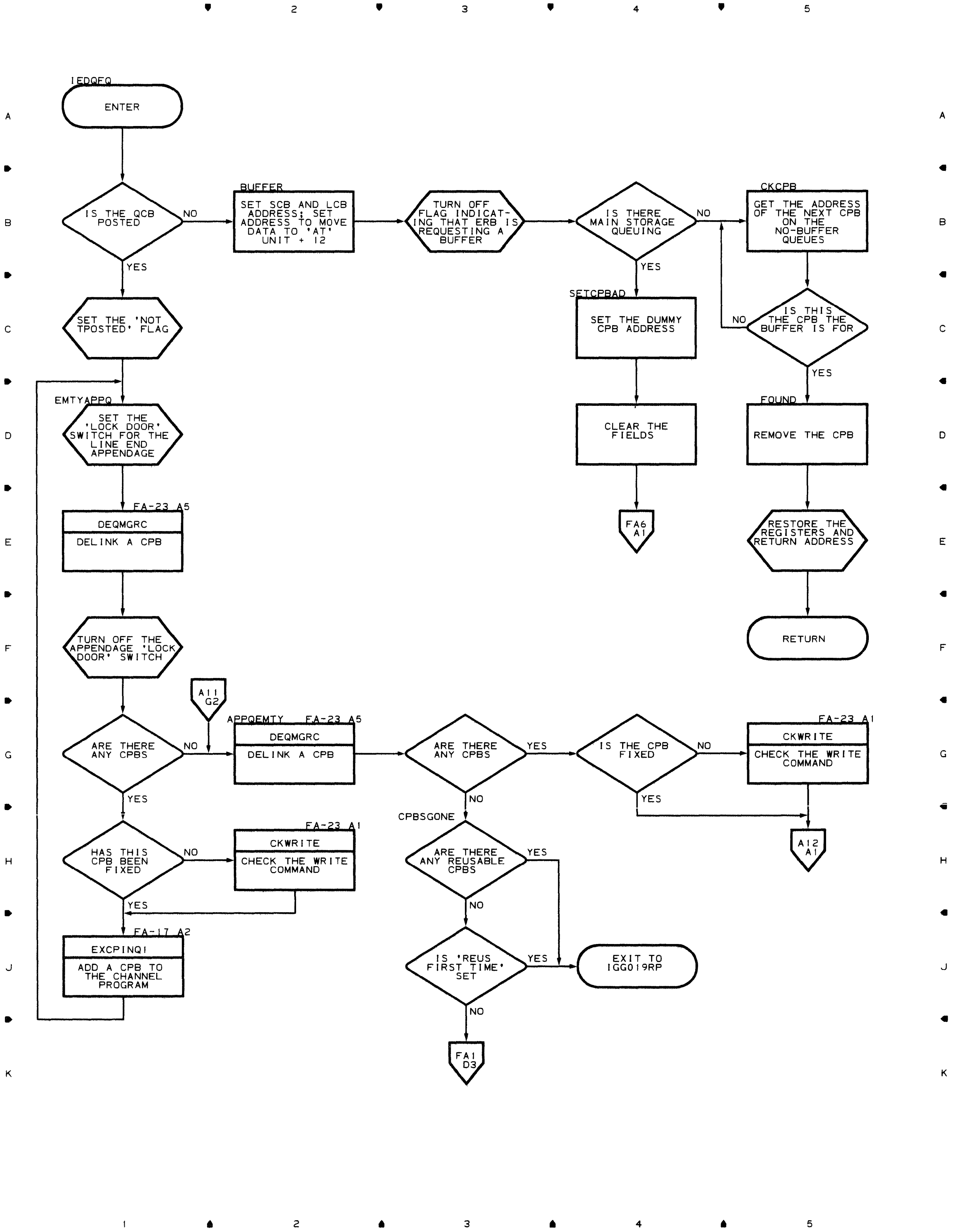

IEDQFQ
ENTER
IS THE QCB POSTED
NO
YES
BUFFER
SET SCB AND LCB ADDRESS; SET ADDRESS TO MOVE DATA TO 'AT' UNIT + 12
TURN OFF FLAG INDICATING THAT ERB IS REQUESTING A BUFFER
IS THERE MAIN STORAGE QUEUING
NO
YES
CKCPB
GET THE ADDRESS OF THE NEXT CPB ON THE NO-BUFFER QUEUES
IS THIS THE CPB THE BUFFER IS FOR
NO
YES
SET THE 'NOT TPOSTED' FLAG
SETCPBAD
SET THE DUMMY CPB ADDRESS
CLEAR THE FIELDS
FA6 A1
EMTYAPPQ
SET THE 'LOCK DOOR' SWITCH FOR THE LINE END APPENDAGE
FA-23 A5
DEQMGRC
DELINK A CPB
FOUND
REMOVE THE CPB
RESTORE THE REGISTERS AND RETURN ADDRESS
RETURN
TURN OFF THE APPENDAGE 'LOCK DOOR' SWITCH
A11 G2
ARE THERE ANY CPBS
NO
YES
APPQEMTY
FA-23 A5
DEQMGRC
DELINK A CPB
ARE THERE ANY CPBS
YES
NO
IS THE CPB FIXED
NO
YES
FA-23 A1
CKWRITE
CHECK THE WRITE COMMAND
A12 A1
HAS THIS CPB BEEN FIXED
NO
YES
FA-23 A1
CKWRITE
CHECK THE WRITE COMMAND
CPBSGONE
ARE THERE ANY REUSABLE CPBS
YES
NO
FA-17 A2
EXCPINQ1
ADD A CPB TO THE CHANNEL PROGRAM
IS 'REUS FIRST TIME' SET
YES
NO
EXIT TO IGG019RP
FA1 D3
A
B
C
D
E
F
G
H
J
K
1
2
3
4
5

Chart FA-12 (A12) CPB INITIALIZATION

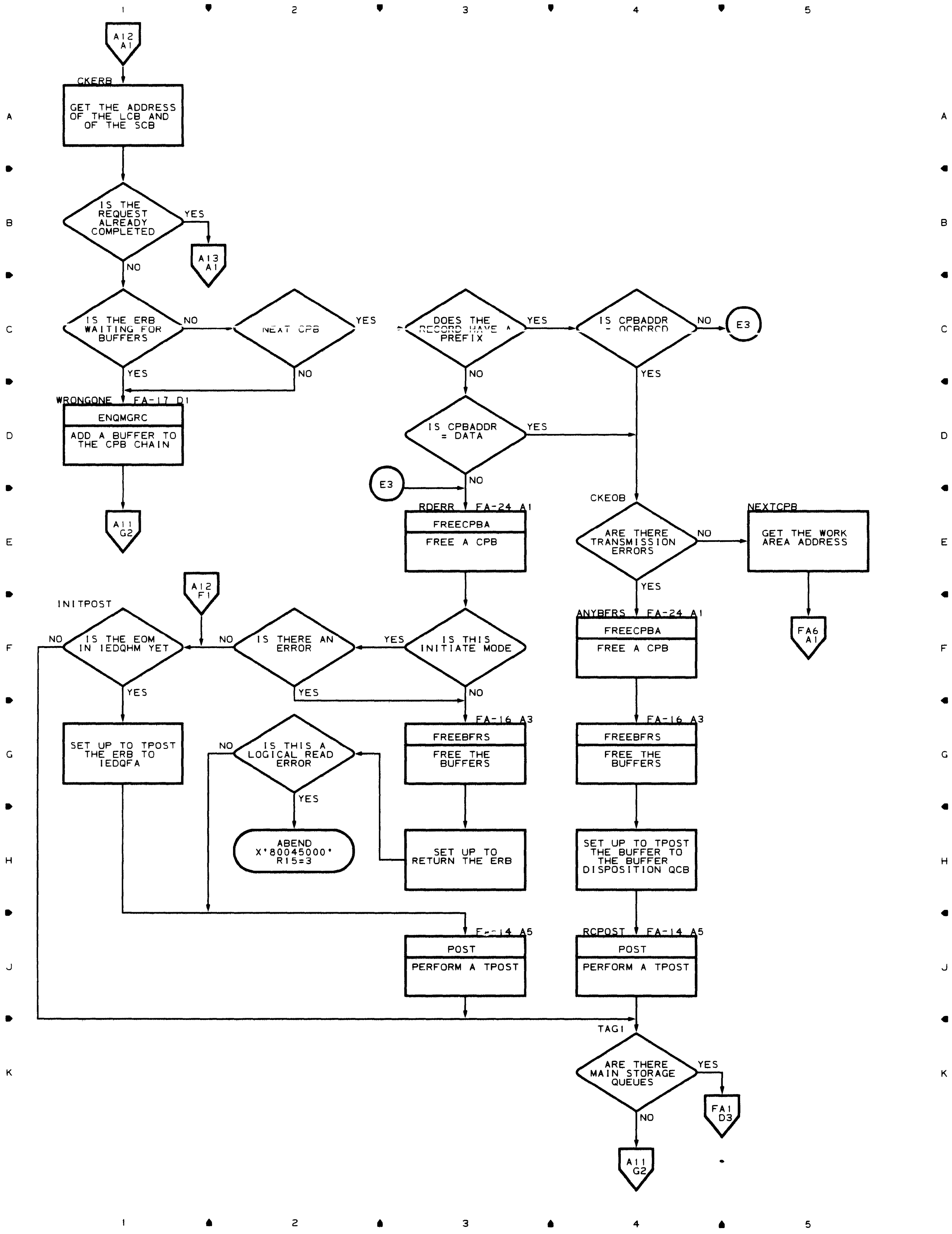


Chart FA-13 (A13) CPB INITIALIZATION

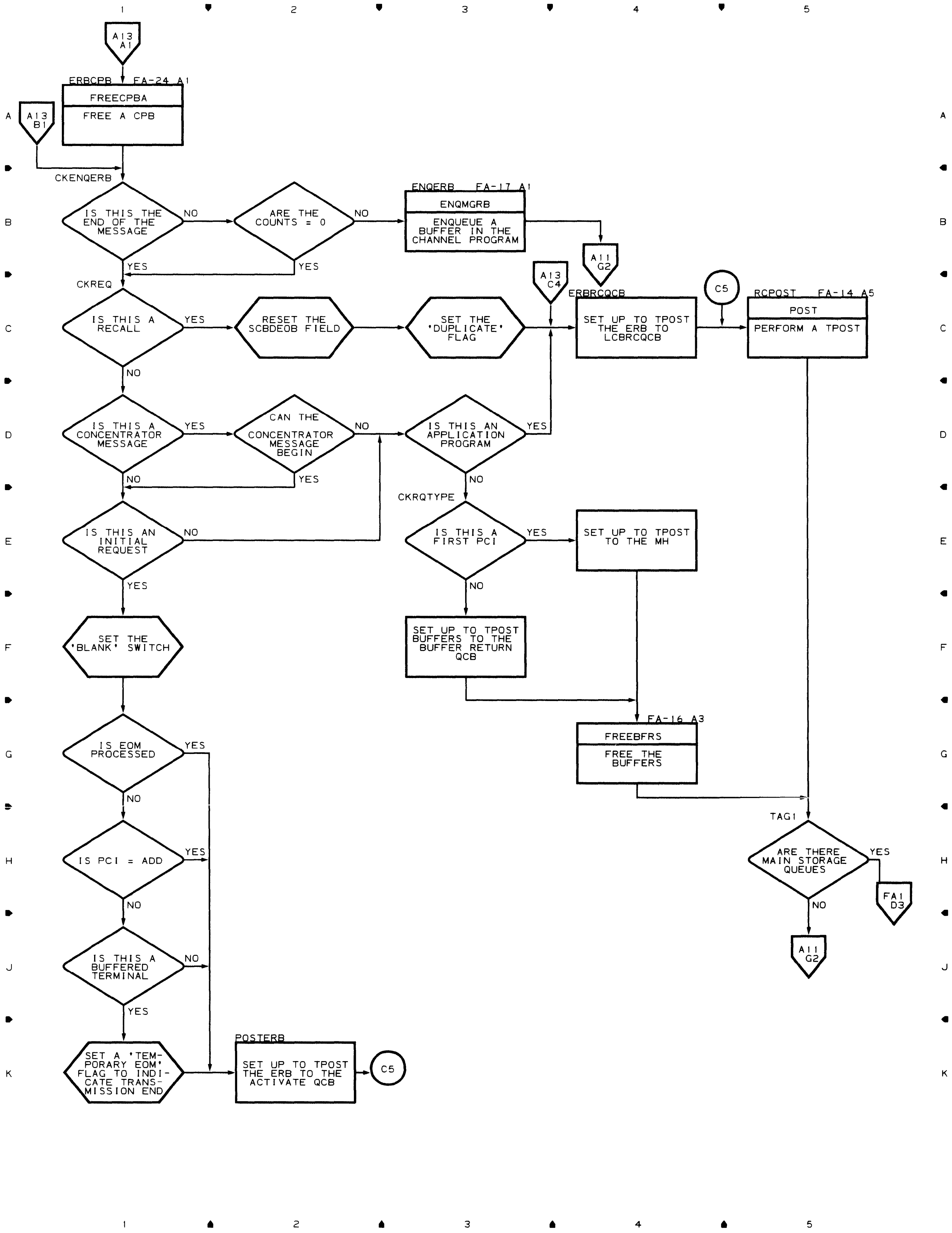

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
A13 A1
ERBCPB FA-24 A1
FREECPBA
FREE A CPB
A13 B1
CKENQERB
IS THIS THE END OF THE MESSAGE
NO
ARE THE COUNTS = 0
NO
ENQERB FA-17 A1
ENQMGRB
ENQUEUE A BUFFER IN THE CHANNEL PROGRAM
A11 G2
YES
YES
CKREQ
IS THIS A RECALL
YES
RESET THE SCBDEOB FIELD
SET THE 'DUPLICATE' FLAG
A13 C4
ERBRCQCB
SET UP TO TPOST THE ERB TO LCBRCQCB
C5
RCPOST FA-14 A5
POST
PERFORM A TPOST
NO
IS THIS A CONCENTRATOR MESSAGE
YES
CAN THE CONCENTRATOR MESSAGE BEGIN
NO
IS THIS AN APPLICATION PROGRAM
YES
NO
YES
NO
CKRQTYPE
IS THIS AN INITIAL REQUEST
NO
IS THIS A FIRST PCI
YES
SET UP TO TPOST TO THE MH
YES
NO
SET THE 'BLANK' SWITCH
SET UP TO TPOST BUFFERS TO THE BUFFER RETURN QCB
FA-16 A3
FREEBFRS
FREE THE BUFFERS
IS EOM PROCESSED
YES
NO
TAG1
IS PCI = ADD
YES
ARE THERE MAIN STORAGE QUEUES
YES
FA1 D3
NO
NO
A11 G2
IS THIS A BUFFERED TERMINAL
NO
YES
POSTERB
SET A 'TEMPORARY EOM' FLAG TO INDICATE TRANSMISSION END
SET UP TO TPOST THE ERB TO THE ACTIVATE QCB
C5

Chart FA-14 (A14) CPB INITIALIZATION

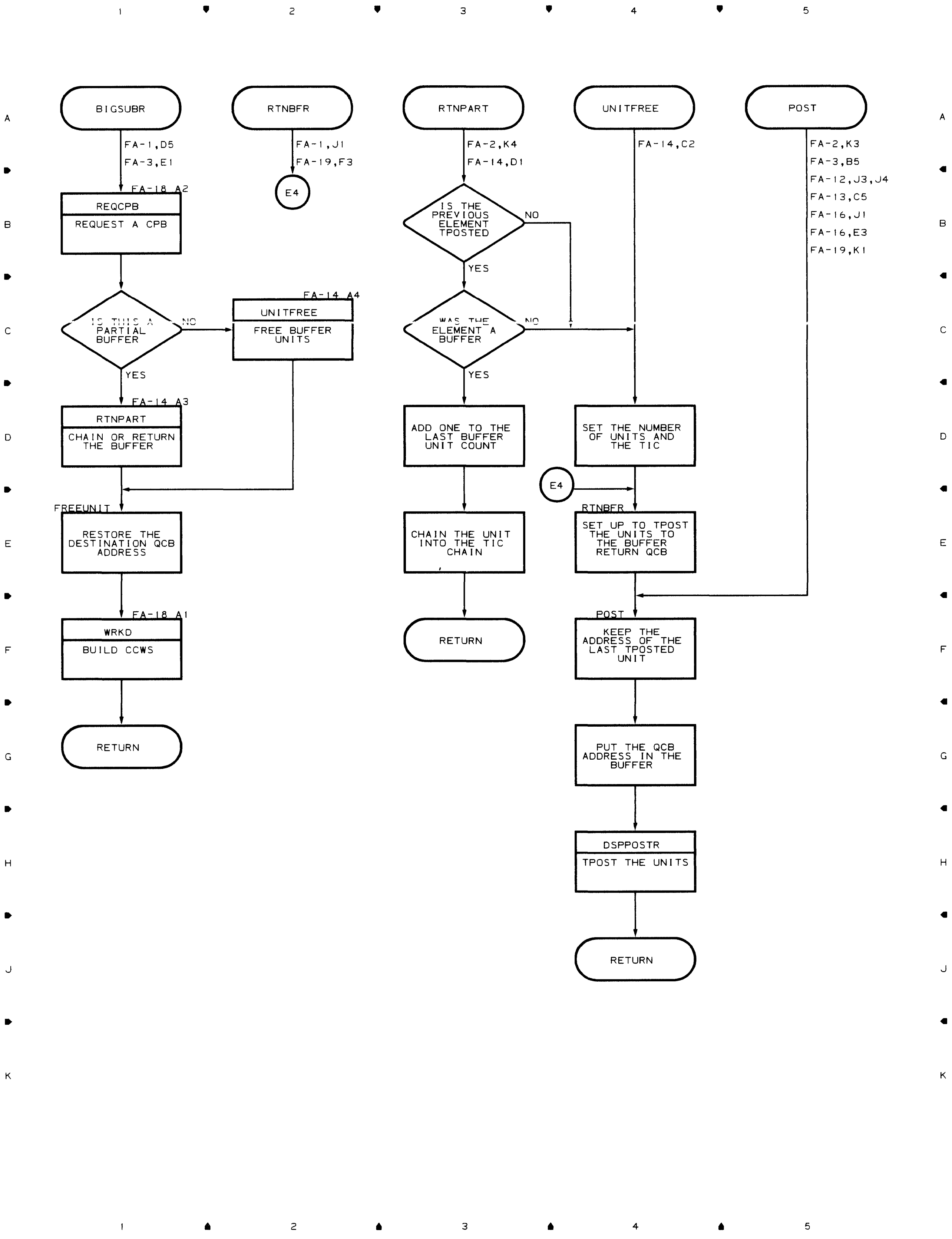


Chart FA-15 (A15) CPB INITIALIZATION

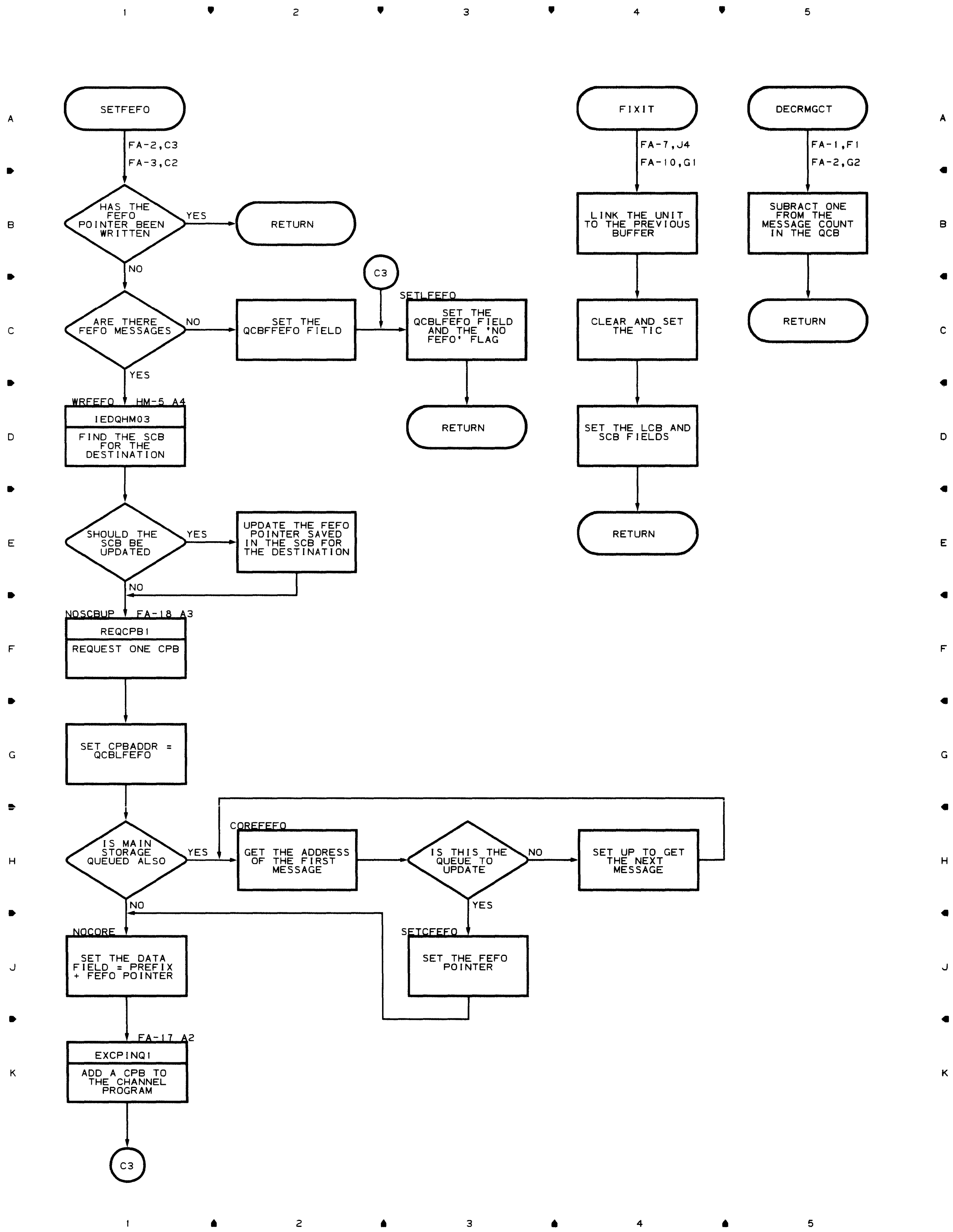
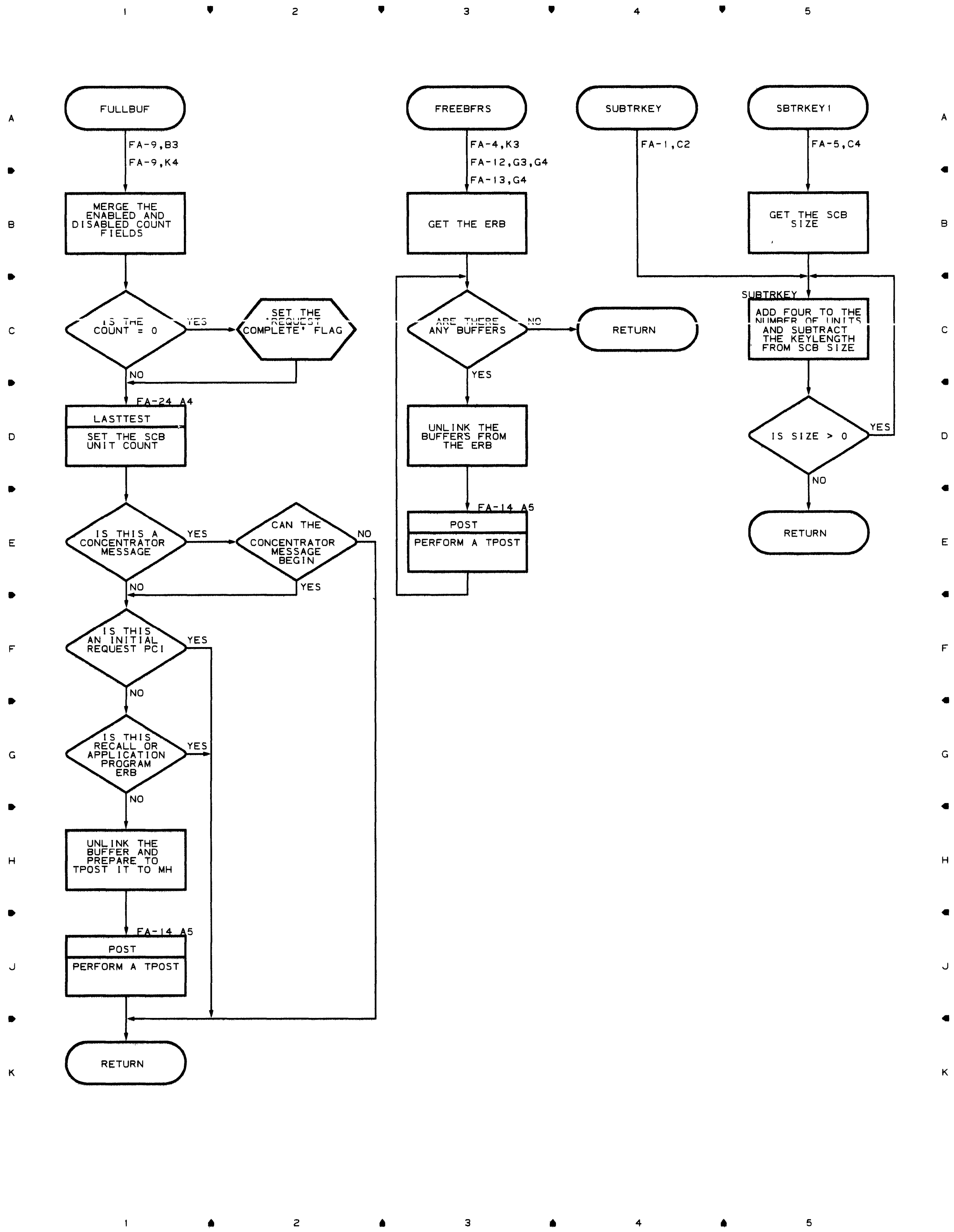

SETFEFO
FA-2,C3
FA-3,C2
HAS THE FEFO POINTER BEEN WRITTEN
YES
RETURN
NO
ARE THERE FEFO MESSAGES
NO
SET THE QCBFFEFO FIELD
C3
SETLFEFO
SET THE QCBLFEFO FIELD AND THE 'NO FEFO' FLAG
RETURN
YES
WRFEFO
HM-5 A4
IEDQHM03
FIND THE SCB FOR THE DESTINATION
SHOULD THE SCB BE UPDATED
YES
UPDATE THE FEFO POINTER SAVED IN THE SCB FOR THE DESTINATION
NO
NOSCBUP
FA-18 A3
REQCPB1
REQUEST ONE CPB
SET CPBADDR = QCBLFEFO
IS MAIN STORAGE QUEUED ALSO
YES
COREFEFO
GET THE ADDRESS OF THE FIRST MESSAGE
IS THIS THE QUEUE TO UPDATE
NO
SET UP TO GET THE NEXT MESSAGE
YES
SETCFEFO
SET THE FEFO POINTER
NO
NOCORE
SET THE DATA FIELD = PREFIX + FEFO POINTER
FA-17 A2
EXCPINQ1
ADD A CPB TO THE CHANNEL PROGRAM
C3
FIXIT
FA-7,J4
FA-10,G1
LINK THE UNIT TO THE PREVIOUS BUFFER
CLEAR AND SET THE TIC
SET THE LCB AND SCB FIELDS
RETURN
DECRMGCT
FA-1,F1
FA-2,G2
SUBRACT ONE FROM THE MESSAGE COUNT IN THE QCB
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA-16 (A16) CPB INITIALIZATION



FULLBUF
FA-9,B3
FA-9,K4
MERGE THE ENABLED AND DISABLED COUNT FIELDS
IS THE COUNT = 0
YES
SET THE 'REQUEST COMPLETE' FLAG
NO
FA-24 A4
LASTTEST
SET THE SCB UNIT COUNT
IS THIS A CONCENTRATOR MESSAGE
YES
CAN THE CONCENTRATOR MESSAGE BEGIN
NO
YES
NO
IS THIS AN INITIAL REQUEST PCI
YES
NO
IS THIS RECALL OR APPLICATION PROGRAM ERB
YES
NO
UNLINK THE BUFFER AND PREPARE TO TPOST IT TO MH
FA-14 A5
POST
PERFORM A TPOST
RETURN
FREEBFRS
FA-4,K3
FA-12,G3,G4
FA-13,G4
GET THE ERB
ARE THERE ANY BUFFERS
NO
RETURN
YES
UNLINK THE BUFFERS FROM THE ERB
FA-14 A5
POST
PERFORM A TPOST
SUBTRKEY
FA-1,C2
SBTRKEY1
FA-5,C4
GET THE SCB SIZE
SUBTRKEY
ADD FOUR TO THE NUMBER OF UNITS AND SUBTRACT THE KEYLENGTH FROM SCB SIZE
IS SIZE > 0
YES
NO
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA-17 (A17) CPB INITIALIZATION

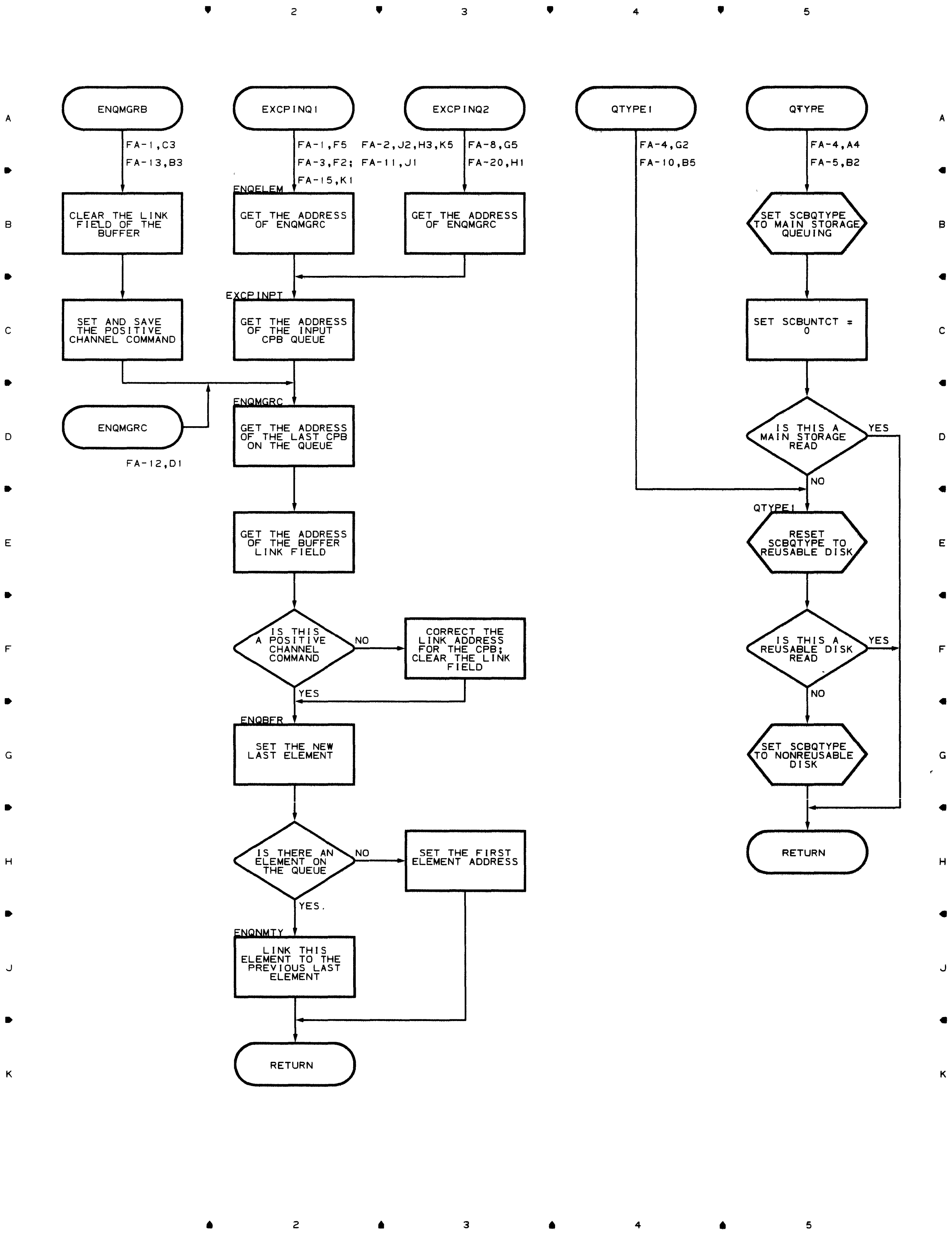

ENQMGRB
FA-1,C3
FA-13,B3
CLEAR THE LINK FIELD OF THE BUFFER
SET AND SAVE THE POSITIVE CHANNEL COMMAND
ENQMGRC
FA-12,D1
EXCPINQ1
FA-1,F5
FA-3,F2; FA-11,J1
FA-15,K1
FA-2,J2,H3,K5
ENQELEM
GET THE ADDRESS OF ENQMGRC
EXCPINPT
GET THE ADDRESS OF THE INPUT CPB QUEUE
ENQMGRC
GET THE ADDRESS OF THE LAST CPB ON THE QUEUE
GET THE ADDRESS OF THE BUFFER LINK FIELD
IS THIS A POSITIVE CHANNEL COMMAND
NO
YES
ENQBFR
SET THE NEW LAST ELEMENT
IS THERE AN ELEMENT ON THE QUEUE
NO
YES.
ENQNMTY
LINK THIS ELEMENT TO THE PREVIOUS LAST ELEMENT
RETURN
EXCPINQ2
FA-8,G5
FA-20,H1
GET THE ADDRESS OF ENQMGRC
CORRECT THE LINK ADDRESS FOR THE CPB; CLEAR THE LINK FIELD
SET THE FIRST ELEMENT ADDRESS
QTYPE1
FA-4,G2
FA-10,B5
QTYPE
FA-4,A4
FA-5,B2
SET SCBQTYPE TO MAIN STORAGE QUEUING
SET SCBUNTCT = 0
IS THIS A MAIN STORAGE READ
YES
NO
QTYPE1
RESET SCBQTYPE TO REUSABLE DISK
IS THIS A REUSABLE DISK READ
YES
NO
SET SCBQTYPE TO NONREUSABLE DISK
RETURN
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA-18 (A18) CPB INITIALIZATION

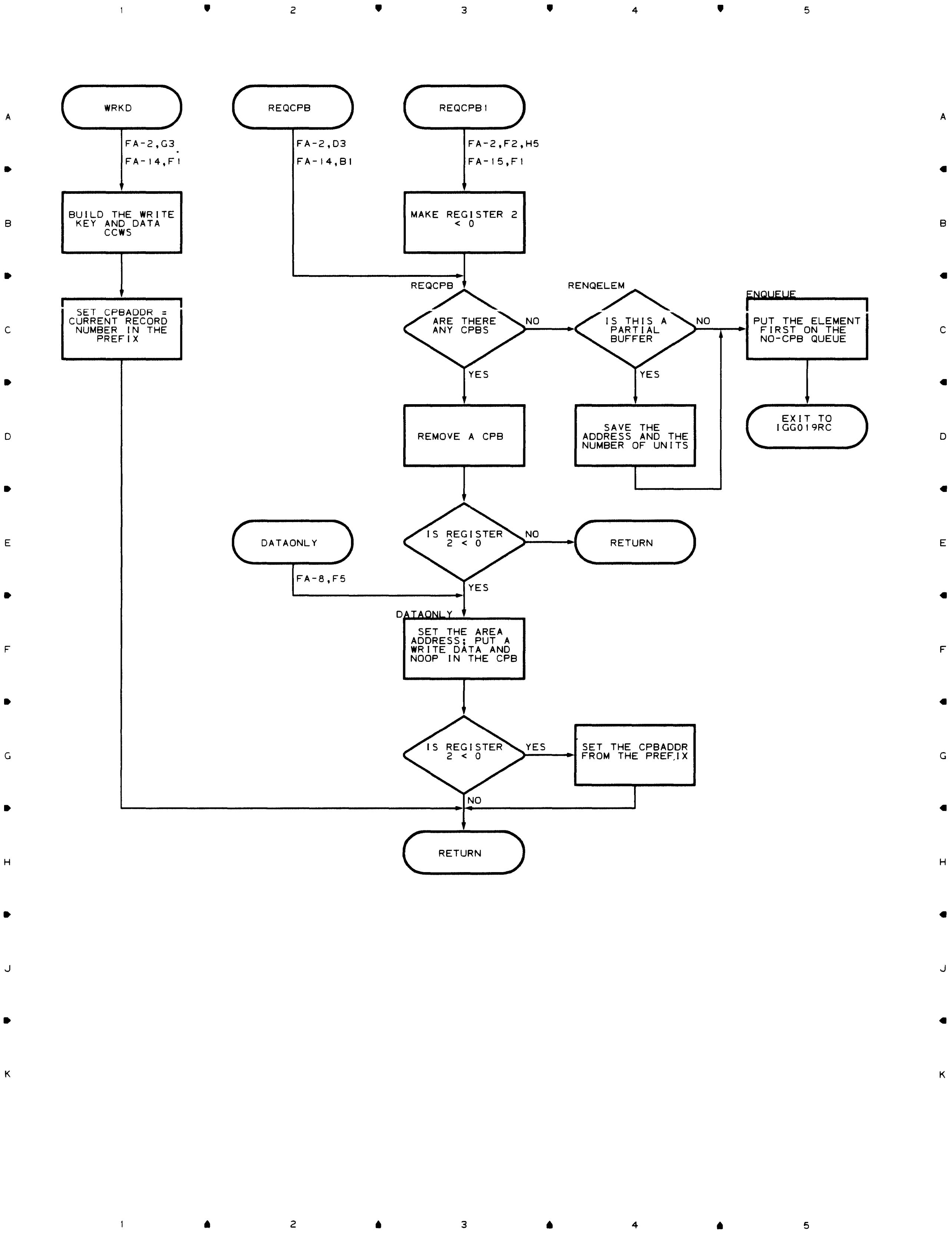


Chart FA-19 (A19) CPB INITIALIZATION

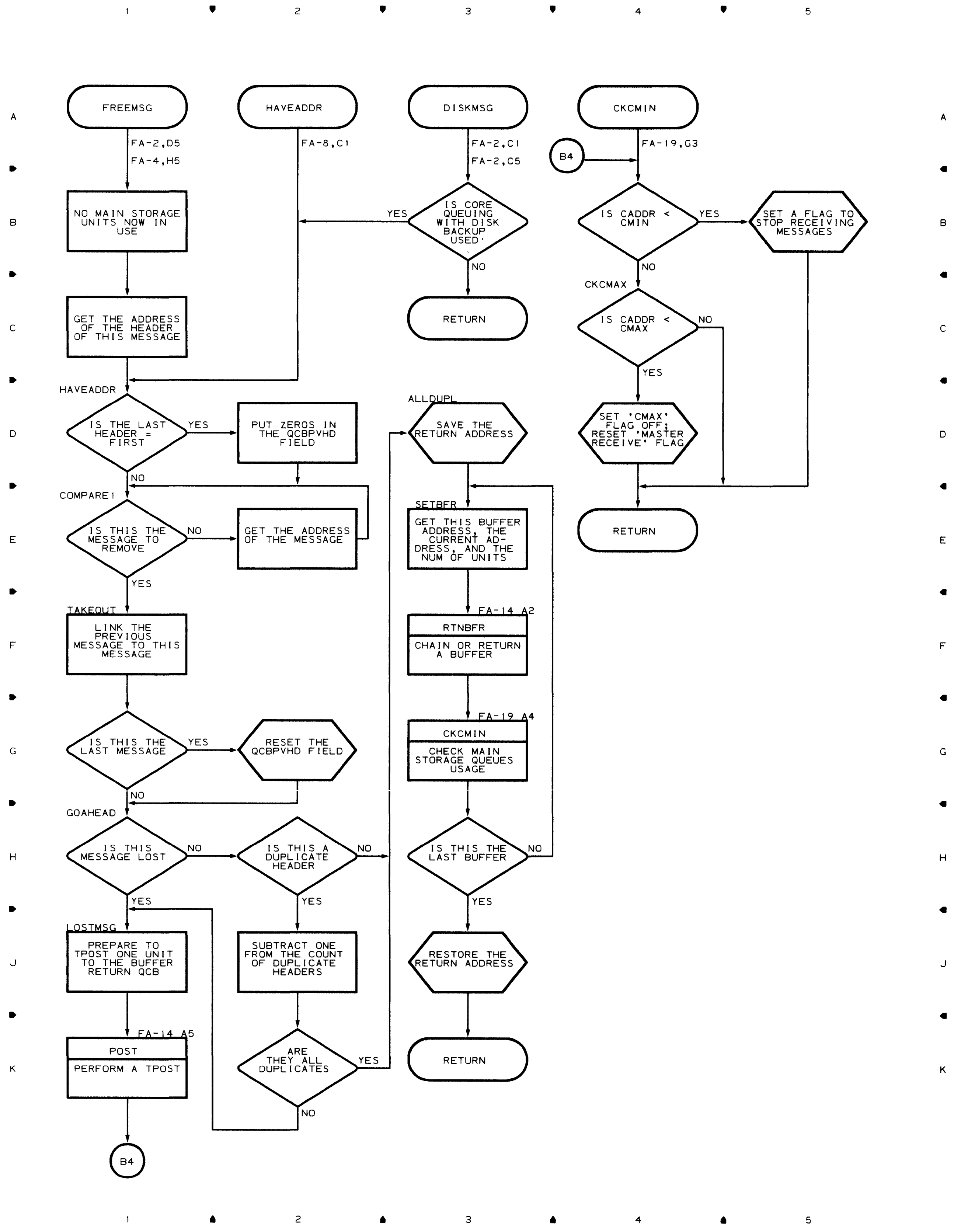

FREEMSG
FA-2,D5
FA-4,H5
NO MAIN STORAGE UNITS NOW IN USE
GET THE ADDRESS OF THE HEADER OF THIS MESSAGE
HAVEADDR
IS THE LAST HEADER = FIRST
YES
PUT ZEROS IN THE QCBPVHD FIELD
NO
COMPARE1
IS THIS THE MESSAGE TO REMOVE
NO
GET THE ADDRESS OF THE MESSAGE
YES
TAKEOUT
LINK THE PREVIOUS MESSAGE TO THIS MESSAGE
IS THIS THE LAST MESSAGE
YES
RESET THE QCBPVHD FIELD
NO
GOAHEAD
IS THIS MESSAGE LOST
NO
IS THIS A DUPLICATE HEADER
NO
YES
YES
LOSTMSG
PREPARE TO TPOST ONE UNIT TO THE BUFFER RETURN QCB
SUBTRACT ONE FROM THE COUNT OF DUPLICATE HEADERS
FA-14 A5
POST
PERFORM A TPOST
ARE THEY ALL DUPLICATES
YES
NO
B4
HAVEADDR
FA-8,C1
DISKMSG
FA-2,C1
FA-2,C5
IS CORE QUEUING WITH DISK BACKUP USED
YES
NO
RETURN
ALLDUPL
SAVE THE RETURN ADDRESS
SETBFR
GET THIS BUFFER ADDRESS, THE CURRENT ADDRESS, AND THE NUM OF UNITS
FA-14 A2
RTNBFR
CHAIN OR RETURN A BUFFER
FA-19 A4
CKCMIN
CHECK MAIN STORAGE QUEUES USAGE
IS THIS THE LAST BUFFER
NO
YES
RESTORE THE RETURN ADDRESS
RETURN
CKCMIN
B4
FA-19,G3
IS CADDR < CMIN
YES
SET A FLAG TO STOP RECEIVING MESSAGES
NO
CKCMAX
IS CADDR < CMAX
NO
YES
SET 'CMAX' FLAG OFF; RESET 'MASTER RECEIVE' FLAG
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA-20 (A20) CPB INITIALIZATION

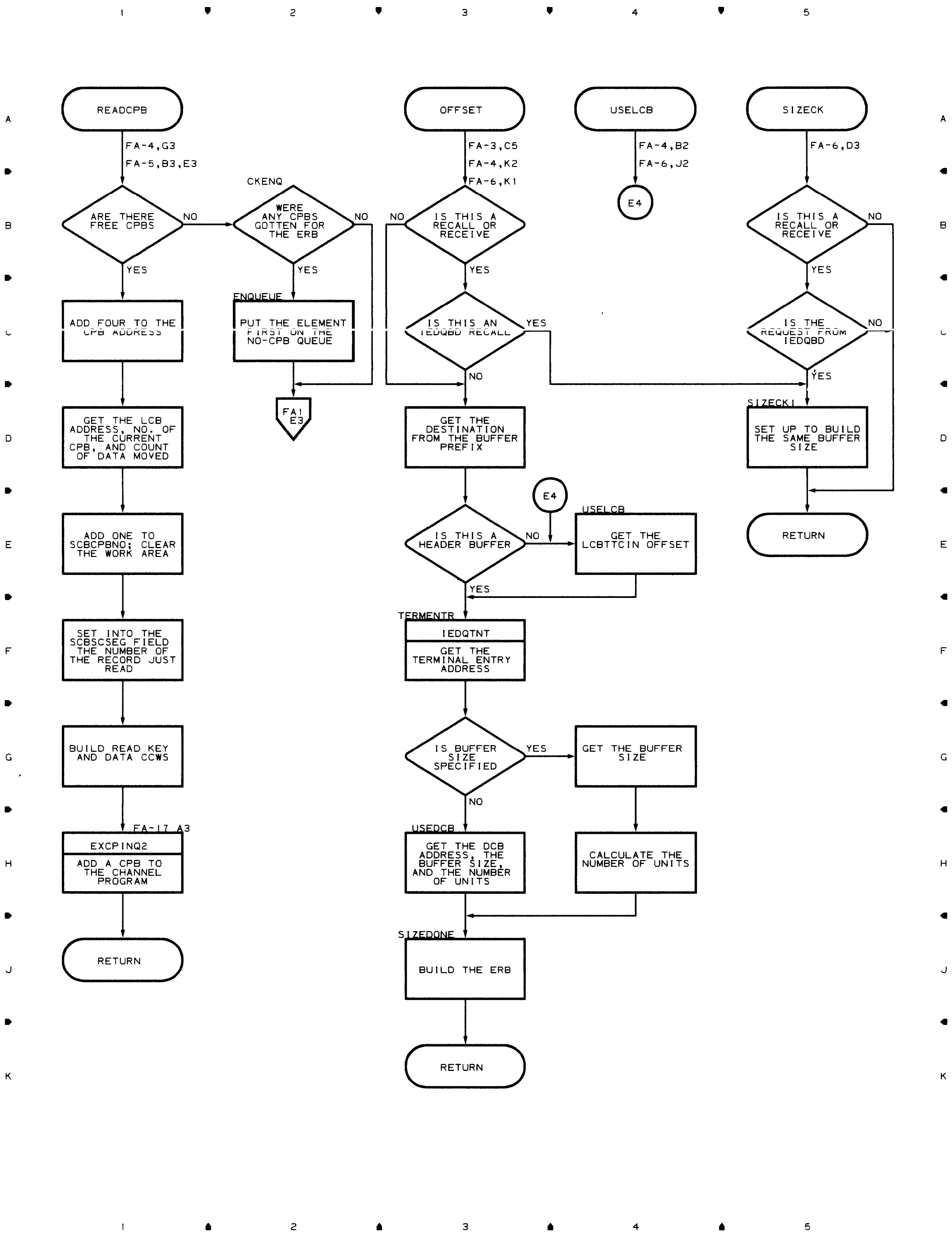

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
READCPB
FA-4,G3
FA-5,B3,E3
ARE THERE FREE CPBS
NO
YES
ADD FOUR TO THE CPB ADDRESS
GET THE LCB ADDRESS, NO. OF THE CURRENT CPB, AND COUNT OF DATA MOVED
ADD ONE TO SCBCPBNO; CLEAR THE WORK AREA
SET INTO THE SCBSCSEG FIELD THE NUMBER OF THE RECORD JUST READ
BUILD READ KEY AND DATA CCWS
FA-17 A3
EXCPINQ2
ADD A CPB TO THE CHANNEL PROGRAM
RETURN
CKENQ
WERE ANY CPBS GOTTEN FOR THE ERB
NO
YES
ENQUEUE
PUT THE ELEMENT FIRST ON THE NO-CPB QUEUE
FA1 E3
OFFSET
FA-3,C5
FA-4,K2
FA-6,K1
NO
IS THIS A RECALL OR RECEIVE
YES
IS THIS AN IEDQBD RECALL
YES
NO
GET THE DESTINATION FROM THE BUFFER PREFIX
IS THIS A HEADER BUFFER
NO
YES
TERMENTR
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
IS BUFFER SIZE SPECIFIED
YES
NO
USEDCB
GET THE DCB ADDRESS, THE BUFFER SIZE, AND THE NUMBER OF UNITS
SIZEDONE
BUILD THE ERB
RETURN
USELCB
FA-4,B2
FA-6,J2
E4
E4
USELCB
GET THE LCBTTCIN OFFSET
GET THE BUFFER SIZE
CALCULATE THE NUMBER OF UNITS
SIZECK
FA-6,D3
IS THIS A RECALL OR RECEIVE
NO
YES
IS THE REQUEST FROM IEDQBD
NO
YES
SIZECK1
SET UP TO BUILD THE SAME BUFFER SIZE
RETURN

Chart FA-21 (A21) CPB INITIALIZATION

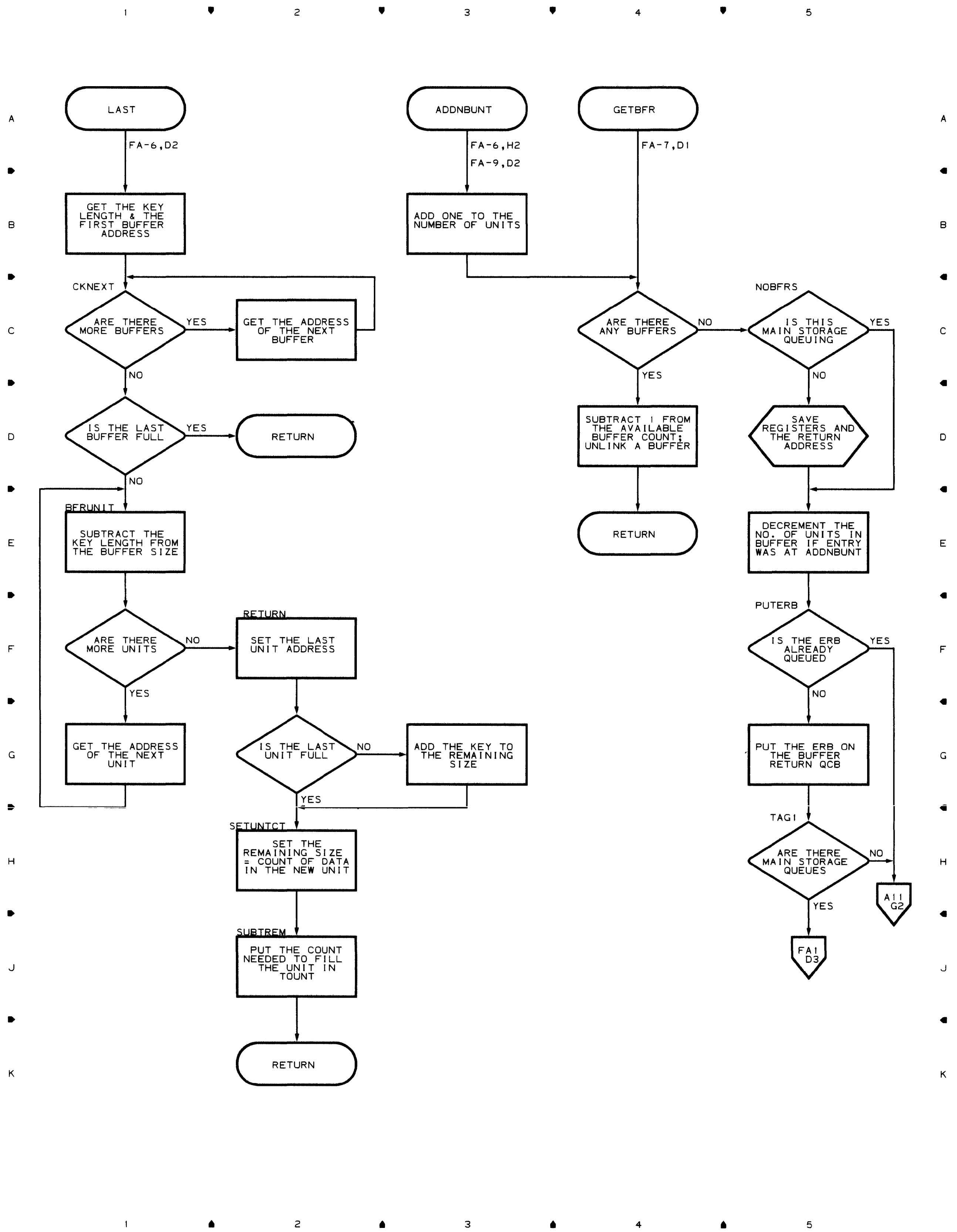
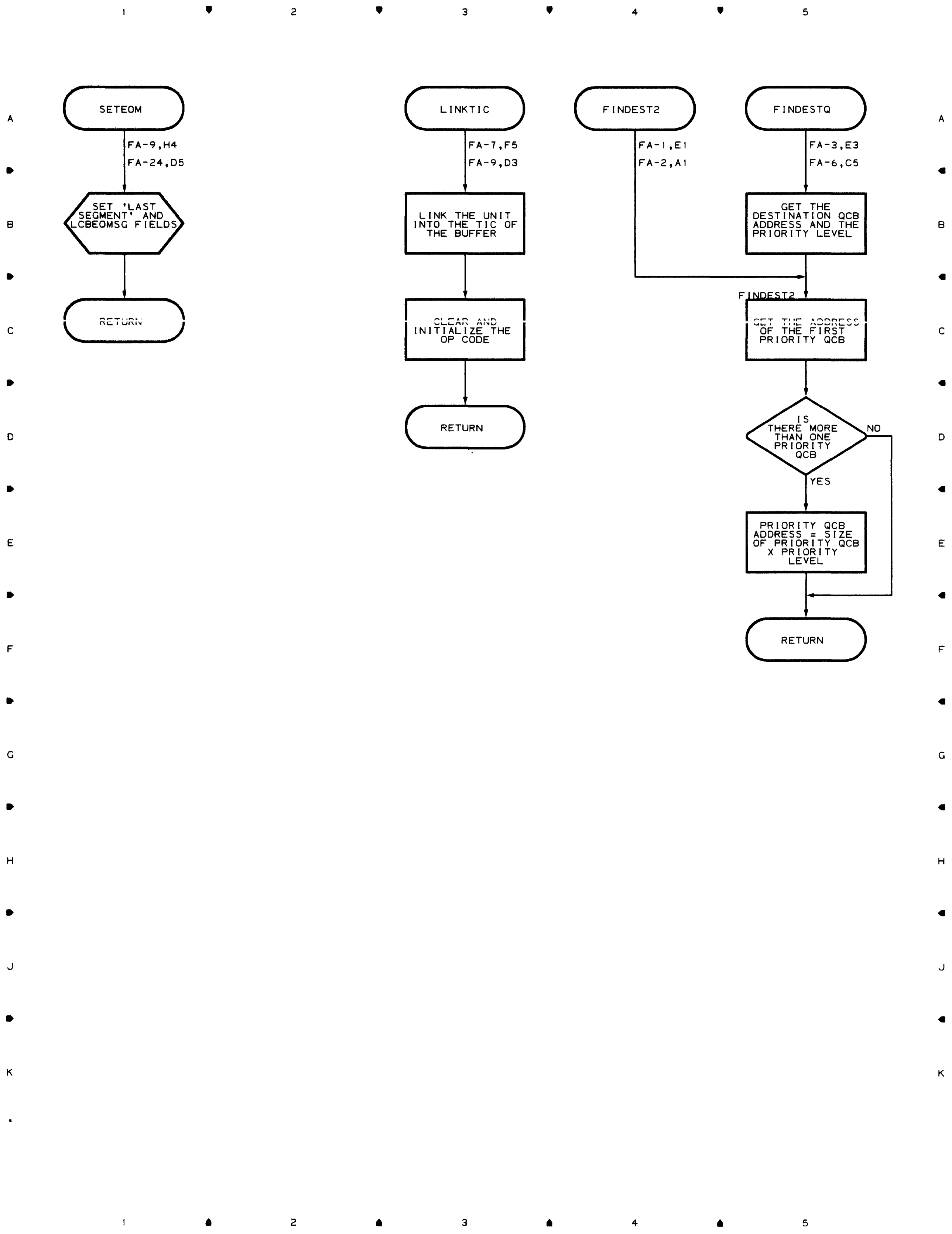

LAST
FA-6,D2
GET THE KEY LENGTH & THE FIRST BUFFER ADDRESS
CKNEXT
ARE THERE MORE BUFFERS
YES
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE LAST BUFFER FULL
YES
RETURN
NO
BFRUNIT
SUBTRACT THE KEY LENGTH FROM THE BUFFER SIZE
ARE THERE MORE UNITS
NO
RETURN
SET THE LAST UNIT ADDRESS
YES
GET THE ADDRESS OF THE NEXT UNIT
IS THE LAST UNIT FULL
NO
ADD THE KEY TO THE REMAINING SIZE
YES
SETUNTCT
SET THE REMAINING SIZE = COUNT OF DATA IN THE NEW UNIT
SUBTREM
PUT THE COUNT NEEDED TO FILL THE UNIT IN TOUNT
RETURN
ADDNBUNT
FA-6,H2
FA-9,D2
ADD ONE TO THE NUMBER OF UNITS
GETBFR
FA-7,D1
ARE THERE ANY BUFFERS
NO
YES
SUBTRACT 1 FROM THE AVAILABLE BUFFER COUNT; UNLINK A BUFFER
RETURN
NOBFRS
IS THIS MAIN STORAGE QUEUING
YES
NO
SAVE REGISTERS AND THE RETURN ADDRESS
DECREMENT THE NO. OF UNITS IN BUFFER IF ENTRY WAS AT ADDNBUNT
PUTERB
IS THE ERB ALREADY QUEUED
YES
NO
PUT THE ERB ON THE BUFFER RETURN QCB
TAG1
ARE THERE MAIN STORAGE QUEUES
NO
A11 G2
YES
FA1 D3
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA-22 (A22) CPB INITIALIZATION



SETEOM
FA-9,H4
FA-24,D5
SET 'LAST SEGMENT' AND LCBEOMSG FIELDS
RETURN
LINKTIC
FA-7,F5
FA-9,D3
LINK THE UNIT INTO THE TIC OF THE BUFFER
CLEAR AND INITIALIZE THE OP CODE
RETURN
FINDEST2
FA-1,E1
FA-2,A1
FINDESTQ
FA-3,E3
FA-6,C5
GET THE DESTINATION QCB ADDRESS AND THE PRIORITY LEVEL
FINDEST2
GET THE ADDRESS OF THE FIRST PRIORITY QCB
IS THERE MORE THAN ONE PRIORITY QCB
NO
YES
PRIORITY QCB ADDRESS = SIZE OF PRIORITY QCB X PRIORITY LEVEL
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA-23 (A23) CPB INITIALIZATION

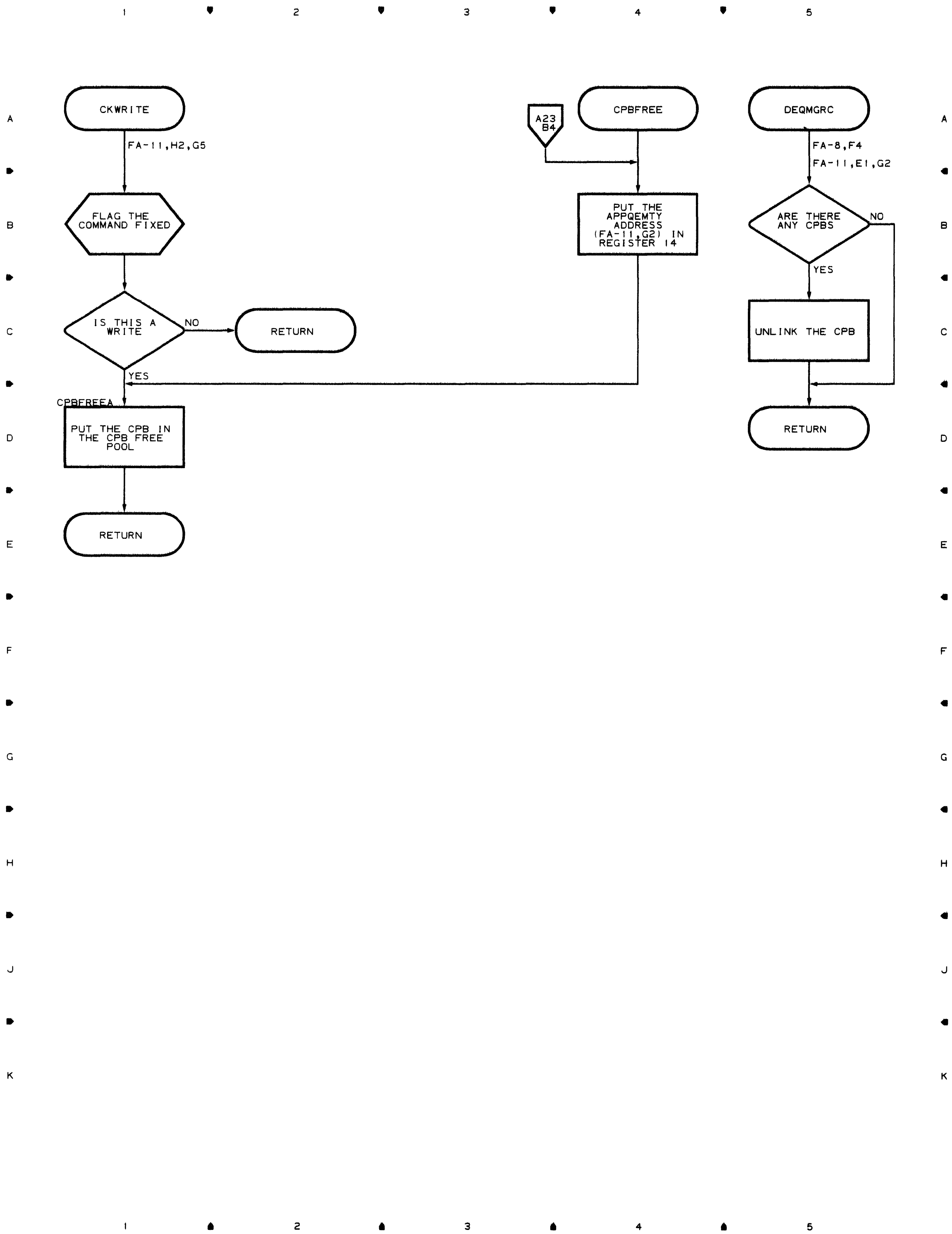


Chart FA-24 (A24) CPB INITIALIZATION

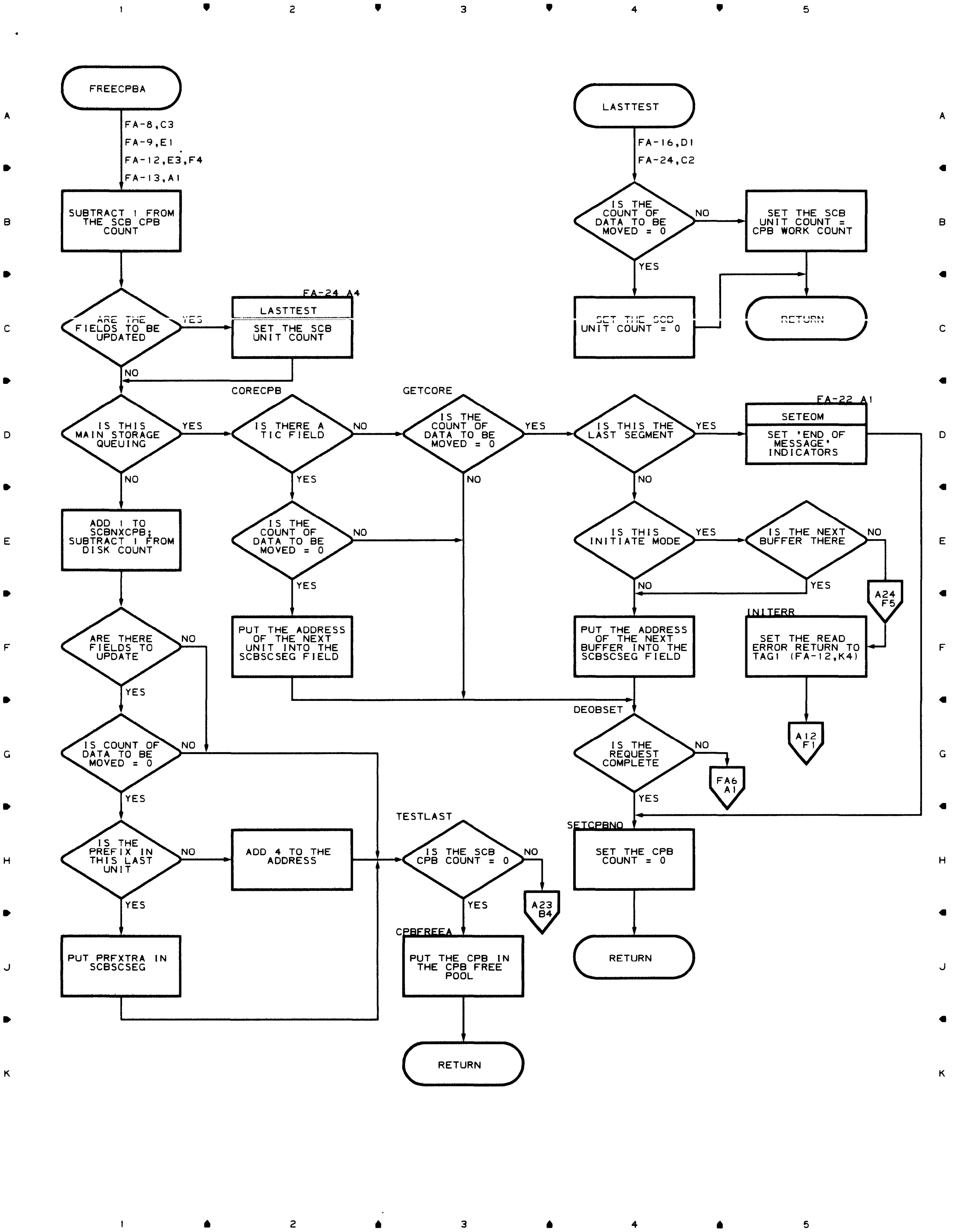

FREECPBA
FA-8,C3
FA-9,E1
FA-12,E3,F4
FA-13,A1
SUBTRACT 1 FROM THE SCB CPB COUNT
ARE THE FIELDS TO BE UPDATED
YES
NO
FA-24 A4
LASTTEST
SET THE SCB UNIT COUNT
IS THIS MAIN STORAGE QUEUING
YES
NO
CORECPB
IS THERE A TIC FIELD
NO
YES
GETCORE
IS THE COUNT OF DATA TO BE MOVED = 0
YES
NO
IS THIS THE LAST SEGMENT
YES
NO
FA-22 A1
SETEOM
SET 'END OF MESSAGE' INDICATORS
ADD 1 TO SCBNXCPB; SUBTRACT 1 FROM DISK COUNT
IS THE COUNT OF DATA TO BE MOVED = 0
NO
YES
IS THIS INITIATE MODE
YES
NO
IS THE NEXT BUFFER THERE
NO
YES
A24 F5
ARE THERE FIELDS TO UPDATE
NO
YES
PUT THE ADDRESS OF THE NEXT UNIT INTO THE SCBSCSEG FIELD
PUT THE ADDRESS OF THE NEXT BUFFER INTO THE SCBSCSEG FIELD
INITERR
SET THE READ ERROR RETURN TO TAG1 (FA-12,K4)
DEOBSET
IS COUNT OF DATA TO BE MOVED = 0
NO
YES
IS THE REQUEST COMPLETE
NO
YES
FA6 A1
A12 F1
TESTLAST
IS THE PREFIX IN THIS LAST UNIT
NO
YES
ADD 4 TO THE ADDRESS
IS THE SCB CPB COUNT = 0
NO
YES
A23 B4
SETCPBNO
SET THE CPB COUNT = 0
CPBFREEA
PUT PRFXTRA IN SCBSCSEG
PUT THE CPB IN THE CPB FREE POOL
RETURN
RETURN
LASTTEST
FA-16,D1
FA-24,C2
IS THE COUNT OF DATA TO BE MOVED = 0
NO
YES
SET THE SCB UNIT COUNT = CPB WORK COUNT
SET THE SCB UNIT COUNT = 0
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-1 (1-1) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

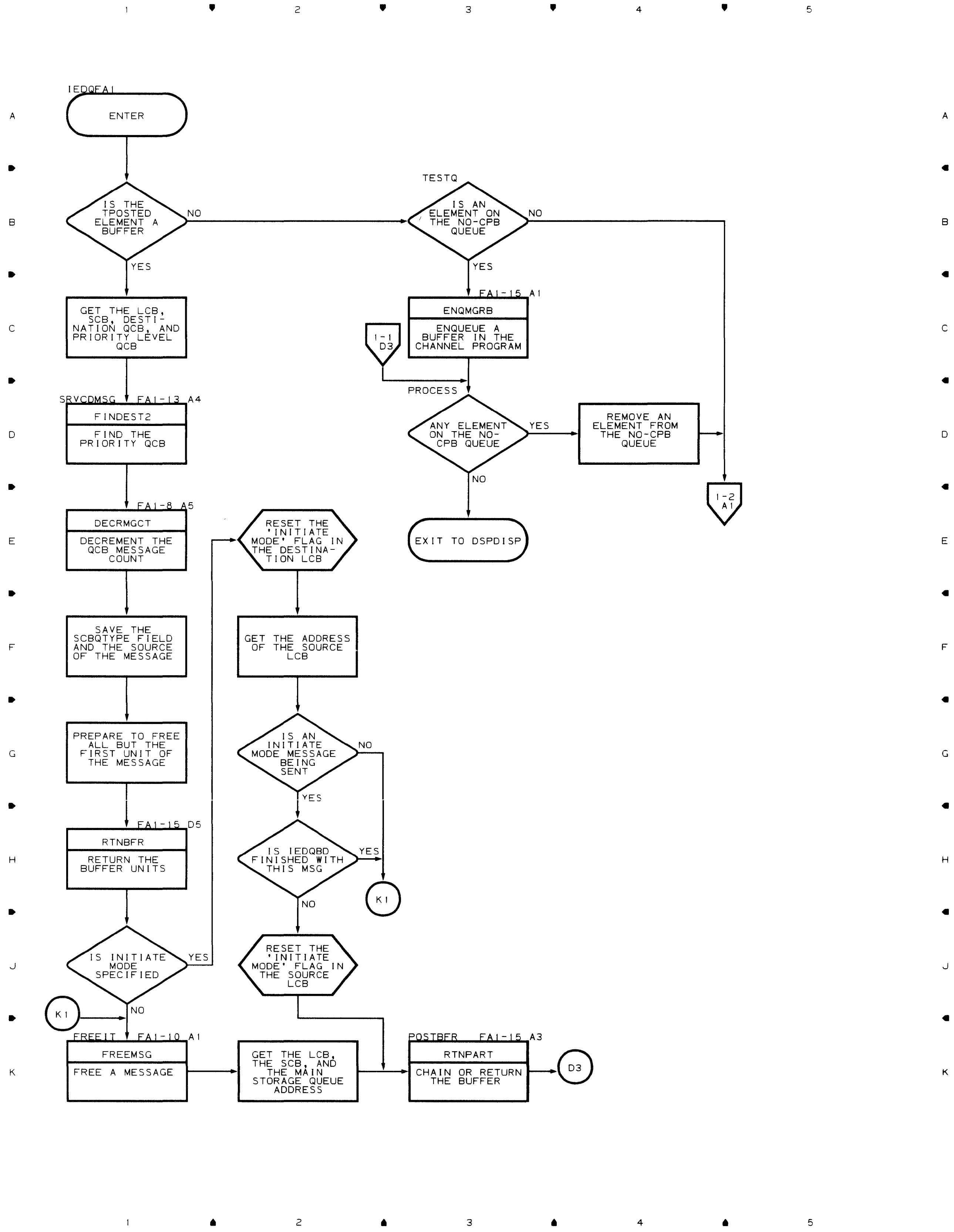
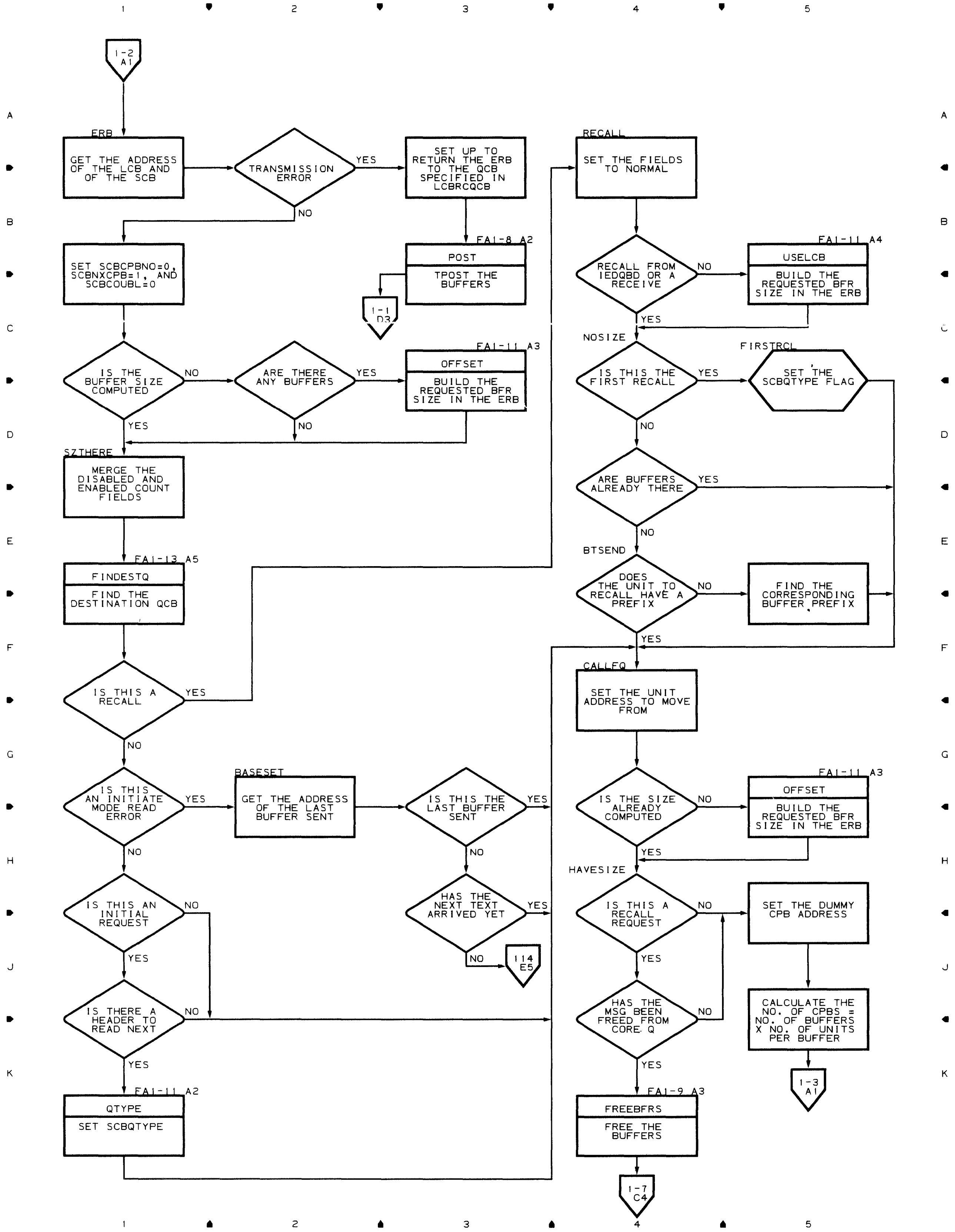

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
IEDQFA1
ENTER
IS THE TPOSTED ELEMENT A BUFFER
NO
YES
GET THE LCB, SCB, DESTINATION QCB, AND PRIORITY LEVEL QCB
SRVCDMSG
FA1-13 A4
FINDEST2
FIND THE PRIORITY QCB
FA1-8 A5
DECRMGCT
DECREMENT THE QCB MESSAGE COUNT
SAVE THE SCBQTYPE FIELD AND THE SOURCE OF THE MESSAGE
PREPARE TO FREE ALL BUT THE FIRST UNIT OF THE MESSAGE
FA1-15 D5
RTNBFR
RETURN THE BUFFER UNITS
IS INITIATE MODE SPECIFIED
YES
NO
K1
FREEIT
FA1-10 A1
FREEMSG
FREE A MESSAGE
RESET THE 'INITIATE MODE' FLAG IN THE DESTINATION LCB
GET THE ADDRESS OF THE SOURCE LCB
IS AN INITIATE MODE MESSAGE BEING SENT
NO
YES
IS IEDQBD FINISHED WITH THIS MSG
YES
NO
K1
RESET THE 'INITIATE MODE' FLAG IN THE SOURCE LCB
GET THE LCB, THE SCB, AND THE MAIN STORAGE QUEUE ADDRESS
TESTQ
IS AN ELEMENT ON THE NO-CPB QUEUE
NO
YES
FA1-15 A1
ENQMGRB
ENQUEUE A BUFFER IN THE CHANNEL PROGRAM
1-1 D3
PROCESS
ANY ELEMENT ON THE NO-CPB QUEUE
YES
NO
REMOVE AN ELEMENT FROM THE NO-CPB QUEUE
1-2 A1
EXIT TO DSPDISP
POSTBFR
FA1-15 A3
RTNPART
CHAIN OR RETURN THE BUFFER
D3

Chart FA1-2 (1-2) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
1-2 A1
ERB
GET THE ADDRESS OF THE LCB AND OF THE SCB
TRANSMISSION ERROR
YES
NO
SET UP TO RETURN THE ERB TO THE QCB SPECIFIED IN LCBRCQCB
FA1-8 A2
POST
TPOST THE BUFFERS
1-1 D3
SET SCBCPBNO=0, SCBNXCPB=1, AND SCBCOUBL=0
IS THE BUFFER SIZE COMPUTED
NO
YES
ARE THERE ANY BUFFERS
YES
NO
FA1-11 A3
OFFSET
BUILD THE REQUESTED BFR SIZE IN THE ERB
SZTHERE
MERGE THE DISABLED AND ENABLED COUNT FIELDS
FA1-13 A5
FINDESTQ
FIND THE DESTINATION QCB
IS THIS A RECALL
YES
NO
IS THIS AN INITIATE MODE READ ERROR
YES
NO
BASESET
GET THE ADDRESS OF THE LAST BUFFER SENT
IS THIS THE LAST BUFFER SENT
YES
NO
HAS THE NEXT TEXT ARRIVED YET
YES
NO
114 E5
IS THIS AN INITIAL REQUEST
NO
YES
IS THERE A HEADER TO READ NEXT
NO
YES
FA1-11 A2
QTYPE
SET SCBQTYPE
RECALL
SET THE FIELDS TO NORMAL
RECALL FROM IEDQBD OR A RECEIVE
NO
YES
FA1-11 A4
USELCB
BUILD THE REQUESTED BFR SIZE IN THE ERB
NOSIZE
IS THIS THE FIRST RECALL
YES
NO
FIRSTRCL
SET THE SCBQTYPE FLAG
ARE BUFFERS ALREADY THERE
YES
NO
BTSEND
DOES THE UNIT TO RECALL HAVE A PREFIX
NO
YES
FIND THE CORRESPONDING BUFFER PREFIX
CALLFQ
SET THE UNIT ADDRESS TO MOVE FROM
IS THE SIZE ALREADY COMPUTED
NO
YES
FA1-11 A3
OFFSET
BUILD THE REQUESTED BFR SIZE IN THE ERB
HAVESIZE
IS THIS A RECALL REQUEST
NO
YES
SET THE DUMMY CPB ADDRESS
HAS THE MSG BEEN FREED FROM CORE Q
NO
YES
CALCULATE THE NO. OF CPBS = NO. OF BUFFERS X NO. OF UNITS PER BUFFER
1-3 A1
FA1-9 A3
FREEBFRS
FREE THE BUFFERS
1-7 C4

Chart FA1-3 (1-3) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

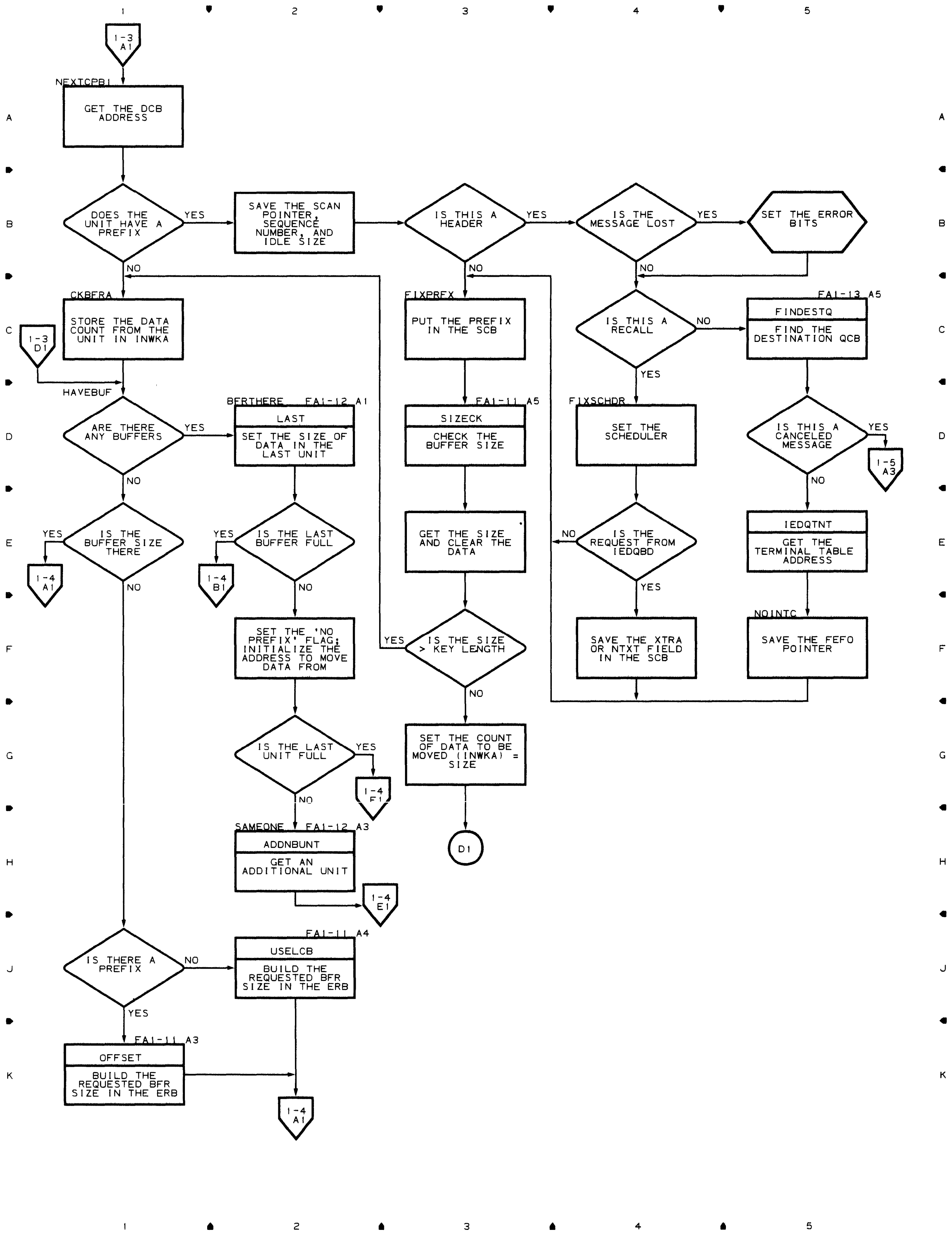


Chart FA1-4 (1-4) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

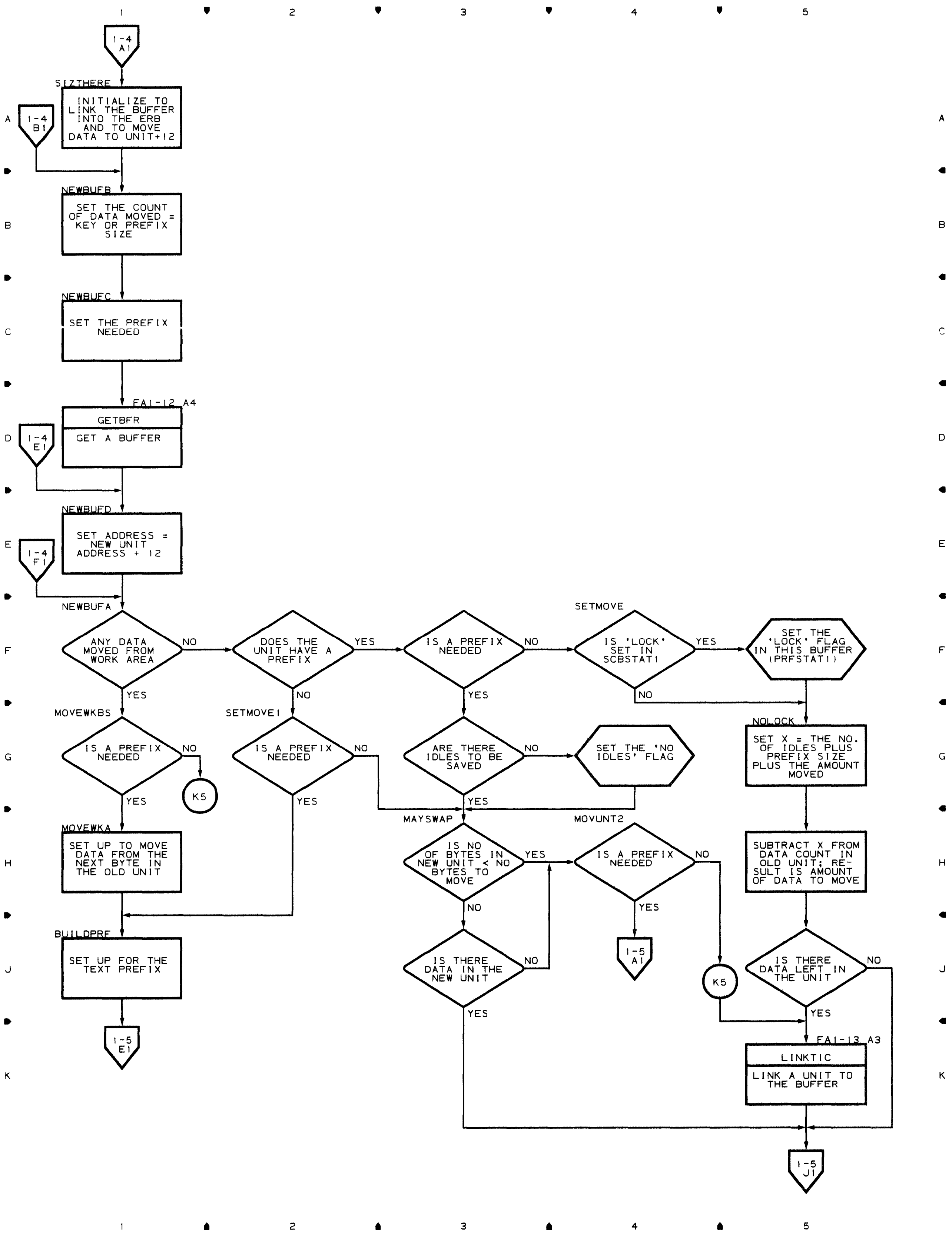

1-4 A1
SIZTHERE
INITIALIZE TO LINK THE BUFFER INTO THE ERB AND TO MOVE DATA TO UNIT+12
1-4 B1
NEWBUFB
SET THE COUNT OF DATA MOVED = KEY OR PREFIX SIZE
NEWBUFC
SET THE PREFIX NEEDED
FA1-12 A4
GETBFR
GET A BUFFER
1-4 E1
NEWBUFD
SET ADDRESS = NEW UNIT ADDRESS + 12
1-4 F1
NEWBUFA
ANY DATA MOVED FROM WORK AREA
NO
DOES THE UNIT HAVE A PREFIX
YES
IS A PREFIX NEEDED
NO
SETMOVE
IS 'LOCK' SET IN SCBSTAT1
YES
SET THE 'LOCK' FLAG IN THIS BUFFER (PRFSTAT1)
YES
MOVEWKBS
IS A PREFIX NEEDED
NO
K5
YES
NO
SETMOVE1
IS A PREFIX NEEDED
NO
YES
YES
ARE THERE IDLES TO BE SAVED
NO
SET THE 'NO IDLES' FLAG
NO
NOLOCK
SET X = THE NO. OF IDLES PLUS PREFIX SIZE PLUS THE AMOUNT MOVED
YES
MOVEWKA
SET UP TO MOVE DATA FROM THE NEXT BYTE IN THE OLD UNIT
MAYSWAP
IS NO OF BYTES IN NEW UNIT < NO BYTES TO MOVE
YES
MOVUNT2
IS A PREFIX NEEDED
NO
SUBTRACT X FROM DATA COUNT IN OLD UNIT; RESULT IS AMOUNT OF DATA TO MOVE
NO
YES
BUILDPRF
SET UP FOR THE TEXT PREFIX
IS THERE DATA IN THE NEW UNIT
NO
1-5 A1
K5
IS THERE DATA LEFT IN THE UNIT
NO
YES
YES
FA1-13 A3
LINKTIC
LINK A UNIT TO THE BUFFER
1-5 E1
1-5 J1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-5 (1-5) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

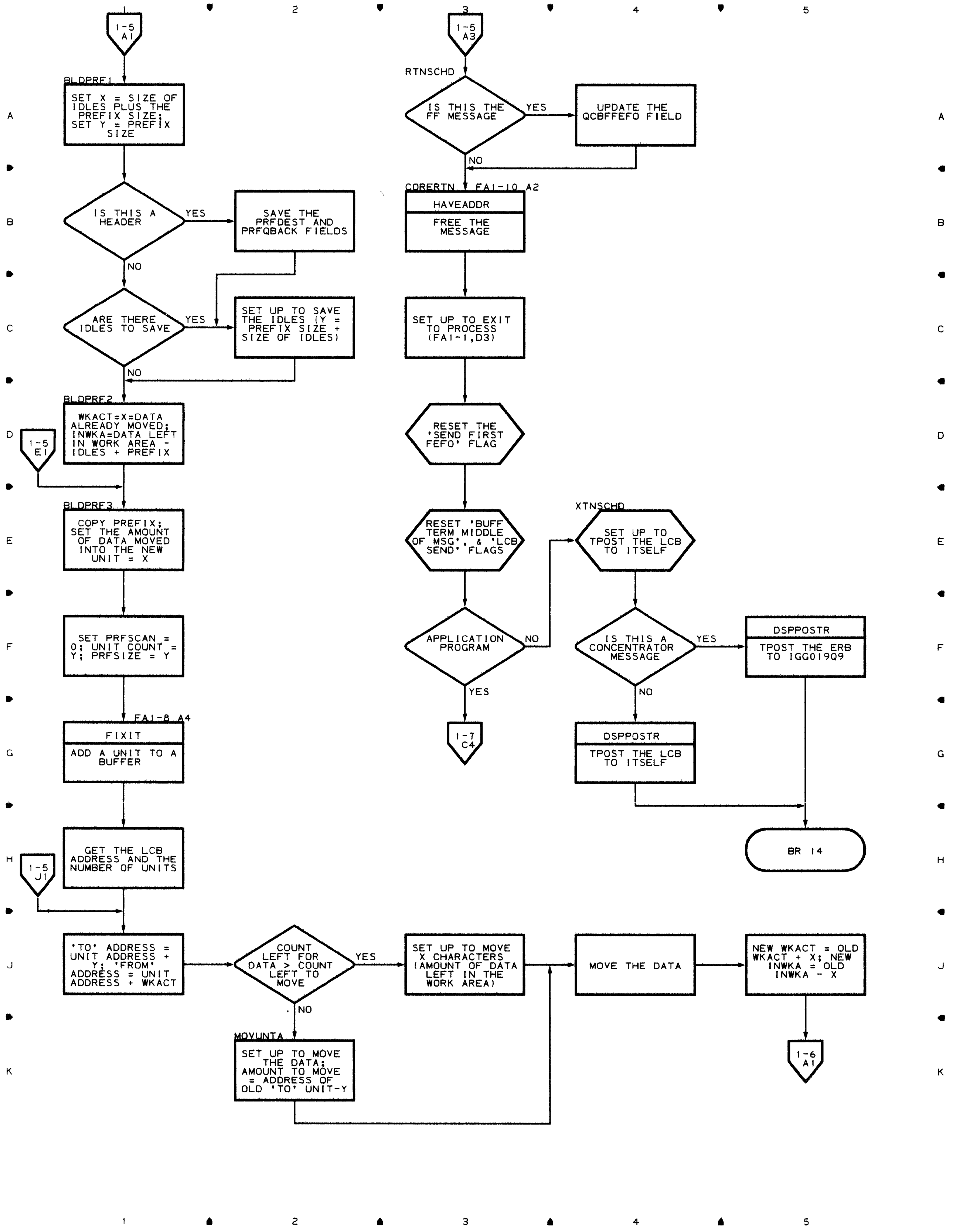

1-5 A1
BLDPREF1
SET X = SIZE OF IDLES PLUS THE PREFIX SIZE; SET Y = PREFIX SIZE
IS THIS A HEADER
YES
SAVE THE PRFDEST AND PRFQBACK FIELDS
NO
ARE THERE IDLES TO SAVE
YES
SET UP TO SAVE THE IDLES (Y = PREFIX SIZE + SIZE OF IDLES)
NO
BLDPREF2
WKACT=X=DATA ALREADY MOVED; INWKA=DATA LEFT IN WORK AREA - IDLES + PREFIX
1-5 E1
BLDPREF3
COPY PREFIX; SET THE AMOUNT OF DATA MOVED INTO THE NEW UNIT = X
SET PRFSCAN = 0; UNIT COUNT = Y; PRFSIZE = Y
FA1-8 A4
FIXIT
ADD A UNIT TO A BUFFER
GET THE LCB ADDRESS AND THE NUMBER OF UNITS
1-5 J1
'TO' ADDRESS = UNIT ADDRESS + Y; 'FROM' ADDRESS = UNIT ADDRESS + WKACT
COUNT LEFT FOR DATA > COUNT LEFT TO MOVE
YES
NO
MOVUNTA
SET UP TO MOVE THE DATA; AMOUNT TO MOVE = ADDRESS OF OLD 'TO' UNIT-Y
1-5 A3
RTNSCHD
IS THIS THE FF MESSAGE
YES
UPDATE THE QCBFFEFO FIELD
NO
CORERTN FA1-10 A2
HAVEADDR
FREE THE MESSAGE
SET UP TO EXIT TO PROCESS (FA1-1,D3)
RESET THE 'SEND FIRST FEFO' FLAG
RESET 'BUFF TERM MIDDLE OF MSG', & 'LCB SEND' FLAGS
APPLICATION PROGRAM
NO
YES
1-7 C4
XTNSCHD
SET UP TO TPOST THE LCB TO ITSELF
IS THIS A CONCENTRATOR MESSAGE
YES
NO
DSPPOSTR
TPOST THE ERB TO IGG019Q9
DSPPOSTR
TPOST THE LCB TO ITSELF
BR 14
SET UP TO MOVE X CHARACTERS (AMOUNT OF DATA LEFT IN THE WORK AREA)
MOVE THE DATA
NEW WKACT = OLD WKACT + X; NEW INWKA = OLD INWKA - X
1-6 A1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-6 (1-6) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

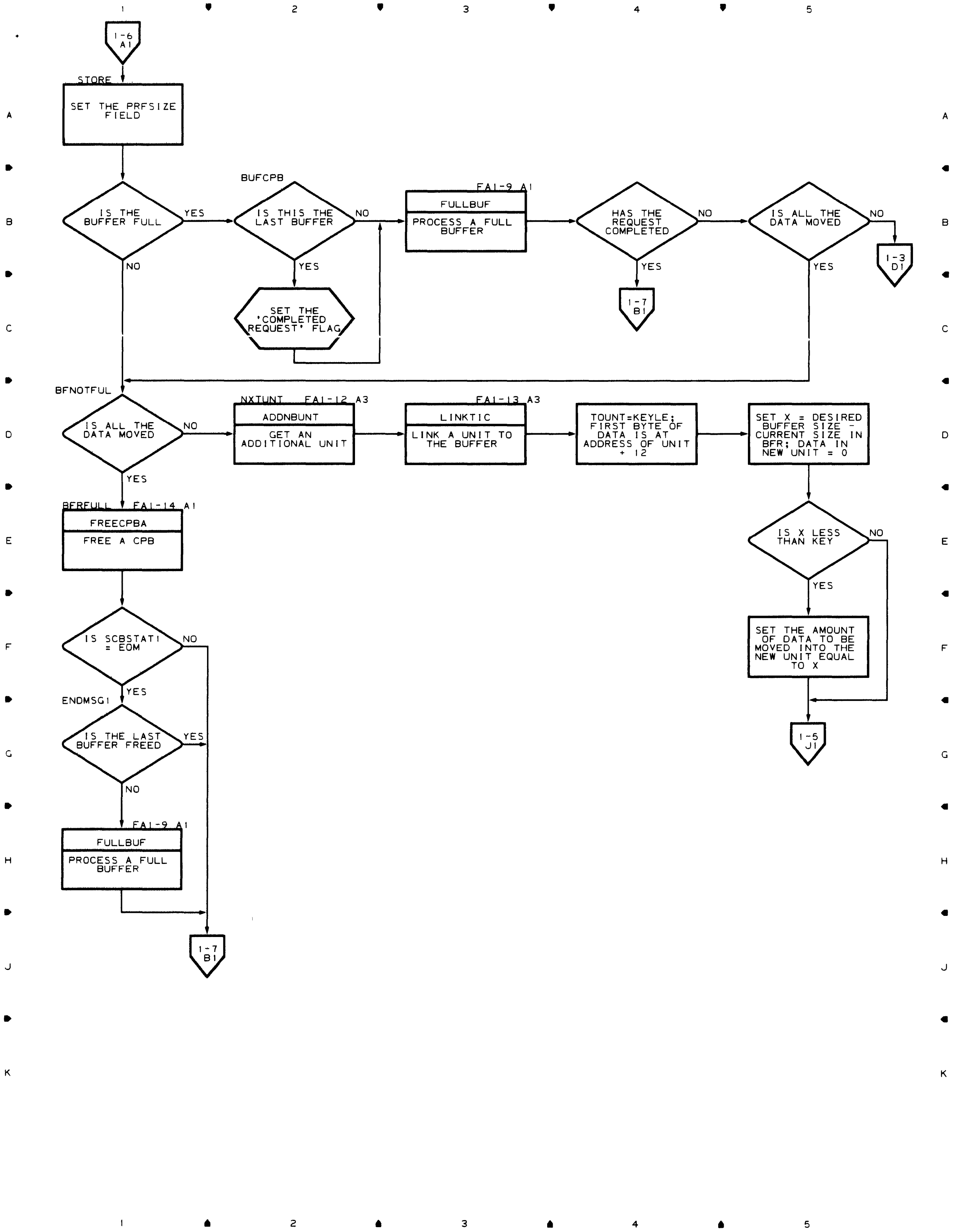


Chart FA1-7 (1-7) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

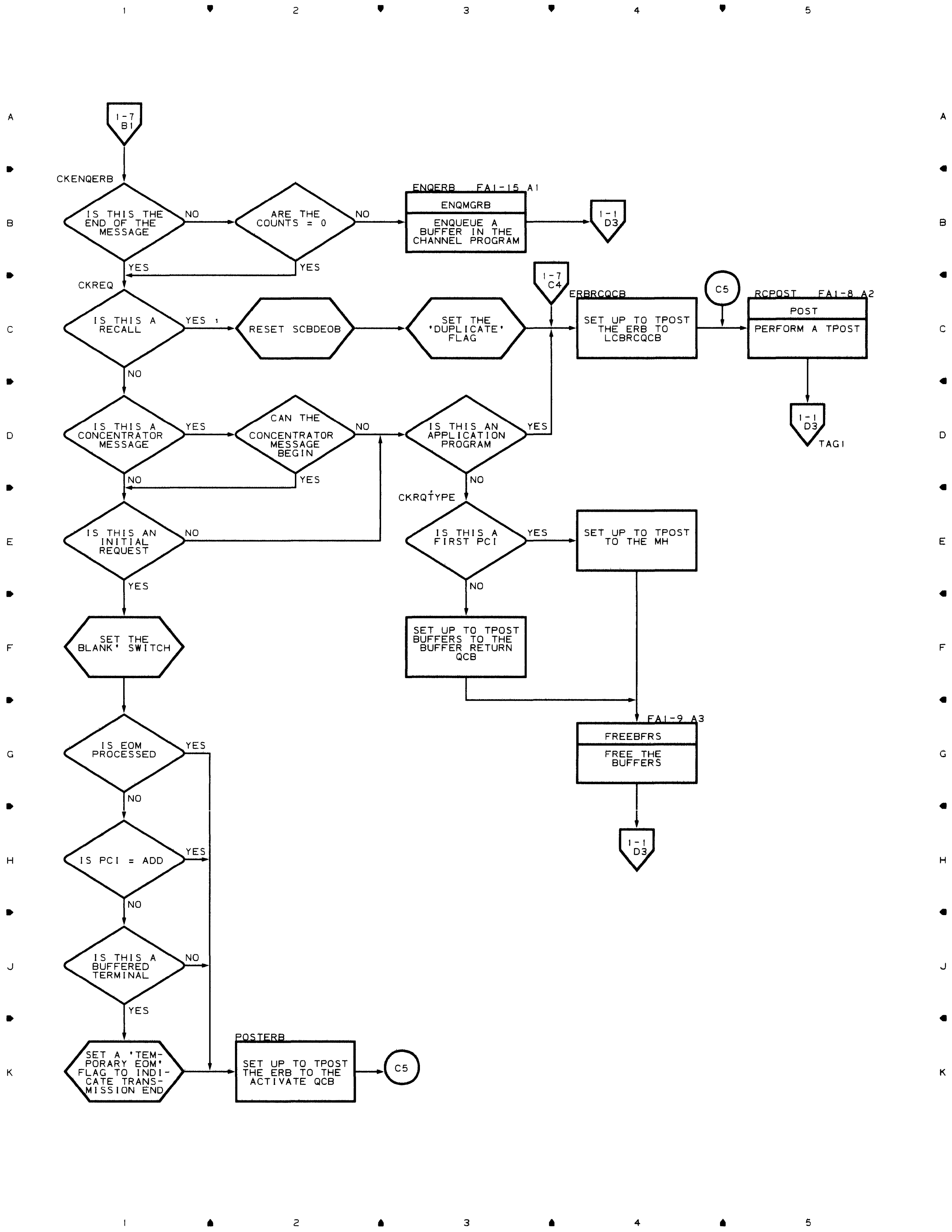
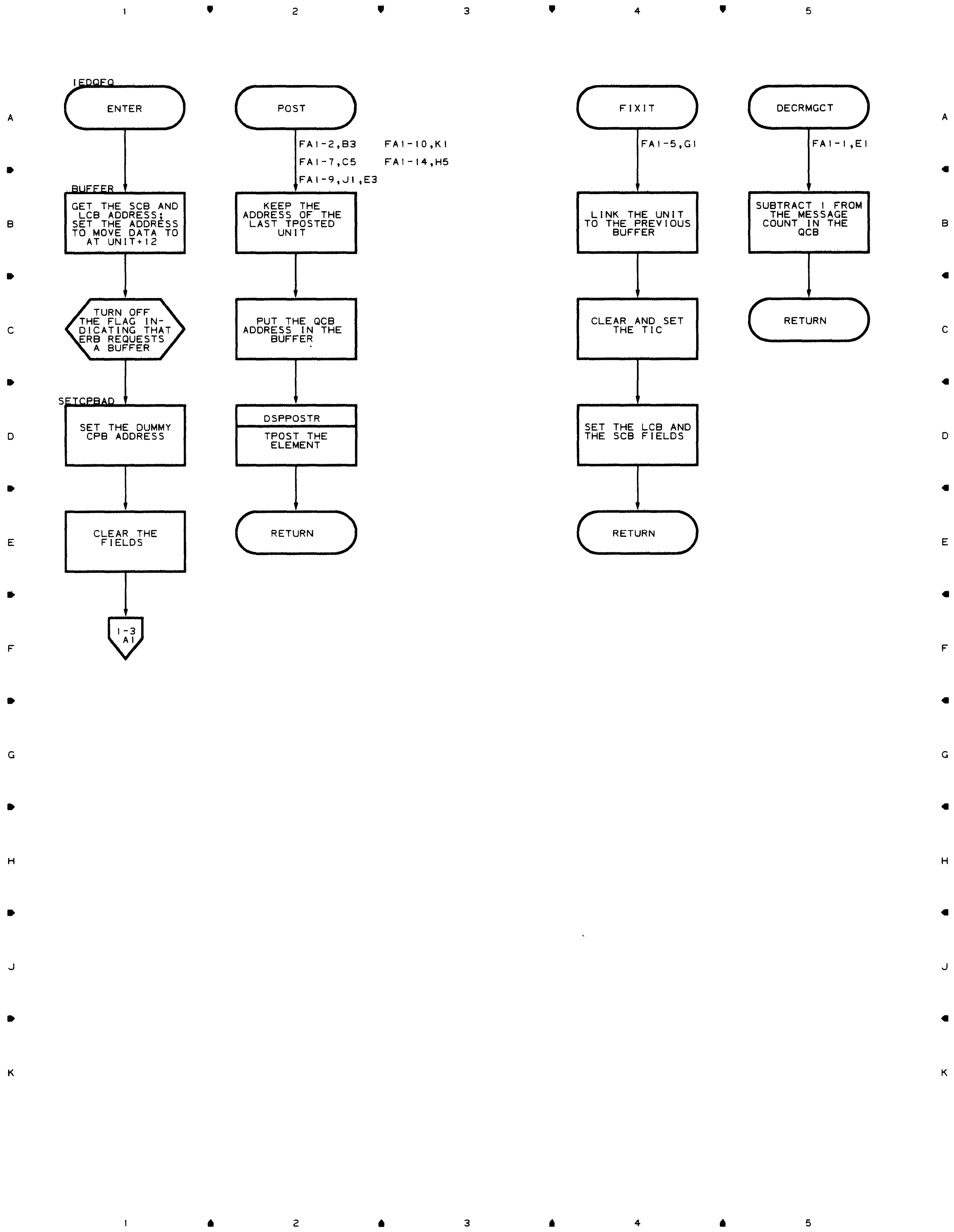

1-7 B1
CKENQERB
IS THIS THE END OF THE MESSAGE
NO
ARE THE COUNTS = 0
NO
ENQERB FA1-15 A1
ENQMGRB
ENQUEUE A BUFFER IN THE CHANNEL PROGRAM
1-1 D3
YES
YES
CKREQ
1-7 C4
C5
ERBRCQCB
RCPOST FA1-8 A2
IS THIS A RECALL
YES
RESET SCBDEOB
SET THE 'DUPLICATE' FLAG
SET UP TO TPOST THE ERB TO LCBRCQCB
POST
PERFORM A TPOST
NO
1-1 D3
TAG1
IS THIS A CONCENTRATOR MESSAGE
YES
CAN THE CONCENTRATOR MESSAGE BEGIN
NO
IS THIS AN APPLICATION PROGRAM
YES
NO
YES
NO
CKRQTYPE
IS THIS AN INITIAL REQUEST
NO
IS THIS A FIRST PCI
YES
SET UP TO TPOST TO THE MH
YES
NO
SET THE BLANK' SWITCH
SET UP TO TPOST BUFFERS TO THE BUFFER RETURN QCB
FA1-9 A3
FREEBFRS
FREE THE BUFFERS
IS EOM PROCESSED
YES
NO
1-1 D3
IS PCI = ADD
YES
NO
IS THIS A BUFFERED TERMINAL
NO
YES
POSTERB
SET A 'TEMPORARY EOM' FLAG TO INDICATE TRANSMISSION END
SET UP TO TPOST THE ERB TO THE ACTIVATE QCB
C5
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-8 (1-8) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING


IEDQFQ
ENTER
BUFFER
GET THE SCB AND LCB ADDRESS; SET THE ADDRESS TO MOVE DATA TO AT UNIT+12
TURN OFF THE FLAG INDICATING THAT ERB REQUESTS A BUFFER
SETCPBAD
SET THE DUMMY CPB ADDRESS
CLEAR THE FIELDS
1-3 A1
POST
FA1-2,B3
FA1-7,C5
FA1-9,J1,E3
FA1-10,K1
FA1-14,H5
KEEP THE ADDRESS OF THE LAST TPOSTED UNIT
PUT THE QCB ADDRESS IN THE BUFFER
DSPPOSTR
TPOST THE ELEMENT
RETURN
FIXIT
FA1-5,G1
LINK THE UNIT TO THE PREVIOUS BUFFER
CLEAR AND SET THE TIC
SET THE LCB AND THE SCB FIELDS
RETURN
DECRMGCT
FA1-1,E1
SUBTRACT 1 FROM THE MESSAGE COUNT IN THE QCB
RETURN

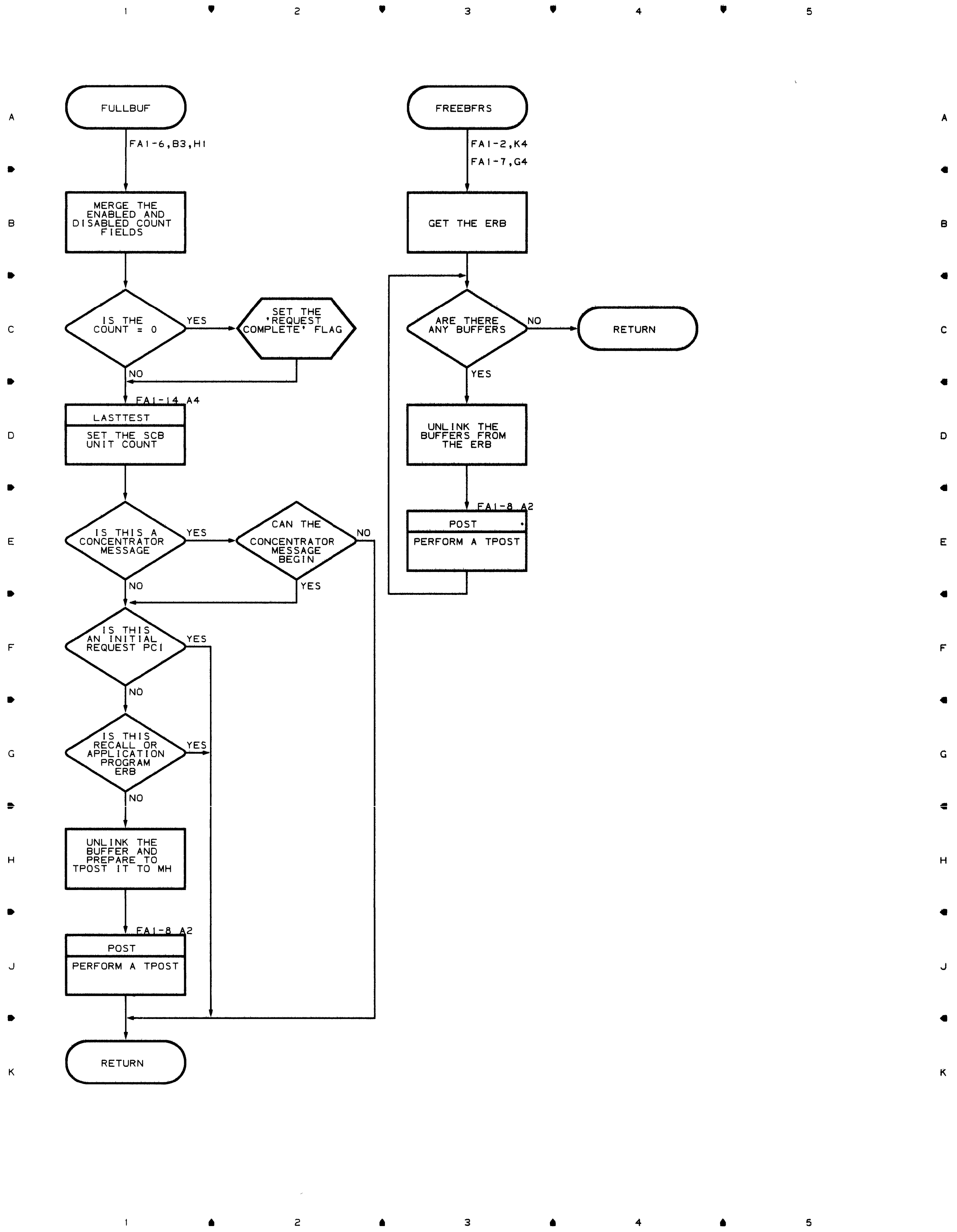
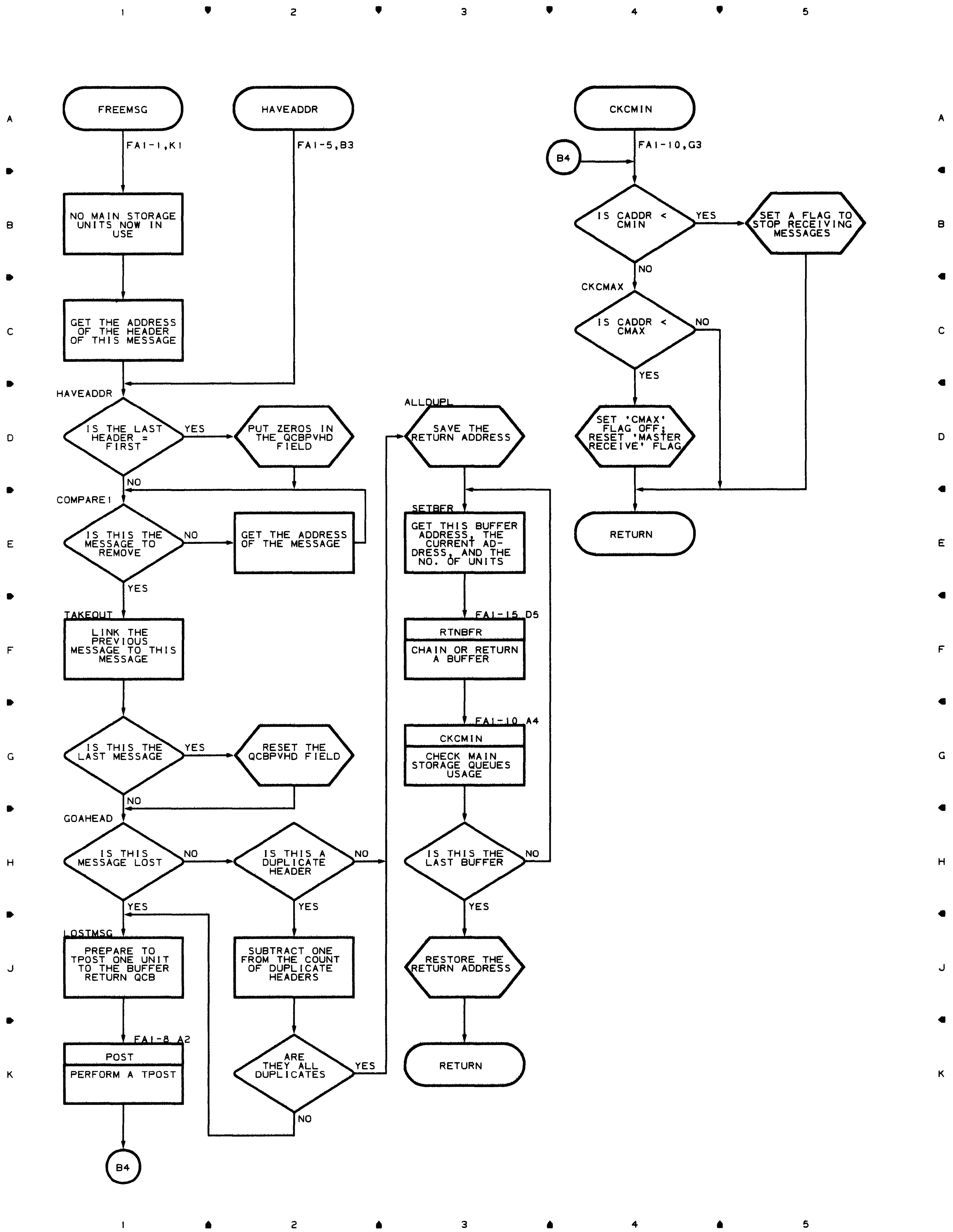

FULLBUF
FA1-6,B3,H1
MERGE THE ENABLED AND DISABLED COUNT FIELDS
IS THE COUNT = 0
YES
SET THE 'REQUEST COMPLETE' FLAG
NO
FA1-14 A4
LASTTEST
SET THE SCB UNIT COUNT
IS THIS A CONCENTRATOR MESSAGE
YES
CAN THE CONCENTRATOR MESSAGE BEGIN
NO
YES
NO
IS THIS AN INITIAL REQUEST PCI
YES
NO
IS THIS RECALL OR APPLICATION PROGRAM ERB
YES
NO
UNLINK THE BUFFER AND PREPARE TO TPOST IT TO MH
FA1-8 A2
POST
PERFORM A TPOST
RETURN
FREEBFRS
FA1-2,K4
FA1-7,G4
GET THE ERB
ARE THERE ANY BUFFERS
NO
RETURN
YES
UNLINK THE BUFFERS FROM THE ERB
FA1-8 A2
POST
PERFORM A TPOST
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-10 (110) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FREEMSG
FA1-1,K1
NO MAIN STORAGE UNITS NOW IN USE
GET THE ADDRESS OF THE HEADER OF THIS MESSAGE
HAVEADDR
FA1-5,B3
HAVEADDR
IS THE LAST HEADER = FIRST
YES
PUT ZEROS IN THE QCBPVHD FIELD
NO
COMPARE1
IS THIS THE MESSAGE TO REMOVE
NO
GET THE ADDRESS OF THE MESSAGE
YES
TAKEOUT
LINK THE PREVIOUS MESSAGE TO THIS MESSAGE
IS THIS THE LAST MESSAGE
YES
RESET THE QCBPVHD FIELD
NO
GOAHEAD
IS THIS MESSAGE LOST
NO
IS THIS A DUPLICATE HEADER
NO
YES
LOSTMSG
PREPARE TO TPOST ONE UNIT TO THE BUFFER RETURN QCB
YES
SUBTRACT ONE FROM THE COUNT OF DUPLICATE HEADERS
FA1-8,A2
POST
PERFORM A TPOST
ARE THEY ALL DUPLICATES
YES
NO
B4
ALLDUPL
SAVE THE RETURN ADDRESS
SETBFR
GET THIS BUFFER ADDRESS, THE CURRENT AD- DRESS, AND THE NO. OF UNITS
FA1-15,D5
RTNBFR
CHAIN OR RETURN A BUFFER
FA1-10,A4
CKCMIN
CHECK MAIN STORAGE QUEUES USAGE
IS THIS THE LAST BUFFER
NO
YES
RESTORE THE RETURN ADDRESS
RETURN
CKCMIN
B4
FA1-10,G3
IS CADDR < CMIN
YES
SET A FLAG TO STOP RECEIVING MESSAGES
NO
CKCMAX
IS CADDR < CMAX
NO
YES
SET 'CMAX' FLAG OFF; RESET 'MASTER RECEIVE' FLAG
RETURN

Chart FA1-11 (111) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

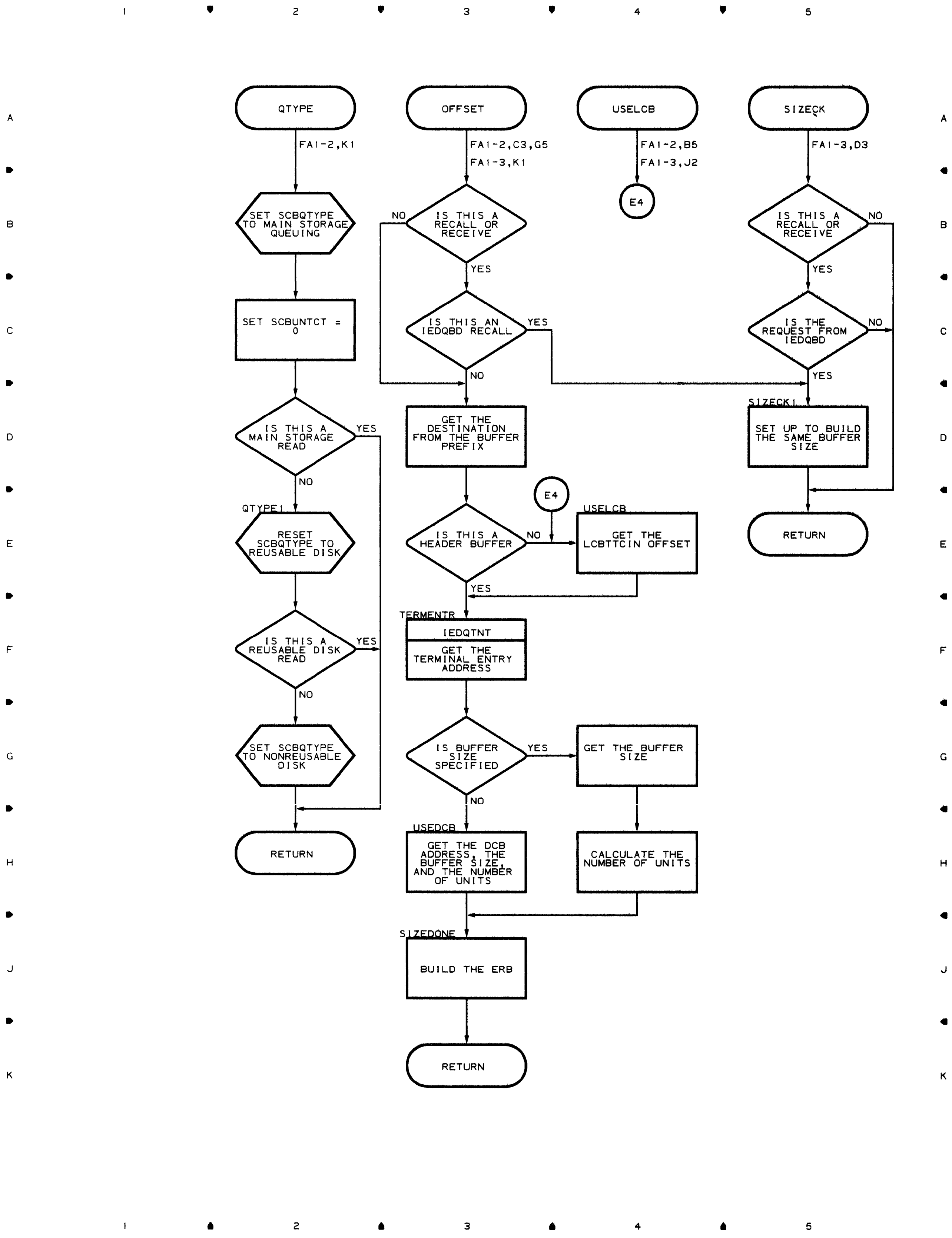

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
QTYPE
FA1-2,K1
SET SCBQTYPE TO MAIN STORAGE QUEUING
SET SCBUNTCT = 0
IS THIS A MAIN STORAGE READ
YES
NO
QTYPE1
RESET SCBQTYPE TO REUSABLE DISK
IS THIS A REUSABLE DISK READ
YES
NO
SET SCBQTYPE TO NONREUSABLE DISK
RETURN
OFFSET
FA1-2,C3,G5
FA1-3,K1
IS THIS A RECALL OR RECEIVE
NO
YES
IS THIS AN IEDQBD RECALL
YES
NO
GET THE DESTINATION FROM THE BUFFER PREFIX
IS THIS A HEADER BUFFER
NO
YES
TERMENTR
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
IS BUFFER SIZE SPECIFIED
YES
NO
USEDCB
GET THE DCB ADDRESS, THE BUFFER SIZE, AND THE NUMBER OF UNITS
SIZEDONE
BUILD THE ERB
RETURN
USELCB
FA1-2,B5
FA1-3,J2
E4
E4
USELCB
GET THE LCBTTCIN OFFSET
GET THE BUFFER SIZE
CALCULATE THE NUMBER OF UNITS
SIZECK
FA1-3,D3
IS THIS A RECALL OR RECEIVE
NO
YES
IS THE REQUEST FROM IEDQBD
NO
YES
SIZECK1
SET UP TO BUILD THE SAME BUFFER SIZE
RETURN

Chart FA1-12 (112) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

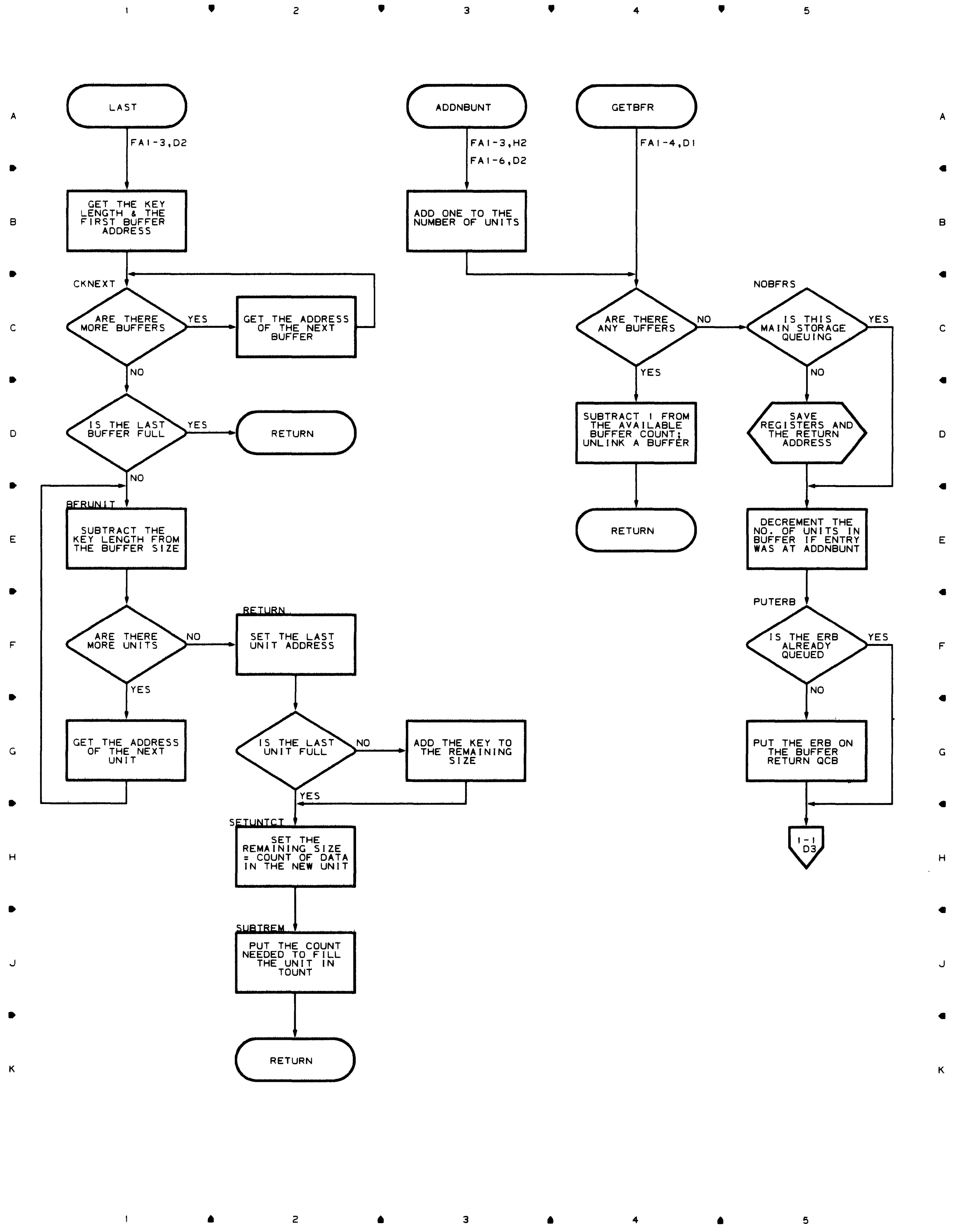

LAST
FA1-3,D2
GET THE KEY LENGTH & THE FIRST BUFFER ADDRESS
CKNEXT
ARE THERE MORE BUFFERS
YES
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE LAST BUFFER FULL
YES
RETURN
NO
BFRUNIT
SUBTRACT THE KEY LENGTH FROM THE BUFFER SIZE
ARE THERE MORE UNITS
NO
RETURN
SET THE LAST UNIT ADDRESS
YES
GET THE ADDRESS OF THE NEXT UNIT
IS THE LAST UNIT FULL
NO
ADD THE KEY TO THE REMAINING SIZE
YES
SETUNTCT
SET THE REMAINING SIZE = COUNT OF DATA IN THE NEW UNIT
SUBTREM
PUT THE COUNT NEEDED TO FILL THE UNIT IN TOUNT
RETURN
ADDNBUNT
FA1-3,H2
FA1-6,D2
ADD ONE TO THE NUMBER OF UNITS
GETBFR
FA1-4,D1
ARE THERE ANY BUFFERS
YES
SUBTRACT 1 FROM THE AVAILABLE BUFFER COUNT; UNLINK A BUFFER
RETURN
NO
NOBFRS
IS THIS MAIN STORAGE QUEUING
YES
NO
SAVE REGISTERS AND THE RETURN ADDRESS
DECREMENT THE NO. OF UNITS IN BUFFER IF ENTRY WAS AT ADDNBUNT
PUTERB
IS THE ERB ALREADY QUEUED
YES
NO
PUT THE ERB ON THE BUFFER RETURN QCB
1-1
D3
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-13 (113) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

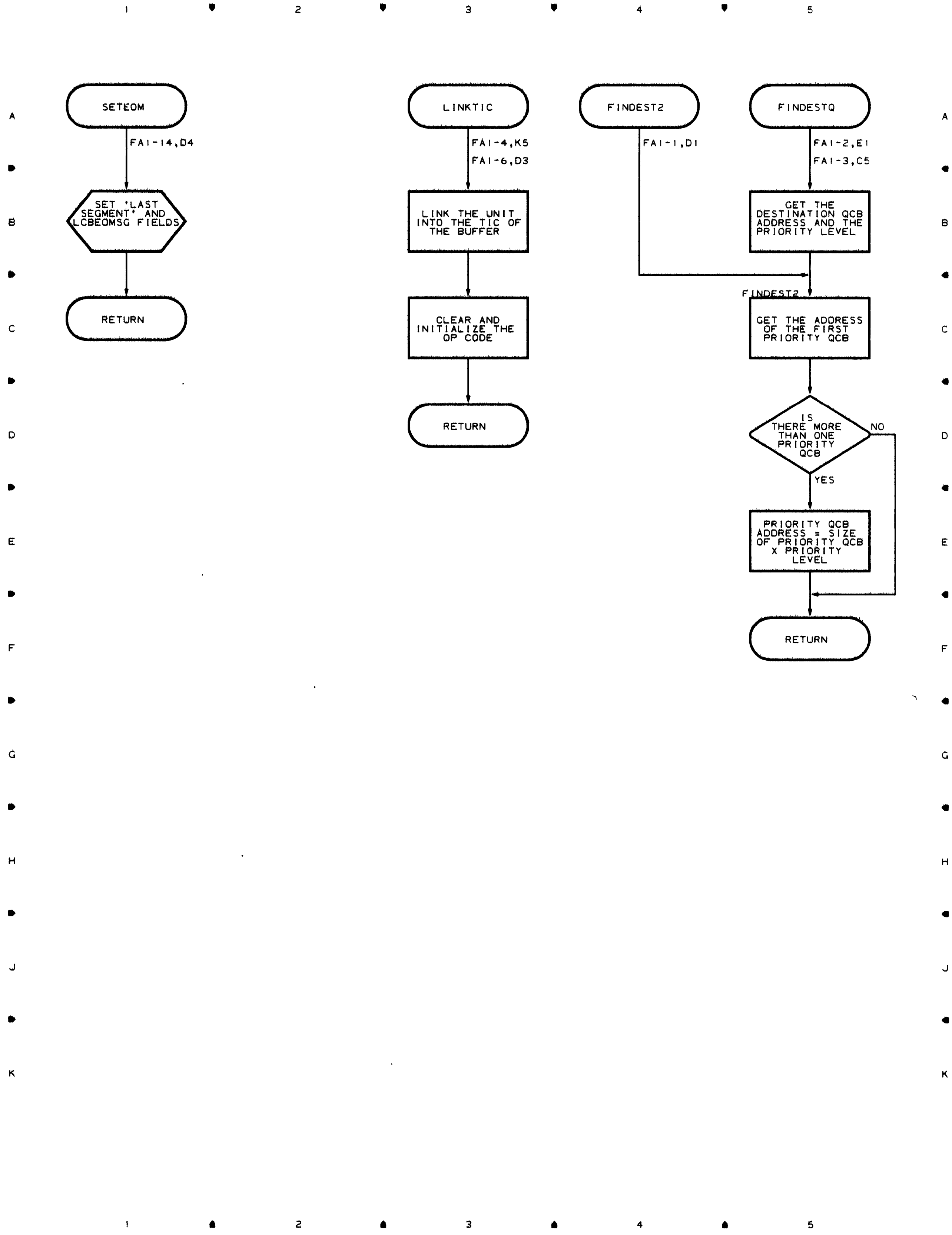

SETEOM
FA1-14,D4
SET 'LAST SEGMENT' AND LCBEOMSG FIELDS
RETURN
LINKTIC
FA1-4,K5
FA1-6,D3
LINK THE UNIT INTO THE TIC OF THE BUFFER
CLEAR AND INITIALIZE THE OP CODE
RETURN
FINDEST2
FA1-1,D1
FINDESTQ
FA1-2,E1
FA1-3,C5
GET THE DESTINATION QCB ADDRESS AND THE PRIORITY LEVEL
FINDEST2
GET THE ADDRESS OF THE FIRST PRIORITY QCB
IS THERE MORE THAN ONE PRIORITY QCB
NO
YES
PRIORITY QCB ADDRESS = SIZE OF PRIORITY QCB X PRIORITY LEVEL
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-14 (114) CPB INITIALIZATION - MAIN STORAGE ONLY QUEUING

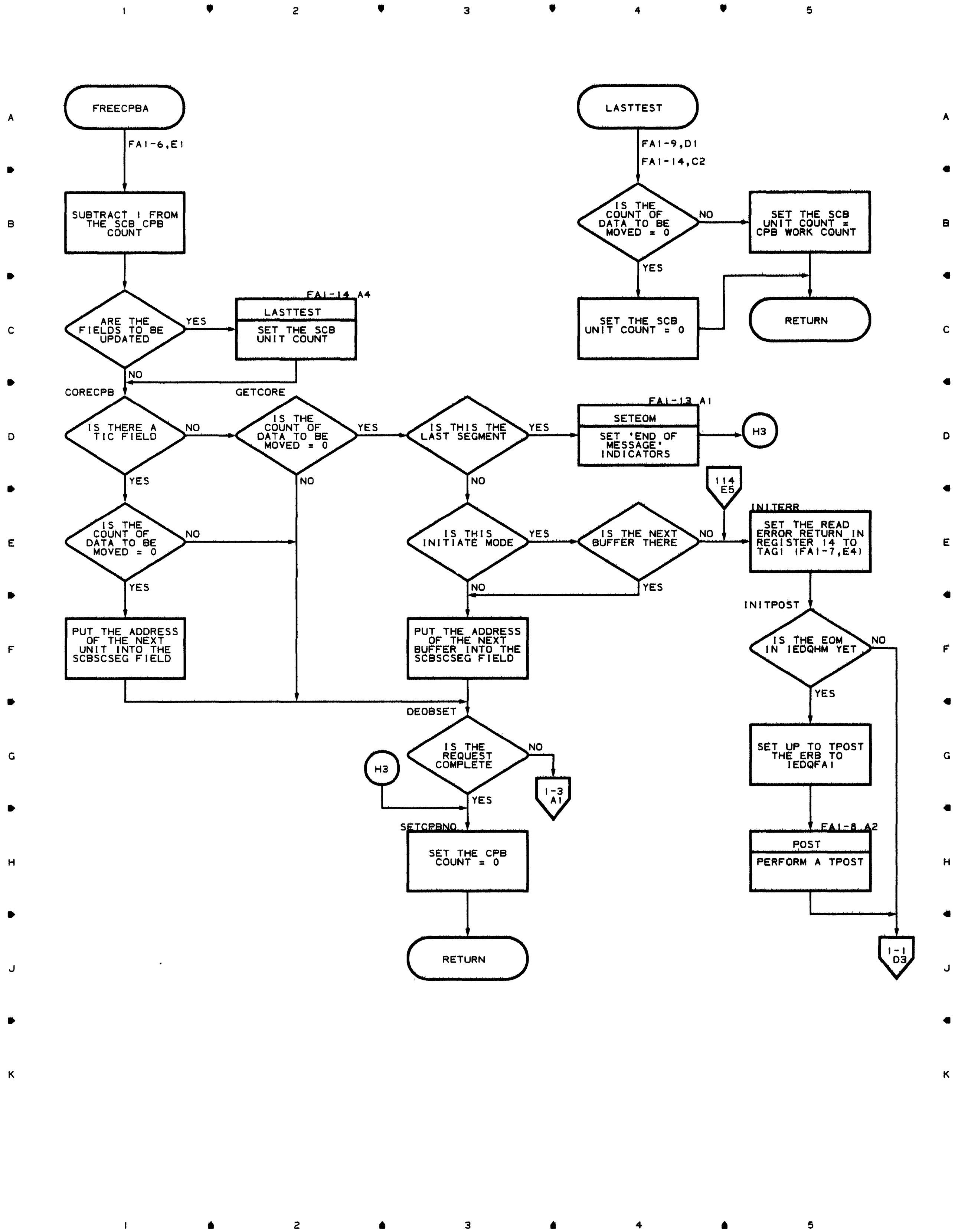

FREECPBA
FAI-6,E1
SUBTRACT 1 FROM THE SCB CPB COUNT
ARE THE FIELDS TO BE UPDATED
YES
NO
FAI-14 A4
LASTTEST
SET THE SCB UNIT COUNT
CORECPB
IS THERE A TIC FIELD
NO
YES
GETCORE
IS THE COUNT OF DATA TO BE MOVED = 0
YES
NO
IS THE COUNT OF DATA TO BE MOVED = 0
NO
YES
PUT THE ADDRESS OF THE NEXT UNIT INTO THE SCBSCSEG FIELD
IS THIS THE LAST SEGMENT
YES
NO
FAI-13 A1
SETEOM
SET 'END OF MESSAGE' INDICATORS
H3
IS THIS INITIATE MODE
YES
NO
IS THE NEXT BUFFER THERE
NO
YES
PUT THE ADDRESS OF THE NEXT BUFFER INTO THE SCBSCSEG FIELD
DEOBSET
IS THE REQUEST COMPLETE
NO
YES
1-3 A1
H3
SETCPBNO
SET THE CPB COUNT = 0
RETURN
LASTTEST
FAI-9,D1
FAI-14,C2
IS THE COUNT OF DATA TO BE MOVED = 0
NO
YES
SET THE SCB UNIT COUNT = CPB WORK COUNT
SET THE SCB UNIT COUNT = 0
RETURN
114 E5
INITERR
SET THE READ ERROR RETURN IN REGISTER 14 TO TAG1 (FAI-7,E4)
INITPOST
IS THE EOM IN IEDQHM YET
NO
YES
SET UP TO TPOST THE ERB TO IEDQFA1
FAI-8 A2
POST
PERFORM A TPOST
1-1 D3
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA1-15 (115) CPB INITIALIZATION - MAIN-STORAGE-ONLY QUEUING

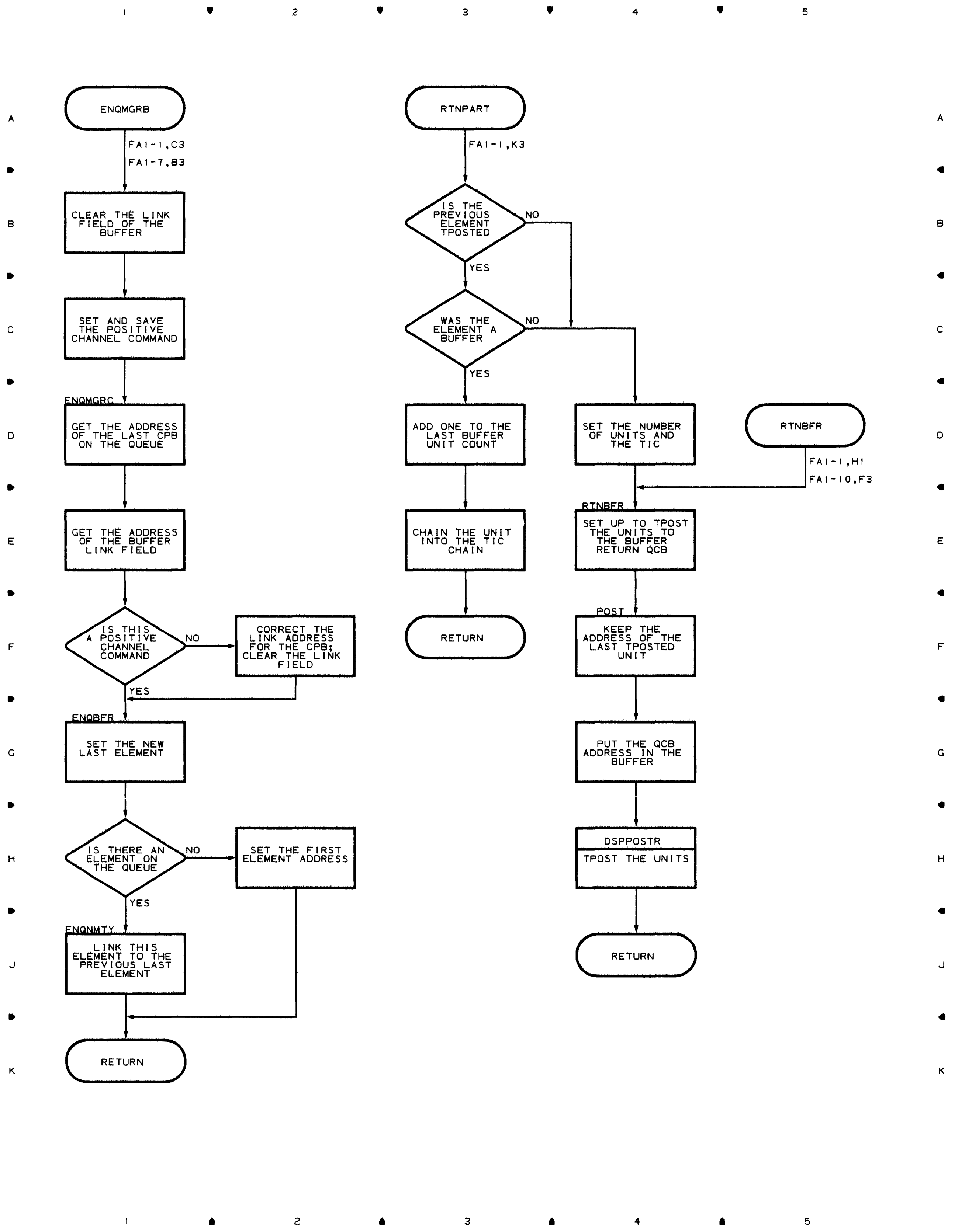
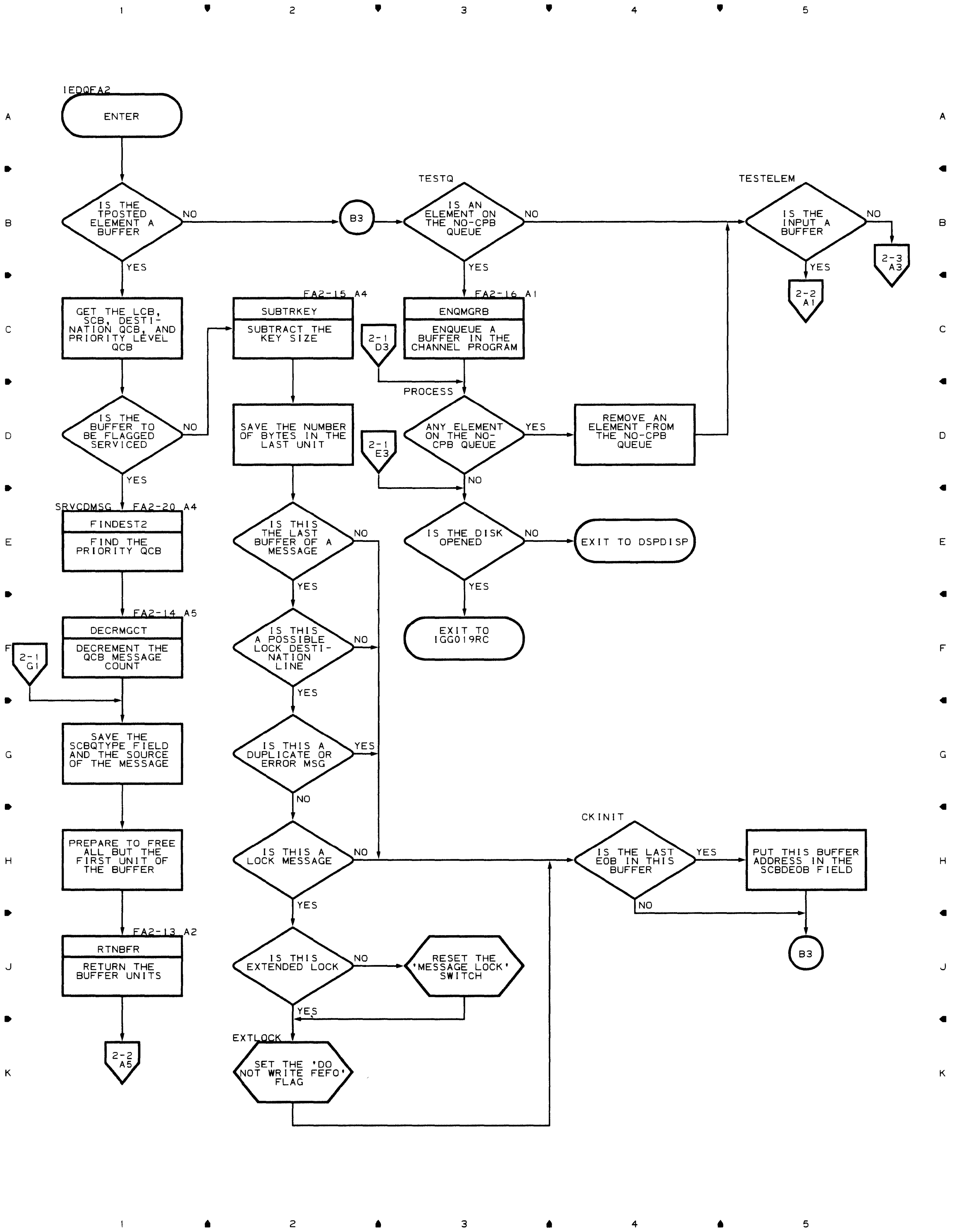
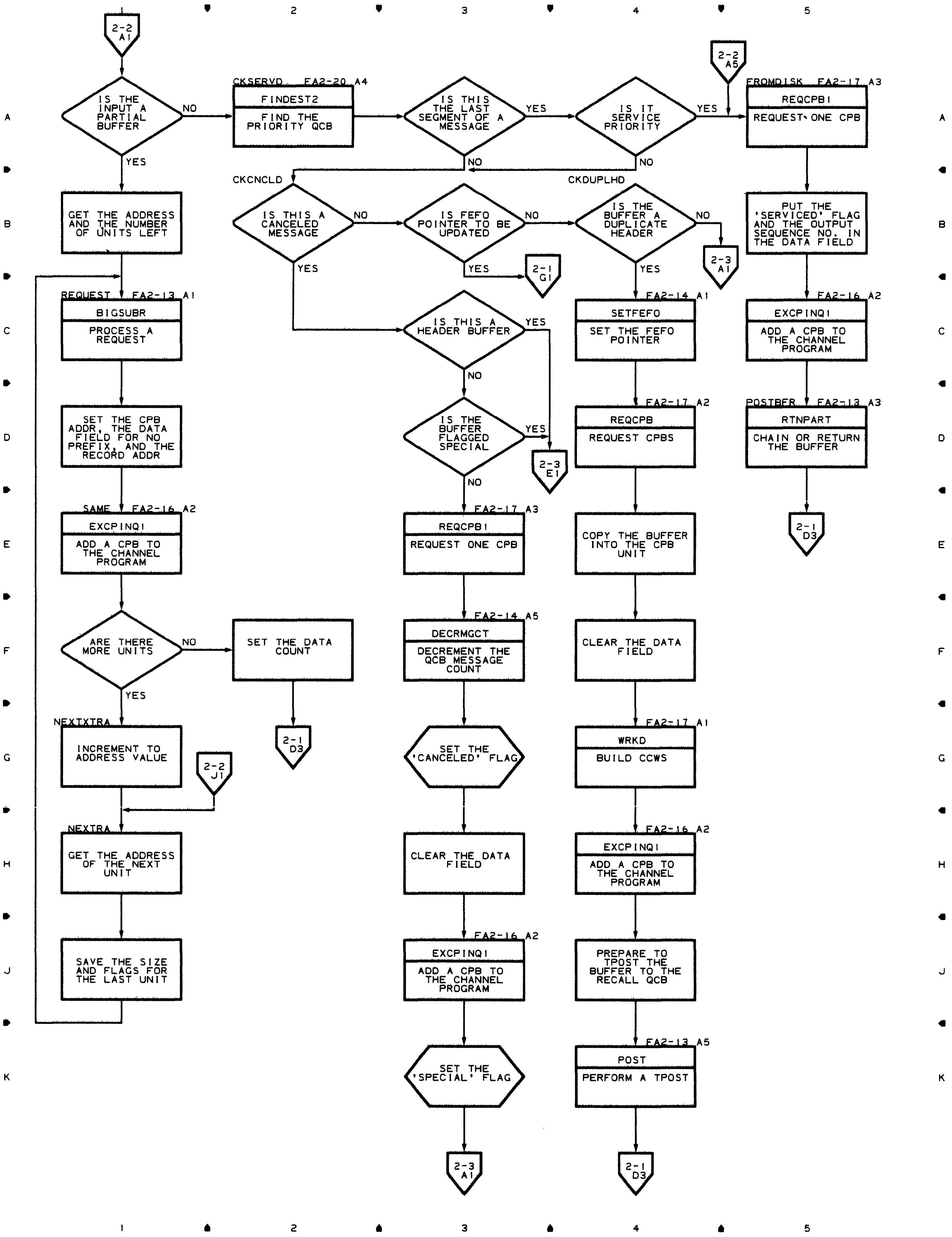

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
ENQMGRB
FA1-1,C3
FA1-7,B3
CLEAR THE LINK FIELD OF THE BUFFER
SET AND SAVE THE POSITIVE CHANNEL COMMAND
ENQMGRC
GET THE ADDRESS OF THE LAST CPB ON THE QUEUE
GET THE ADDRESS OF THE BUFFER LINK FIELD
IS THIS A POSITIVE CHANNEL COMMAND
NO
CORRECT THE LINK ADDRESS FOR THE CPB; CLEAR THE LINK FIELD
YES
ENQBFR
SET THE NEW LAST ELEMENT
IS THERE AN ELEMENT ON THE QUEUE
NO
SET THE FIRST ELEMENT ADDRESS
YES
ENQNMTY
LINK THIS ELEMENT TO THE PREVIOUS LAST ELEMENT
RETURN
RTNPART
FA1-1,K3
IS THE PREVIOUS ELEMENT TPOSTED
NO
YES
WAS THE ELEMENT A BUFFER
NO
YES
ADD ONE TO THE LAST BUFFER UNIT COUNT
CHAIN THE UNIT INTO THE TIC CHAIN
RETURN
SET THE NUMBER OF UNITS AND THE TIC
RTNBFR
FA1-1,H1
FA1-10,F3
RTNBFR
SET UP TO TPOST THE UNITS TO THE BUFFER RETURN QCB
POST
KEEP THE ADDRESS OF THE LAST TPOSTED UNIT
PUT THE QCB ADDRESS IN THE BUFFER
DSPPOSTR
TPOST THE UNITS
RETURN

Chart FA2-1 (2-1) CPB INITIALIZATION - DISK-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
IEDQFA2
ENTER
IS THE TPOSTED ELEMENT A BUFFER
NO
YES
B3
TESTQ
IS AN ELEMENT ON THE NO-CPB QUEUE
NO
YES
TESTELEM
IS THE INPUT A BUFFER
NO
YES
2-3 A3
2-2 A1
GET THE LCB, SCB, DESTINATION QCB, AND PRIORITY LEVEL QCB
FA2-15 A4
SUBTRKEY
SUBTRACT THE KEY SIZE
2-1 D3
FA2-16 A1
ENQMGRB
ENQUEUE A BUFFER IN THE CHANNEL PROGRAM
PROCESS
IS THE BUFFER TO BE FLAGGED SERVICED
NO
YES
SAVE THE NUMBER OF BYTES IN THE LAST UNIT
2-1 E3
ANY ELEMENT ON THE NO-CPB QUEUE
YES
NO
REMOVE AN ELEMENT FROM THE NO-CPB QUEUE
SRVCDMSG
FA2-20 A4
FINDEST2
FIND THE PRIORITY QCB
IS THIS THE LAST BUFFER OF A MESSAGE
NO
YES
IS THE DISK OPENED
NO
YES
EXIT TO DSPDISP
FA2-14 A5
DECRMGCT
DECREMENT THE QCB MESSAGE COUNT
2-1 G1
IS THIS A POSSIBLE LOCK DESTINATION LINE
NO
YES
EXIT TO IGG019RC
SAVE THE SCBQTYPE FIELD AND THE SOURCE OF THE MESSAGE
IS THIS A DUPLICATE OR ERROR MSG
YES
NO
CKINIT
PREPARE TO FREE ALL BUT THE FIRST UNIT OF THE BUFFER
IS THIS A LOCK MESSAGE
NO
YES
IS THE LAST EOB IN THIS BUFFER
YES
NO
PUT THIS BUFFER ADDRESS IN THE SCBDEOB FIELD
FA2-13 A2
RTNBFR
RETURN THE BUFFER UNITS
IS THIS EXTENDED LOCK
NO
YES
RESET THE 'MESSAGE LOCK' SWITCH
B3
EXTLOCK
2-2 A5
SET THE 'DO NOT WRITE FEFO' FLAG

Chart FA2-2 (2-2) CPB INITIALIZATION - DISK-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
2-2 A1
IS THE INPUT A PARTIAL BUFFER
NO
YES
GET THE ADDRESS AND THE NUMBER OF UNITS LEFT
REQUEST FA2-13 A1
BIGSUBR
PROCESS A REQUEST
SET THE CPB ADDR, THE DATA FIELD FOR NO PREFIX, AND THE RECORD ADDR
SAME FA2-16 A2
EXCPINQI
ADD A CPB TO THE CHANNEL PROGRAM
ARE THERE MORE UNITS
NO
YES
SET THE DATA COUNT
2-1 D3
NEXTXTRA
INCREMENT TO ADDRESS VALUE
2-2 J1
NEXTRA
GET THE ADDRESS OF THE NEXT UNIT
SAVE THE SIZE AND FLAGS FOR THE LAST UNIT
CKSERVD FA2-20 A4
FINDEST2
FIND THE PRIORITY QCB
IS THIS THE LAST SEGMENT OF A MESSAGE
YES
NO
IS IT SERVICE PRIORITY
YES
NO
2-2 A5
FROMDISK FA2-17 A3
REQCPBI
REQUEST ONE CPB
CKCNCLD
IS THIS A CANCELED MESSAGE
NO
YES
IS FEFO POINTER TO BE UPDATED
NO
YES
2-1 G1
CKDUPLHD
IS THE BUFFER A DUPLICATE HEADER
NO
YES
2-3 A1
PUT THE 'SERVICED' FLAG AND THE OUTPUT SEQUENCE NO. IN THE DATA FIELD
FA2-16 A2
EXCPINQI
ADD A CPB TO THE CHANNEL PROGRAM
POSTBFR FA2-13 A3
RTNPART
CHAIN OR RETURN THE BUFFER
2-1 D3
IS THIS A HEADER BUFFER
YES
NO
IS THE BUFFER FLAGGED SPECIAL
YES
NO
2-3 E1
FA2-17 A3
REQCPBI
REQUEST ONE CPB
FA2-14 A5
DECRMGCT
DECREMENT THE QCB MESSAGE COUNT
SET THE 'CANCELED' FLAG
CLEAR THE DATA FIELD
FA2-16 A2
EXCPINQI
ADD A CPB TO THE CHANNEL PROGRAM
SET THE 'SPECIAL' FLAG
2-3 A1
FA2-14 A1
SETFEFO
SET THE FEFO POINTER
FA2-17 A2
REQCPB
REQUEST CPBS
COPY THE BUFFER INTO THE CPB UNIT
CLEAR THE DATA FIELD
FA2-17 A1
WRKD
BUILD CCWS
FA2-16 A2
EXCPINQI
ADD A CPB TO THE CHANNEL PROGRAM
PREPARE TO TPOST THE BUFFER TO THE RECALL QCB
FA2-13 A5
POST
PERFORM A TPOST
2-1 D3

Chart FA2-3 (2-3) CPB INITIALIZATION - DISK-ONLY QUEUING

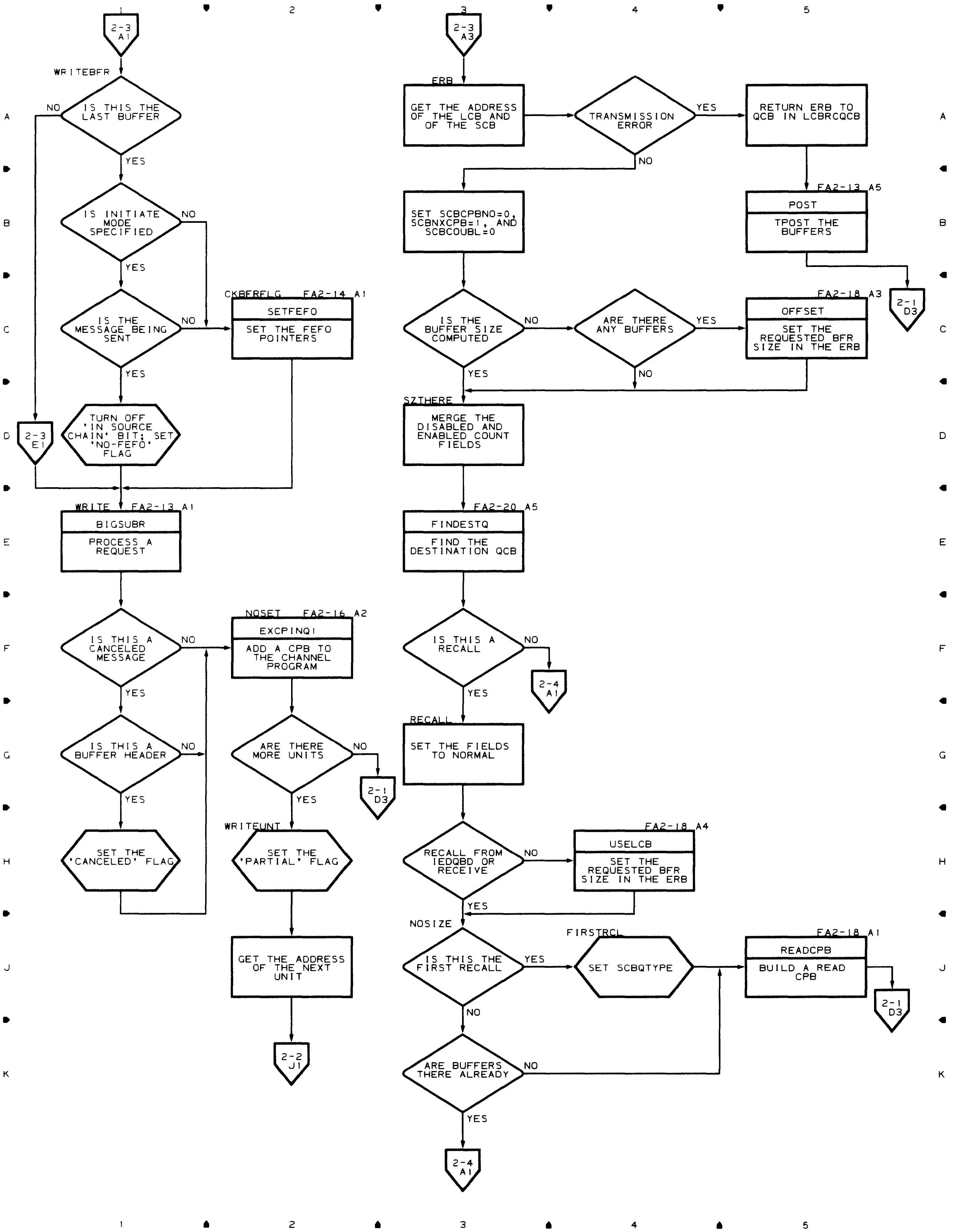


Chart FA2-4 (2-4) CPB INITIALIZATION - DISK-ONLY QUEUING

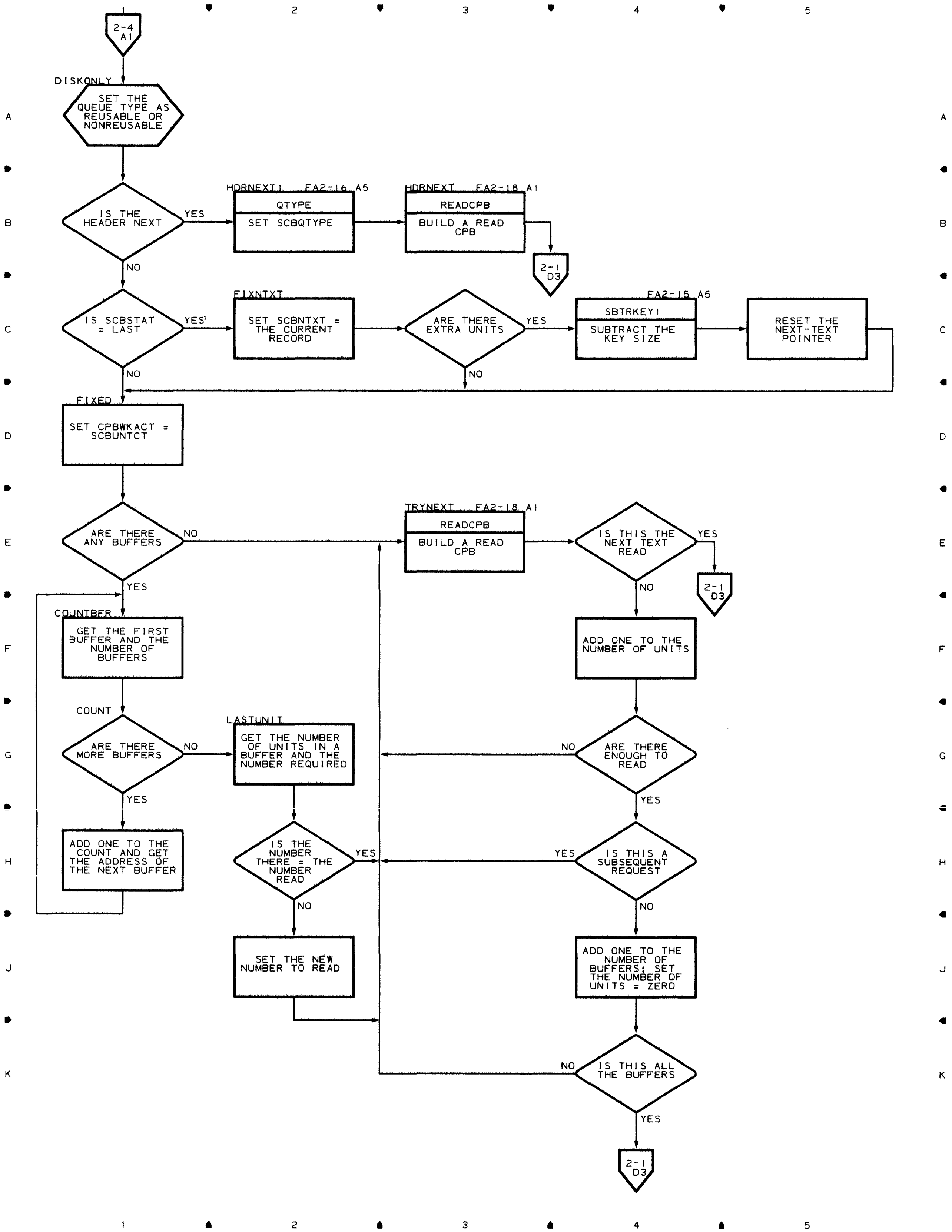

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
2-4 A1
DISKONLY
SET THE QUEUE TYPE AS REUSABLE OR NONREUSABLE
IS THE HEADER NEXT
YES
NO
HDRNEXT1 FA2-16 A5
QTYPE
SET SCBQTYPE
HDRNEXT FA2-18 A1
READCPB
BUILD A READ CPB
2-1 D3
IS SCBSTAT = LAST
YES
NO
FIXNTXT
SET SCBNTXT = THE CURRENT RECORD
ARE THERE EXTRA UNITS
YES
NO
FA2-15 A5
SBTRKEY1
SUBTRACT THE KEY SIZE
RESET THE NEXT-TEXT POINTER
FIXED
SET CPBWKACT = SCBUNTCT
ARE THERE ANY BUFFERS
NO
YES
TRYNEXT FA2-18 A1
READCPB
BUILD A READ CPB
IS THIS THE NEXT TEXT READ
YES
NO
2-1 D3
COUNTBFR
GET THE FIRST BUFFER AND THE NUMBER OF BUFFERS
ADD ONE TO THE NUMBER OF UNITS
COUNT
ARE THERE MORE BUFFERS
NO
YES
LASTUNIT
GET THE NUMBER OF UNITS IN A BUFFER AND THE NUMBER REQUIRED
ARE THERE ENOUGH TO READ
NO
YES
ADD ONE TO THE COUNT AND GET THE ADDRESS OF THE NEXT BUFFER
IS THE NUMBER THERE = THE NUMBER READ
YES
NO
IS THIS A SUBSEQUENT REQUEST
YES
NO
SET THE NEW NUMBER TO READ
ADD ONE TO THE NUMBER OF BUFFERS; SET THE NUMBER OF UNITS = ZERO
IS THIS ALL THE BUFFERS
NO
YES
2-1 D3

Chart FA2-5 (2-5) CPB INITIALIZATION - DISK-ONLY QUEUING

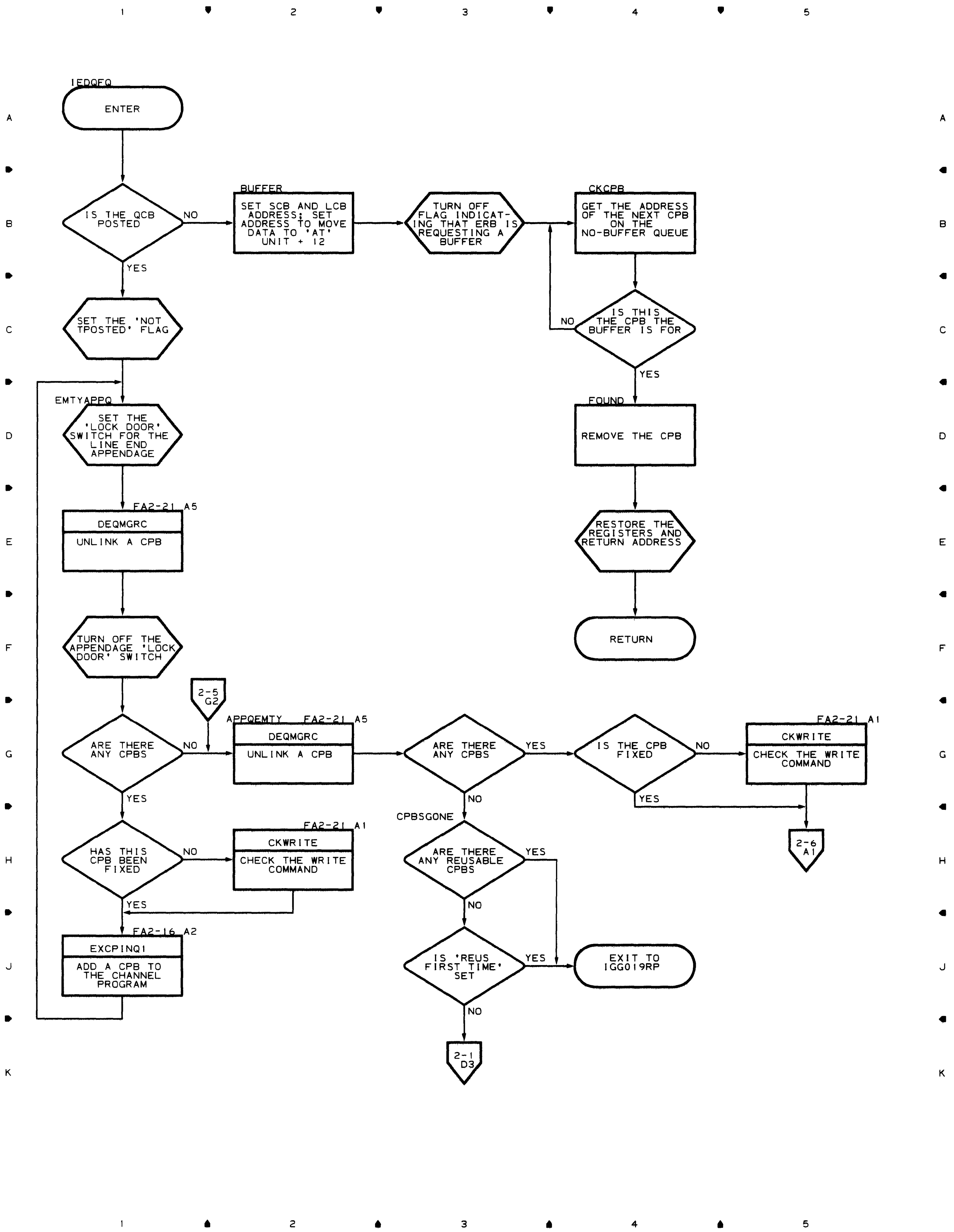


Chart FA2-6 (2-6) CPB INITIALIZATION - DISK-ONLY QUEUING

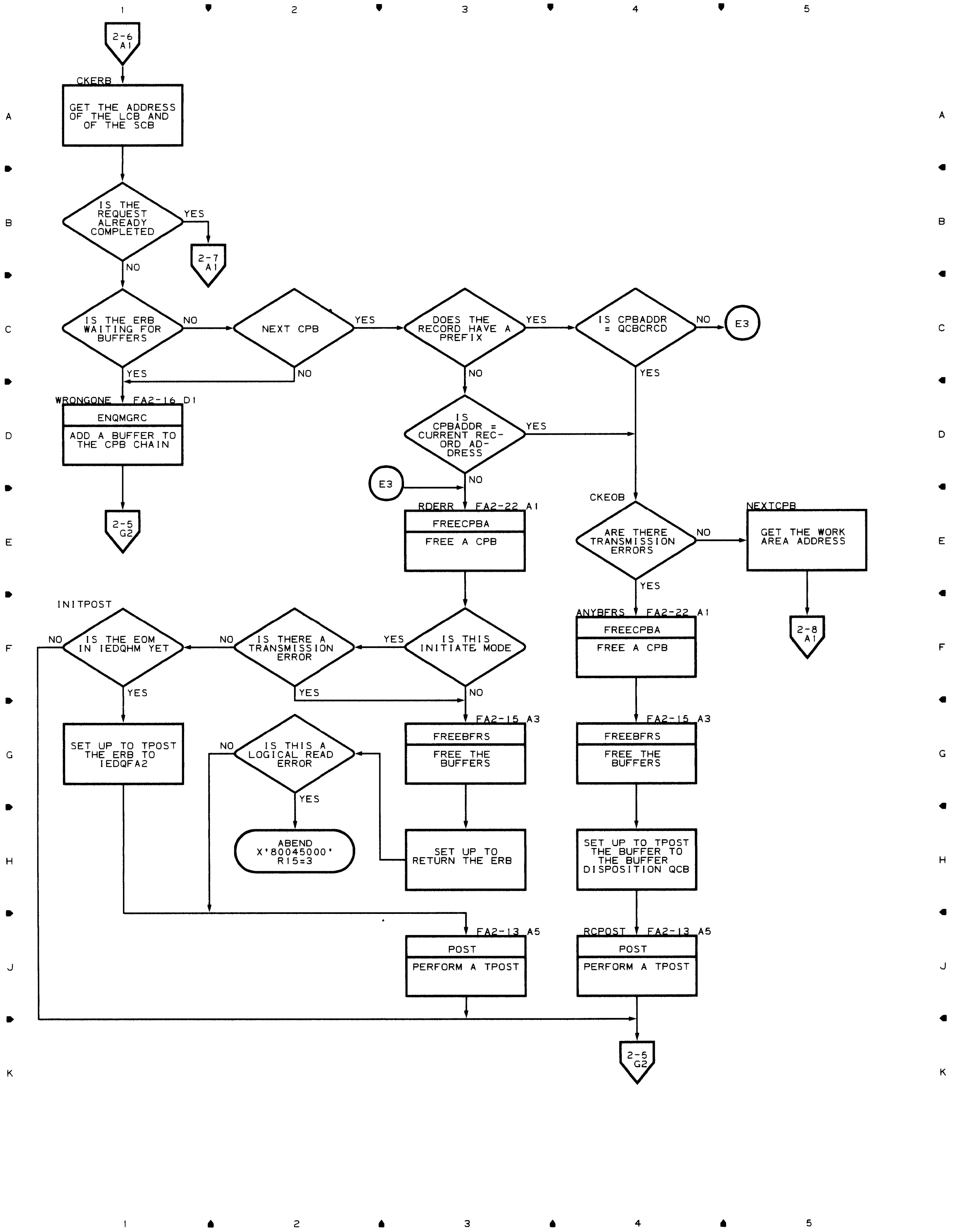


Chart FA2-7 (2-7) CPB INITIALIZATION - DISK-ONLY QUEUING

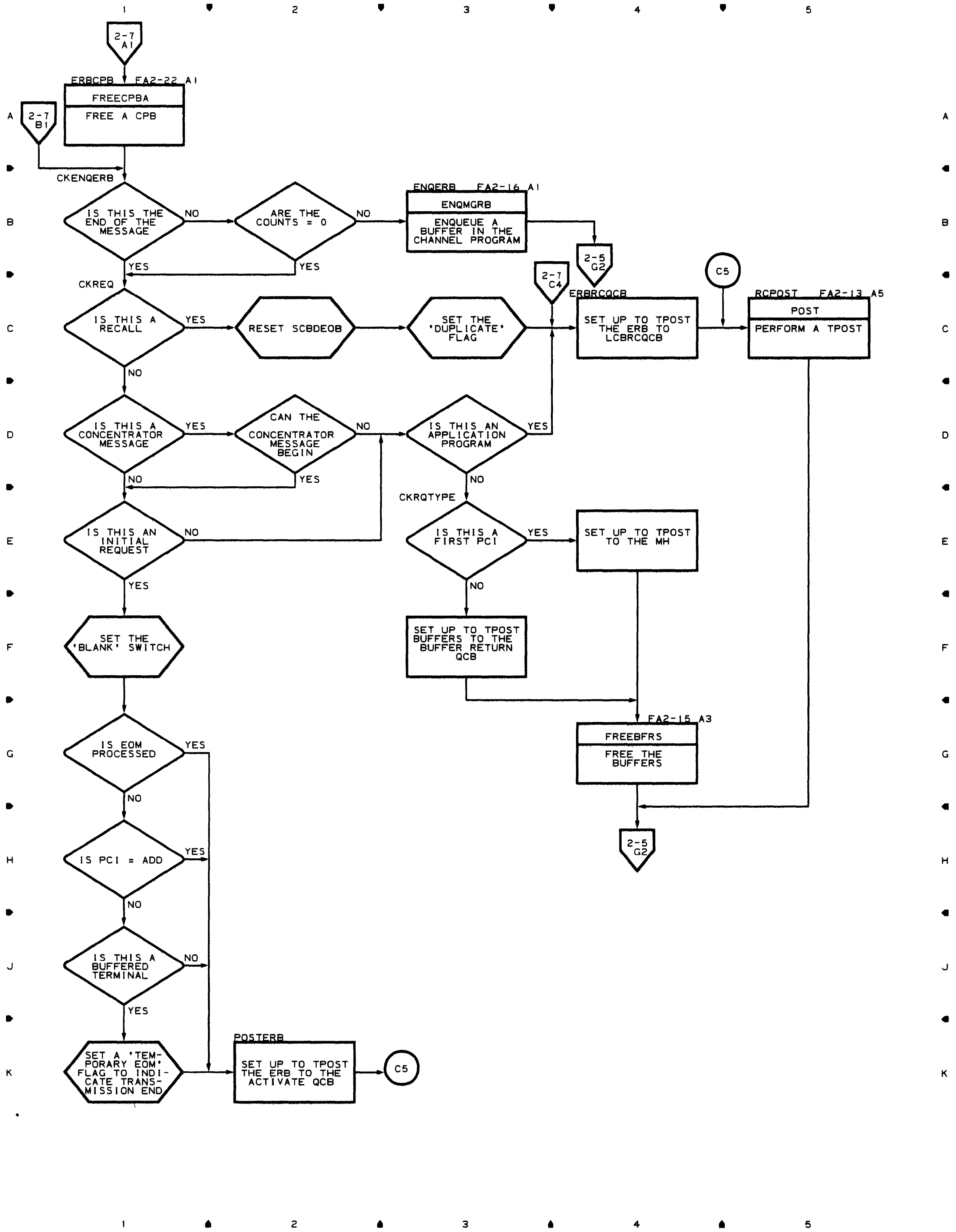

2-7 A1
ERBCPB FA2-22 A1
FREECPBA
FREE A CPB
2-7 B1
CKENQERB
IS THIS THE END OF THE MESSAGE
NO
ARE THE COUNTS = 0
NO
ENQERB FA2-16 A1
ENQMGRB
ENQUEUE A BUFFER IN THE CHANNEL PROGRAM
2-5 G2
YES
YES
CKREQ
2-7 C4
C5
IS THIS A RECALL
YES
RESET SCBDEOB
SET THE 'DUPLICATE' FLAG
ERBRCQCB
SET UP TO TPOST THE ERB TO LCBRCQCB
RCPOST FA2-13 A5
POST
PERFORM A TPOST
NO
IS THIS A CONCENTRATOR MESSAGE
YES
CAN THE CONCENTRATOR MESSAGE BEGIN
NO
IS THIS AN APPLICATION PROGRAM
YES
NO
YES
NO
CKRQTYPE
IS THIS AN INITIAL REQUEST
NO
IS THIS A FIRST PCI
YES
SET UP TO TPOST TO THE MH
YES
NO
SET THE 'BLANK' SWITCH
SET UP TO TPOST BUFFERS TO THE BUFFER RETURN QCB
FA2-15 A3
FREEBFRS
FREE THE BUFFERS
IS EOM PROCESSED
YES
NO
2-5 G2
IS PCI = ADD
YES
NO
IS THIS A BUFFERED TERMINAL
NO
YES
POSTERB
SET A 'TEMPORARY EOM' FLAG TO INDICATE TRANSMISSION END
SET UP TO TPOST THE ERB TO THE ACTIVATE QCB
C5
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA2-8 (2-8) CPB INITIALIZATION - DISK-ONLY QUEUING

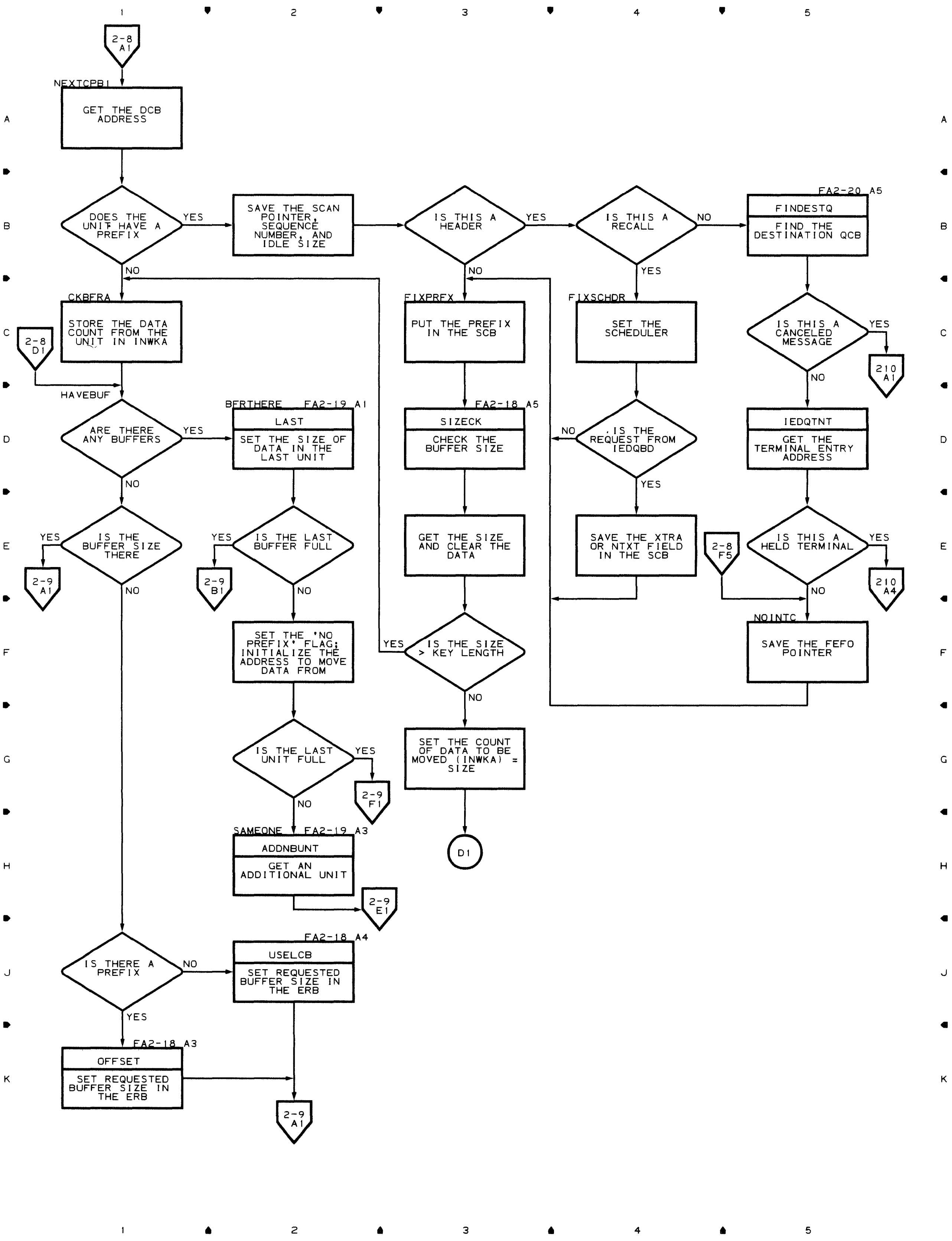
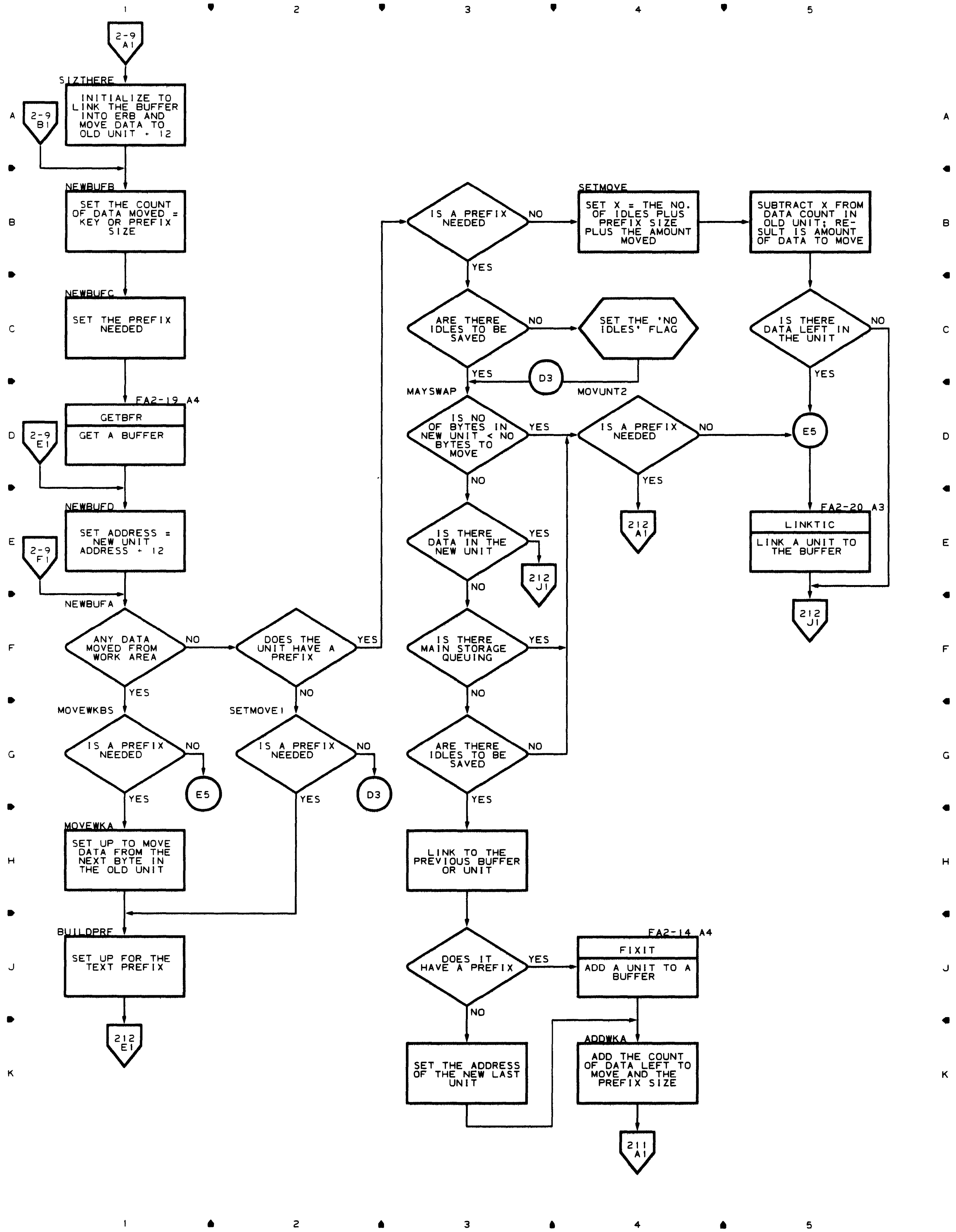
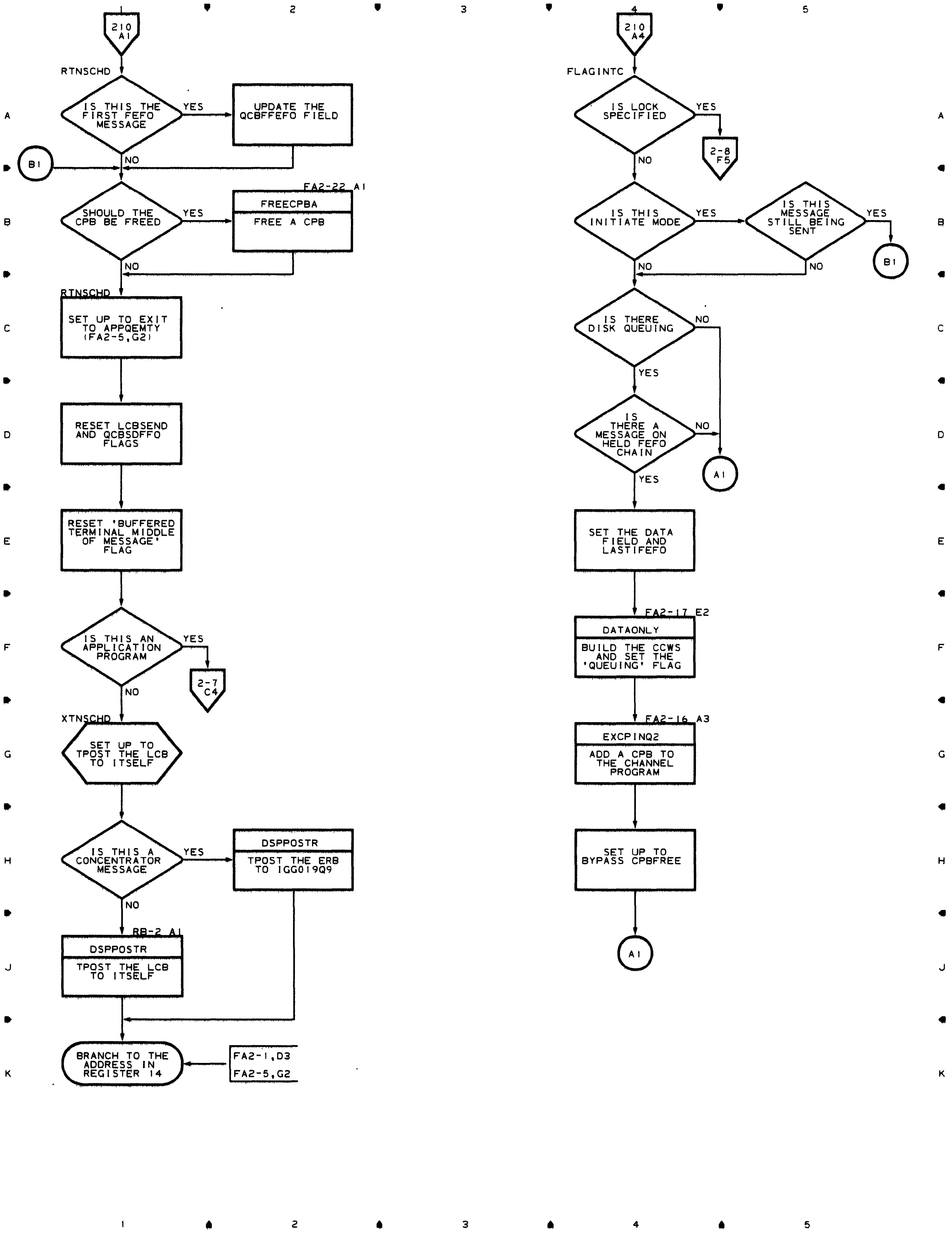


Chart FA2-9 (2-9) CPB INITIALIZATION - DISK-ONLY QUEUING



2-9 A1
SIZTHERE
INITIALIZE TO LINK THE BUFFER INTO ERB AND MOVE DATA TO OLD UNIT + 12
2-9 B1
NEWBUFB
SET THE COUNT OF DATA MOVED = KEY OR PREFIX SIZE
NEWBUFC
SET THE PREFIX NEEDED
FA2-19 A4
GETBFR
GET A BUFFER
2-9 E1
NEWBUFD
SET ADDRESS = NEW UNIT ADDRESS + 12
2-9 F1
NEWBUFA
ANY DATA MOVED FROM WORK AREA
NO
YES
MOVEWKBS
IS A PREFIX NEEDED
NO
E5
YES
MOVEWKA
SET UP TO MOVE DATA FROM THE NEXT BYTE IN THE OLD UNIT
BUILDPREF
SET UP FOR THE TEXT PREFIX
212 E1
DOES THE UNIT HAVE A PREFIX
YES
NO
SETMOVE1
IS A PREFIX NEEDED
NO
D3
YES
IS A PREFIX NEEDED
NO
YES
ARE THERE IDLES TO BE SAVED
NO
YES
D3
MAYSWAP
IS NO OF BYTES IN NEW UNIT < NO BYTES TO MOVE
YES
NO
IS THERE DATA IN THE NEW UNIT
YES
212 J1
NO
IS THERE MAIN STORAGE QUEUING
YES
NO
ARE THERE IDLES TO BE SAVED
NO
YES
LINK TO THE PREVIOUS BUFFER OR UNIT
DOES IT HAVE A PREFIX
YES
NO
SET THE ADDRESS OF THE NEW LAST UNIT
SETMOVE
SET X = THE NO. OF IDLES PLUS PREFIX SIZE PLUS THE AMOUNT MOVED
SET THE 'NO IDLES' FLAG
MOVUNT2
IS A PREFIX NEEDED
NO
YES
212 A1
FA2-14 A4
FIXIT
ADD A UNIT TO A BUFFER
ADDWKA
ADD THE COUNT OF DATA LEFT TO MOVE AND THE PREFIX SIZE
211 A1
SUBTRACT X FROM DATA COUNT IN OLD UNIT; RESULT IS AMOUNT OF DATA TO MOVE
IS THERE DATA LEFT IN THE UNIT
NO
YES
E5
FA2-20 A3
LINKTIC
LINK A UNIT TO THE BUFFER
212 J1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA2-10 (210) CPB INITIALIZATION - DISK-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
210 A1
RTNSCHD
IS THIS THE FIRST FEFO MESSAGE
YES
UPDATE THE QCBFFEFO FIELD
B1
NO
FA2-22 A1
SHOULD THE CPB BE FREED
YES
FREECPBA
FREE A CPB
NO
RTNSCHD
SET UP TO EXIT TO APPQEMTY (FA2-5,G2)
RESET LCBSEND AND QCBSDFFO FLAGS
RESET 'BUFFERED TERMINAL MIDDLE OF MESSAGE' FLAG
IS THIS AN APPLICATION PROGRAM
YES
2-7 C4
NO
XTNSCHD
SET UP TO TPOST THE LCB TO ITSELF
IS THIS A CONCENTRATOR MESSAGE
YES
DSPPOSTR
TPOST THE ERB TO IGG019Q9
NO
RB-2 A1
DSPPOSTR
TPOST THE LCB TO ITSELF
BRANCH TO THE ADDRESS IN REGISTER 14
FA2-1,D3
FA2-5,G2
210 A4
FLAGINTC
IS LOCK SPECIFIED
YES
2-8 F5
NO
IS THIS INITIATE MODE
YES
IS THIS MESSAGE STILL BEING SENT
YES
B1
NO
NO
IS THERE DISK QUEUING
NO
YES
IS THERE A MESSAGE ON HELD FEFO CHAIN
NO
A1
YES
SET THE DATA FIELD AND LASTIFEFO
FA2-17 E2
DATAONLY
BUILD THE CCWS AND SET THE 'QUEUING' FLAG
FA2-16 A3
EXCPINQ2
ADD A CPB TO THE CHANNEL PROGRAM
SET UP TO BYPASS CPBFREE
A1

Chart FA2-11 (211) CPB INITIALIZATION - DISK-ONLY QUEUING

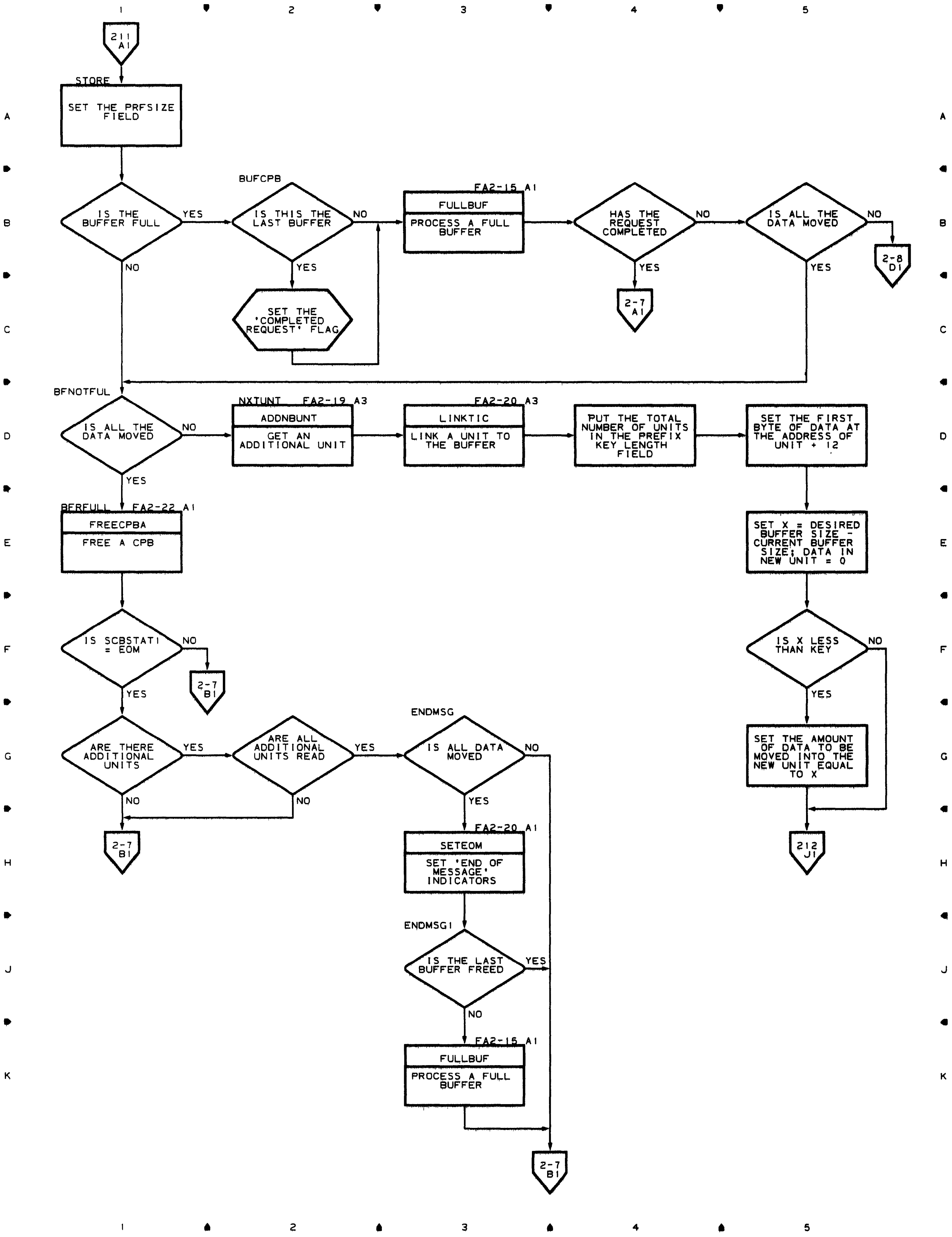


Chart FA2-12 (212) CPB INITIALIZATION - DISK-ONLY QUEUING

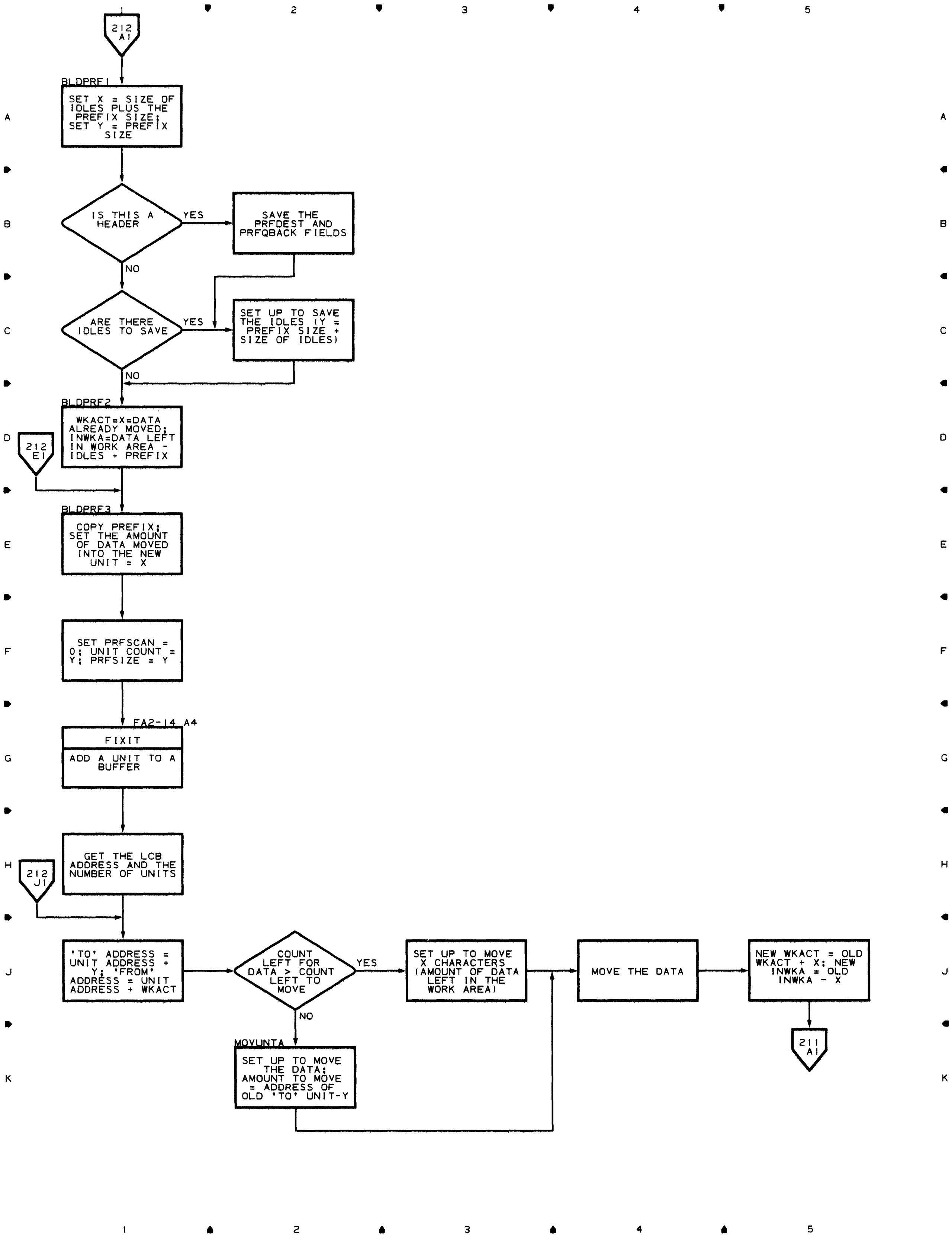

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
212 A1
BLDPRF1
SET X = SIZE OF IDLES PLUS THE PREFIX SIZE; SET Y = PREFIX SIZE
IS THIS A HEADER
YES
SAVE THE PRFDEST AND PRFQBACK FIELDS
NO
ARE THERE IDLES TO SAVE
YES
SET UP TO SAVE THE IDLES (Y = PREFIX SIZE + SIZE OF IDLES)
NO
BLDPRF2
WKACT=X=DATA ALREADY MOVED; INWKA=DATA LEFT IN WORK AREA - IDLES + PREFIX
212 E1
BLDPRF3
COPY PREFIX; SET THE AMOUNT OF DATA MOVED INTO THE NEW UNIT = X
SET PRFSCAN = 0; UNIT COUNT = Y; PRFSIZE = Y
FA2-14 A4
FIXIT
ADD A UNIT TO A BUFFER
GET THE LCB ADDRESS AND THE NUMBER OF UNITS
212 J1
'TO' ADDRESS = UNIT ADDRESS + Y; 'FROM' ADDRESS = UNIT ADDRESS + WKACT
COUNT LEFT FOR DATA > COUNT LEFT TO MOVE
YES
SET UP TO MOVE X CHARACTERS (AMOUNT OF DATA LEFT IN THE WORK AREA)
MOVE THE DATA
NEW WKACT = OLD WKACT + X; NEW INWKA = OLD INWKA - X
211 A1
NO
MOVUNTA
SET UP TO MOVE THE DATA; AMOUNT TO MOVE = ADDRESS OF OLD 'TO' UNIT-Y

Chart FA2-13 (213) CPB INITIALIZATION - DISK-ONLY QUEUING

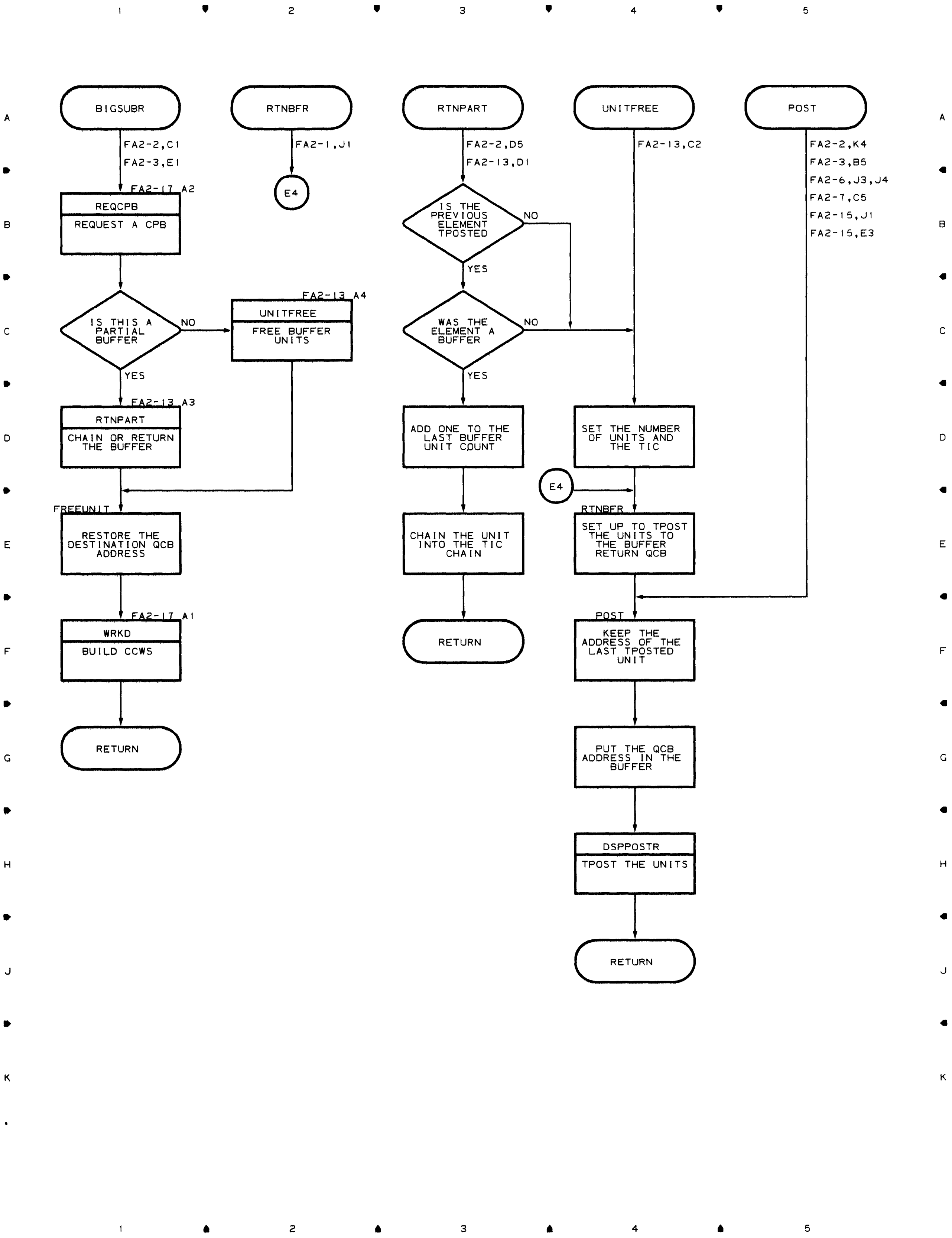
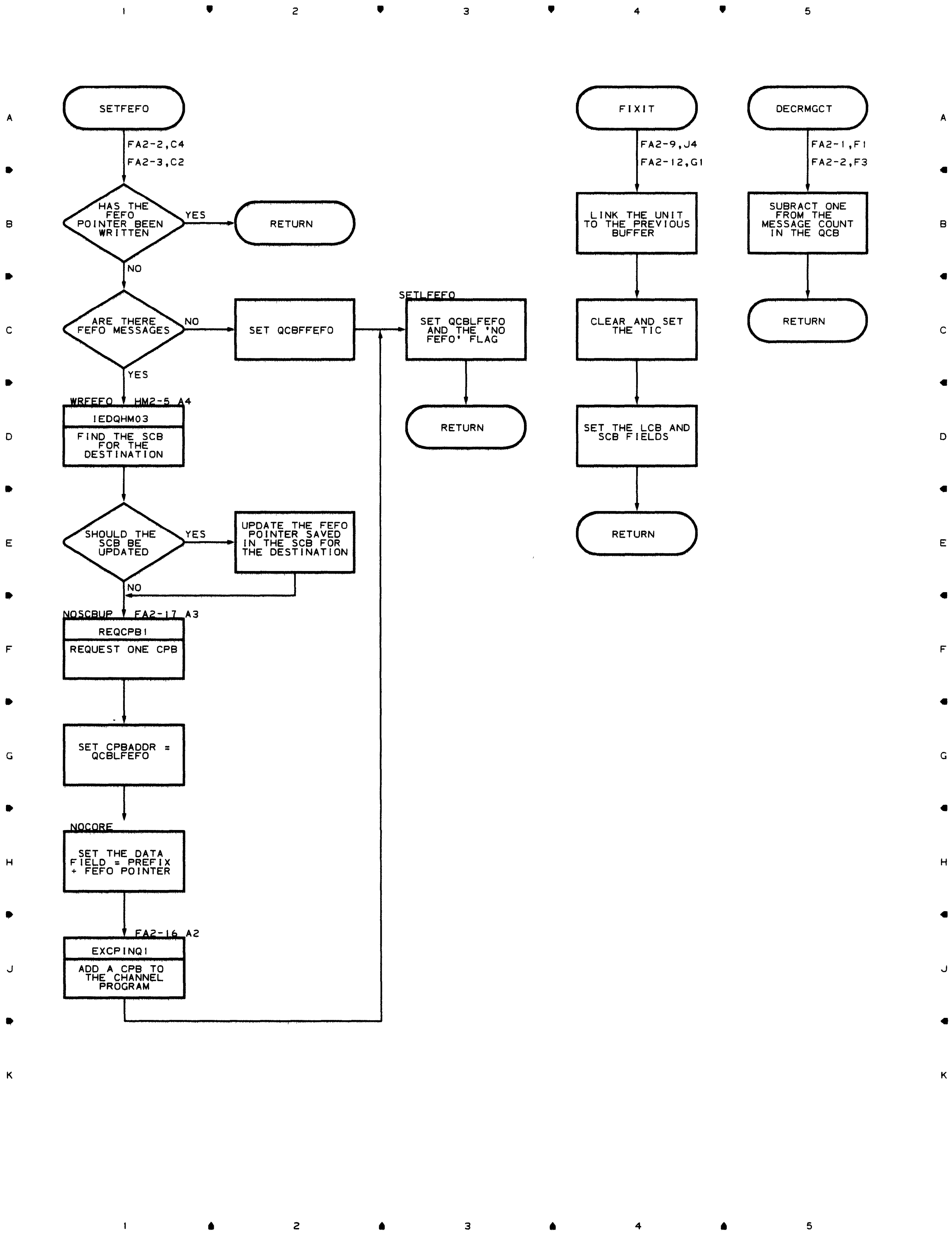


Chart FA2-14 (214) CPB INITIALIZATION - DISK-ONLY QUEUING



SETFEFO
FA2-2,C4
FA2-3,C2
HAS THE FEFO POINTER BEEN WRITTEN
YES
RETURN
NO
ARE THERE FEFO MESSAGES
NO
SET QCBFFEFO
SETLFEFO
SET QCBLFEFO AND THE 'NO FEFO' FLAG
RETURN
YES
WRFEFO
HM2-5 A4
IEDQHM03
FIND THE SCB FOR THE DESTINATION
SHOULD THE SCB BE UPDATED
YES
UPDATE THE FEFO POINTER SAVED IN THE SCB FOR THE DESTINATION
NO
NOSCBUP
FA2-17 A3
REQCPB1
REQUEST ONE CPB
SET CPBADDR = QCBLFEFO
NOCORE
SET THE DATA FIELD = PREFIX + FEFO POINTER
FA2-16 A2
EXCPINQ1
ADD A CPB TO THE CHANNEL PROGRAM
FIXIT
FA2-9,J4
FA2-12,G1
LINK THE UNIT TO THE PREVIOUS BUFFER
CLEAR AND SET THE TIC
SET THE LCB AND SCB FIELDS
RETURN
DECRMGCT
FA2-1,F1
FA2-2,F3
SUBRACT ONE FROM THE MESSAGE COUNT IN THE QCB
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart FA2-15 (215) CPB INITIALIZATION - DISK-ONLY QUEUING

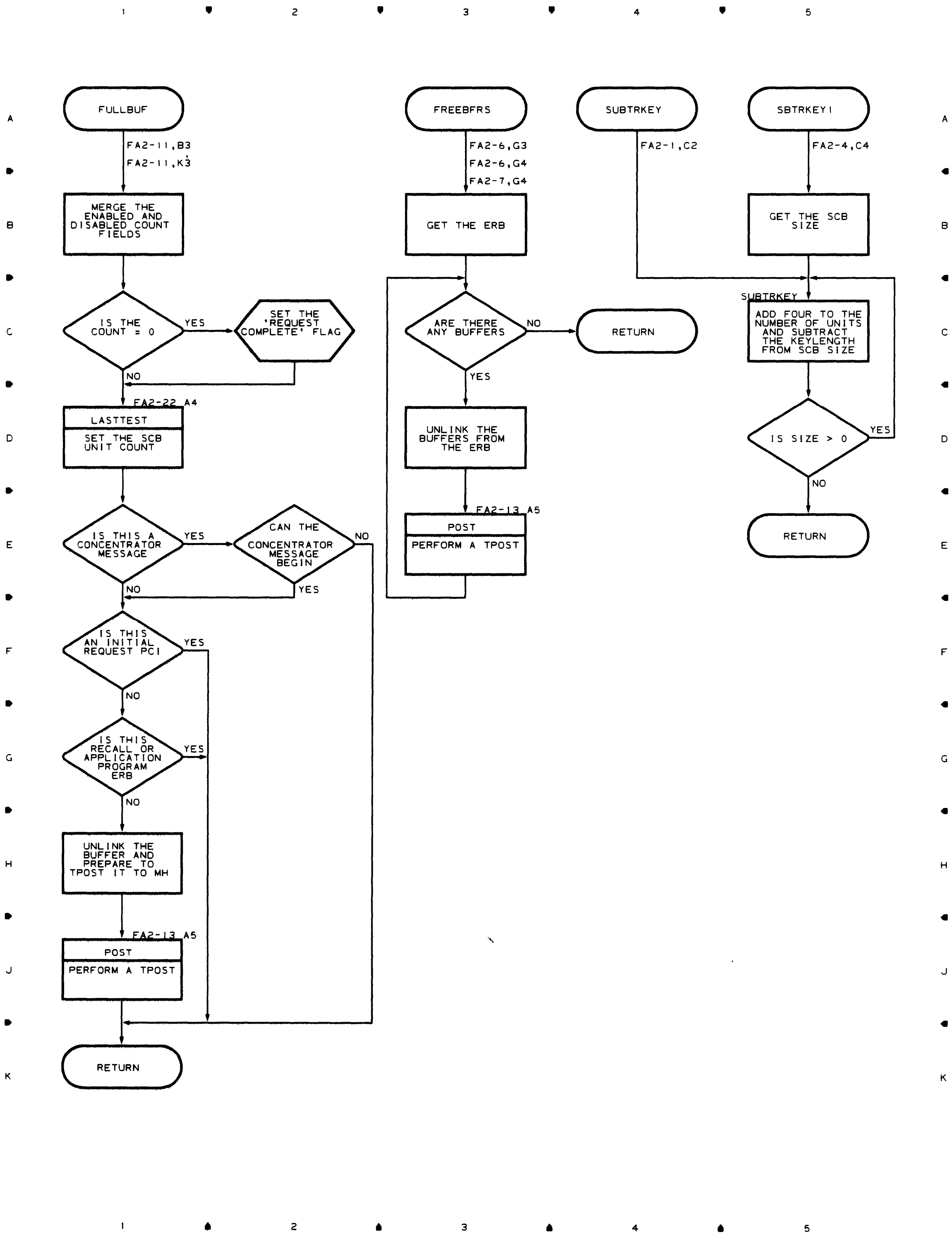

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FULLBUF
FA2-11,B3
FA2-11,K3
MERGE THE ENABLED AND DISABLED COUNT FIELDS
IS THE COUNT = 0
YES
SET THE 'REQUEST COMPLETE' FLAG
NO
FA2-22 A4
LASTTEST
SET THE SCB UNIT COUNT
IS THIS A CONCENTRATOR MESSAGE
YES
CAN THE CONCENTRATOR MESSAGE BEGIN
NO
YES
NO
IS THIS AN INITIAL REQUEST PCI
YES
NO
IS THIS RECALL OR APPLICATION PROGRAM ERB
YES
NO
UNLINK THE BUFFER AND PREPARE TO TPOST IT TO MH
FA2-13 A5
POST
PERFORM A TPOST
RETURN
FREEBFRS
FA2-6,G3
FA2-6,G4
FA2-7,G4
GET THE ERB
ARE THERE ANY BUFFERS
NO
RETURN
YES
UNLINK THE BUFFERS FROM THE ERB
FA2-13 A5
POST
PERFORM A TPOST
SUBTRKEY
FA2-1,C2
SBTRKEY1
FA2-4,C4
GET THE SCB SIZE
SUBTRKEY
ADD FOUR TO THE NUMBER OF UNITS AND SUBTRACT THE KEYLENGTH FROM SCB SIZE
IS SIZE > 0
YES
NO
RETURN

Chart FA2-16 (216) CPB INITIALIZATION - DISK ONLY QUEUING

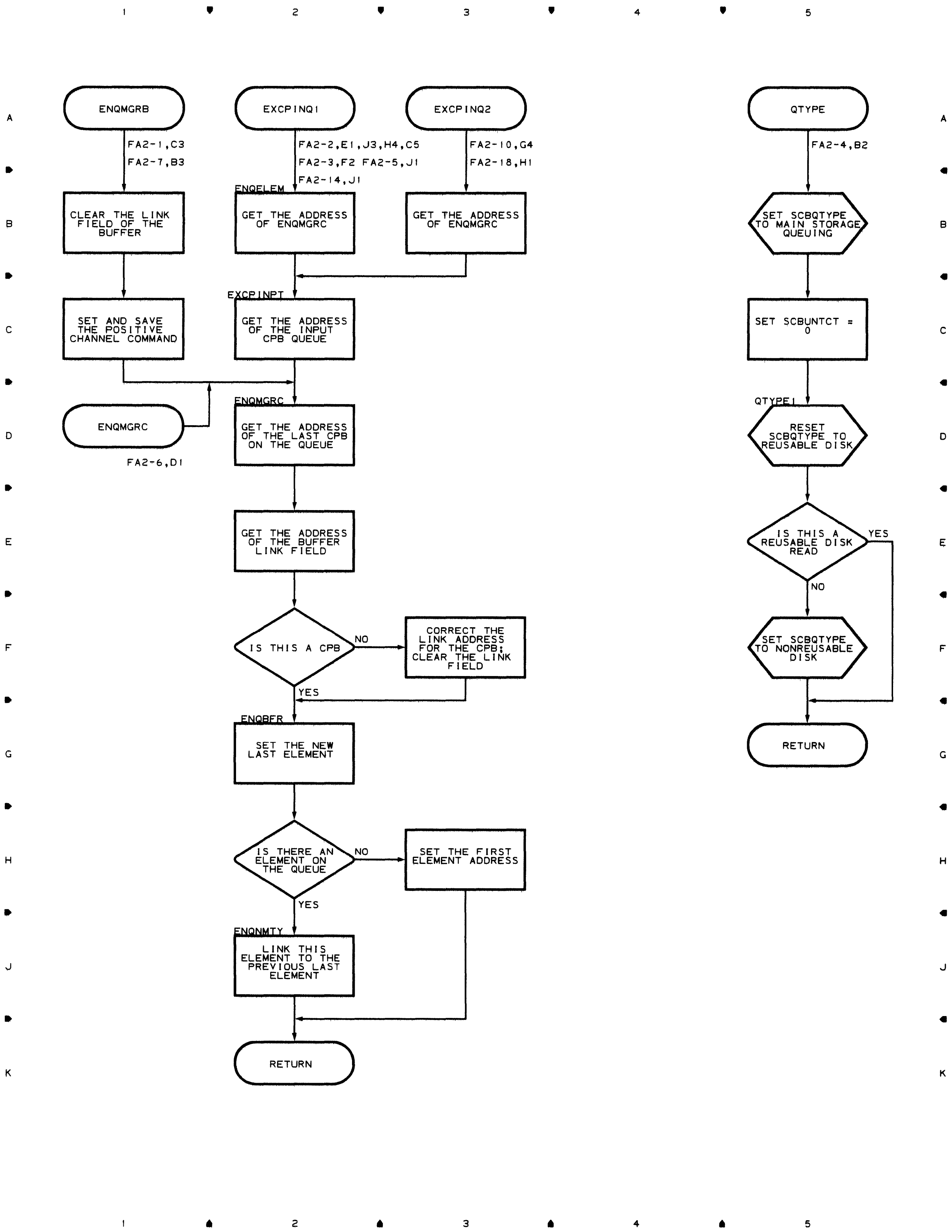

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
ENQMGRB
FA2-1,C3
FA2-7,B3
CLEAR THE LINK FIELD OF THE BUFFER
SET AND SAVE THE POSITIVE CHANNEL COMMAND
ENQMGRC
FA2-6,D1
EXCPINQ1
FA2-2,E1,J3,H4,C5
FA2-3,F2 FA2-5,J1
FA2-14,J1
ENQELEM
GET THE ADDRESS OF ENQMGRC
EXCPINPT
GET THE ADDRESS OF THE INPUT CPB QUEUE
ENQMGRC
GET THE ADDRESS OF THE LAST CPB ON THE QUEUE
GET THE ADDRESS OF THE BUFFER LINK FIELD
IS THIS A CPB
NO
YES
ENQBFR
SET THE NEW LAST ELEMENT
IS THERE AN ELEMENT ON THE QUEUE
NO
YES
ENQNMTY
LINK THIS ELEMENT TO THE PREVIOUS LAST ELEMENT
RETURN
EXCPINQ2
FA2-10,G4
FA2-18,H1
GET THE ADDRESS OF ENQMGRC
CORRECT THE LINK ADDRESS FOR THE CPB; CLEAR THE LINK FIELD
SET THE FIRST ELEMENT ADDRESS
QTYPE
FA2-4,B2
SET SCBQTYPE TO MAIN STORAGE QUEUING
SET SCBUNTCT = 0
QTYPE1
RESET SCBQTYPE TO REUSABLE DISK
IS THIS A REUSABLE DISK READ
YES
NO
SET SCBQTYPE TO NONREUSABLE DISK
RETURN

Chart FA2-17 (217) CPB INITIALIZATION - DISK-ONLY QUEUING

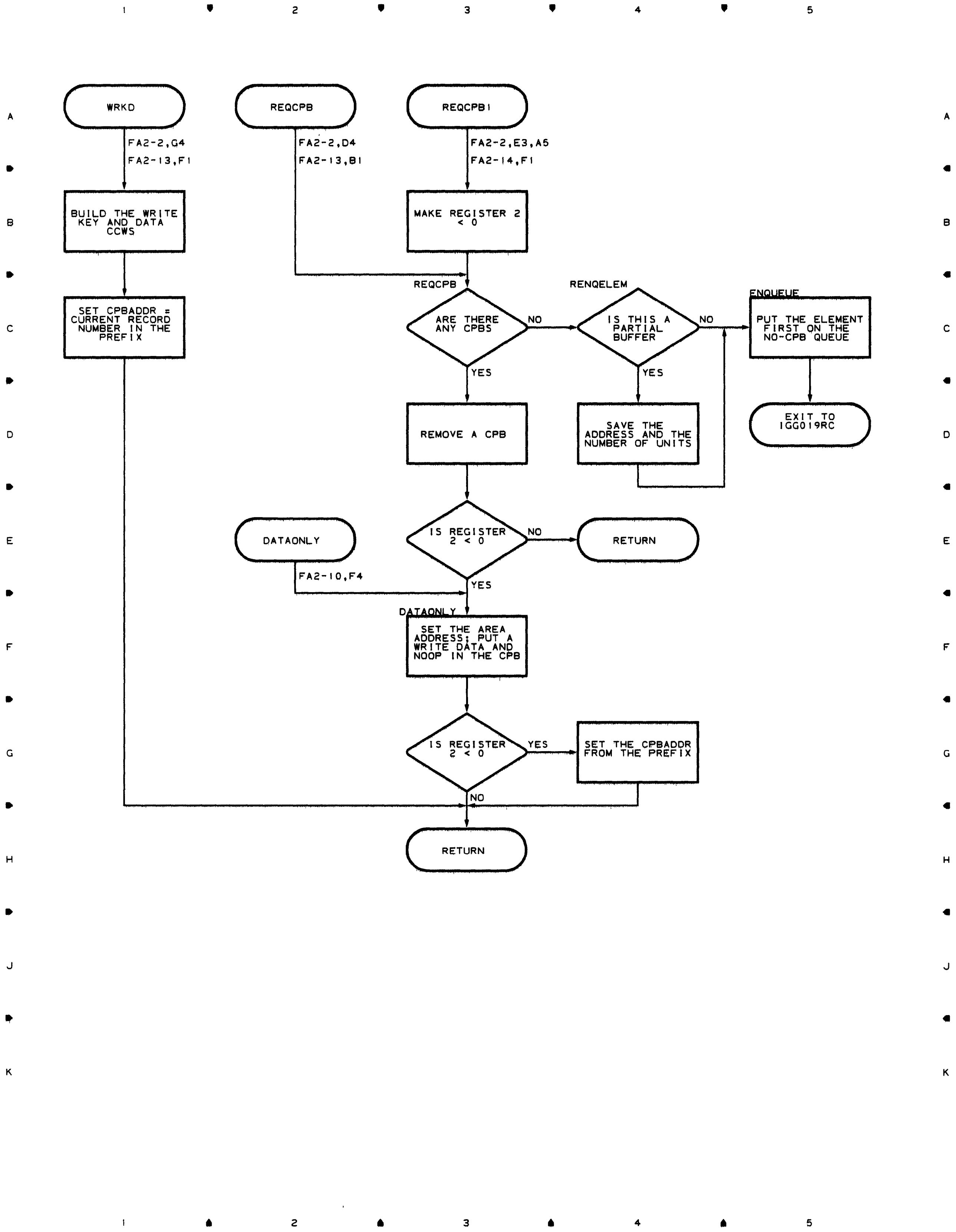
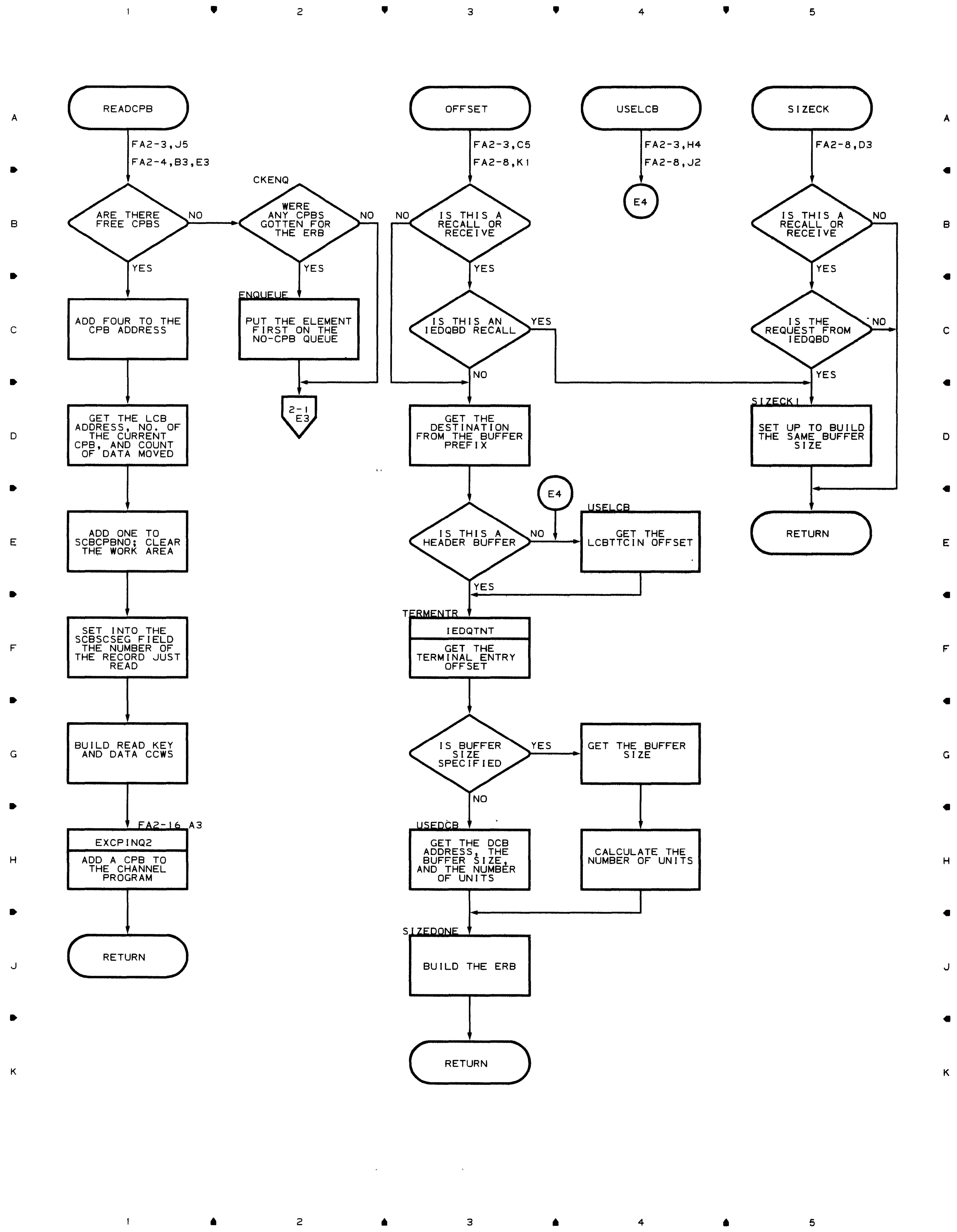


Chart FA2-18 (218) CPB INITIALIZATION - DISK-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
READCPB
FA2-3,J5
FA2-4,B3,E3
ARE THERE FREE CPBS
NO
YES
ADD FOUR TO THE CPB ADDRESS
GET THE LCB ADDRESS, NO. OF THE CURRENT CPB, AND COUNT OF DATA MOVED
ADD ONE TO SCBCPBNO; CLEAR THE WORK AREA
SET INTO THE SCBSCSEG FIELD THE NUMBER OF THE RECORD JUST READ
BUILD READ KEY AND DATA CCWS
FA2-16 A3
EXCPINQ2
ADD A CPB TO THE CHANNEL PROGRAM
RETURN
CKENQ
WERE ANY CPBS GOTTEN FOR THE ERB
NO
YES
ENQUEUE
PUT THE ELEMENT FIRST ON THE NO-CPB QUEUE
2-1 E3
OFFSET
FA2-3,C5
FA2-8,K1
NO
IS THIS A RECALL OR RECEIVE
YES
IS THIS AN IEDQBD RECALL
YES
NO
GET THE DESTINATION FROM THE BUFFER PREFIX
IS THIS A HEADER BUFFER
NO
YES
TERMENTR
IEDQTNT
GET THE TERMINAL ENTRY OFFSET
IS BUFFER SIZE SPECIFIED
YES
NO
USEDCB
GET THE DCB ADDRESS, THE BUFFER SIZE, AND THE NUMBER OF UNITS
SIZEDONE
BUILD THE ERB
RETURN
USELCB
FA2-3,H4
FA2-8,J2
E4
E4
USELCB
GET THE LCBTTCIN OFFSET
GET THE BUFFER SIZE
CALCULATE THE NUMBER OF UNITS
SIZECK
FA2-8,D3
IS THIS A RECALL OR RECEIVE
NO
YES
IS THE REQUEST FROM IEDQBD
NO
YES
SIZECK1
SET UP TO BUILD THE SAME BUFFER SIZE
RETURN

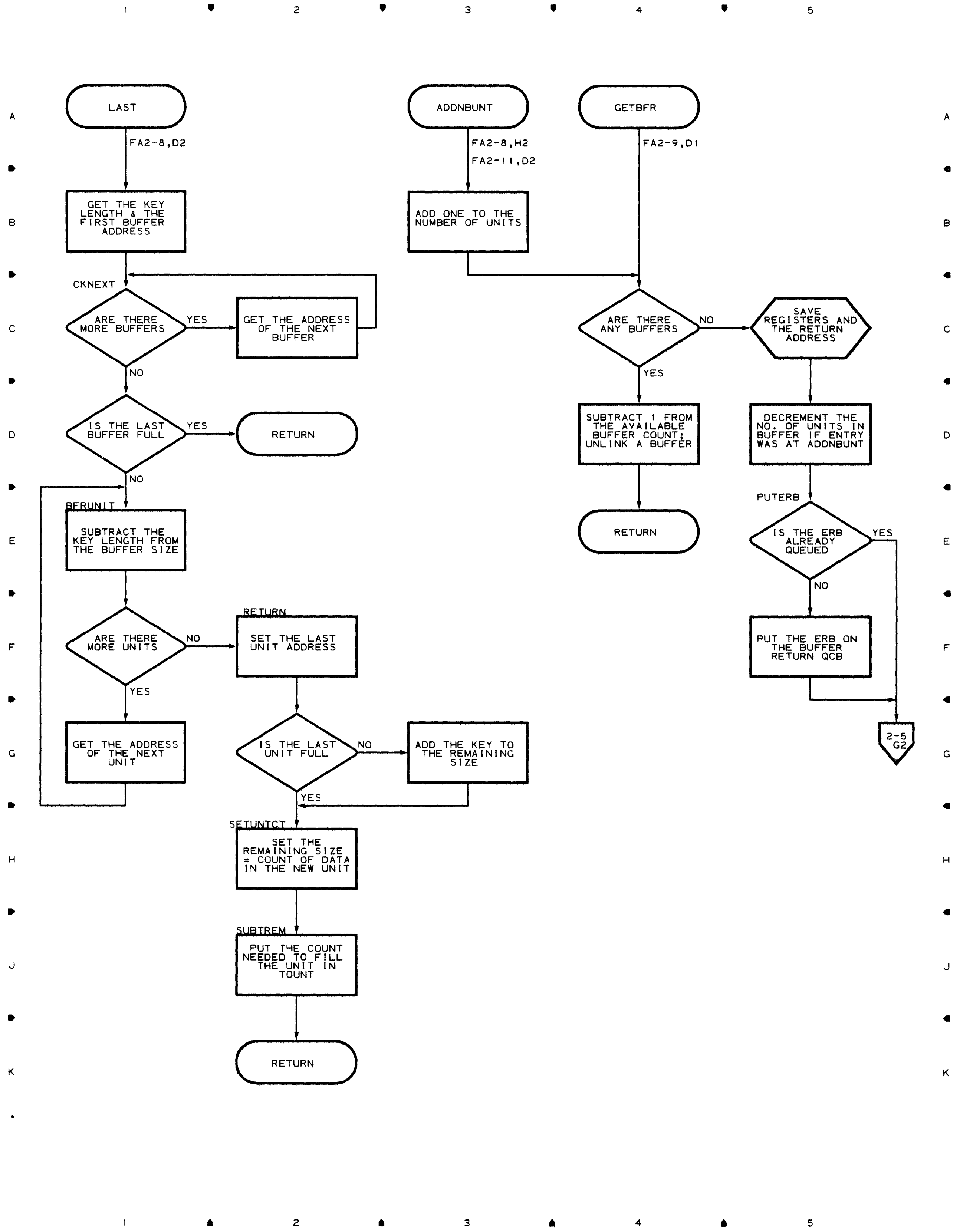
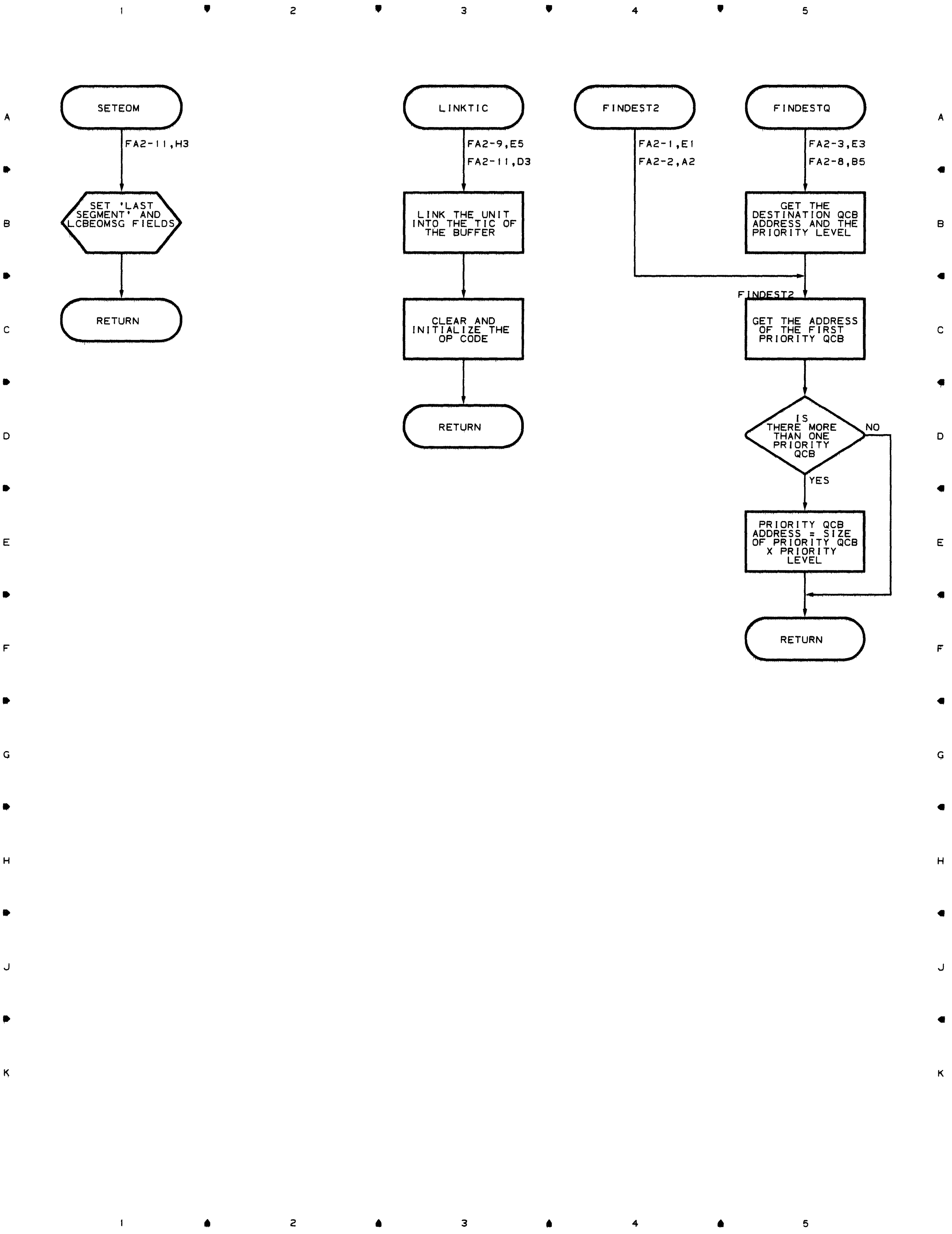

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
LAST
FA2-8,D2
GET THE KEY LENGTH & THE FIRST BUFFER ADDRESS
CKNEXT
ARE THERE MORE BUFFERS
YES
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE LAST BUFFER FULL
YES
RETURN
NO
BFRUNIT
SUBTRACT THE KEY LENGTH FROM THE BUFFER SIZE
ARE THERE MORE UNITS
NO
RETURN
SET THE LAST UNIT ADDRESS
YES
GET THE ADDRESS OF THE NEXT UNIT
IS THE LAST UNIT FULL
NO
ADD THE KEY TO THE REMAINING SIZE
YES
SETUNTCT
SET THE REMAINING SIZE = COUNT OF DATA IN THE NEW UNIT
SUBTREM
PUT THE COUNT NEEDED TO FILL THE UNIT IN TOUNT
RETURN
ADDNBUNT
FA2-8,H2
FA2-11,D2
ADD ONE TO THE NUMBER OF UNITS
GETBFR
FA2-9,D1
ARE THERE ANY BUFFERS
NO
SAVE REGISTERS AND THE RETURN ADDRESS
YES
SUBTRACT 1 FROM THE AVAILABLE BUFFER COUNT; UNLINK A BUFFER
RETURN
DECREMENT THE NO. OF UNITS IN BUFFER IF ENTRY WAS AT ADDNBUNT
PUTERB
IS THE ERB ALREADY QUEUED
YES
NO
PUT THE ERB ON THE BUFFER RETURN QCB
2-5 G2

Chart FA2-20 (220) CPB INITIALIZATION - DISK-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
SETEOM
FA2-11,H3
SET 'LAST SEGMENT' AND LCBEOMSG FIELDS
RETURN
LINKTIC
FA2-9,E5
FA2-11,D3
LINK THE UNIT INTO THE TIC OF THE BUFFER
CLEAR AND INITIALIZE THE OP CODE
RETURN
FINDEST2
FA2-1,E1
FA2-2,A2
FINDESTQ
FA2-3,E3
FA2-8,B5
GET THE DESTINATION QCB ADDRESS AND THE PRIORITY LEVEL
FINDEST2
GET THE ADDRESS OF THE FIRST PRIORITY QCB
IS THERE MORE THAN ONE PRIORITY QCB
NO
YES
PRIORITY QCB ADDRESS = SIZE OF PRIORITY QCB X PRIORITY LEVEL
RETURN

Chart FA2-21 (221) CPB INITIALIZATION - DISK-ONLY QUEUING

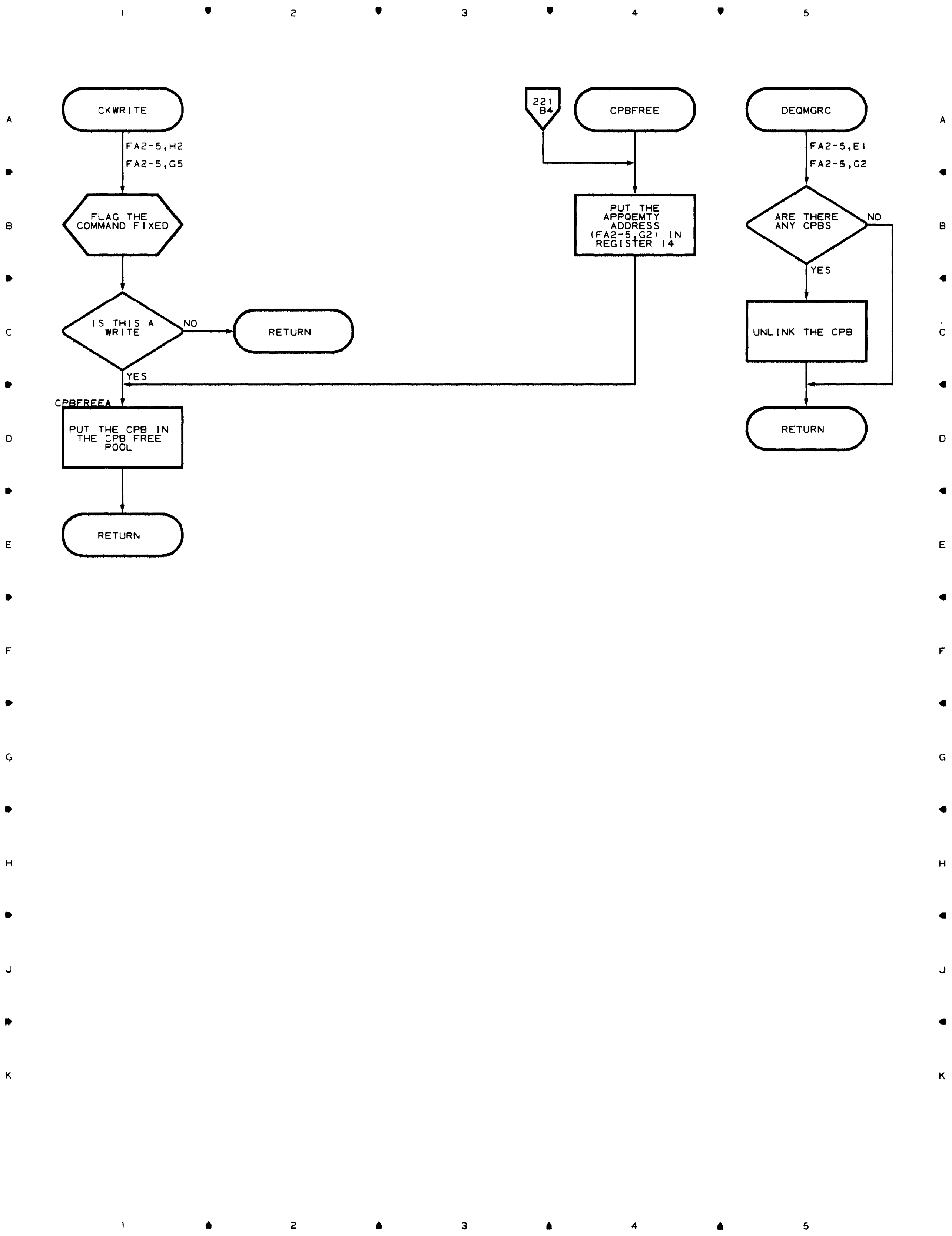
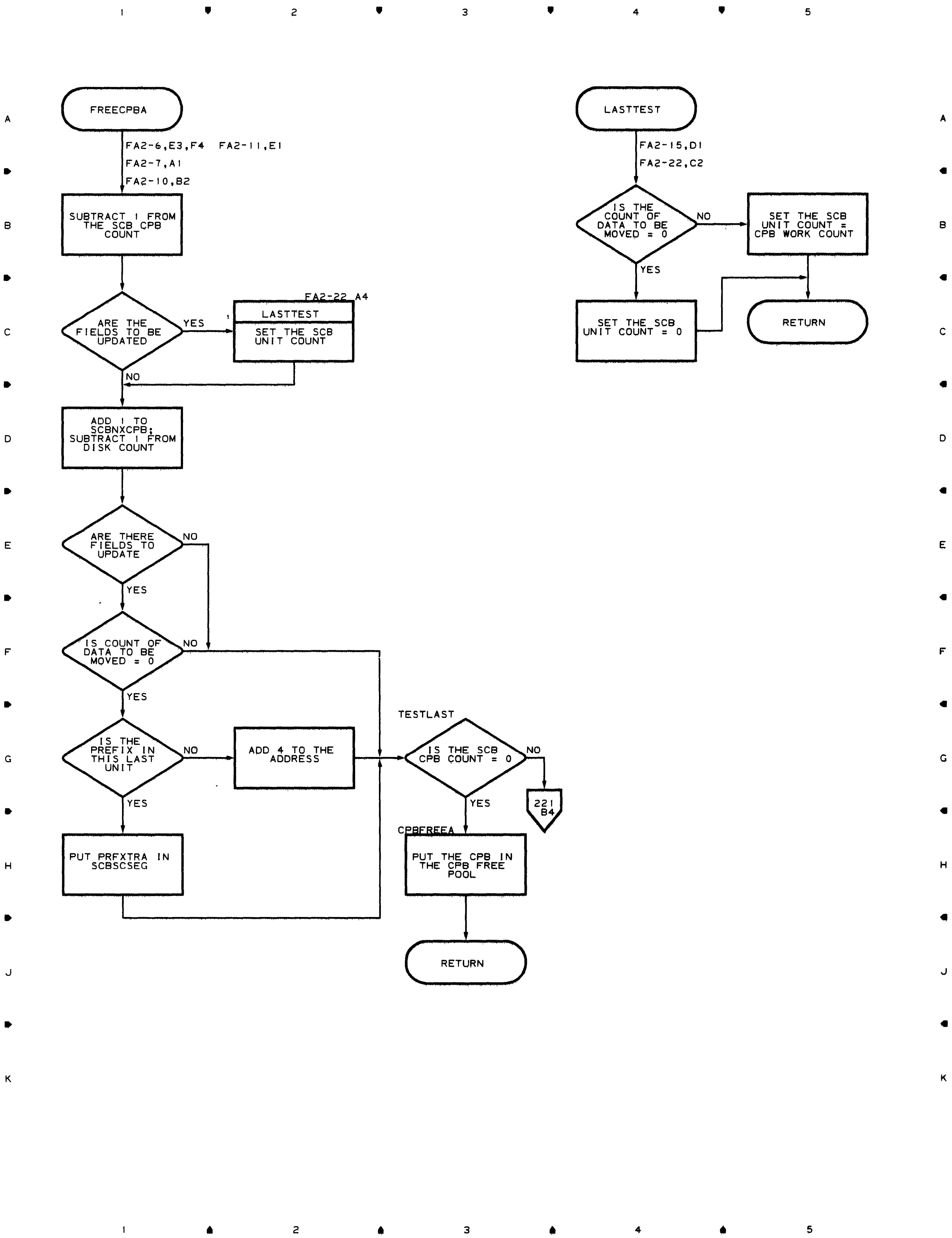


Chart FA2-22 (222) CPB INITIALIZATION - DISK ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FREECPBA
FA2-6,E3,F4 FA2-11,E1
FA2-7,A1
FA2-10,B2
SUBTRACT 1 FROM THE SCB CPB COUNT
ARE THE FIELDS TO BE UPDATED
YES
NO
FA2-22 A4
LASTTEST
SET THE SCB UNIT COUNT
ADD 1 TO SCBNXCPB; SUBTRACT 1 FROM DISK COUNT
ARE THERE FIELDS TO UPDATE
NO
YES
IS COUNT OF DATA TO BE MOVED = 0
NO
YES
IS THE PREFIX IN THIS LAST UNIT
NO
ADD 4 TO THE ADDRESS
YES
PUT PRFXTRA IN SCBSCSEG
TESTLAST
IS THE SCB CPB COUNT = 0
NO
221 B4
YES
CPBFREEA
PUT THE CPB IN THE CPB FREE POOL
RETURN
LASTTEST
FA2-15,D1
FA2-22,C2
IS THE COUNT OF DATA TO BE MOVED = 0
NO
SET THE SCB UNIT COUNT = CPB WORK COUNT
YES
SET THE SCB UNIT COUNT = 0
RETURN

Chart HM-1 (HM1) DESTINATION SCHEDULER

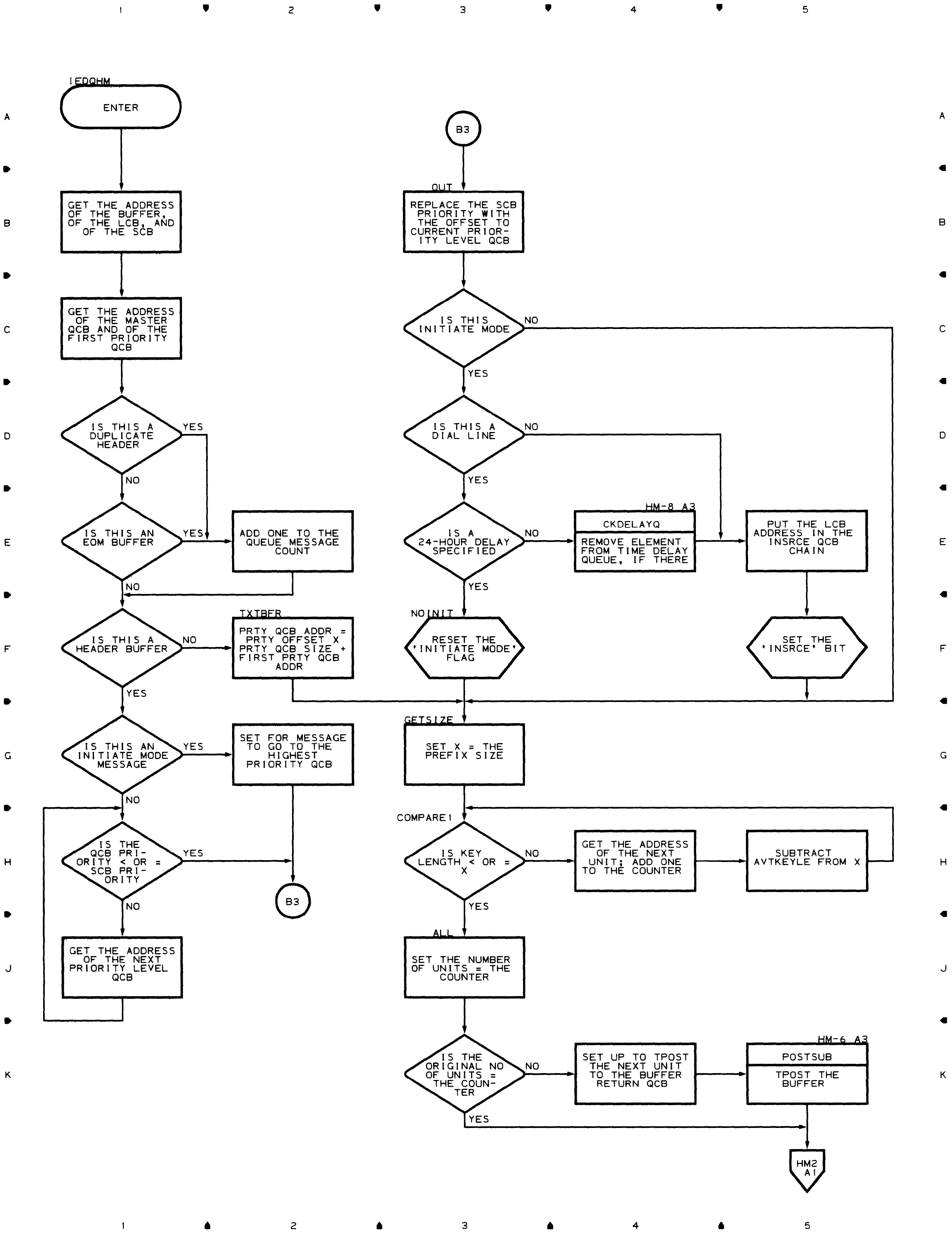

IEDQHM
ENTER
GET THE ADDRESS OF THE BUFFER, OF THE LCB, AND OF THE SCB
GET THE ADDRESS OF THE MASTER QCB AND OF THE FIRST PRIORITY QCB
IS THIS A DUPLICATE HEADER
YES
NO
IS THIS AN EOM BUFFER
YES
ADD ONE TO THE QUEUE MESSAGE COUNT
NO
IS THIS A HEADER BUFFER
NO
TXTBFR
PRTY QCB ADDR = PRTY OFFSET X PRTY QCB SIZE + FIRST PRTY QCB ADDR
YES
IS THIS AN INITIATE MODE MESSAGE
YES
SET FOR MESSAGE TO GO TO THE HIGHEST PRIORITY QCB
NO
IS THE QCB PRI- ORITY < OR = SCB PRI- ORITY
YES
B3
NO
GET THE ADDRESS OF THE NEXT PRIORITY LEVEL QCB
B3
OUT
REPLACE THE SCB PRIORITY WITH THE OFFSET TO CURRENT PRIOR- ITY LEVEL QCB
IS THIS INITIATE MODE
NO
YES
IS THIS A DIAL LINE
NO
YES
IS A 24-HOUR DELAY SPECIFIED
NO
HM-8 A3
CKDELAYQ
REMOVE ELEMENT FROM TIME DELAY QUEUE, IF THERE
PUT THE LCB ADDRESS IN THE INSRCE QCB CHAIN
YES
NOINIT
RESET THE 'INITIATE MODE' FLAG
SET THE 'INSRCE' BIT
GETSIZE
SET X = THE PREFIX SIZE
COMPARE1
IS KEY LENGTH < OR = X
NO
GET THE ADDRESS OF THE NEXT UNIT; ADD ONE TO THE COUNTER
SUBTRACT AVTKEYLE FROM X
YES
ALL
SET THE NUMBER OF UNITS = THE COUNTER
IS THE ORIGINAL NO OF UNITS = THE COUN- TER
NO
SET UP TO TPOST THE NEXT UNIT TO THE BUFFER RETURN QCB
HM-6 A3
POSTSUB
TPOST THE BUFFER
YES
HM2 A1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart HM-2 (HM2) DESTINATION SCHEDULER

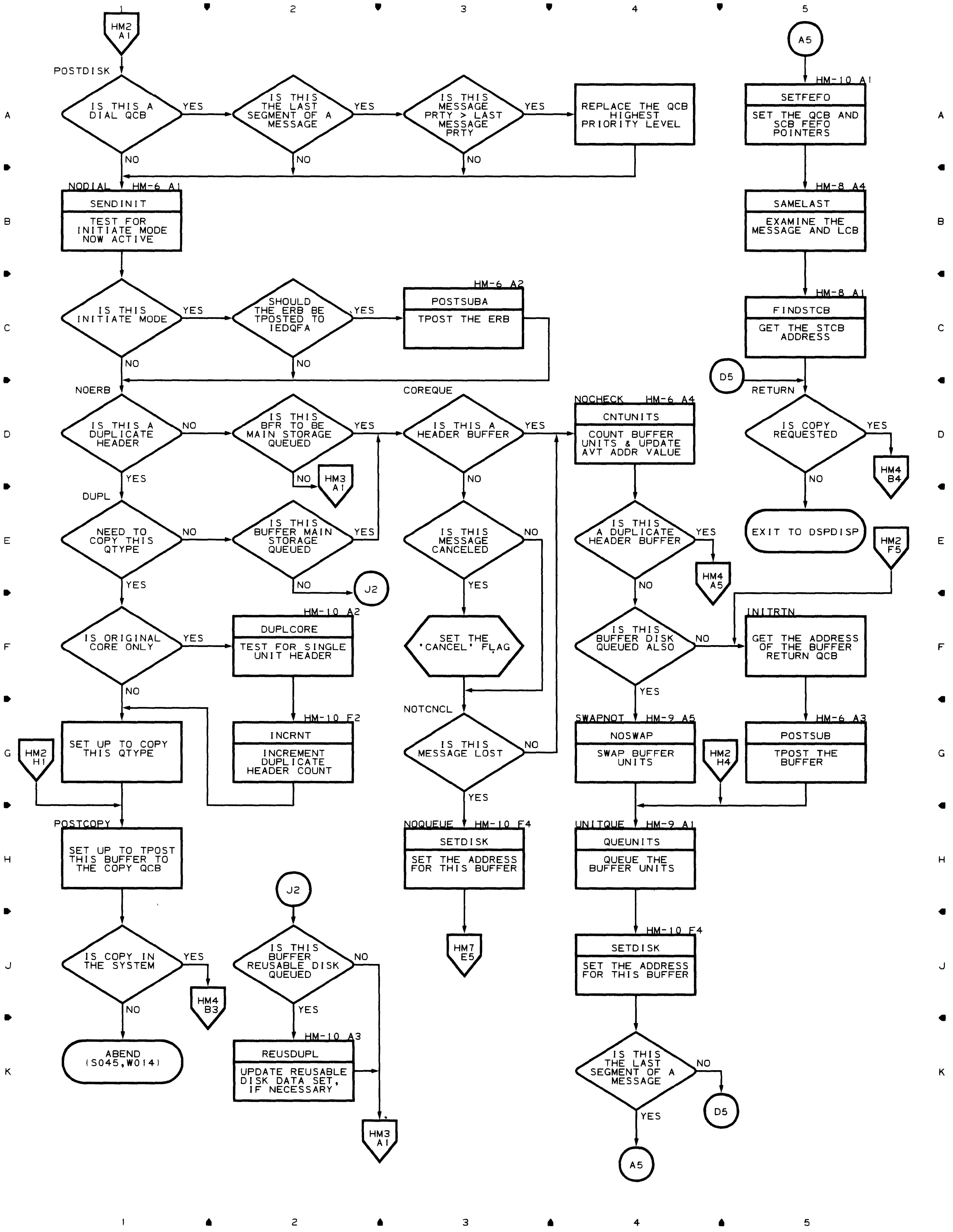
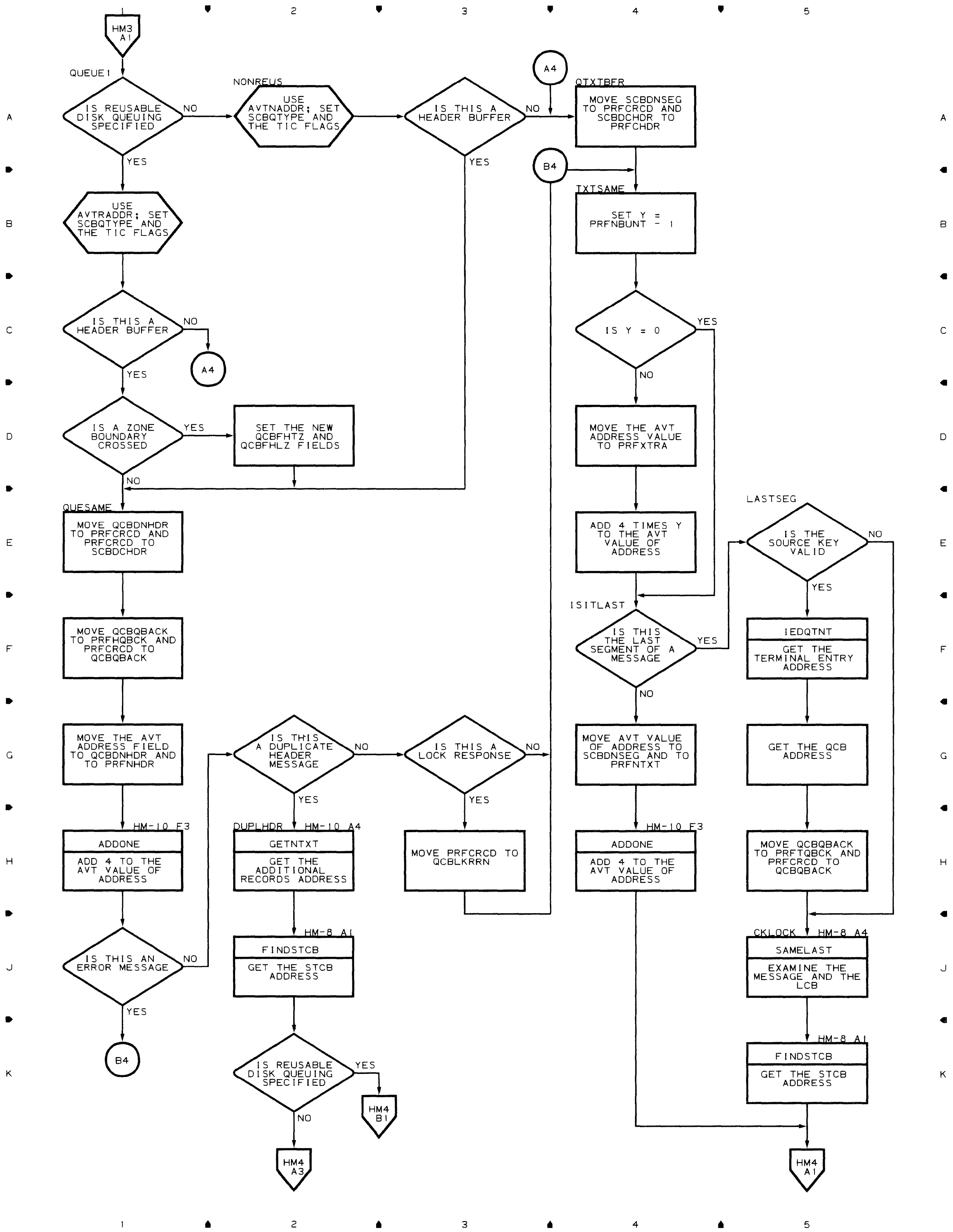


Chart HM-3 (HM3) DESTINATION SCHEDULER



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
HM3 A1
QUEUE1
IS REUSABLE DISK QUEUING SPECIFIED
NO
YES
NONREUS
USE AVTNADDR; SET SCBQTYPE AND THE TIC FLAGS
IS THIS A HEADER BUFFER
NO
YES
A4
QTXTBFR
MOVE SCBDNSEG TO PRFCRCD AND SCBDCHDR TO PRFCHDR
B4
TXTSAME
SET Y = PRFNBUNT - 1
USE AVTRADDR; SET SCBQTYPE AND THE TIC FLAGS
IS THIS A HEADER BUFFER
NO
A4
YES
IS Y = 0
YES
NO
IS A ZONE BOUNDARY CROSSED
YES
SET THE NEW QCBFHTZ AND QCBFHLZ FIELDS
NO
MOVE THE AVT ADDRESS VALUE TO PRFXTRA
QUESAME
MOVE QCBDNHDR TO PRFCRCD AND PRFCRCD TO SCBDCHDR
ADD 4 TIMES Y TO THE AVT VALUE OF ADDRESS
LASTSEG
IS THE SOURCE KEY VALID
NO
YES
ISITLAST
MOVE QCBQBACK TO PRFHQBCK AND PRFCRCD TO QCBQBACK
IS THIS THE LAST SEGMENT OF A MESSAGE
YES
NO
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
MOVE THE AVT ADDRESS FIELD TO QCBDNHDR AND TO PRFNHDR
IS THIS A DUPLICATE HEADER MESSAGE
NO
YES
IS THIS A LOCK RESPONSE
NO
YES
MOVE AVT VALUE OF ADDRESS TO SCBDNSEG AND TO PRFNTXT
GET THE QCB ADDRESS
HM-10 F3
ADDONE
ADD 4 TO THE AVT VALUE OF ADDRESS
DUPLHDR
HM-10 A4
GETNTXT
GET THE ADDITIONAL RECORDS ADDRESS
MOVE PRFCRCD TO QCBLKRRN
HM-10 F3
ADDONE
ADD 4 TO THE AVT VALUE OF ADDRESS
MOVE QCBQBACK TO PRFTQBCK AND PRFCRCD TO QCBQBACK
IS THIS AN ERROR MESSAGE
NO
YES
HM-8 A1
FINDSTCB
GET THE STCB ADDRESS
CKLOCK
HM-8 A4
SAMELAST
EXAMINE THE MESSAGE AND THE LCB
B4
IS REUSABLE DISK QUEUING SPECIFIED
YES
NO
HM4 B1
HM-8 A1
FINDSTCB
GET THE STCB ADDRESS
HM4 A3
HM4 A1

Chart HM-4 (HM4) DESTINATION SCHEDULER

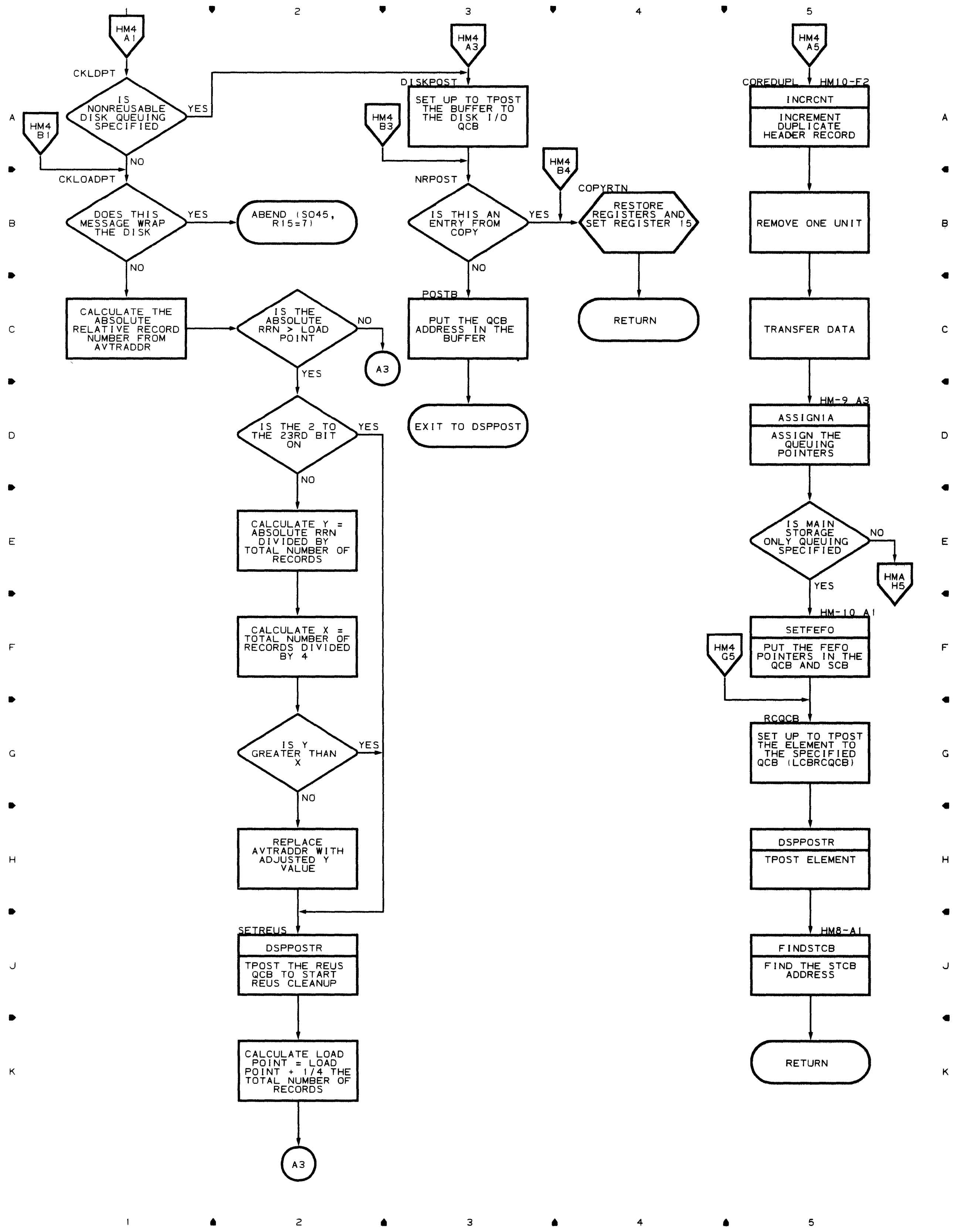
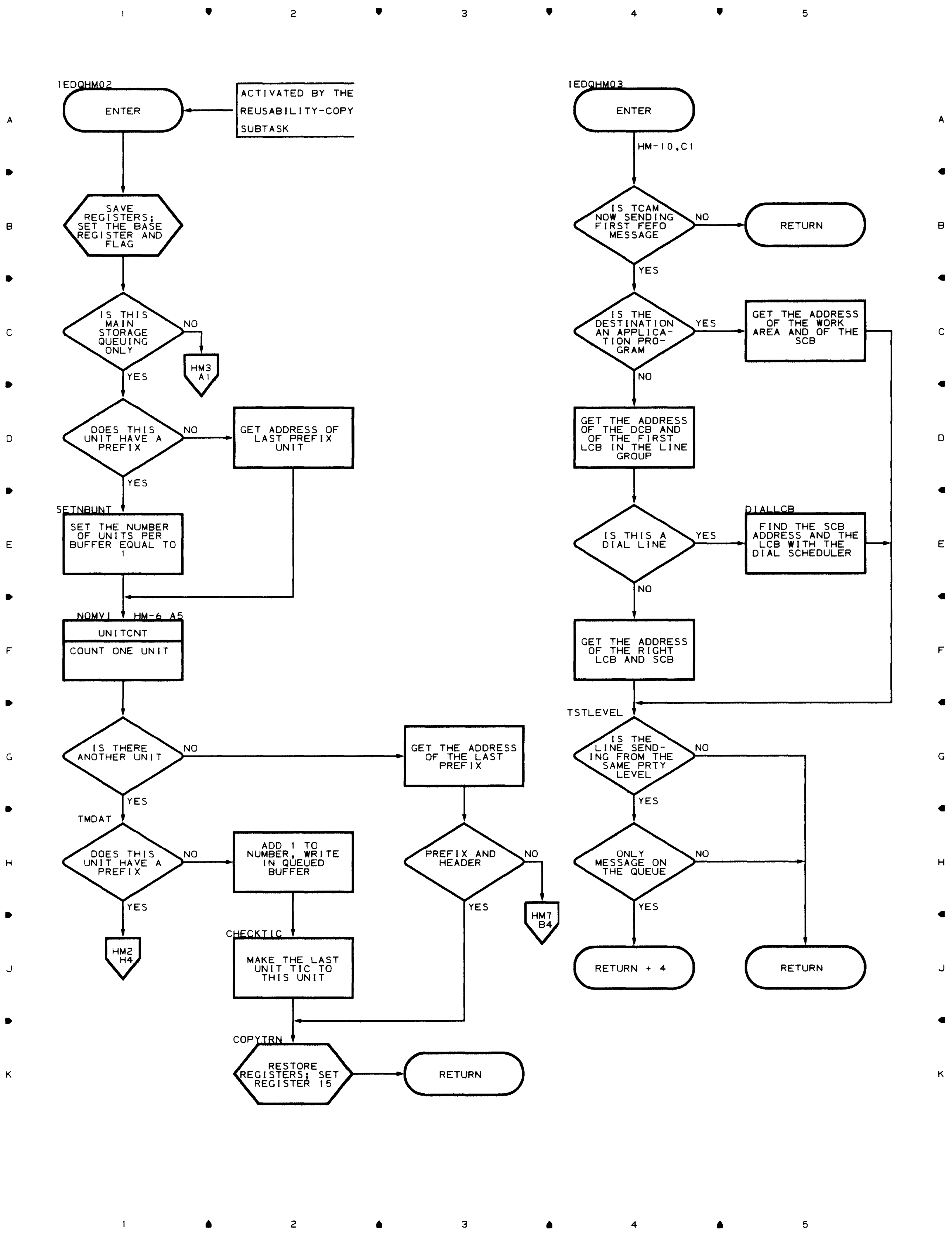

HM4 A1
CKLDPT
IS NONREUSABLE DISK QUEUING SPECIFIED
YES
NO
HM4 B1
CKLOADPT
DOES THIS MESSAGE WRAP THE DISK
YES
ABEND (S045, R15=7)
NO
CALCULATE THE ABSOLUTE RELATIVE RECORD NUMBER FROM AVTRADDR
IS THE ABSOLUTE RRN > LOAD POINT
NO
A3
YES
IS THE 2 TO THE 23RD BIT ON
YES
NO
CALCULATE Y = ABSOLUTE RRN DIVIDED BY TOTAL NUMBER OF RECORDS
CALCULATE X = TOTAL NUMBER OF RECORDS DIVIDED BY 4
IS Y GREATER THAN X
YES
NO
REPLACE AVTRADDR WITH ADJUSTED Y VALUE
SETREUS
DSPPOSTR
TPOST THE REUS QCB TO START REUS CLEANUP
CALCULATE LOAD POINT = LOAD POINT + 1/4 THE TOTAL NUMBER OF RECORDS
A3
HM4 A3
DISKPOST
SET UP TO TPOST THE BUFFER TO THE DISK I/O QCB
HM4 B3
NRPOST
IS THIS AN ENTRY FROM COPY
YES
NO
POSTB
PUT THE QCB ADDRESS IN THE BUFFER
EXIT TO DSPPOST
HM4 B4
COPYRTN
RESTORE REGISTERS AND SET REGISTER 15
RETURN
HM4 A5
COREDUPL HM10-F2
INCRCNT
INCREMENT DUPLICATE HEADER RECORD
REMOVE ONE UNIT
TRANSFER DATA
HM-9 A3
ASSIGNIA
ASSIGN THE QUEUING POINTERS
IS MAIN STORAGE ONLY QUEUING SPECIFIED
NO
HMA H5
YES
HM-10 A1
SETFEFO
PUT THE FEFO POINTERS IN THE QCB AND SCB
HM4 G5
RCQCB
SET UP TO TPOST THE ELEMENT TO THE SPECIFIED QCB (LCBRCQCB)
DSPPOSTR
TPOST ELEMENT
HM8-A1
FINDSTCB
FIND THE STCB ADDRESS
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart HM-5 (HM5) DESTINATION SCHEDULER



IEDQHM02
ENTER
ACTIVATED BY THE REUSABILITY-COPY SUBTASK
SAVE REGISTERS; SET THE BASE REGISTER AND FLAG
IS THIS MAIN STORAGE QUEUING ONLY
NO
HM3 A1
YES
DOES THIS UNIT HAVE A PREFIX
NO
GET ADDRESS OF LAST PREFIX UNIT
YES
SETNBUNT
SET THE NUMBER OF UNITS PER BUFFER EQUAL TO 1
NOMVI
HM-6 A5
UNITCNT
COUNT ONE UNIT
IS THERE ANOTHER UNIT
NO
GET THE ADDRESS OF THE LAST PREFIX
YES
TMDAT
DOES THIS UNIT HAVE A PREFIX
NO
ADD 1 TO NUMBER, WRITE IN QUEUED BUFFER
PREFIX AND HEADER
NO
HM7 B4
YES
YES
HM2 H4
CHECKTIC
MAKE THE LAST UNIT TIC TO THIS UNIT
COPYTRN
RESTORE REGISTERS; SET REGISTER 15
RETURN
IEDQHM03
ENTER
HM-10,C1
IS TCAM NOW SENDING FIRST FEFO MESSAGE
NO
RETURN
YES
IS THE DESTINATION AN APPLICA-TION PRO-GRAM
YES
GET THE ADDRESS OF THE WORK AREA AND OF THE SCB
NO
GET THE ADDRESS OF THE DCB AND OF THE FIRST LCB IN THE LINE GROUP
DIALLCB
IS THIS A DIAL LINE
YES
FIND THE SCB ADDRESS AND THE LCB WITH THE DIAL SCHEDULER
NO
GET THE ADDRESS OF THE RIGHT LCB AND SCB
TSTLEVEL
IS THE LINE SEND-ING FROM THE SAME PRTY LEVEL
NO
YES
ONLY MESSAGE ON THE QUEUE
NO
YES
RETURN + 4
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart HM-6 (HM6) DESTINATION SCHEDULER

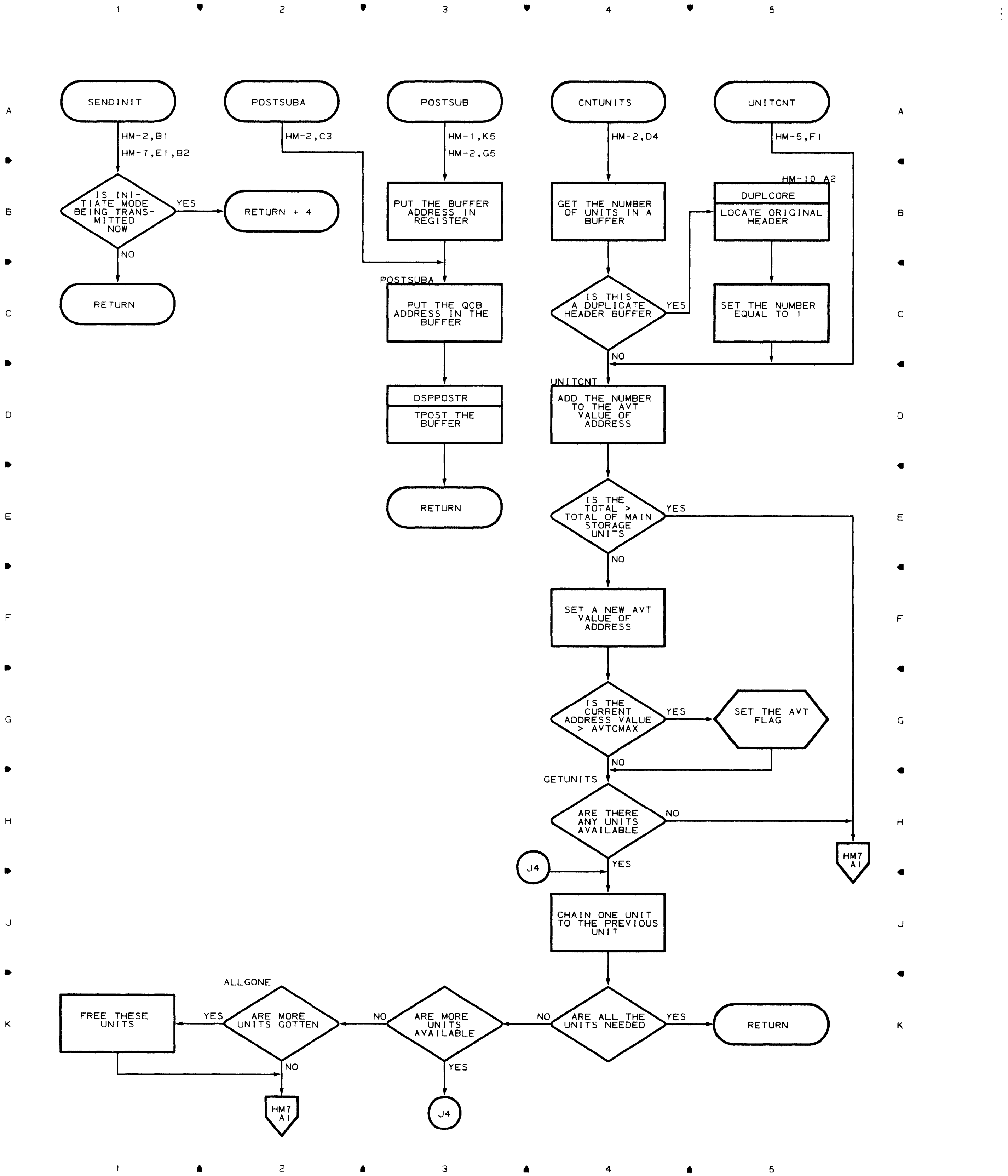

SENDINIT
HM-2,B1
HM-7,E1,B2
IS INITIATE MODE BEING TRANSMITTED NOW
YES
RETURN + 4
NO
RETURN
POSTSUBA
HM-2,C3
POSTSUB
HM-1,K5
HM-2,G5
PUT THE BUFFER ADDRESS IN REGISTER
POSTSUBA
PUT THE QCB ADDRESS IN THE BUFFER
DSPPOSTR
TPOST THE BUFFER
RETURN
CNTUNITS
HM-2,D4
GET THE NUMBER OF UNITS IN A BUFFER
IS THIS A DUPLICATE HEADER BUFFER
YES
NO
UNITCNT
HM-5,F1
HM-10 A2
DUPLCORE
LOCATE ORIGINAL HEADER
SET THE NUMBER EQUAL TO 1
UNITCNT
ADD THE NUMBER TO THE AVT VALUE OF ADDRESS
IS THE TOTAL > TOTAL OF MAIN STORAGE UNITS
YES
NO
SET A NEW AVT VALUE OF ADDRESS
IS THE CURRENT ADDRESS VALUE > AVTCMAX
YES
SET THE AVT FLAG
NO
GETUNITS
ARE THERE ANY UNITS AVAILABLE
NO
HM7 A1
J4
YES
CHAIN ONE UNIT TO THE PREVIOUS UNIT
ARE ALL THE UNITS NEEDED
YES
RETURN
NO
ARE MORE UNITS AVAILABLE
YES
J4
NO
ALLGONE
ARE MORE UNITS GOTTEN
YES
FREE THESE UNITS
NO
HM7 A1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart HM-7 (HM7) DESTINATION SCHEDULER

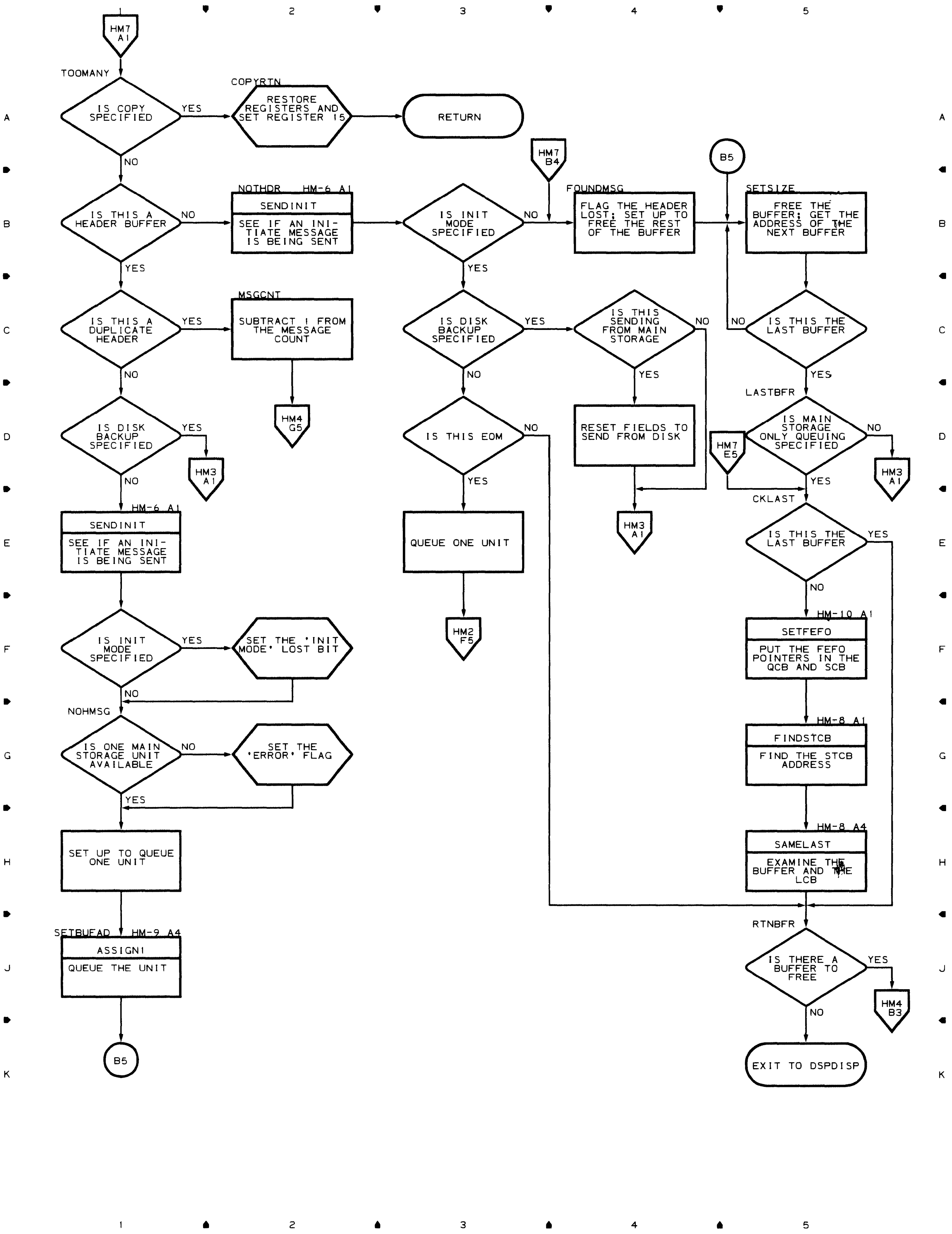


Chart HM-8 (HM8) DESTINATION SCHEDULER

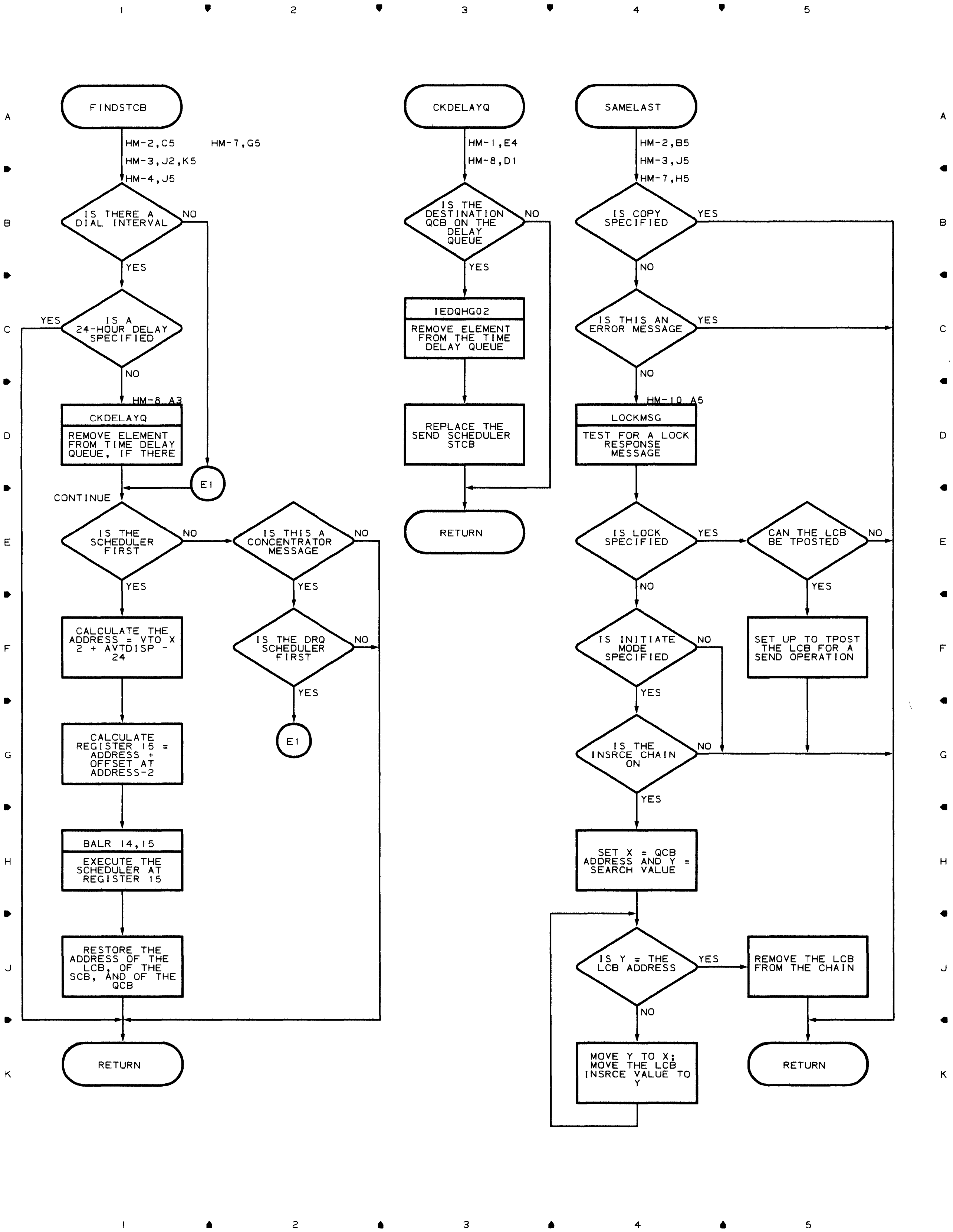
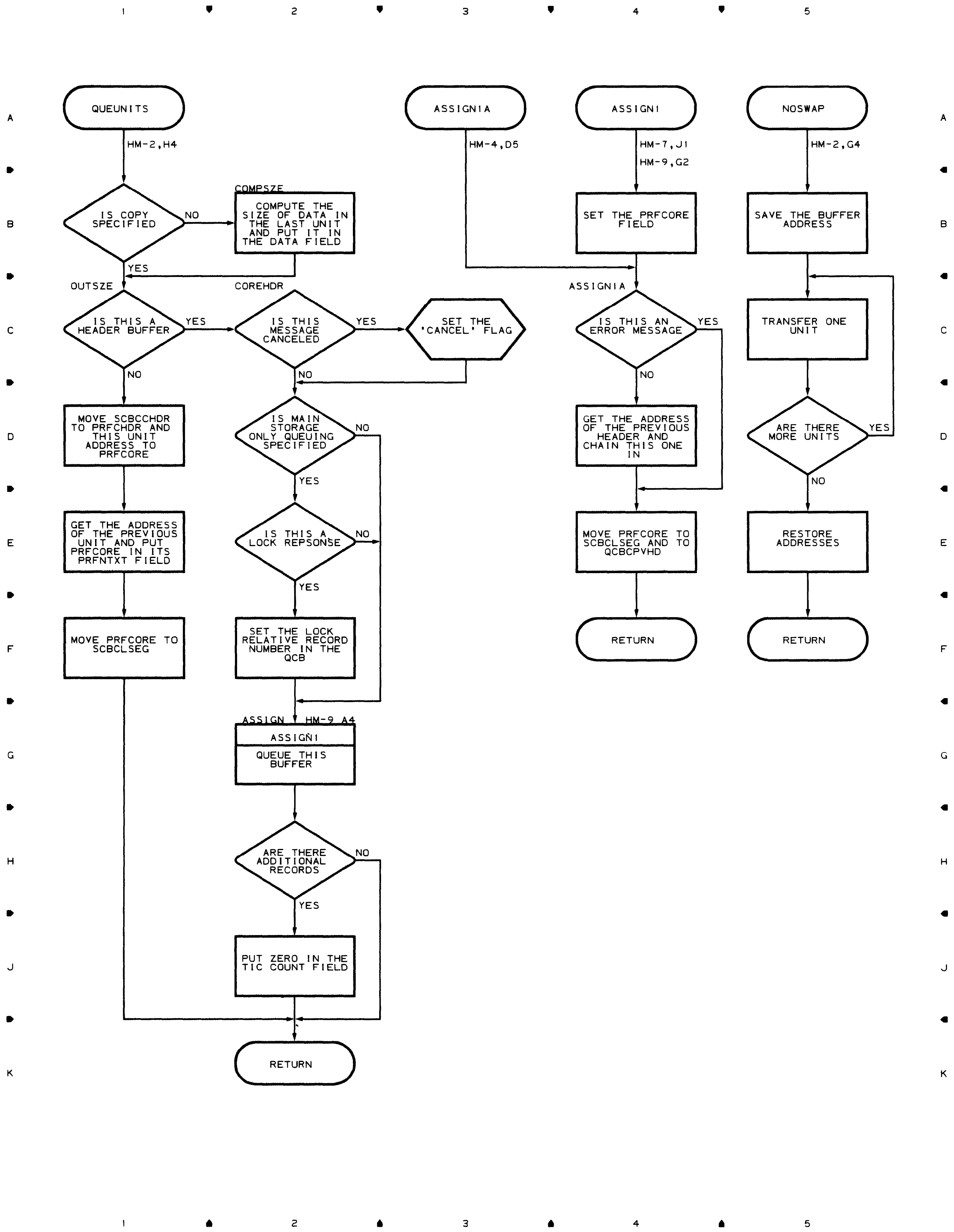
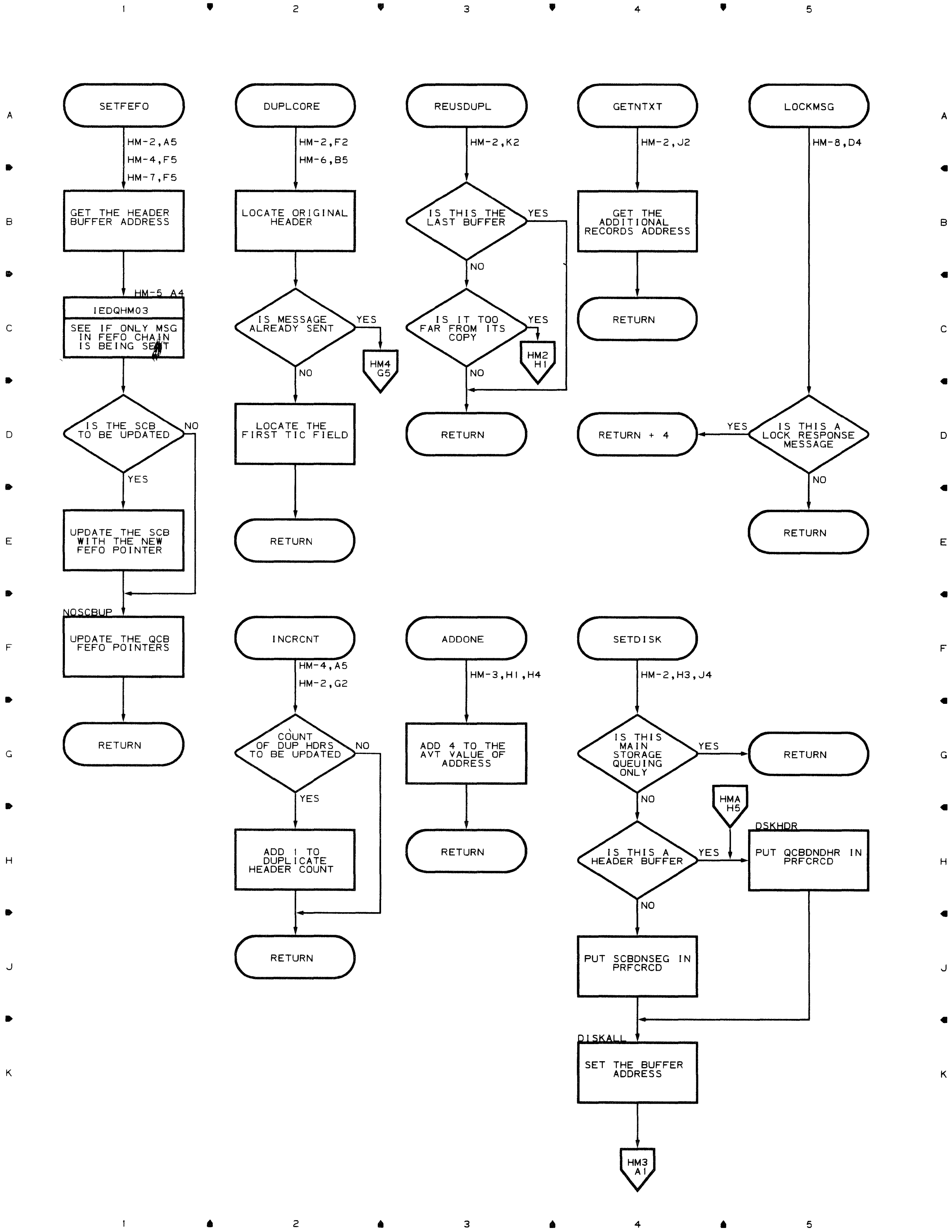

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FINDSTCB
HM-2,C5
HM-7,G5
HM-3,J2,K5
HM-4,J5
IS THERE A DIAL INTERVAL
NO
YES
IS A 24-HOUR DELAY SPECIFIED
YES
NO
HM-8 A3
CKDELAYQ
REMOVE ELEMENT FROM TIME DELAY QUEUE, IF THERE
E1
CONTINUE
IS THE SCHEDULER FIRST
NO
YES
IS THIS A CONCENTRATOR MESSAGE
NO
YES
IS THE DRQ SCHEDULER FIRST
NO
YES
E1
CALCULATE THE ADDRESS = VTO X 2 + AVTDISP - 24
CALCULATE REGISTER 15 = ADDRESS + OFFSET AT ADDRESS-2
BALR 14,15
EXECUTE THE SCHEDULER AT REGISTER 15
RESTORE THE ADDRESS OF THE LCB, OF THE SCB, AND OF THE QCB
RETURN
CKDELAYQ
HM-1,E4
HM-8,D1
IS THE DESTINATION QCB ON THE DELAY QUEUE
NO
YES
IEDQHG02
REMOVE ELEMENT FROM THE TIME DELAY QUEUE
REPLACE THE SEND SCHEDULER STCB
RETURN
SAMELAST
HM-2,B5
HM-3,J5
HM-7,H5
IS COPY SPECIFIED
YES
NO
IS THIS AN ERROR MESSAGE
YES
NO
HM-10 A5
LOCKMSG
TEST FOR A LOCK RESPONSE MESSAGE
IS LOCK SPECIFIED
YES
NO
CAN THE LCB BE TPOSTED
NO
YES
SET UP TO TPOST THE LCB FOR A SEND OPERATION
IS INITIATE MODE SPECIFIED
NO
YES
IS THE INSRCE CHAIN ON
NO
YES
SET X = QCB ADDRESS AND Y = SEARCH VALUE
IS Y = THE LCB ADDRESS
YES
NO
REMOVE THE LCB FROM THE CHAIN
MOVE Y TO X; MOVE THE LCB INSRCE VALUE TO Y
RETURN

Chart HM-9 (HM9) DESTINATION SCHEDULER





1 2 3 4 5
A B C D E F G H J K
SETFEFO
HM-2,A5
HM-4,F5
HM-7,F5
GET THE HEADER BUFFER ADDRESS
HM-5 A4
IEDQHM03
SEE IF ONLY MSG IN FEFO CHAIN IS BEING SENT
IS THE SCB TO BE UPDATED
NO
YES
UPDATE THE SCB WITH THE NEW FEFO POINTER
NOSCBUP
UPDATE THE QCB FEFO POINTERS
RETURN
DUPLCORE
HM-2,F2
HM-6,B5
LOCATE ORIGINAL HEADER
IS MESSAGE ALREADY SENT
YES
HM4 G5
NO
LOCATE THE FIRST TIC FIELD
RETURN
REUSDUPL
HM-2,K2
IS THIS THE LAST BUFFER
YES
NO
IS IT TOO FAR FROM ITS COPY
YES
HM2 H1
NO
RETURN
GETNTXT
HM-2,J2
GET THE ADDITIONAL RECORDS ADDRESS
RETURN
RETURN + 4
LOCKMSG
HM-8,D4
IS THIS A LOCK RESPONSE MESSAGE
YES
NO
RETURN
INCRCNT
HM-4,A5
HM-2,G2
COUNT OF DUP HDRS TO BE UPDATED
NO
YES
ADD 1 TO DUPLICATE HEADER COUNT
RETURN
ADDONE
HM-3,H1,H4
ADD 4 TO THE AVT VALUE OF ADDRESS
RETURN
SETDISK
HM-2,H3,J4
IS THIS MAIN STORAGE QUEUING ONLY
YES
RETURN
NO
HMA H5
DSKHDR
IS THIS A HEADER BUFFER
YES
PUT QCBDNDHR IN PRFCRCD
NO
PUT SCBDNSEG IN PRFCRCD
DISKALL
SET THE BUFFER ADDRESS
HM3 A1

Chart HM1-1 (H11) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

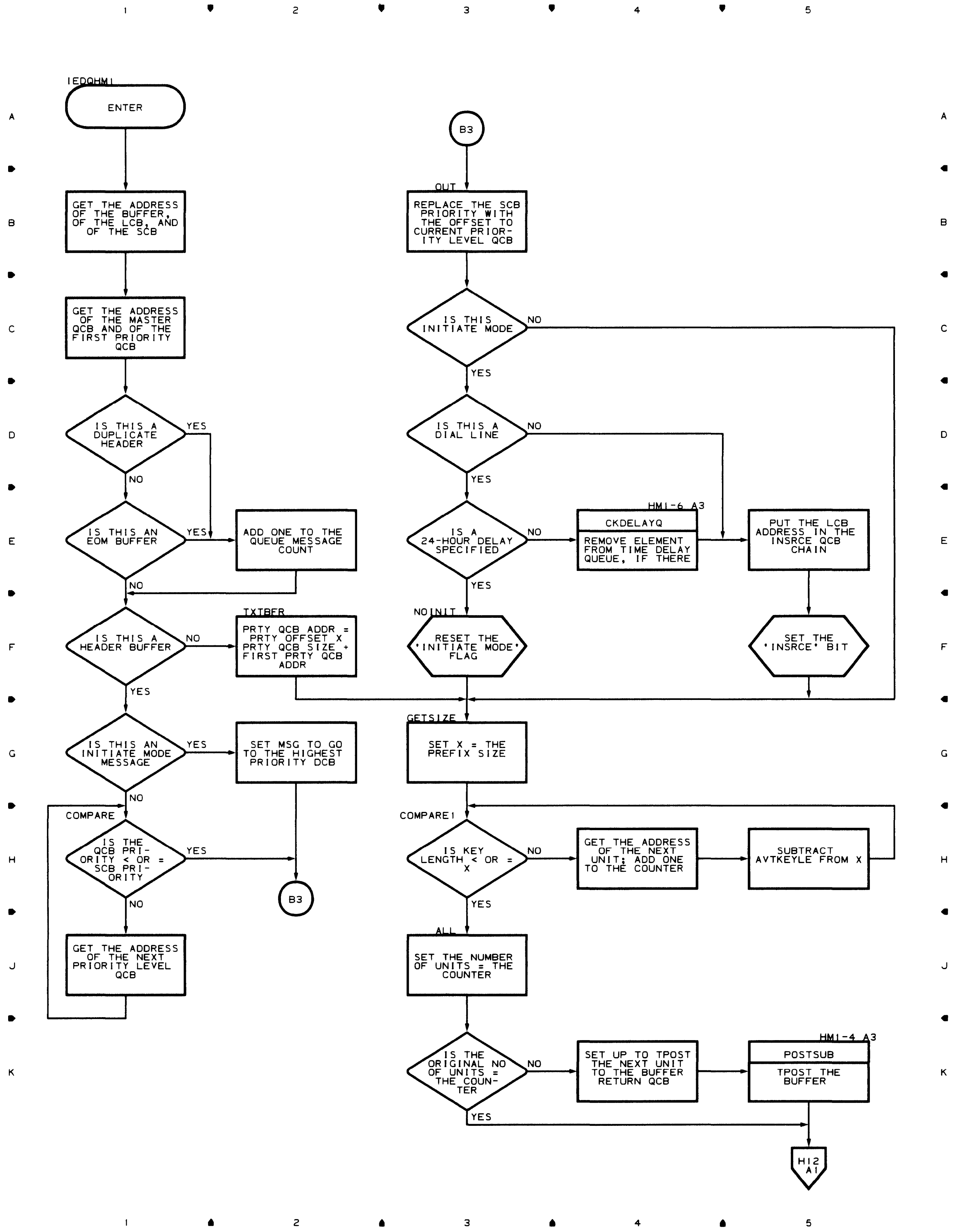

IEDQHM1
ENTER
GET THE ADDRESS OF THE BUFFER, OF THE LCB, AND OF THE SCB
GET THE ADDRESS OF THE MASTER QCB AND OF THE FIRST PRIORITY QCB
IS THIS A DUPLICATE HEADER
YES
NO
IS THIS AN EOM BUFFER
YES
ADD ONE TO THE QUEUE MESSAGE COUNT
NO
IS THIS A HEADER BUFFER
NO
TXTBFR
PRTY QCB ADDR = PRTY OFFSET X PRTY QCB SIZE + FIRST PRTY QCB ADDR
YES
IS THIS AN INITIATE MODE MESSAGE
YES
SET MSG TO GO TO THE HIGHEST PRIORITY DCB
NO
COMPARE
IS THE QCB PRIORITY < OR = SCB PRIORITY
YES
B3
NO
GET THE ADDRESS OF THE NEXT PRIORITY LEVEL QCB
B3
OUT
REPLACE THE SCB PRIORITY WITH THE OFFSET TO CURRENT PRIORITY LEVEL QCB
IS THIS INITIATE MODE
NO
YES
IS THIS A DIAL LINE
NO
YES
IS A 24-HOUR DELAY SPECIFIED
NO
HM1-6 A3
CKDELAYQ
REMOVE ELEMENT FROM TIME DELAY QUEUE, IF THERE
PUT THE LCB ADDRESS IN THE INSRCE QCB CHAIN
YES
NOINIT
RESET THE 'INITIATE MODE' FLAG
SET THE 'INSRCE' BIT
GETSIZE
SET X = THE PREFIX SIZE
COMPARE1
IS KEY LENGTH < OR = X
NO
GET THE ADDRESS OF THE NEXT UNIT; ADD ONE TO THE COUNTER
SUBTRACT AVTKEYLE FROM X
YES
ALL
SET THE NUMBER OF UNITS = THE COUNTER
IS THE ORIGINAL NO OF UNITS = THE COUNTER
NO
SET UP TO TPOST THE NEXT UNIT TO THE BUFFER RETURN QCB
HM1-4 A3
POSTSUB
TPOST THE BUFFER
YES
H12 A1
1 2 3 4 5
A B C D E F G H J K

Chart HM1-2 (H12) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

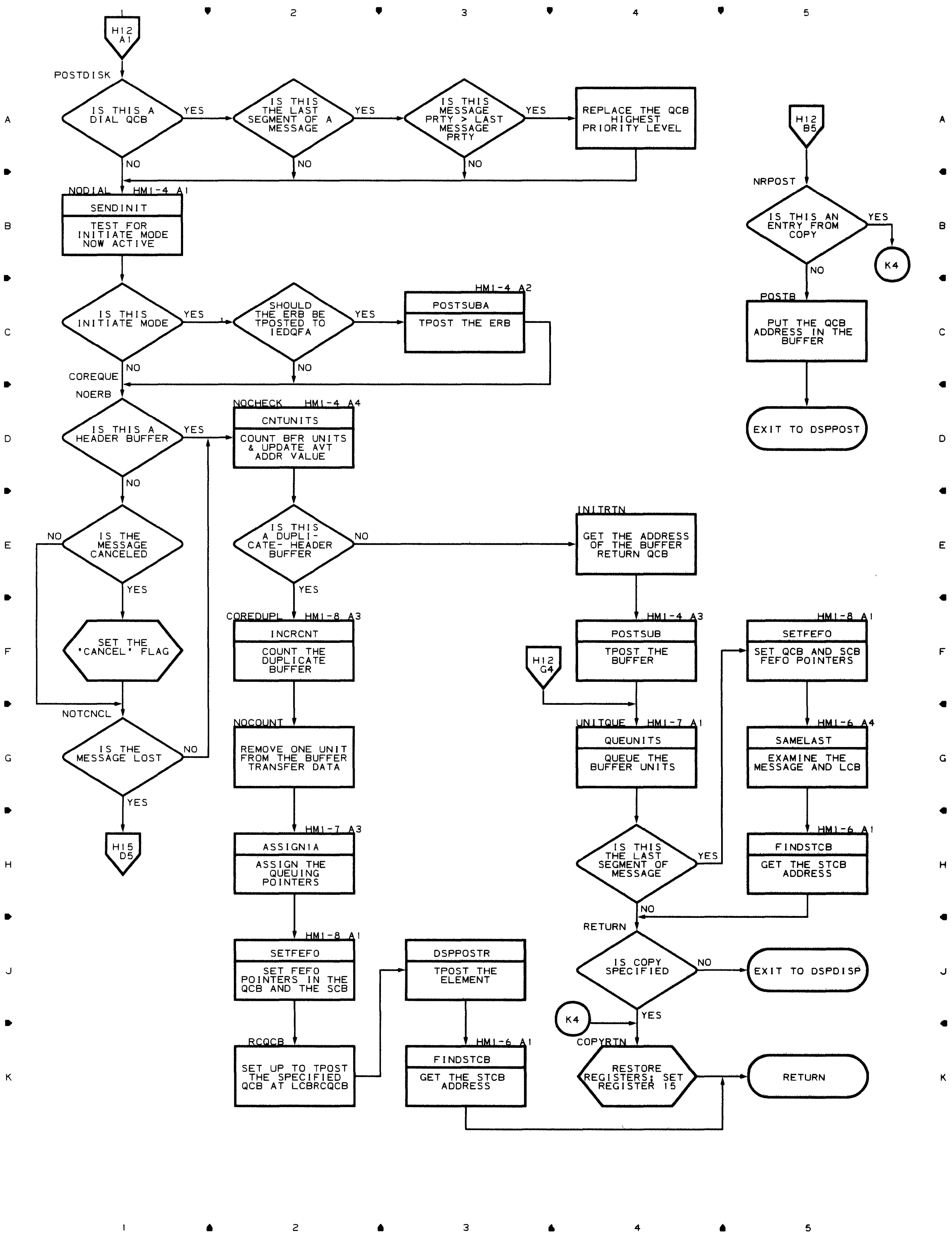

H12 A1
POSTDISK
IS THIS A DIAL QCB
YES
NO
IS THIS THE LAST SEGMENT OF A MESSAGE
IS THIS MESSAGE PRTY > LAST MESSAGE PRTY
REPLACE THE QCB HIGHEST PRIORITY LEVEL
NODIAL HM1-4 A1
SENDINIT
TEST FOR INITIATE MODE NOW ACTIVE
IS THIS INITIATE MODE
SHOULD THE ERB BE TPOSTED TO IEDQFA
HM1-4 A2
POSTSUBA
TPOST THE ERB
COREQUE
NOERB
IS THIS A HEADER BUFFER
NOCHECK HM1-4 A4
CNTUNITS
COUNT BFR UNITS & UPDATE AVT ADDR VALUE
IS THE MESSAGE CANCELED
SET THE 'CANCEL' FLAG
NOTCNCL
IS THE MESSAGE LOST
H15 D5
IS THIS A DUPLI-CATE- HEADER BUFFER
INITRTN
GET THE ADDRESS OF THE BUFFER RETURN QCB
COREDUPL HM1-8 A3
INCRCNT
COUNT THE DUPLICATE BUFFER
NOCOUNT
REMOVE ONE UNIT FROM THE BUFFER TRANSFER DATA
HM1-7 A3
ASSIGNIA
ASSIGN THE QUEUING POINTERS
HM1-8 A1
SETFEFO
SET FEFO POINTERS IN THE QCB AND THE SCB
RCQCB
SET UP TO TPOST THE SPECIFIED QCB AT LCBRCQCB
DSPPOSTR
TPOST THE ELEMENT
HM1-6 A1
FINDSTCB
GET THE STCB ADDRESS
HM1-4 A3
POSTSUB
TPOST THE BUFFER
H12 G4
UNITQUE HM1-7 A1
QUEUNITS
QUEUE THE BUFFER UNITS
IS THIS THE LAST SEGMENT OF MESSAGE
RETURN
IS COPY SPECIFIED
K4
COPYRTN
RESTORE REGISTERS; SET REGISTER 15
RETURN
EXIT TO DSPDISP
HM1-8 A1
SETFEFO
SET QCB AND SCB FEFO POINTERS
HM1-6 A4
SAMELAST
EXAMINE THE MESSAGE AND LCB
HM1-6 A1
FINDSTCB
GET THE STCB ADDRESS
H12 B5
NRPOST
IS THIS AN ENTRY FROM COPY
K4
POSTB
PUT THE QCB ADDRESS IN THE BUFFER
EXIT TO DSPPOST
1 2 3 4 5
A B C D E F G H J K

Chart HM1-3 (H13) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

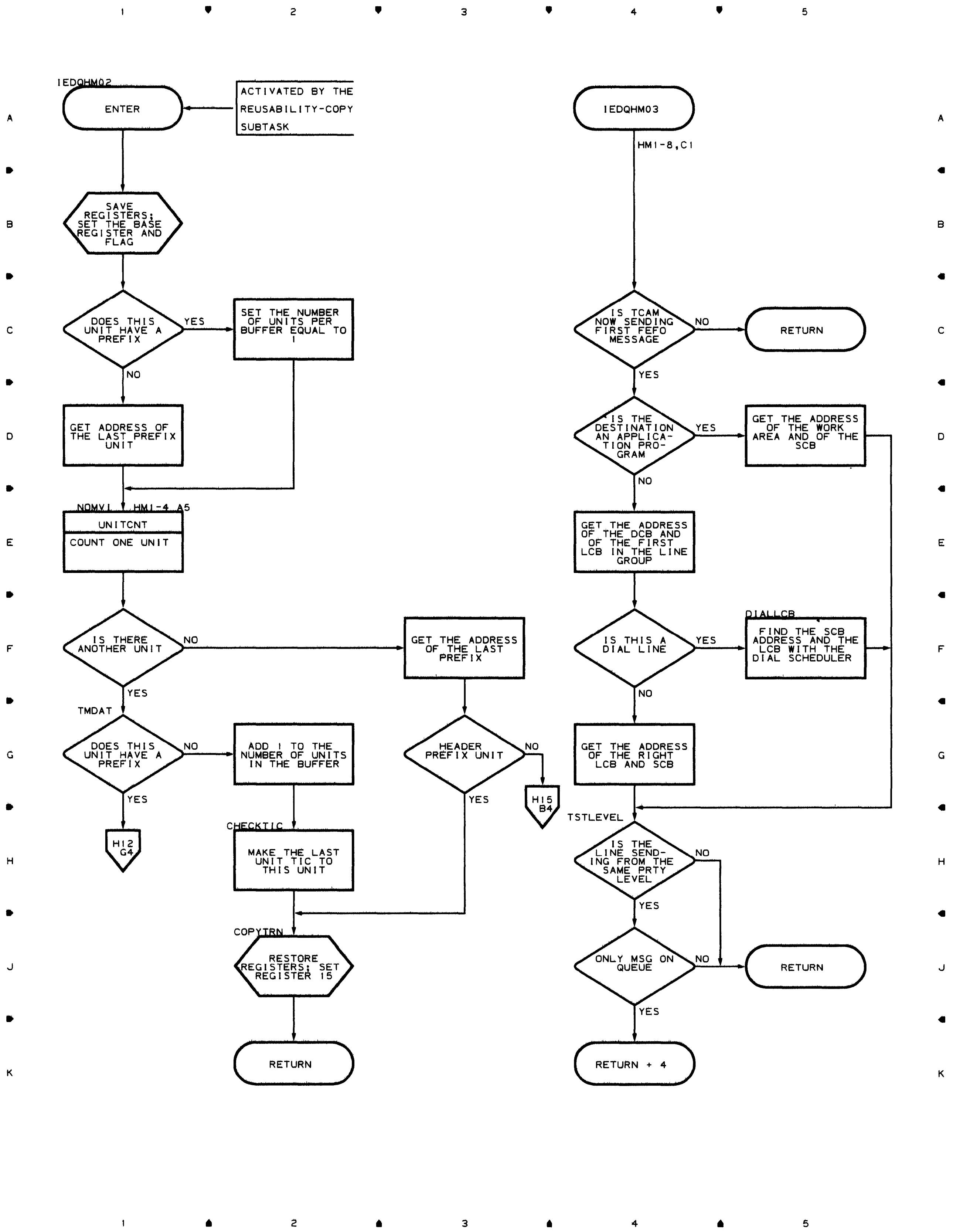

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
IEDQHM02
ENTER
ACTIVATED BY THE REUSABILITY-COPY SUBTASK
SAVE REGISTERS; SET THE BASE REGISTER AND FLAG
DOES THIS UNIT HAVE A PREFIX
YES
SET THE NUMBER OF UNITS PER BUFFER EQUAL TO 1
NO
GET ADDRESS OF THE LAST PREFIX UNIT
NOMVL
HM1-4 A5
UNITCNT
COUNT ONE UNIT
IS THERE ANOTHER UNIT
NO
GET THE ADDRESS OF THE LAST PREFIX
YES
TMDAT
DOES THIS UNIT HAVE A PREFIX
NO
ADD 1 TO THE NUMBER OF UNITS IN THE BUFFER
YES
H12 G4
CHECKTIC
MAKE THE LAST UNIT TIC TO THIS UNIT
HEADER PREFIX UNIT
NO
H15 B4
YES
COPYTRN
RESTORE REGISTERS; SET REGISTER 15
RETURN
IEDQHM03
HM1-8,C1
IS TCAM NOW SENDING FIRST FEFO MESSAGE
NO
RETURN
YES
IS THE DESTINATION AN APPLICATION PROGRAM
YES
GET THE ADDRESS OF THE WORK AREA AND OF THE SCB
NO
GET THE ADDRESS OF THE DCB AND OF THE FIRST LCB IN THE LINE GROUP
IS THIS A DIAL LINE
YES
DIALLCB
FIND THE SCB ADDRESS AND THE LCB WITH THE DIAL SCHEDULER
NO
GET THE ADDRESS OF THE RIGHT LCB AND SCB
TSTLEVEL
IS THE LINE SENDING FROM THE SAME PRTY LEVEL
NO
YES
ONLY MSG ON QUEUE
NO
RETURN
YES
RETURN + 4

Chart HM1-4 (H14) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

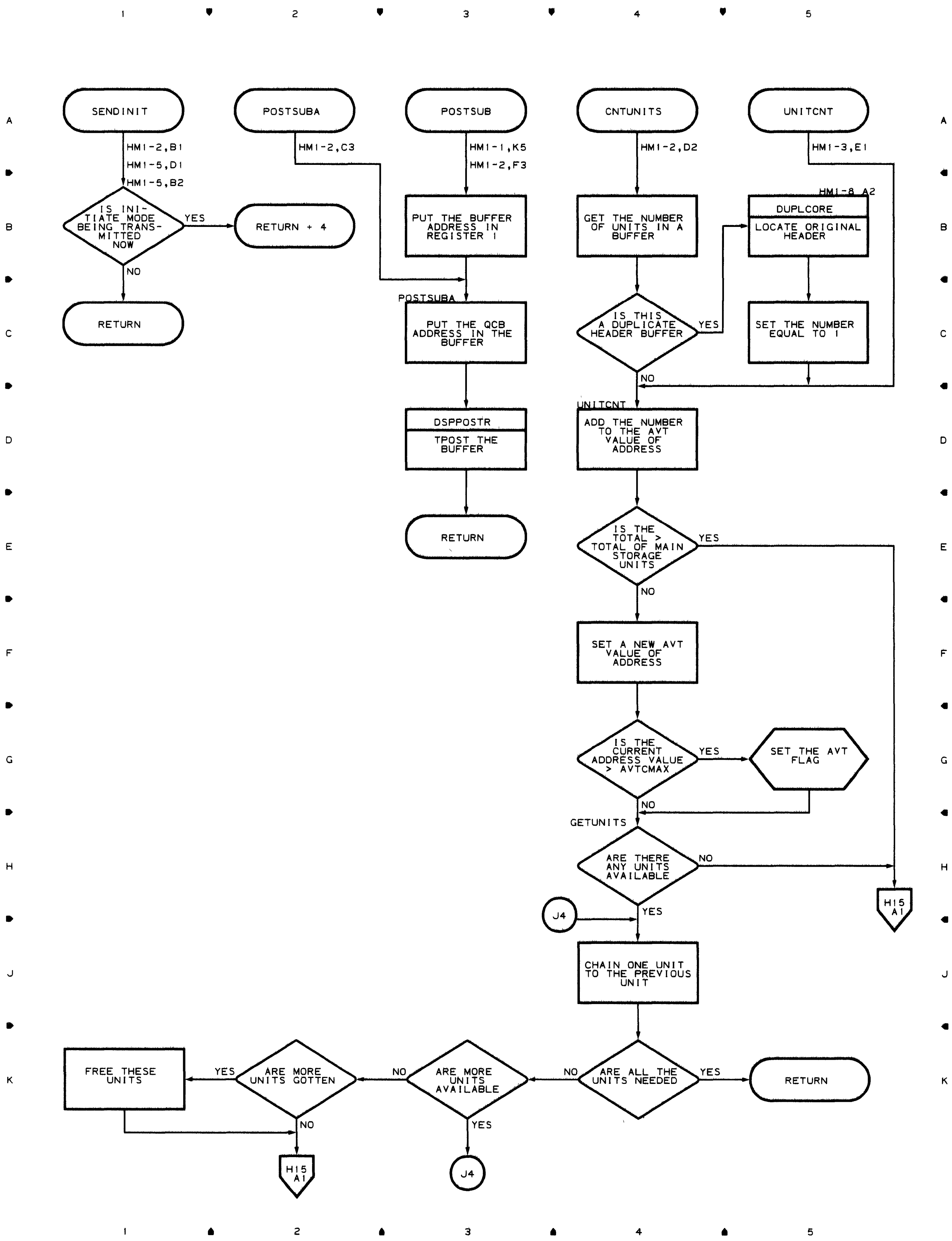

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
SENDINIT
HM1-2,B1
HM1-5,D1
HM1-5,B2
IS INITIATE MODE BEING TRANSMITTED NOW
YES
RETURN + 4
NO
RETURN
POSTSUBA
HM1-2,C3
POSTSUB
HM1-1,K5
HM1-2,F3
PUT THE BUFFER ADDRESS IN REGISTER 1
POSTSUBA
PUT THE QCB ADDRESS IN THE BUFFER
DSPPOSTR
TPOST THE BUFFER
RETURN
CNTUNITS
HM1-2,D2
GET THE NUMBER OF UNITS IN A BUFFER
IS THIS A DUPLICATE HEADER BUFFER
YES
NO
UNITCNT
HM1-3,E1
HM1-8,A2
DUPLCORE
LOCATE ORIGINAL HEADER
SET THE NUMBER EQUAL TO 1
UNITCNT
ADD THE NUMBER TO THE AVT VALUE OF ADDRESS
IS THE TOTAL > TOTAL OF MAIN STORAGE UNITS
YES
NO
SET A NEW AVT VALUE OF ADDRESS
IS THE CURRENT ADDRESS VALUE > AVTCMAX
YES
SET THE AVT FLAG
NO
GETUNITS
ARE THERE ANY UNITS AVAILABLE
NO
H15 A1
J4
YES
CHAIN ONE UNIT TO THE PREVIOUS UNIT
ARE ALL THE UNITS NEEDED
YES
RETURN
NO
ARE MORE UNITS AVAILABLE
YES
J4
NO
ARE MORE UNITS GOTTEN
YES
FREE THESE UNITS
NO
H15 A1

Chart HM1-5 (H15) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

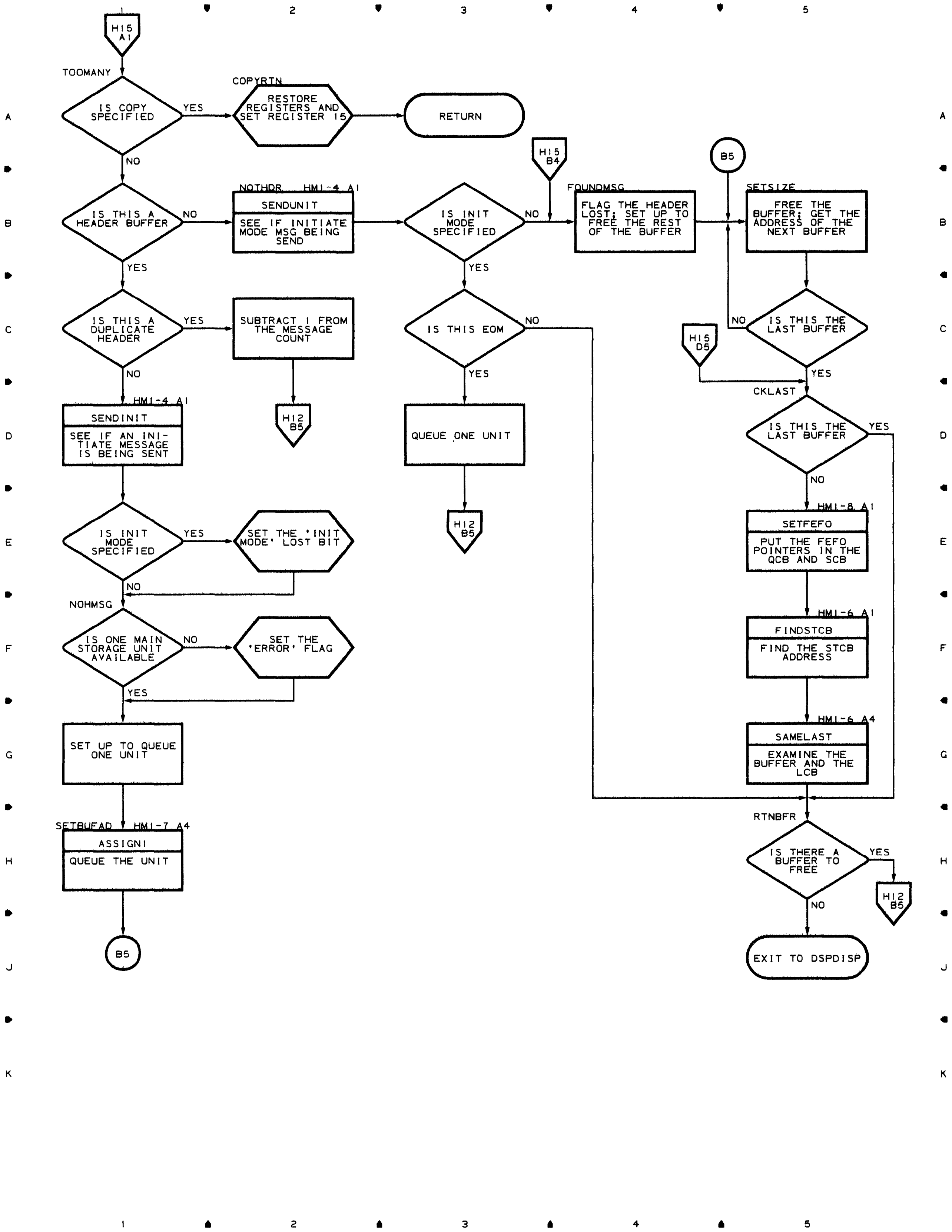


Chart HM1-6 (H16) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

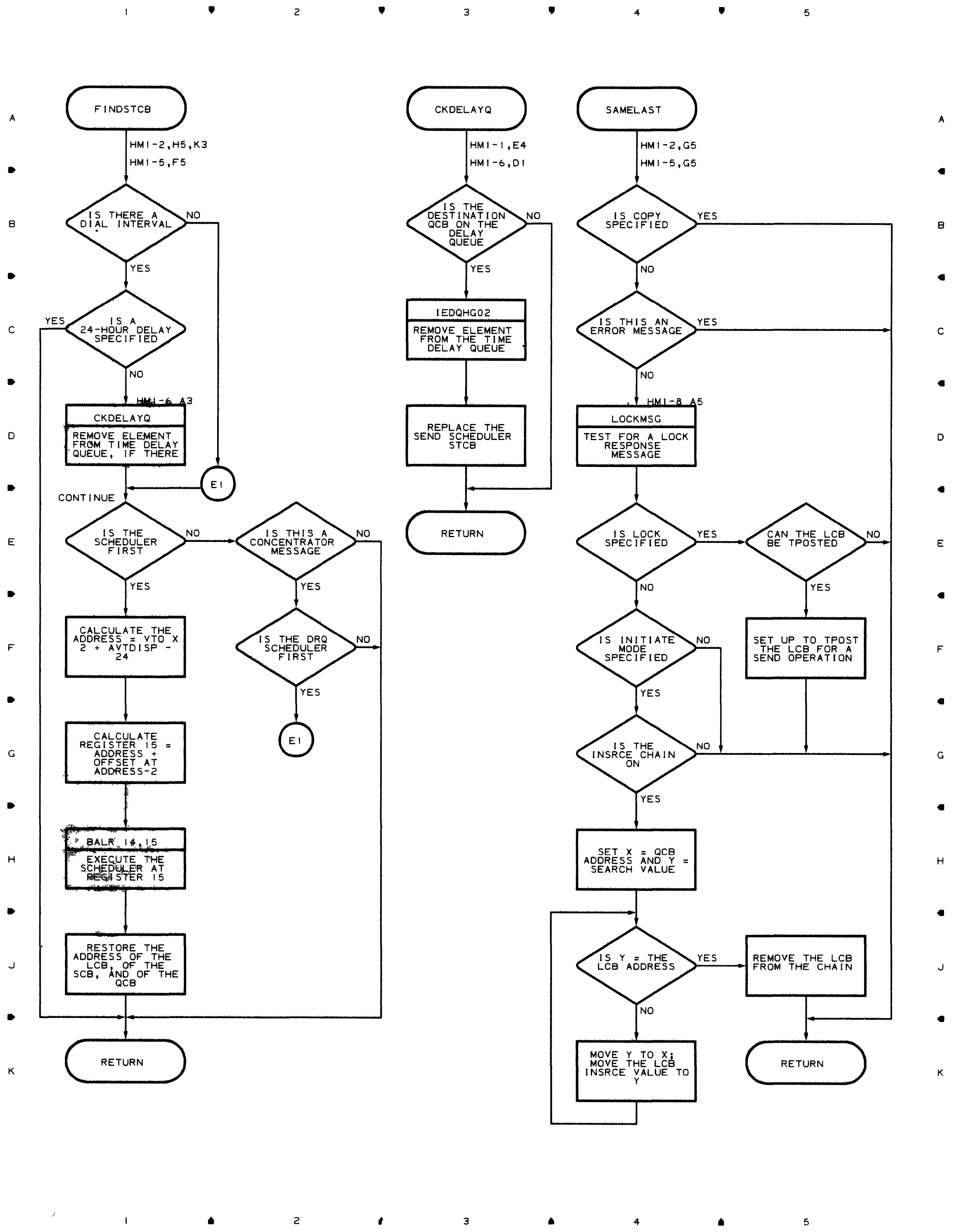

FINDSTCB
HM1-2,H5,K3
HM1-5,F5
IS THERE A DIAL INTERVAL
NO
YES
IS A 24-HOUR DELAY SPECIFIED
YES
NO
HM1-6 A3
CKDELAYQ
REMOVE ELEMENT FROM TIME DELAY QUEUE, IF THERE
E1
CONTINUE
IS THE SCHEDULER FIRST
NO
YES
CALCULATE THE ADDRESS = VTO X 2 + AVTDISP - 24
CALCULATE REGISTER 15 = ADDRESS + OFFSET AT ADDRESS-2
BALR 14,15
EXECUTE THE SCHEDULER AT REGISTER 15
RESTORE THE ADDRESS OF THE LCB, OF THE SCB, AND OF THE QCB
RETURN
IS THIS A CONCENTRATOR MESSAGE
NO
YES
IS THE DRQ SCHEDULER FIRST
NO
YES
E1
CKDELAYQ
HM1-1,E4
HM1-6,D1
IS THE DESTINATION QCB ON THE DELAY QUEUE
NO
YES
IEDQHG02
REMOVE ELEMENT FROM THE TIME DELAY QUEUE
REPLACE THE SEND SCHEDULER STCB
RETURN
SAMELAST
HM1-2,G5
HM1-5,G5
IS COPY SPECIFIED
YES
NO
IS THIS AN ERROR MESSAGE
YES
NO
HM1-8 A5
LOCKMSG
TEST FOR A LOCK RESPONSE MESSAGE
IS LOCK SPECIFIED
YES
NO
CAN THE LCB BE TPOSTED
NO
YES
SET UP TO TPOST THE LCB FOR A SEND OPERATION
IS INITIATE MODE SPECIFIED
NO
YES
IS THE INSRCE CHAIN ON
NO
YES
SET X = QCB ADDRESS AND Y = SEARCH VALUE
IS Y = THE LCB ADDRESS
YES
NO
REMOVE THE LCB FROM THE CHAIN
MOVE Y TO X; MOVE THE LCB INSRCE VALUE TO Y
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart HM1-7 (H17) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

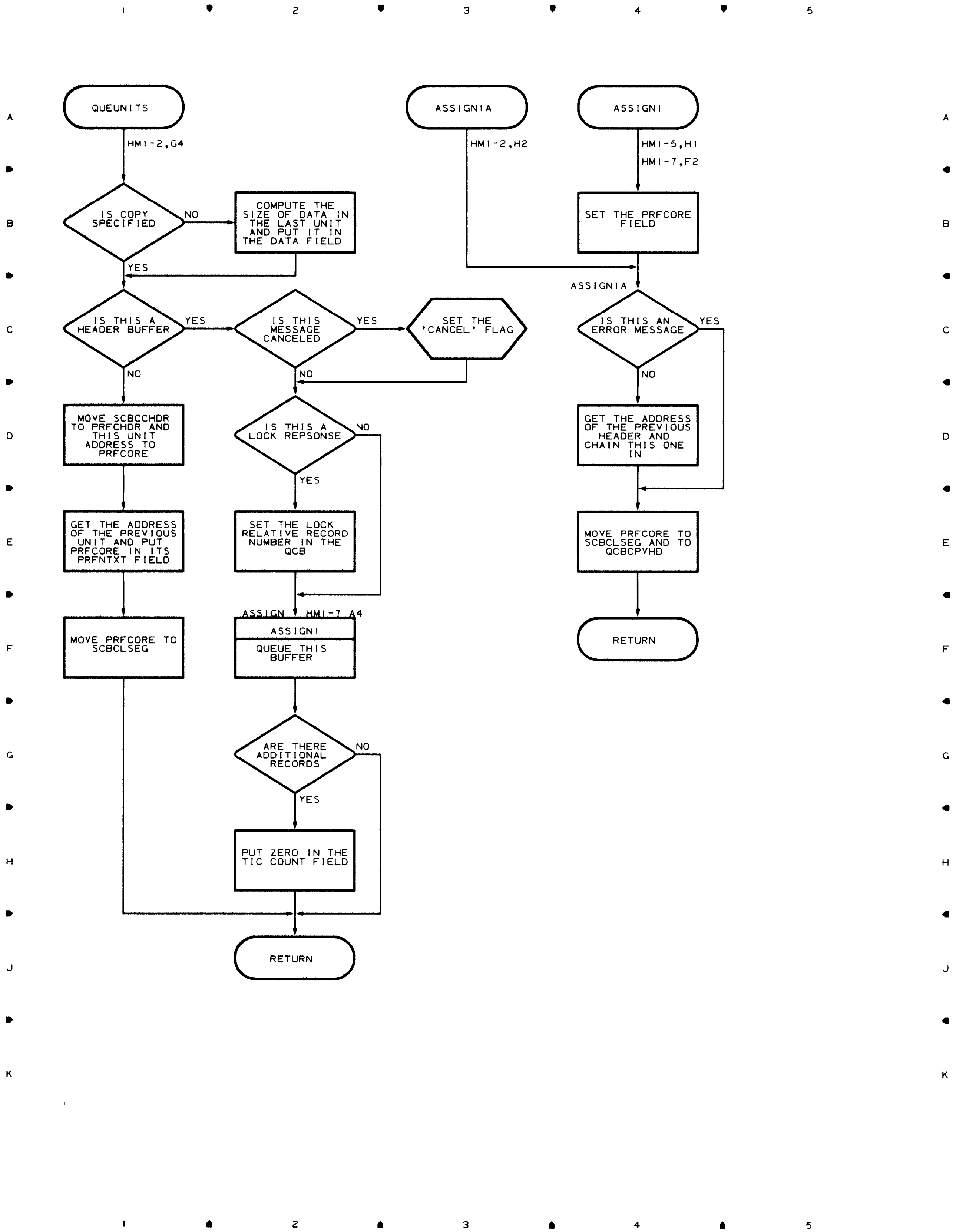

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
QUEUNITS
HM1-2,G4
IS COPY SPECIFIED
NO
COMPUTE THE SIZE OF DATA IN THE LAST UNIT AND PUT IT IN THE DATA FIELD
YES
IS THIS A HEADER BUFFER
YES
IS THIS MESSAGE CANCELED
YES
SET THE 'CANCEL' FLAG
NO
NO
MOVE SCBCCHDR TO PRFCHDR AND THIS UNIT ADDRESS TO PRFCORE
IS THIS A LOCK REPSONSE
NO
YES
GET THE ADDRESS OF THE PREVIOUS UNIT AND PUT PRFCORE IN ITS PRFNTXT FIELD
SET THE LOCK RELATIVE RECORD NUMBER IN THE QCB
ASSIGN
HM1-7 A4
ASSIGN1
QUEUE THIS BUFFER
MOVE PRFCORE TO SCBCLSEG
ARE THERE ADDITIONAL RECORDS
NO
YES
PUT ZERO IN THE TIC COUNT FIELD
RETURN
ASSIGN1A
HM1-2,H2
ASSIGN1
HM1-5,H1
HM1-7,F2
SET THE PRFCORE FIELD
ASSIGN1A
IS THIS AN ERROR MESSAGE
YES
NO
GET THE ADDRESS OF THE PREVIOUS HEADER AND CHAIN THIS ONE IN
MOVE PRFCORE TO SCBCLSEG AND TO QCBCPVHD
RETURN

Chart HM1-8 (H18) DESTINATION SCHEDULER - MAIN-STORAGE-ONLY QUEUING

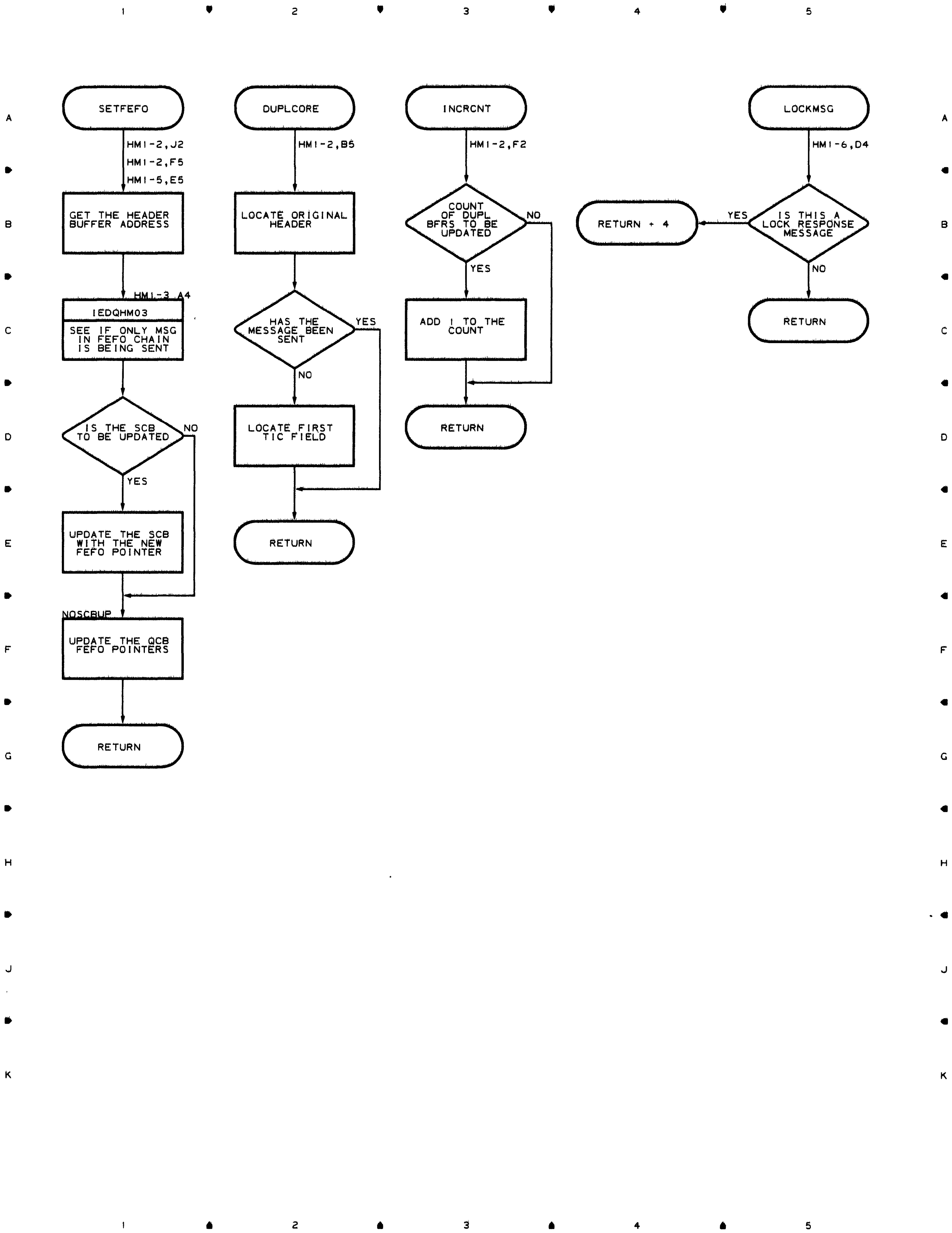


Chart HM2-1 (H21) DESTINATION SCHEDULER - DISK-ONLY QUEUING

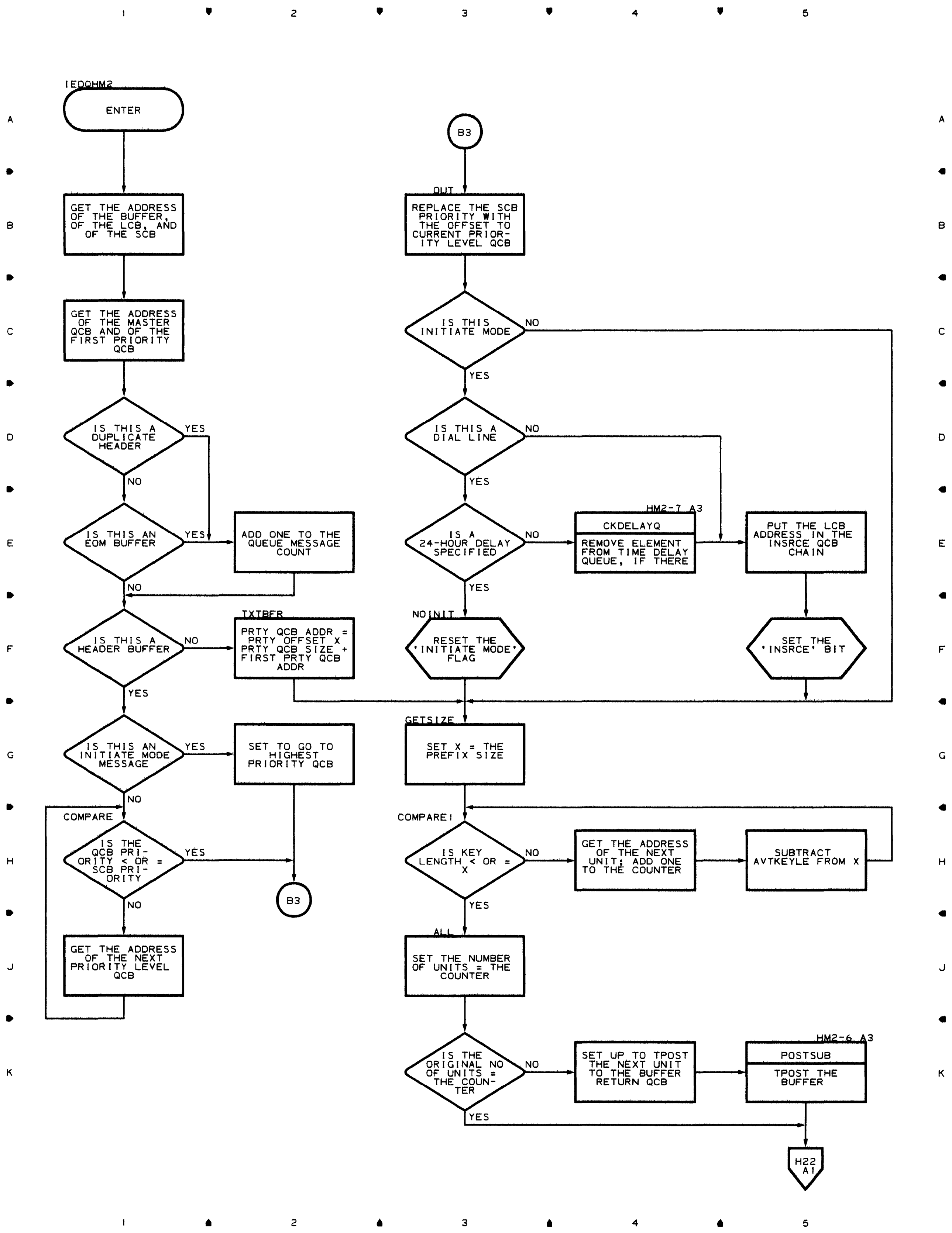

IEDQHM2
ENTER
GET THE ADDRESS OF THE BUFFER, OF THE LCB, AND OF THE SCB
GET THE ADDRESS OF THE MASTER QCB AND OF THE FIRST PRIORITY QCB
IS THIS A DUPLICATE HEADER
YES
NO
IS THIS AN EOM BUFFER
YES
ADD ONE TO THE QUEUE MESSAGE COUNT
NO
IS THIS A HEADER BUFFER
NO
TXTBFR
PRTY QCB ADDR = PRTY OFFSET X PRTY QCB SIZE + FIRST PRTY QCB ADDR
YES
IS THIS AN INITIATE MODE MESSAGE
YES
SET TO GO TO HIGHEST PRIORITY QCB
NO
COMPARE
IS THE QCB PRIORITY < OR = SCB PRIORITY
YES
B3
NO
GET THE ADDRESS OF THE NEXT PRIORITY LEVEL QCB
B3
OUT
REPLACE THE SCB PRIORITY WITH THE OFFSET TO CURRENT PRIORITY LEVEL QCB
IS THIS INITIATE MODE
NO
YES
IS THIS A DIAL LINE
NO
YES
IS A 24-HOUR DELAY SPECIFIED
NO
HM2-7 A3
CKDELAYQ
REMOVE ELEMENT FROM TIME DELAY QUEUE, IF THERE
PUT THE LCB ADDRESS IN THE INSRCE QCB CHAIN
YES
NOINIT
RESET THE 'INITIATE MODE' FLAG
SET THE 'INSRCE' BIT
GETSIZE
SET X = THE PREFIX SIZE
COMPARE1
IS KEY LENGTH < OR = X
NO
GET THE ADDRESS OF THE NEXT UNIT; ADD ONE TO THE COUNTER
SUBTRACT AVTKEYLE FROM X
YES
ALL
SET THE NUMBER OF UNITS = THE COUNTER
IS THE ORIGINAL NO OF UNITS = THE COUNTER
NO
SET UP TO TPOST THE NEXT UNIT TO THE BUFFER RETURN QCB
HM2-6 A3
POSTSUB
TPOST THE BUFFER
YES
H22 A1
1 2 3 4 5
A B C D E F G H J K

Chart HM2-2 (H22) DESTINATION SCHEDULER - DISK-ONLY QUEUING

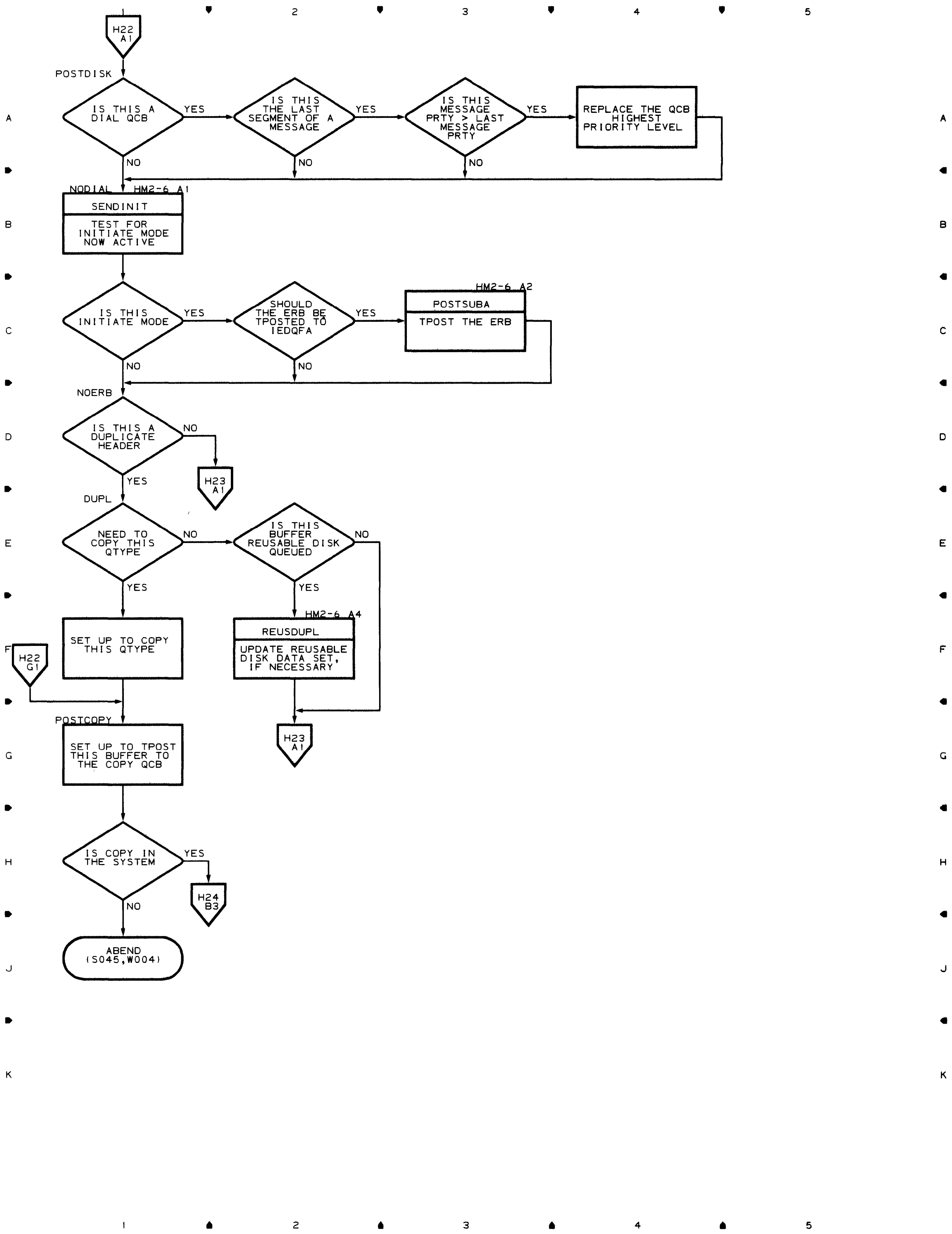

1
2
3
4
5
H22 A1
POSTDISK
IS THIS A DIAL QCB
YES
IS THIS THE LAST SEGMENT OF A MESSAGE
YES
IS THIS MESSAGE PRTY > LAST MESSAGE PRTY
YES
REPLACE THE QCB HIGHEST PRIORITY LEVEL
NO
NO
NO
NODIAL
HM2-6 A1
SENDINIT
TEST FOR INITIATE MODE NOW ACTIVE
IS THIS INITIATE MODE
YES
SHOULD THE ERB BE TPOSTED TO IEDQFA
YES
HM2-6 A2
POSTSUBA
TPOST THE ERB
NO
NO
NOERB
IS THIS A DUPLICATE HEADER
NO
H23 A1
YES
DUPL
NEED TO COPY THIS QTYPE
NO
IS THIS BUFFER REUSABLE DISK QUEUED
NO
YES
YES
HM2-6 A4
REUSDUPL
UPDATE REUSABLE DISK DATA SET, IF NECESSARY
SET UP TO COPY THIS QTYPE
H22 G1
POSTCOPY
SET UP TO TPOST THIS BUFFER TO THE COPY QCB
H23 A1
IS COPY IN THE SYSTEM
YES
H24 B3
NO
ABEND (S045,W004)
A
B
C
D
E
F
G
H
J
K

Chart HM2-3 (H23) DESTINATION SCHEDULER - DISK-ONLY QUEUING

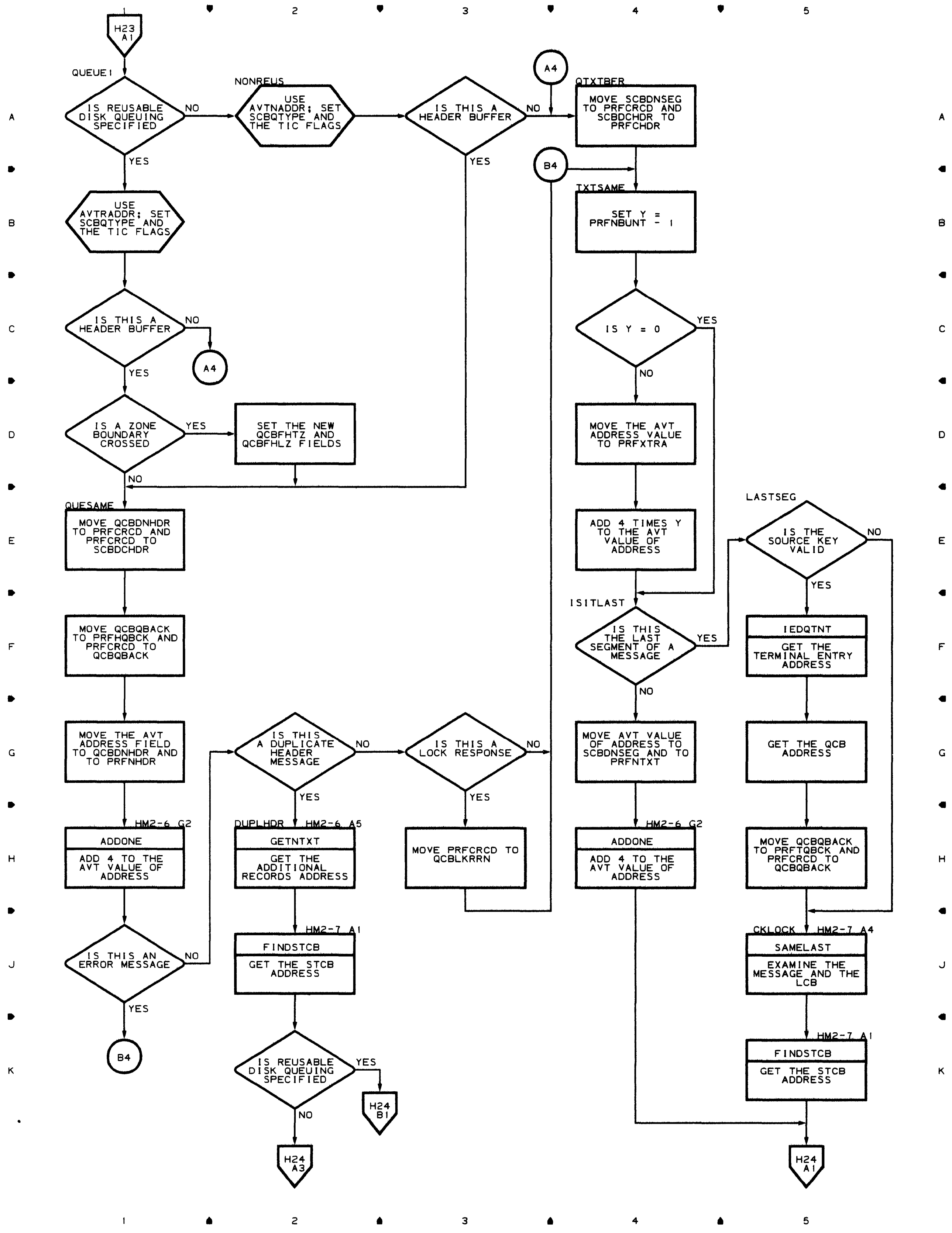
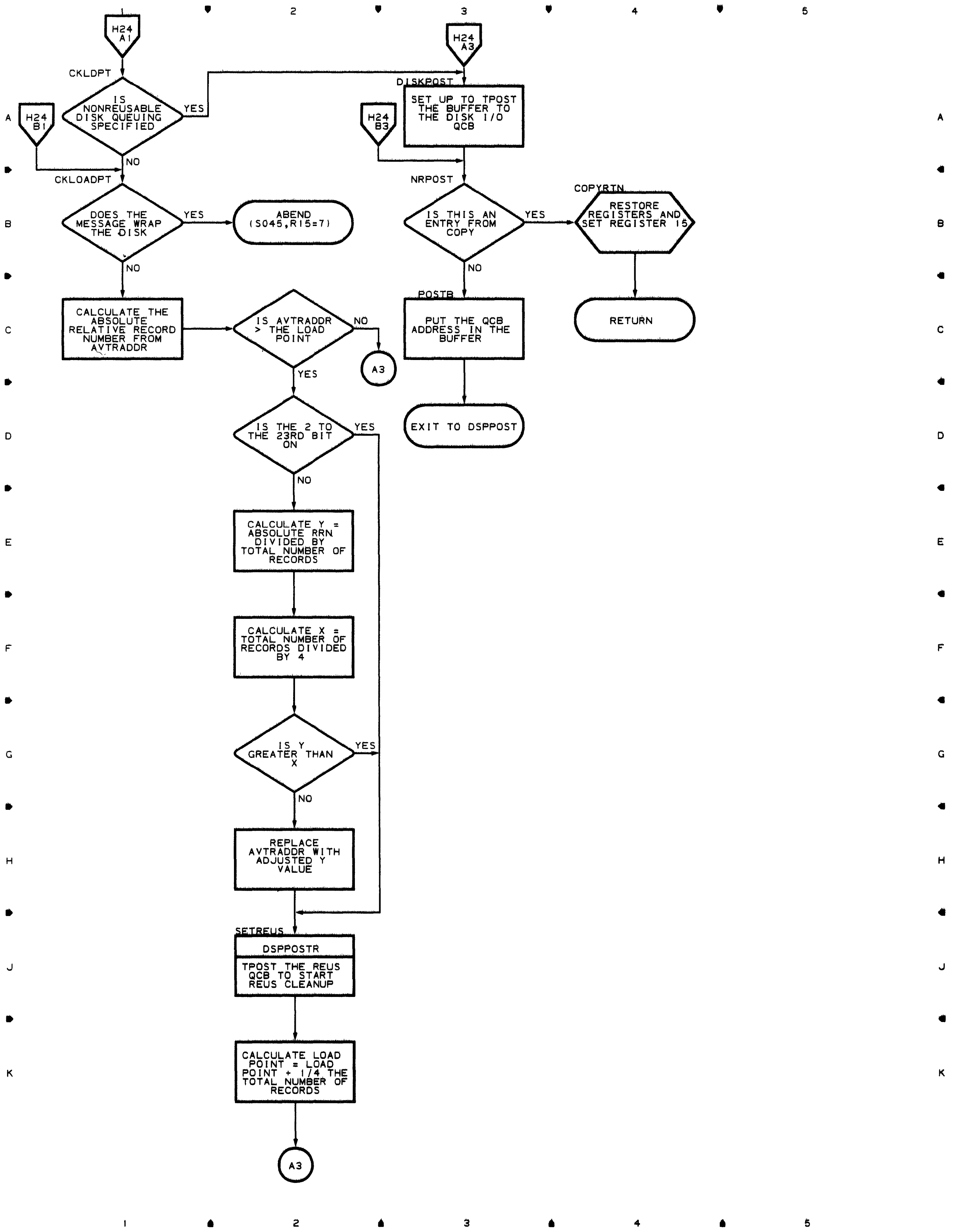

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
H23 A1
QUEUE1
IS REUSABLE DISK QUEUING SPECIFIED
NO
YES
NONREUS
USE AVTNADDR; SET SCBQTYPE AND THE TIC FLAGS
IS THIS A HEADER BUFFER
NO
YES
A4
QTXTBFR
MOVE SCBDNSEG TO PRFCRCD AND SCBDCHDR TO PRFCHDR
B4
TXTSAME
SET Y = PRFNBUNT - 1
USE AVTRADDR; SET SCBQTYPE AND THE TIC FLAGS
IS THIS A HEADER BUFFER
NO
YES
A4
IS Y = 0
YES
NO
IS A ZONE BOUNDARY CROSSED
YES
NO
SET THE NEW QCBFHTZ AND QCBFHLZ FIELDS
MOVE THE AVT ADDRESS VALUE TO PRFXTRA
LASTSEG
QUESAME
MOVE QCBDNHDR TO PRFCRCD AND PRFCRCD TO SCBDCHDR
ADD 4 TIMES Y TO THE AVT VALUE OF ADDRESS
IS THE SOURCE KEY VALID
NO
YES
ISITLAST
MOVE QCBQBACK TO PRFHQBCK AND PRFCRCD TO QCBQBACK
IS THIS THE LAST SEGMENT OF A MESSAGE
YES
NO
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
MOVE THE AVT ADDRESS FIELD TO QCBDNHDR AND TO PRFNHDR
IS THIS A DUPLICATE HEADER MESSAGE
NO
YES
IS THIS A LOCK RESPONSE
NO
YES
MOVE AVT VALUE OF ADDRESS TO SCBDNSEG AND TO PRFNTXT
GET THE QCB ADDRESS
HM2-6 G2
ADDONE
ADD 4 TO THE AVT VALUE OF ADDRESS
DUPLHDR
HM2-6 A5
GETNTXT
GET THE ADDITIONAL RECORDS ADDRESS
MOVE PRFCRCD TO QCBLKRRN
HM2-6 G2
ADDONE
ADD 4 TO THE AVT VALUE OF ADDRESS
MOVE QCBQBACK TO PRFTQBCK AND PRFCRCD TO QCBQBACK
HM2-7 A1
FINDSTCB
GET THE STCB ADDRESS
IS THIS AN ERROR MESSAGE
NO
YES
CKLOCK
HM2-7 A4
SAMELAST
EXAMINE THE MESSAGE AND THE LCB
B4
IS REUSABLE DISK QUEUING SPECIFIED
YES
NO
H24 B1
HM2-7 A1
FINDSTCB
GET THE STCB ADDRESS
H24 A3
H24 A1

Chart HM2-4 (H24) DESTINATION SCHEDULER - DISK-ONLY QUEUING


H24 A1
CKLDPT
IS NONREUSABLE DISK QUEUING SPECIFIED
YES
NO
H24 B1
CKLOADPT
DOES THE MESSAGE WRAP THE DISK
YES
ABEND (S045,R15=7)
NO
CALCULATE THE ABSOLUTE RELATIVE RECORD NUMBER FROM AVTRADDR
IS AVTRADDR > THE LOAD POINT
NO
A3
YES
IS THE 2 TO THE 23RD BIT ON
YES
NO
CALCULATE Y = ABSOLUTE RRN DIVIDED BY TOTAL NUMBER OF RECORDS
CALCULATE X = TOTAL NUMBER OF RECORDS DIVIDED BY 4
IS Y GREATER THAN X
YES
NO
REPLACE AVTRADDR WITH ADJUSTED Y VALUE
SETREUS
DSPPOSTR
TPOST THE REUS QCB TO START REUS CLEANUP
CALCULATE LOAD POINT = LOAD POINT + 1/4 THE TOTAL NUMBER OF RECORDS
A3
H24 A3
DISKPOST
SET UP TO TPOST THE BUFFER TO THE DISK I/O QCB
H24 B3
NRPOST
IS THIS AN ENTRY FROM COPY
YES
COPYRTN
RESTORE REGISTERS AND SET REGISTER 15
RETURN
NO
POSTB
PUT THE QCB ADDRESS IN THE BUFFER
EXIT TO DSPPOST
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

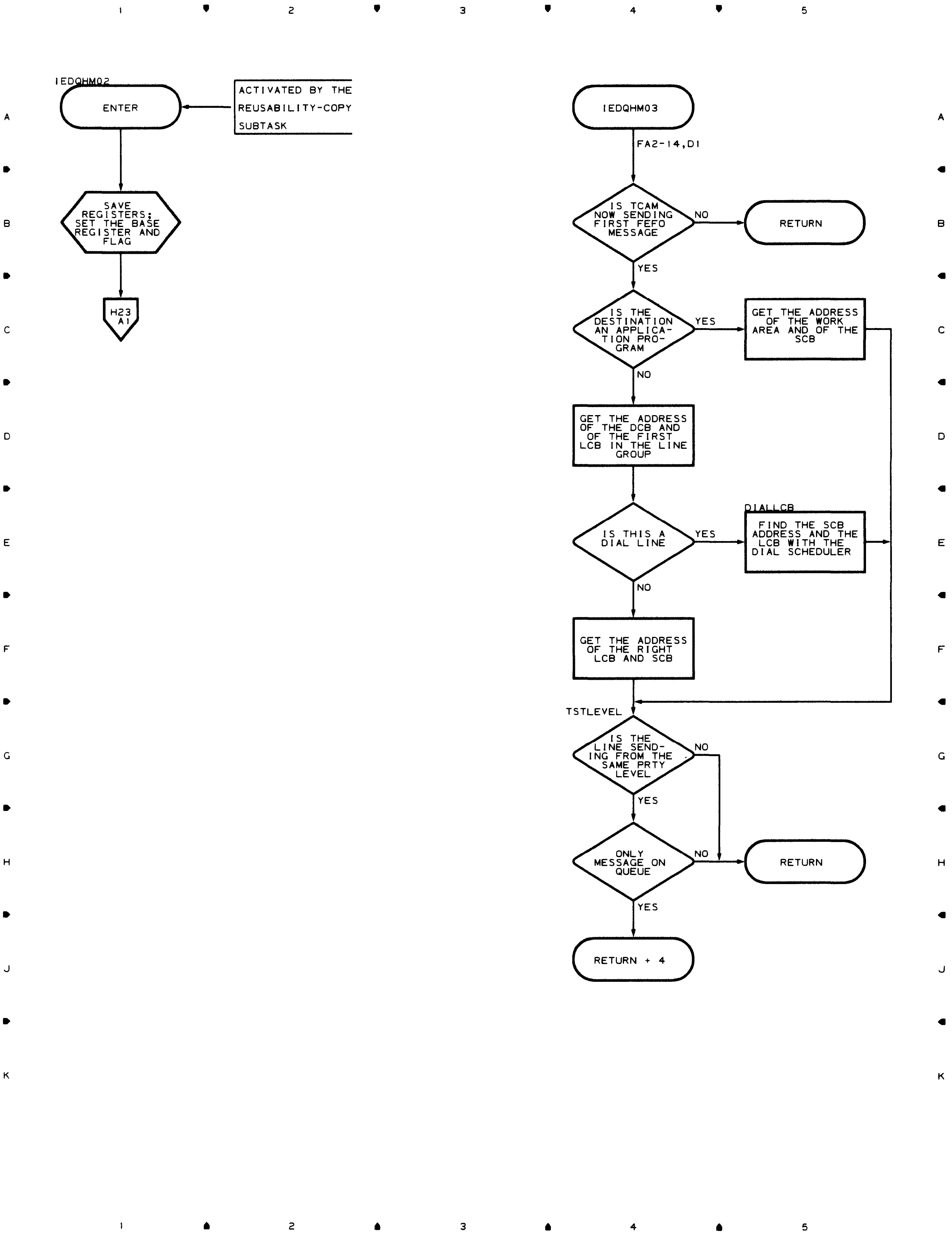

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
IEDQHM02
ENTER
ACTIVATED BY THE REUSABILITY-COPY SUBTASK
SAVE REGISTERS; SET THE BASE REGISTER AND FLAG
H23 A1
IEDQHM03
FA2-14,D1
IS TCAM NOW SENDING FIRST FEFO MESSAGE
NO
RETURN
YES
IS THE DESTINATION AN APPLICA-TION PRO-GRAM
YES
GET THE ADDRESS OF THE WORK AREA AND OF THE SCB
NO
GET THE ADDRESS OF THE DCB AND OF THE FIRST LCB IN THE LINE GROUP
IS THIS A DIAL LINE
YES
DIALLCB
FIND THE SCB ADDRESS AND THE LCB WITH THE DIAL SCHEDULER
NO
GET THE ADDRESS OF THE RIGHT LCB AND SCB
TSTLEVEL
IS THE LINE SEND-ING FROM THE SAME PRTY LEVEL
NO
YES
ONLY MESSAGE ON QUEUE
NO
RETURN
YES
RETURN + 4

Chart HM2-6 (H26) DESTINATION SCHEDULER - DISK-ONLY QUEUING

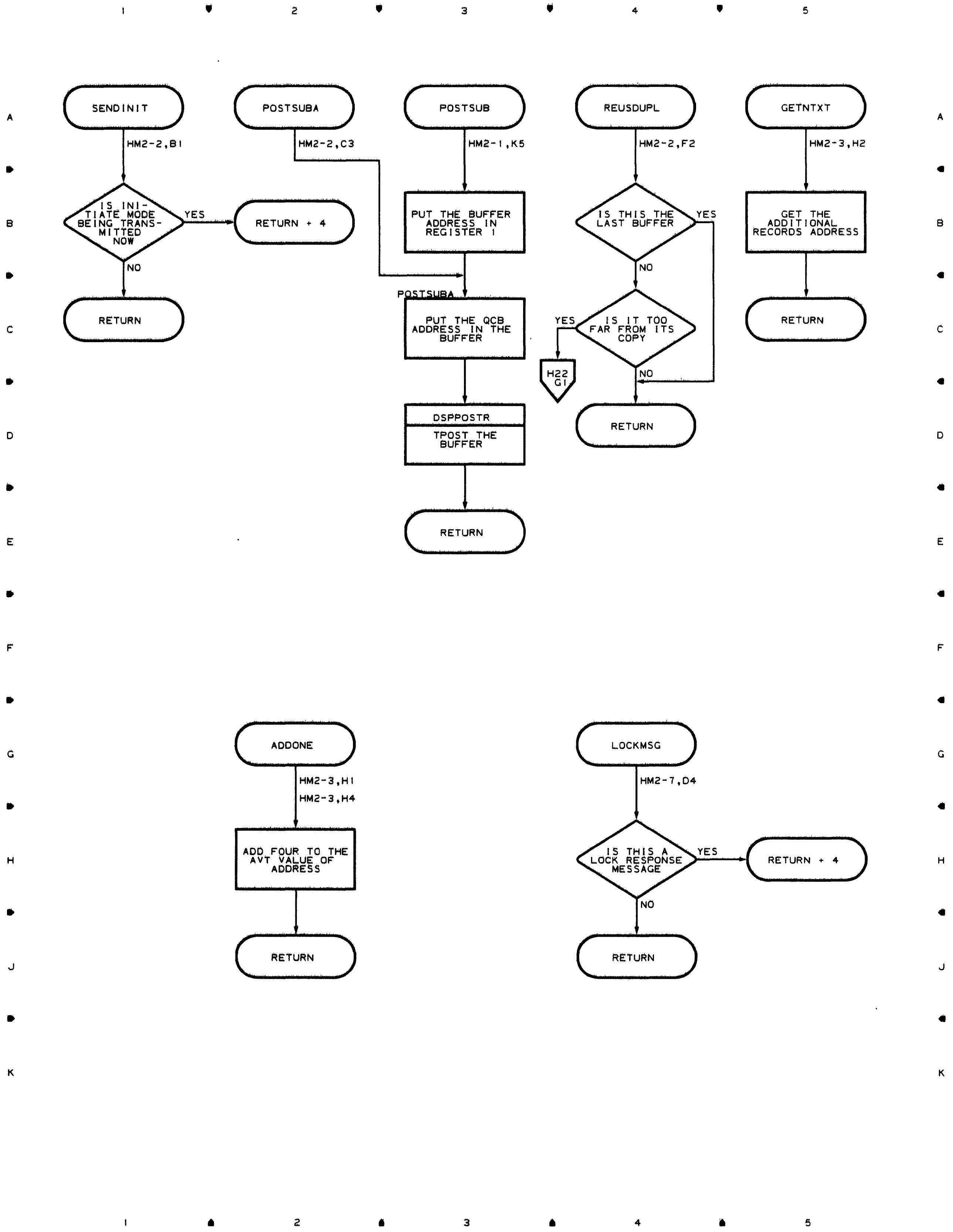
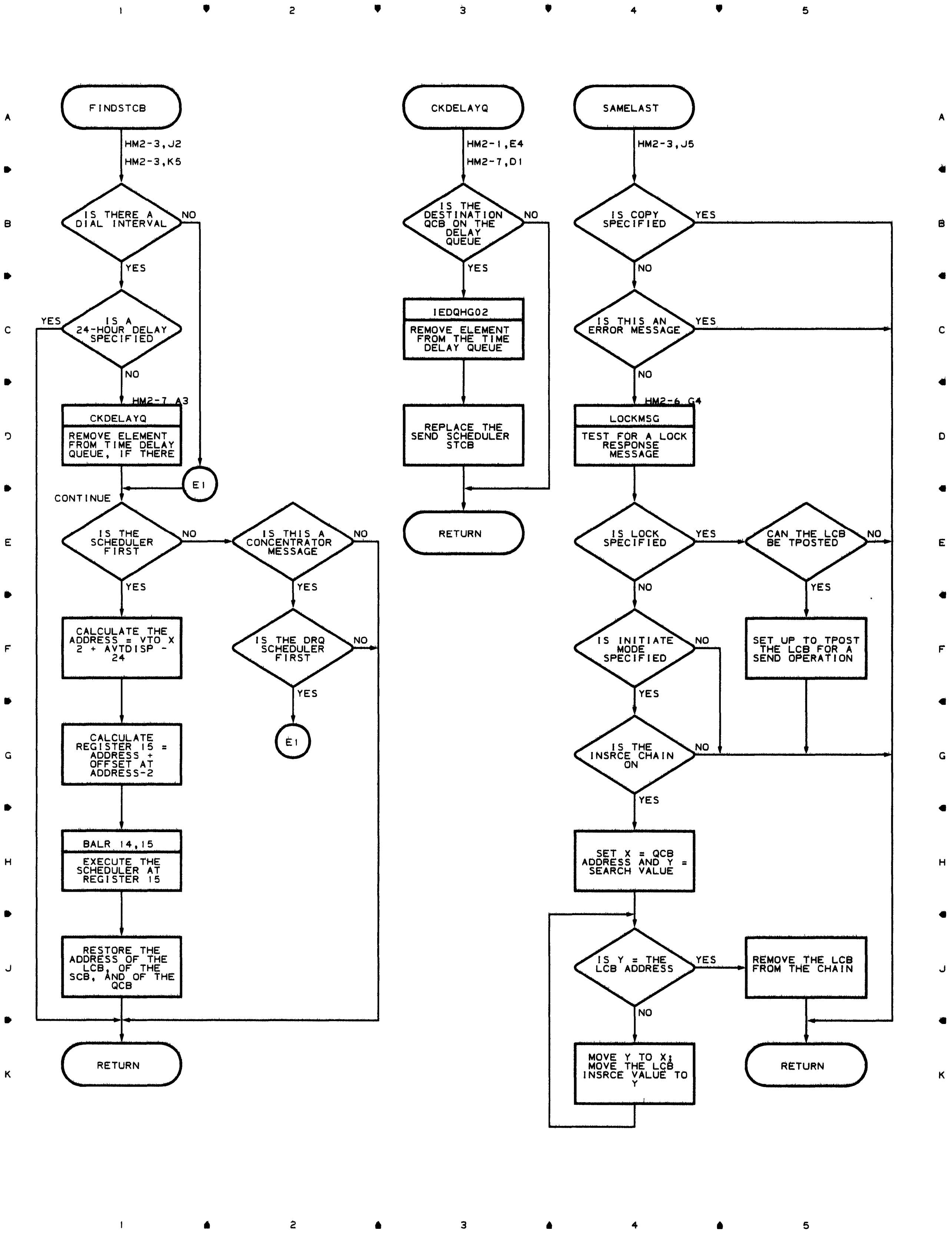


Chart HM2-7 (H27) DESTINATION SCHEDULER - DISK-ONLY QUEUING



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FINDSTCB
HM2-3,J2
HM2-3,K5
IS THERE A DIAL INTERVAL
NO
YES
IS A 24-HOUR DELAY SPECIFIED
YES
NO
HM2-7,A3
CKDELAYQ
REMOVE ELEMENT FROM TIME DELAY QUEUE, IF THERE
E1
CONTINUE
IS THE SCHEDULER FIRST
NO
YES
CALCULATE THE ADDRESS = VTO X 2 + AVTDISP - 24
CALCULATE REGISTER 15 = ADDRESS + OFFSET AT ADDRESS-2
BALR 14,15
EXECUTE THE SCHEDULER AT REGISTER 15
RESTORE THE ADDRESS OF THE LCB, OF THE SCB, AND OF THE QCB
RETURN
IS THIS A CONCENTRATOR MESSAGE
NO
YES
IS THE DRQ SCHEDULER FIRST
NO
YES
E1
CKDELAYQ
HM2-1,E4
HM2-7,D1
IS THE DESTINATION QCB ON THE DELAY QUEUE
NO
YES
IEDQHG02
REMOVE ELEMENT FROM THE TIME DELAY QUEUE
REPLACE THE SEND SCHEDULER STCB
RETURN
SAMELAST
HM2-3,J5
IS COPY SPECIFIED
YES
NO
IS THIS AN ERROR MESSAGE
YES
NO
HM2-6,G4
LOCKMSG
TEST FOR A LOCK RESPONSE MESSAGE
IS LOCK SPECIFIED
YES
NO
CAN THE LCB BE TPOSTED
NO
YES
SET UP TO TPOST THE LCB FOR A SEND OPERATION
IS INITIATE MODE SPECIFIED
NO
YES
IS THE INSRCE CHAIN ON
NO
YES
SET X = QCB ADDRESS AND Y = SEARCH VALUE
IS Y = THE LCB ADDRESS
YES
NO
REMOVE THE LCB FROM THE CHAIN
MOVE Y TO X; MOVE THE LCB INSRCE VALUE TO Y
RETURN

Chart KA-1 (KA1) ACTIVATE-I/O GENERATOR SUBTASK

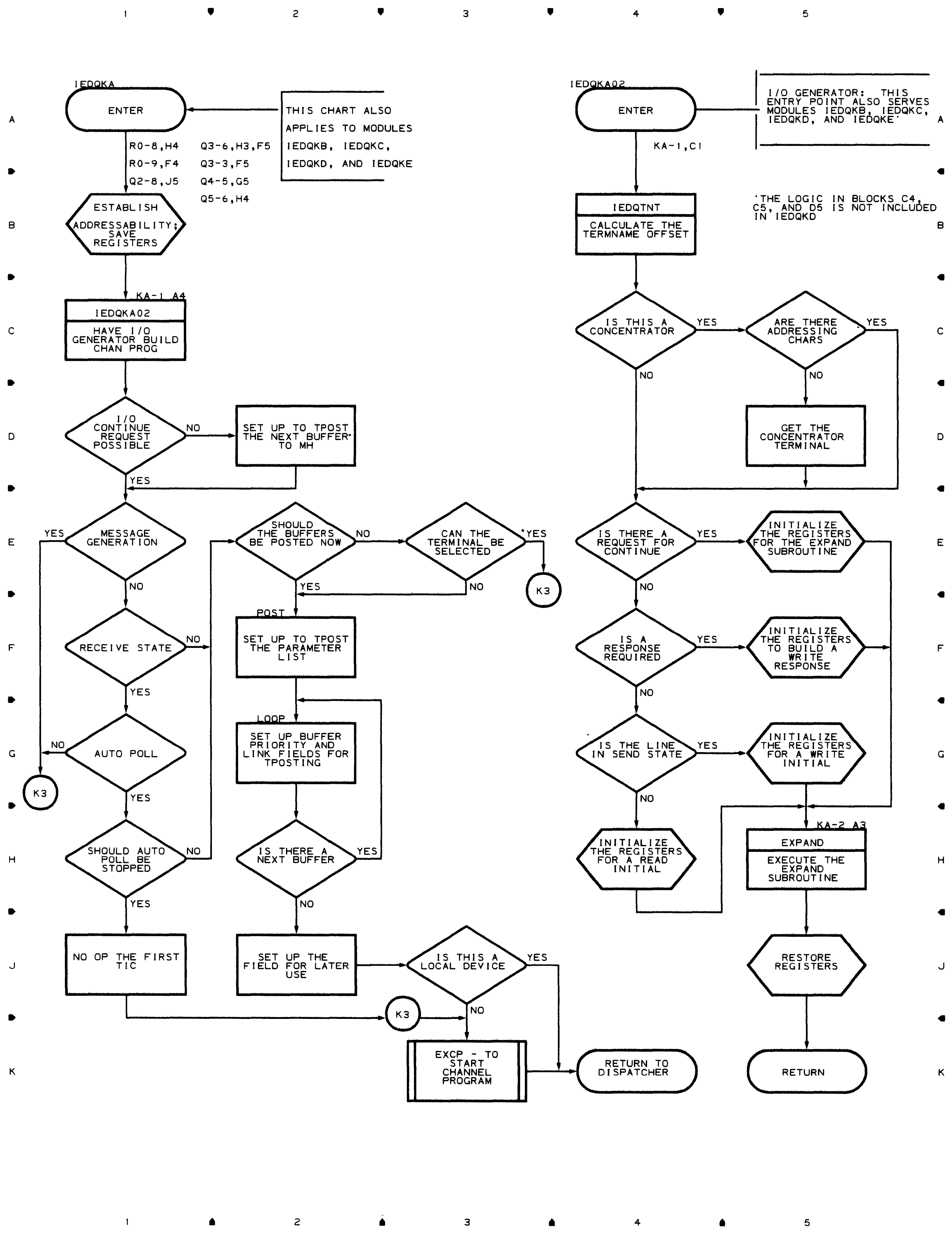

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
IEDQKA
ENTER
THIS CHART ALSO APPLIES TO MODULES IEDQKB, IEDQKC, IEDQKD, AND IEDQKE
R0-8,H4
R0-9,F4
Q2-8,J5
Q3-6,H3,F5
Q3-3,F5
Q4-5,G5
Q5-6,H4
ESTABLISH ADDRESSABILITY; SAVE REGISTERS
KA-1 A4
IEDQKA02
HAVE I/O GENERATOR BUILD CHAN PROG
I/O CONTINUE REQUEST POSSIBLE
NO
SET UP TO TPOST THE NEXT BUFFER TO MH
YES
MESSAGE GENERATION
YES
NO
RECEIVE STATE
NO
YES
AUTO POLL
NO
YES
K3
SHOULD AUTO POLL BE STOPPED
NO
YES
NO OP THE FIRST TIC
SHOULD THE BUFFERS BE POSTED NOW
NO
YES
CAN THE TERMINAL BE SELECTED
*YES
NO
K3
POST
SET UP TO TPOST THE PARAMETER LIST
LOOP
SET UP BUFFER PRIORITY AND LINK FIELDS FOR TPOSTING
IS THERE A NEXT BUFFER
YES
NO
SET UP THE FIELD FOR LATER USE
IS THIS A LOCAL DEVICE
YES
NO
K3
EXCP - TO START CHANNEL PROGRAM
RETURN TO DISPATCHER
IEDQKA02
ENTER
I/O GENERATOR: THIS ENTRY POINT ALSO SERVES MODULES IEDQKB, IEDQKC, IEDQKD, AND IEDQKE*
KA-1,C1
IEDQTNT
CALCULATE THE TERMNAME OFFSET
*THE LOGIC IN BLOCKS C4, C5, AND D5 IS NOT INCLUDED IN IEDQKD
IS THIS A CONCENTRATOR
YES
NO
ARE THERE ADDRESSING CHARS
YES
NO
GET THE CONCENTRATOR TERMINAL
IS THERE A REQUEST FOR CONTINUE
YES
NO
INITIALIZE THE REGISTERS FOR THE EXPAND SUBROUTINE
IS A RESPONSE REQUIRED
YES
NO
INITIALIZE THE REGISTERS TO BUILD A WRITE RESPONSE
IS THE LINE IN SEND STATE
YES
NO
INITIALIZE THE REGISTERS FOR A WRITE INITIAL
INITIALIZE THE REGISTERS FOR A READ INITIAL
KA-2 A3
EXPAND
EXECUTE THE EXPAND SUBROUTINE
RESTORE REGISTERS
RETURN

Chart KA-2 (KA2) ACTIVATE-I/O GENERATOR SUBTASK

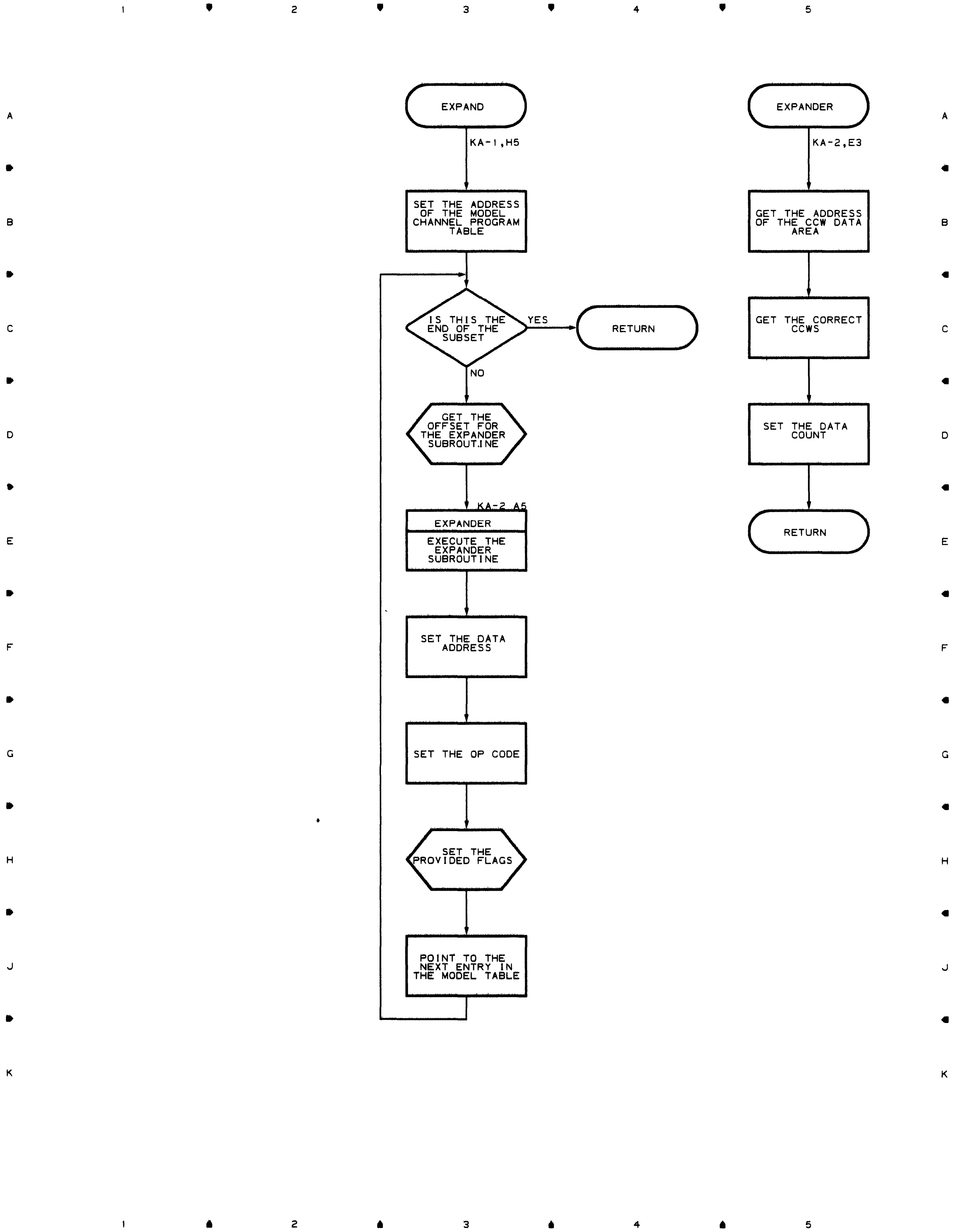

EXPAND
KA-1,H5
SET THE ADDRESS OF THE MODEL CHANNEL PROGRAM TABLE
IS THIS THE END OF THE SUBSET
YES
RETURN
NO
GET THE OFFSET FOR THE EXPANDER SUBROUT.INE
KA-2 A5
EXPANDER
EXECUTE THE EXPANDER SUBROUTINE
SET THE DATA ADDRESS
SET THE OP CODE
SET THE PROVIDED FLAGS
POINT TO THE NEXT ENTRY IN THE MODEL TABLE
EXPANDER
KA-2,E3
GET THE ADDRESS OF THE CCW DATA AREA
GET THE CORRECT CCWS
SET THE DATA COUNT
RETURN

Chart Q2-1 (Q21) LINE END APPENDAGE FOR BSC LINES

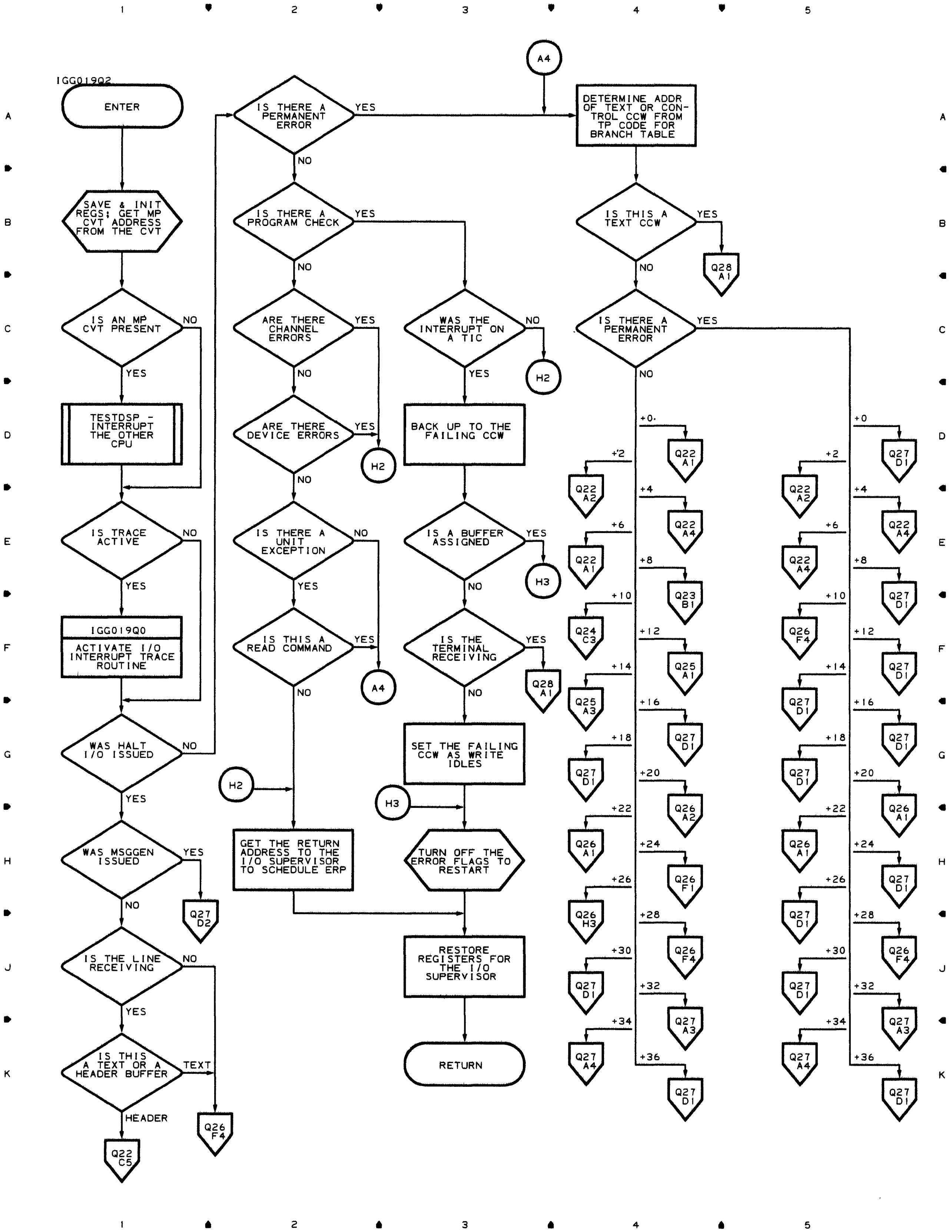
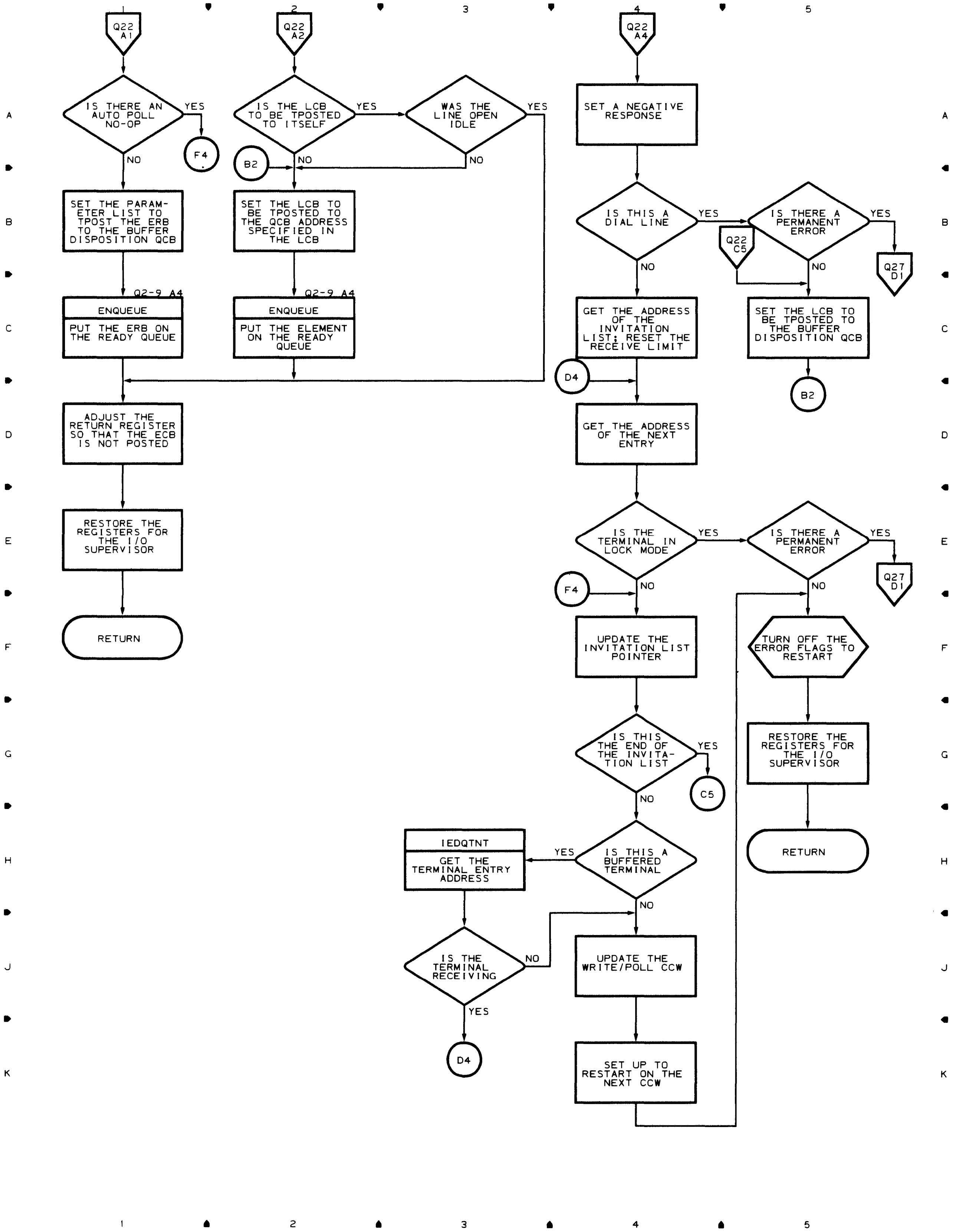


Chart Q2-2 (Q22) LINE END APPENDAGE FOR BSC LINES



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
Q22 A1
IS THERE AN AUTO POLL NO-OP
YES
F4
NO
SET THE PARAMETER LIST TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q2-9 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q22 A2
IS THE LCB TO BE TPOSTED TO ITSELF
YES
B2
NO
SET THE LCB TO BE TPOSTED TO THE QCB ADDRESS SPECIFIED IN THE LCB
Q2-9 A4
ENQUEUE
PUT THE ELEMENT ON THE READY QUEUE
WAS THE LINE OPEN IDLE
YES
NO
Q22 A4
SET A NEGATIVE RESPONSE
IS THIS A DIAL LINE
YES
NO
Q22 C5
IS THERE A PERMANENT ERROR
YES
Q27 D1
NO
SET THE LCB TO BE TPOSTED TO THE BUFFER DISPOSITION QCB
B2
GET THE ADDRESS OF THE INVITATION LIST; RESET THE RECEIVE LIMIT
D4
GET THE ADDRESS OF THE NEXT ENTRY
IS THE TERMINAL IN LOCK MODE
YES
IS THERE A PERMANENT ERROR
YES
Q27 D1
F4
NO
NO
UPDATE THE INVITATION LIST POINTER
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
IS THIS THE END OF THE INVITATION LIST
YES
C5
NO
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
YES
IS THIS A BUFFERED TERMINAL
NO
IS THE TERMINAL RECEIVING
NO
YES
D4
UPDATE THE WRITE/POLL CCW
SET UP TO RESTART ON THE NEXT CCW

Chart Q2-3 (Q23) LINE END APPENDAGE FOR BSC LINES

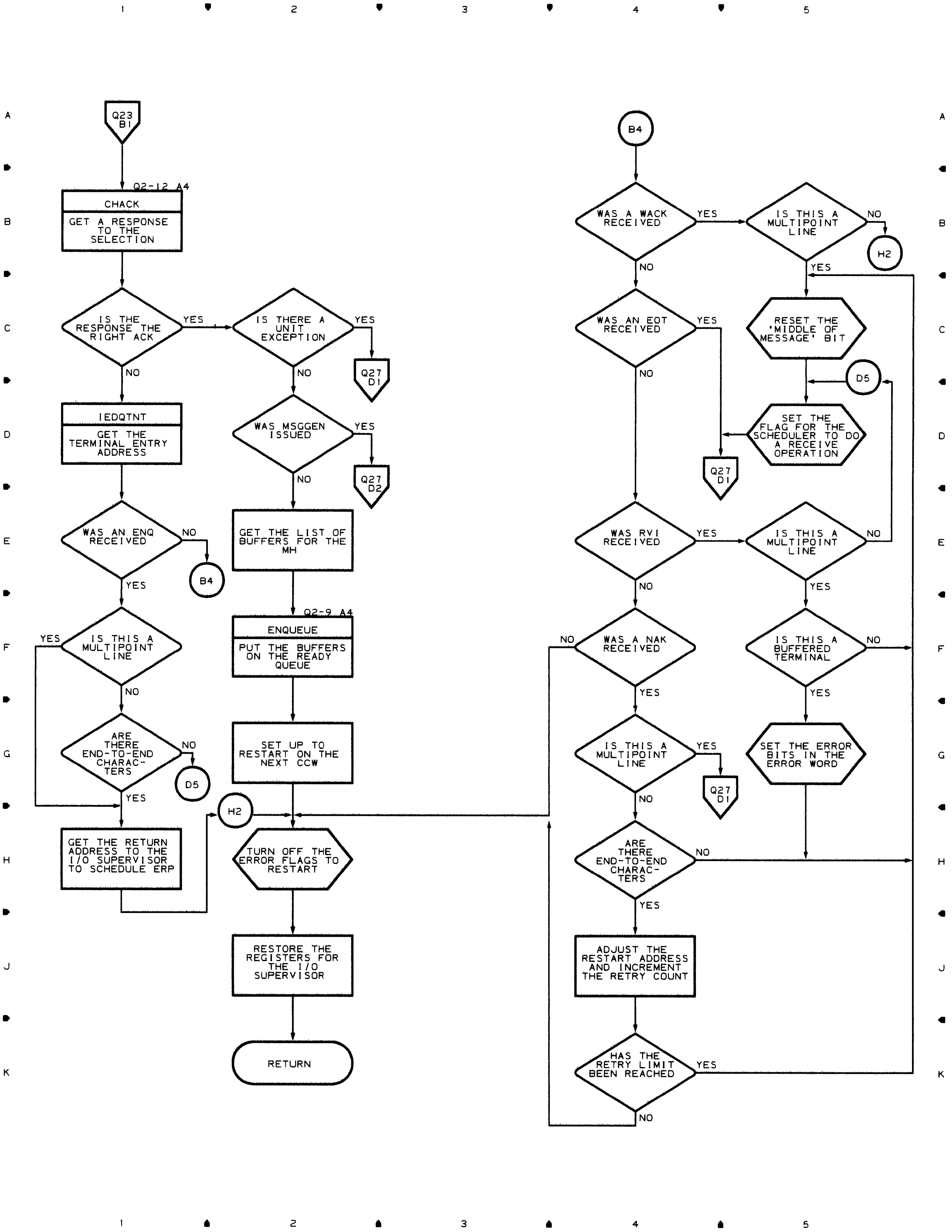

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
Q23 B1
Q2-12 A4
CHACK
GET A RESPONSE TO THE SELECTION
IS THE RESPONSE THE RIGHT ACK
YES
NO
IS THERE A UNIT EXCEPTION
YES
Q27 D1
NO
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
WAS MSGGEN ISSUED
YES
Q27 D2
NO
WAS AN ENQ RECEIVED
NO
B4
YES
GET THE LIST OF BUFFERS FOR THE MH
Q2-9 A4
ENQUEUE
PUT THE BUFFERS ON THE READY QUEUE
IS THIS A MULTIPOINT LINE
YES
NO
ARE THERE END-TO-END CHARAC-TERS
NO
D5
YES
SET UP TO RESTART ON THE NEXT CCW
H2
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
B4
WAS A WACK RECEIVED
YES
IS THIS A MULTIPOINT LINE
NO
H2
YES
NO
WAS AN EOT RECEIVED
YES
RESET THE 'MIDDLE OF MESSAGE' BIT
D5
SET THE FLAG FOR THE SCHEDULER TO DO A RECEIVE OPERATION
Q27 D1
NO
WAS RVI RECEIVED
YES
IS THIS A MULTIPOINT LINE
NO
NO
YES
WAS A NAK RECEIVED
NO
IS THIS A BUFFERED TERMINAL
NO
YES
YES
IS THIS A MULTIPOINT LINE
YES
Q27 D1
SET THE ERROR BITS IN THE ERROR WORD
NO
ARE THERE END-TO-END CHARAC-TERS
NO
YES
ADJUST THE RESTART ADDRESS AND INCREMENT THE RETRY COUNT
HAS THE RETRY LIMIT BEEN REACHED
YES
NO

Chart Q2-4 (Q24) LINE END APPENDAGE FOR BSC LINES

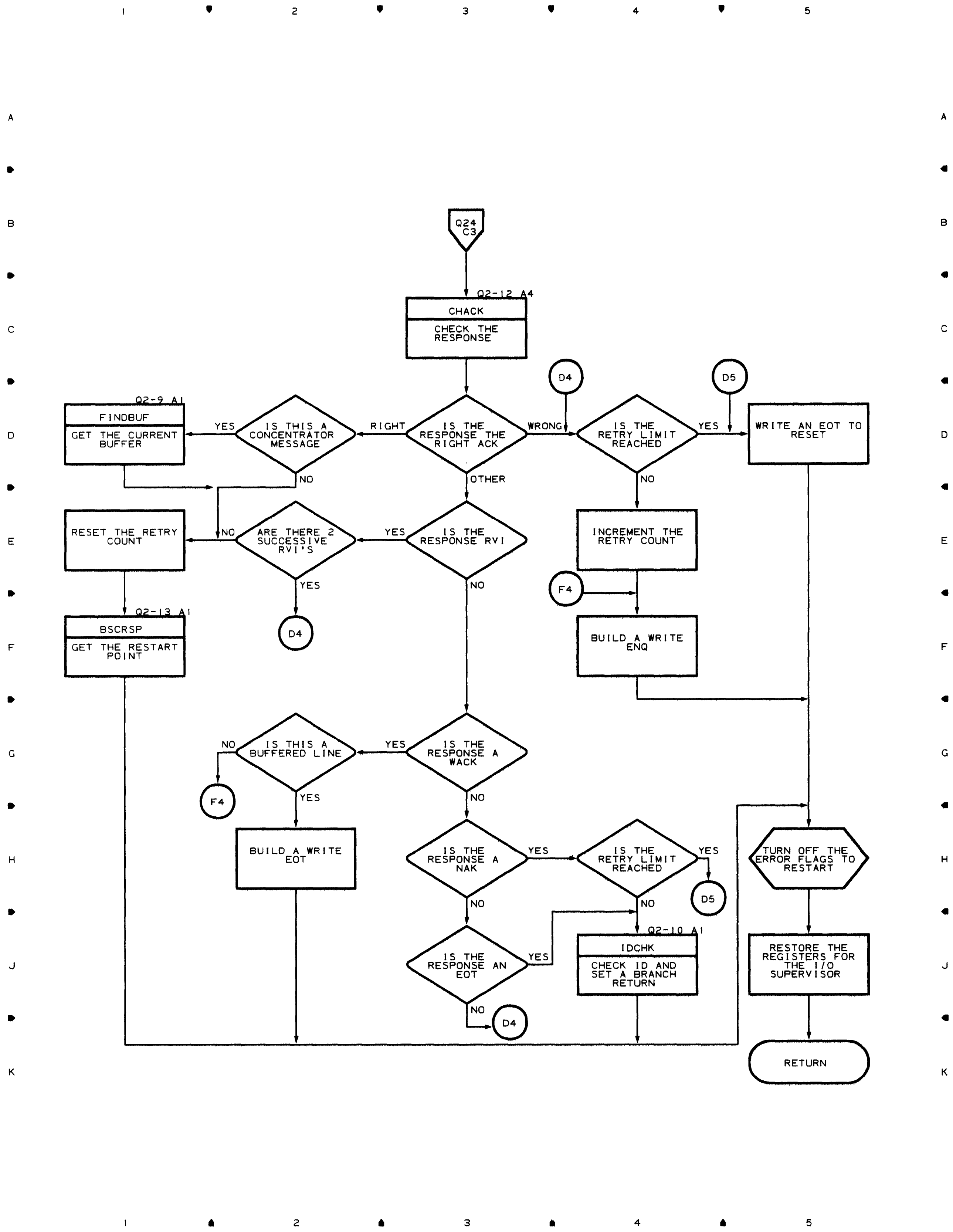


Chart Q2-5 (Q25) LINE END APPENDAGE FOR BSC LINES

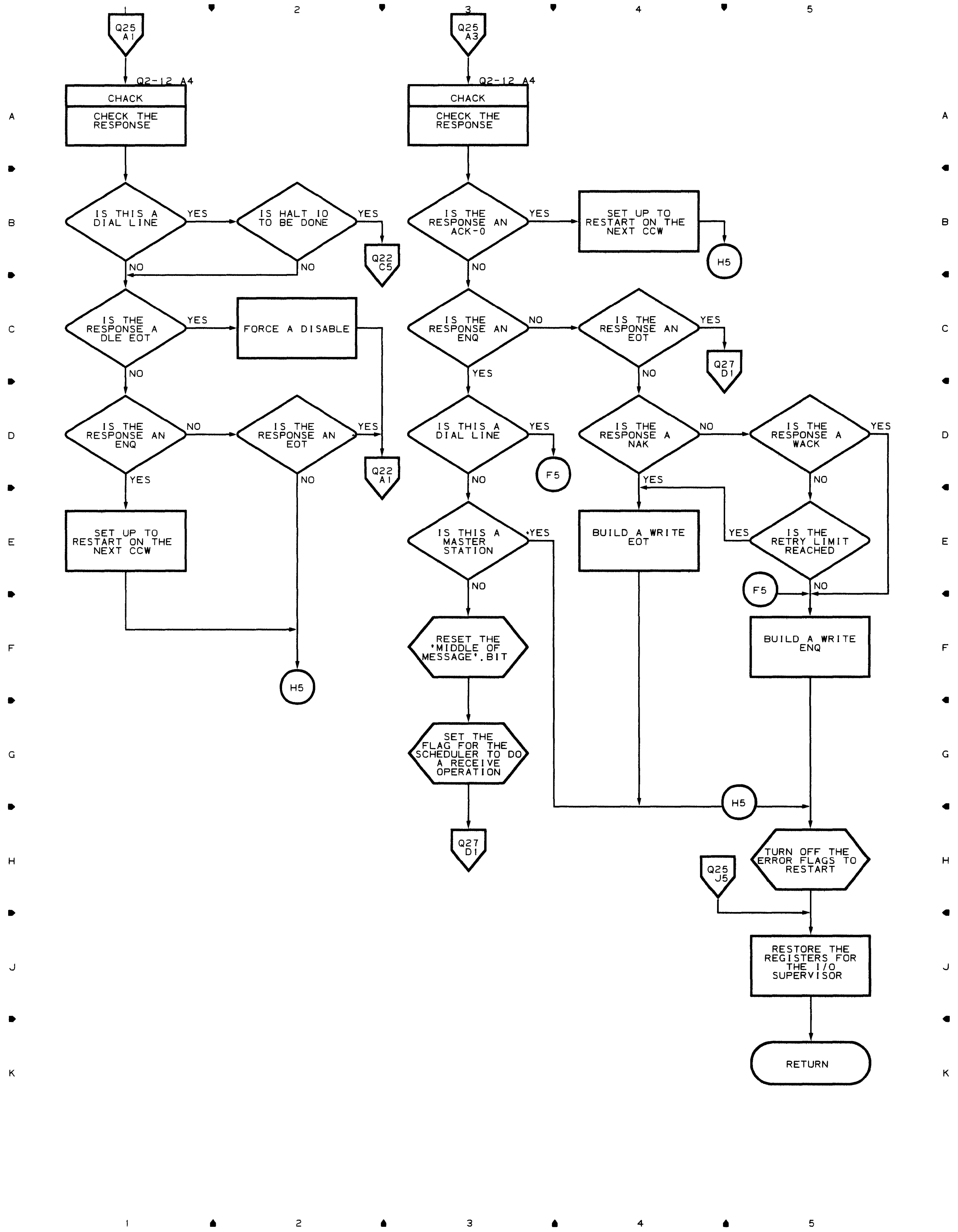
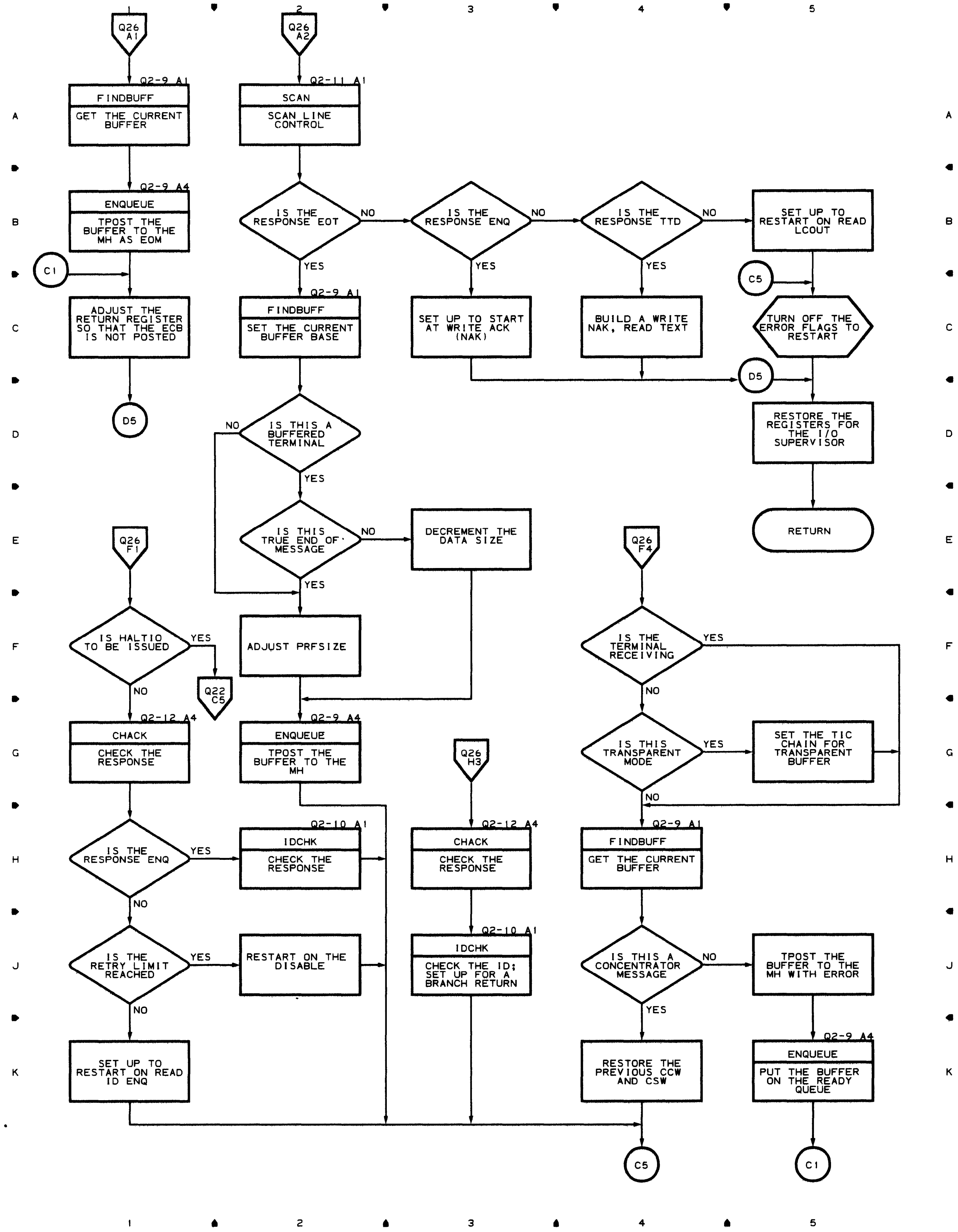


Chart Q2-6 (Q26) LINE END APPENDAGE FOR BSC LINES



Q26 A1
Q2-9 A1
FINDBUFF
GET THE CURRENT BUFFER
Q2-9 A4
ENQUEUE
TPOST THE BUFFER TO THE MH AS EOM
C1
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
D5
Q26 A2
Q2-11 A1
SCAN
SCAN LINE CONTROL
IS THE RESPONSE EOT
NO
YES
Q2-9 A1
FINDBUFF
SET THE CURRENT BUFFER BASE
IS THIS A BUFFERED TERMINAL
NO
YES
IS THIS TRUE END OF MESSAGE
NO
YES
DECREMENT THE DATA SIZE
ADJUST PRFSIZE
Q2-9 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
IS THE RESPONSE ENQ
NO
YES
SET UP TO START AT WRITE ACK (NAK)
IS THE RESPONSE TTD
NO
YES
BUILD A WRITE NAK, READ TEXT
SET UP TO RESTART ON READ LCOUT
C5
TURN OFF THE ERROR FLAGS TO RESTART
D5
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q26 F1
IS HALTIO TO BE ISSUED
YES
Q22 C5
NO
Q2-12 A4
CHACK
CHECK THE RESPONSE
IS THE RESPONSE ENQ
YES
NO
Q2-10 A1
IDCHK
CHECK THE RESPONSE
IS THE RETRY LIMIT REACHED
YES
NO
RESTART ON THE DISABLE
SET UP TO RESTART ON READ ID ENQ
Q26 H3
Q2-12 A4
CHACK
CHECK THE RESPONSE
Q2-10 A1
IDCHK
CHECK THE ID; SET UP FOR A BRANCH RETURN
Q26 F4
IS THE TERMINAL RECEIVING
YES
NO
IS THIS TRANSPARENT MODE
YES
SET THE TIC CHAIN FOR TRANSPARENT BUFFER
NO
Q2-9 A1
FINDBUFF
GET THE CURRENT BUFFER
IS THIS A CONCENTRATOR MESSAGE
NO
YES
TPOST THE BUFFER TO THE MH WITH ERROR
Q2-9 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
RESTORE THE PREVIOUS CCW AND CSW
C5
C1
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q2-7 (Q27) LINE END APPENDAGE FOR BSC LINES

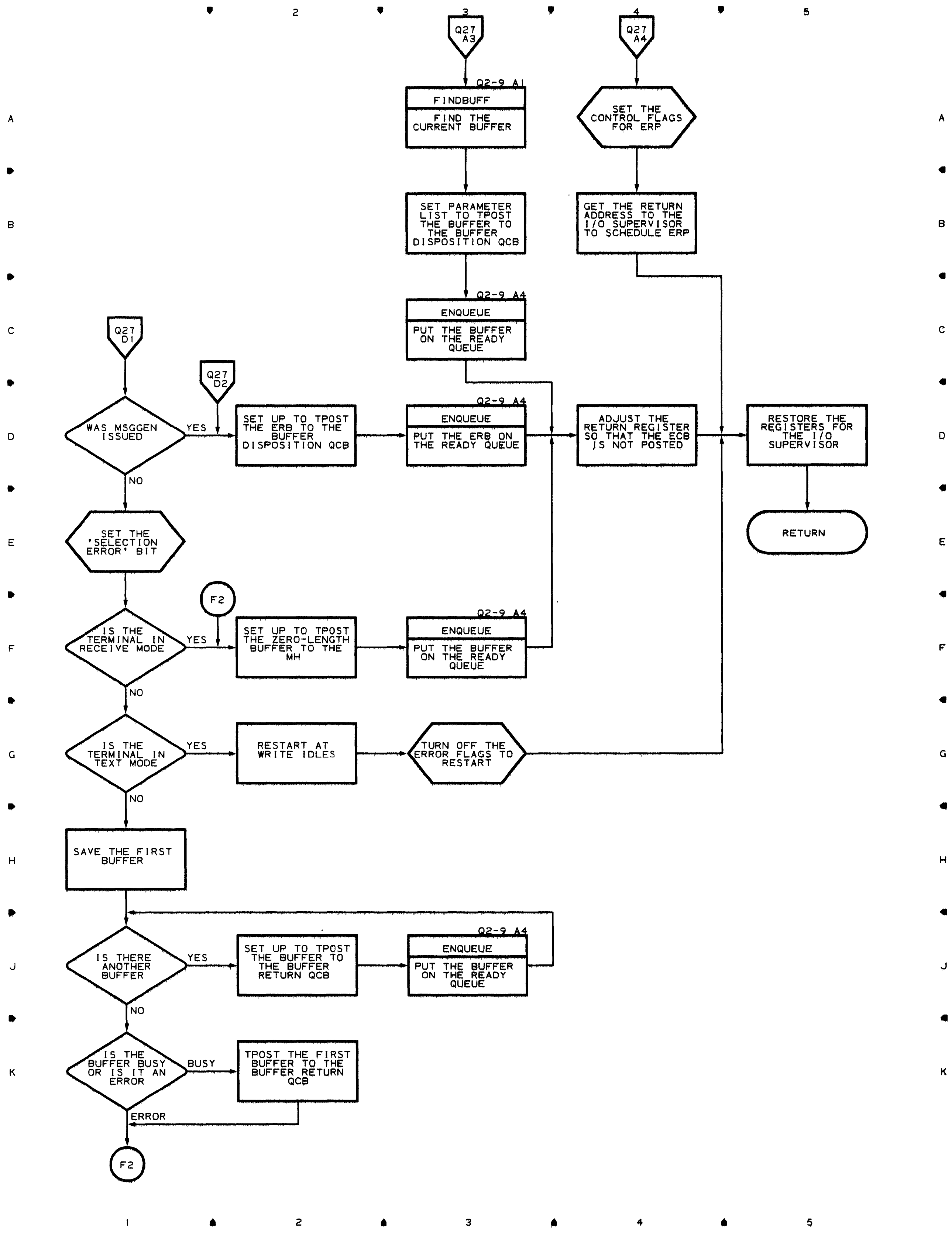
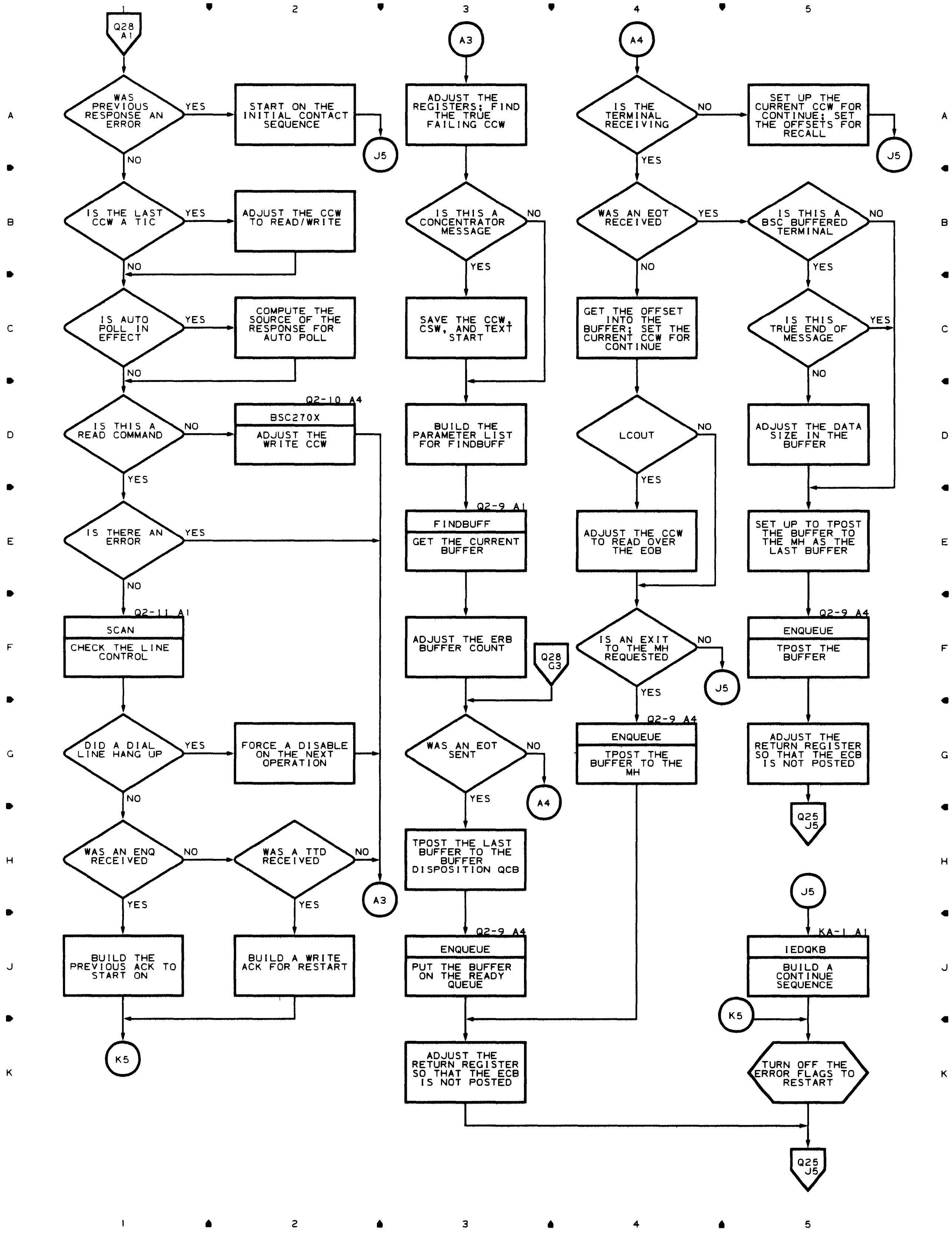

Q27 A3
Q2-9 A1
FINDBUFF
FIND THE CURRENT BUFFER
SET PARAMETER LIST TO TPOST THE BUFFER TO THE BUFFER DISPOSITION QCB
Q2-9 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
Q27 A4
SET THE CONTROL FLAGS FOR ERP
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
Q27 D1
Q27 D2
WAS MSGGEN ISSUED
YES
NO
SET UP TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q2-9 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
SET THE 'SELECTION ERROR' BIT
F2
IS THE TERMINAL IN RECEIVE MODE
YES
NO
SET UP TO TPOST THE ZERO-LENGTH BUFFER TO THE MH
Q2-9 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
IS THE TERMINAL IN TEXT MODE
YES
NO
RESTART AT WRITE IDLES
TURN OFF THE ERROR FLAGS TO RESTART
SAVE THE FIRST BUFFER
IS THERE ANOTHER BUFFER
YES
NO
SET UP TO TPOST THE BUFFER TO THE BUFFER RETURN QCB
Q2-9 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
IS THE BUFFER BUSY OR IS IT AN ERROR
BUSY
ERROR
TPOST THE FIRST BUFFER TO THE BUFFER RETURN QCB
F2
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q2-8 (Q28) LINE END APPENDAGE FOR BSC LINES



Q28 A1
WAS PREVIOUS RESPONSE AN ERROR
YES
START ON THE INITIAL CONTACT SEQUENCE
J5
NO
IS THE LAST CCW A TIC
YES
ADJUST THE CCW TO READ/WRITE
NO
IS AUTO POLL IN EFFECT
YES
COMPUTE THE SOURCE OF THE RESPONSE FOR AUTO POLL
NO
IS THIS A READ COMMAND
NO
Q2-10 A4
BSC270X
ADJUST THE WRITE CCW
YES
IS THERE AN ERROR
YES
NO
Q2-11 A1
SCAN
CHECK THE LINE CONTROL
DID A DIAL LINE HANG UP
YES
FORCE A DISABLE ON THE NEXT OPERATION
NO
WAS AN ENQ RECEIVED
NO
WAS A TTD RECEIVED
NO
A3
YES
YES
BUILD THE PREVIOUS ACK TO START ON
BUILD A WRITE ACK FOR RESTART
K5
A3
ADJUST THE REGISTERS; FIND THE TRUE FAILING CCW
IS THIS A CONCENTRATOR MESSAGE
NO
YES
SAVE THE CCW, CSW, AND TEXT START
BUILD THE PARAMETER LIST FOR FINDBUFF
Q2-9 A1
FINDBUFF
GET THE CURRENT BUFFER
ADJUST THE ERB BUFFER COUNT
Q28 G3
WAS AN EOT SENT
NO
A4
YES
TPOST THE LAST BUFFER TO THE BUFFER DISPOSITION QCB
Q2-9 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
A4
IS THE TERMINAL RECEIVING
NO
SET UP THE CURRENT CCW FOR CONTINUE; SET THE OFFSETS FOR RECALL
J5
YES
WAS AN EOT RECEIVED
YES
IS THIS A BSC BUFFERED TERMINAL
NO
NO
GET THE OFFSET INTO THE BUFFER; SET THE CURRENT CCW FOR CONTINUE
YES
IS THIS TRUE END OF MESSAGE
YES
NO
LCOUT
NO
ADJUST THE DATA SIZE IN THE BUFFER
YES
ADJUST THE CCW TO READ OVER THE EOB
SET UP TO TPOST THE BUFFER TO THE MH AS THE LAST BUFFER
IS AN EXIT TO THE MH REQUESTED
NO
J5
YES
Q2-9 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
Q2-9 A4
ENQUEUE
TPOST THE BUFFER
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
Q25 J5
J5
KA-1 A1
IEDQKB
BUILD A CONTINUE SEQUENCE
K5
TURN OFF THE ERROR FLAGS TO RESTART
Q25 J5
1 2 3 4 5
A B C D E F G H J K

Chart Q2-9 (Q29) LINE END APPENDAGE FOR BSC LINES

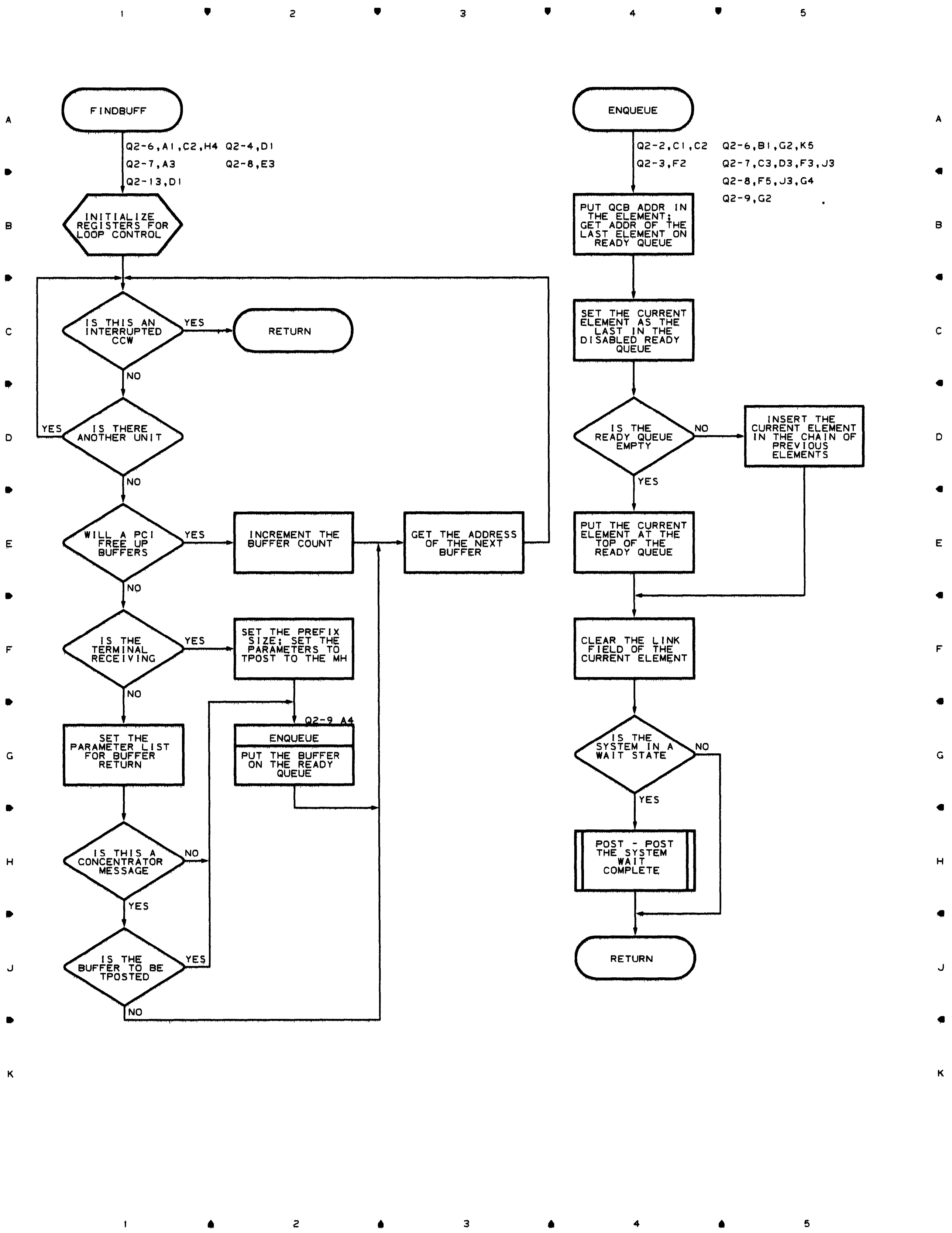

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FINDBUFF
Q2-6,A1,C2,H4 Q2-4,D1
Q2-7,A3 Q2-8,E3
Q2-13,D1
INITIALIZE REGISTERS FOR LOOP CONTROL
IS THIS AN INTERRUPTED CCW
YES
RETURN
NO
IS THERE ANOTHER UNIT
YES
NO
WILL A PCI FREE UP BUFFERS
YES
INCREMENT THE BUFFER COUNT
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE TERMINAL RECEIVING
YES
SET THE PREFIX SIZE; SET THE PARAMETERS TO TPOST TO THE MH
NO
SET THE PARAMETER LIST FOR BUFFER RETURN
Q2-9 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
IS THIS A CONCENTRATOR MESSAGE
NO
YES
IS THE BUFFER TO BE TPOSTED
YES
NO
ENQUEUE
Q2-2,C1,C2 Q2-6,B1,G2,K5
Q2-3,F2 Q2-7,C3,D3,F3,J3
Q2-8,F5,J3,G4
Q2-9,G2
PUT QCB ADDR IN THE ELEMENT; GET ADDR OF THE LAST ELEMENT ON READY QUEUE
SET THE CURRENT ELEMENT AS THE LAST IN THE DISABLED READY QUEUE
IS THE READY QUEUE EMPTY
NO
INSERT THE CURRENT ELEMENT IN THE CHAIN OF PREVIOUS ELEMENTS
YES
PUT THE CURRENT ELEMENT AT THE TOP OF THE READY QUEUE
CLEAR THE LINK FIELD OF THE CURRENT ELEMENT
IS THE SYSTEM IN A WAIT STATE
NO
YES
POST - POST THE SYSTEM WAIT COMPLETE
RETURN

Chart Q2-10 (Q2A) LINE END APPENDAGE FOR BSC LINES

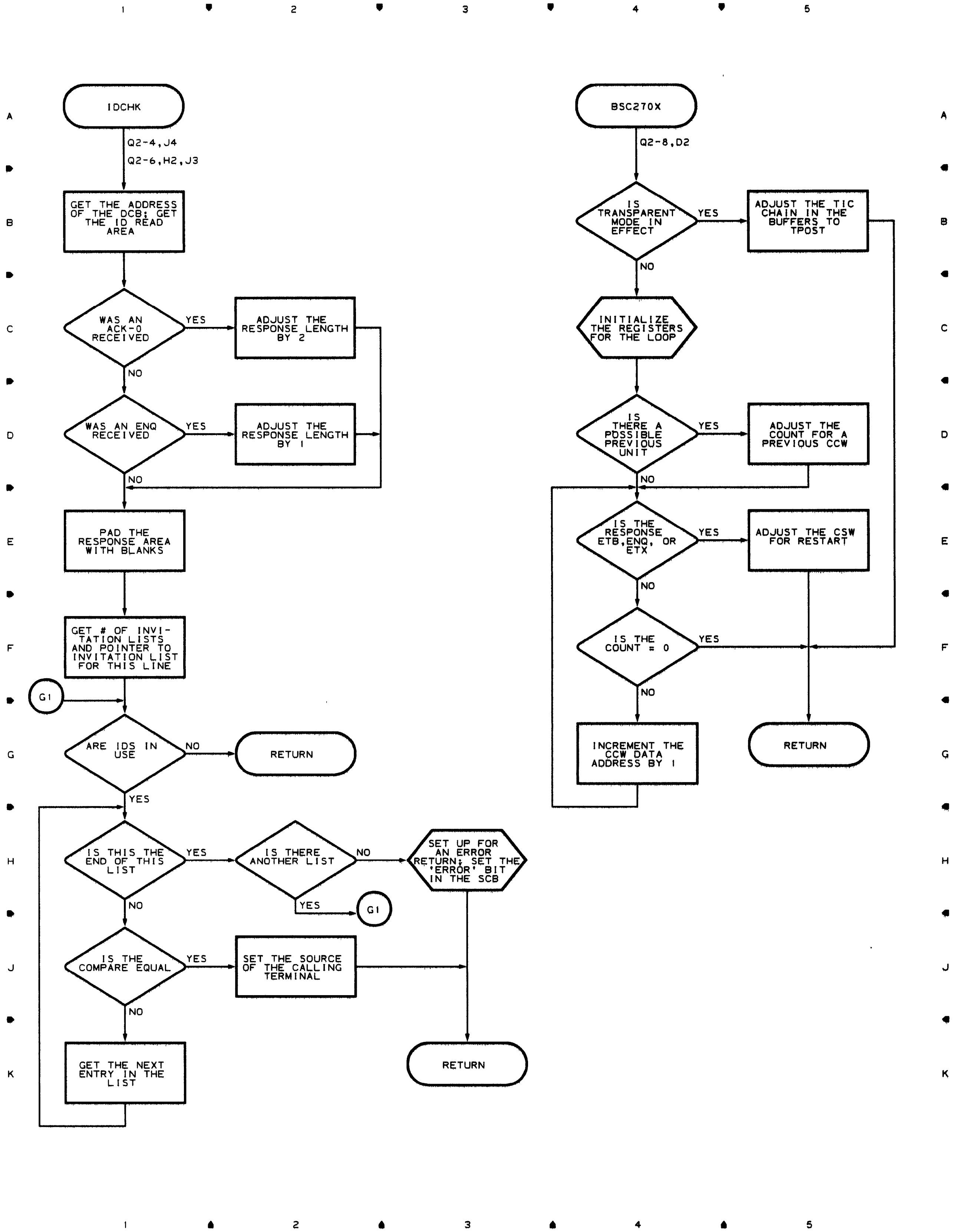
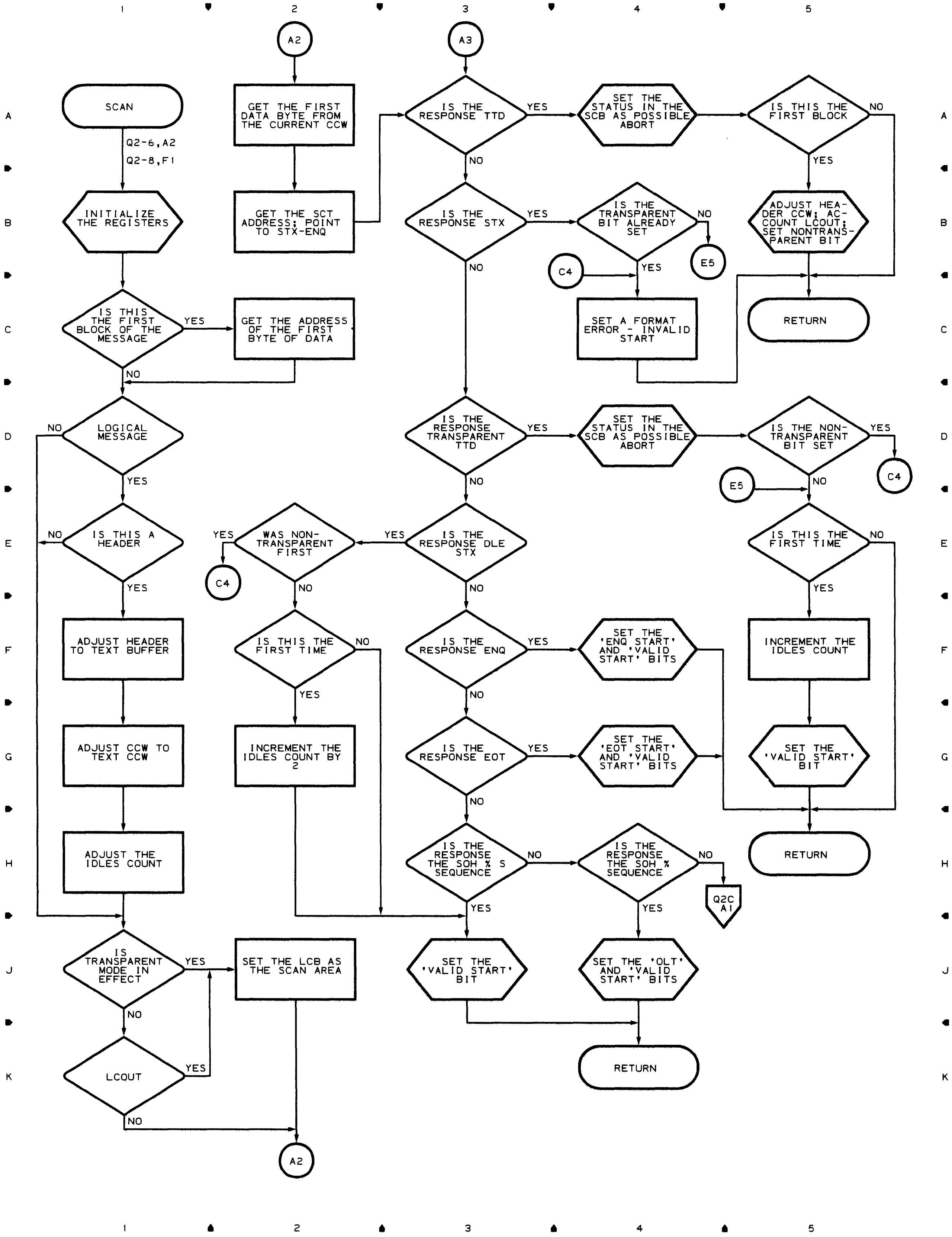
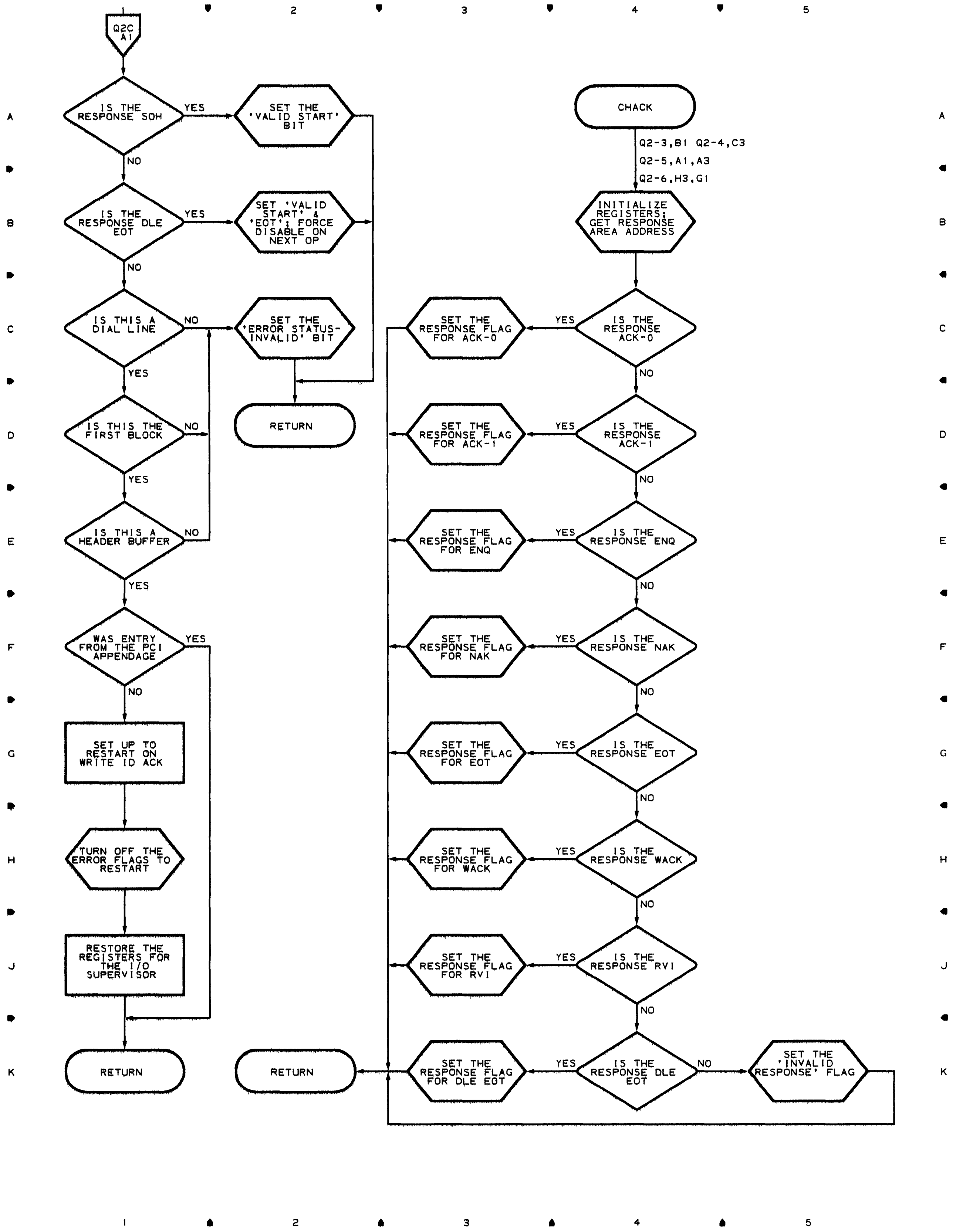

IDCHK
Q2-4,J4
Q2-6,H2,J3
GET THE ADDRESS OF THE DCB; GET THE ID READ AREA
WAS AN ACK-0 RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 2
NO
WAS AN ENQ RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 1
NO
PAD THE RESPONSE AREA WITH BLANKS
GET # OF INVITATION LISTS AND POINTER TO INVITATION LIST FOR THIS LINE
G1
ARE IDS IN USE
NO
RETURN
YES
IS THIS THE END OF THIS LIST
YES
IS THERE ANOTHER LIST
NO
SET UP FOR AN ERROR RETURN; SET THE 'ERROR' BIT IN THE SCB
YES
G1
NO
IS THE COMPARE EQUAL
YES
SET THE SOURCE OF THE CALLING TERMINAL
NO
GET THE NEXT ENTRY IN THE LIST
RETURN
BSC270X
Q2-8,D2
IS TRANSPARENT MODE IN EFFECT
YES
ADJUST THE TIC CHAIN IN THE BUFFERS TO TPOST
NO
INITIALIZE THE REGISTERS FOR THE LOOP
IS THERE A POSSIBLE PREVIOUS UNIT
YES
ADJUST THE COUNT FOR A PREVIOUS CCW
NO
IS THE RESPONSE ETB,ENQ, OR ETX
YES
ADJUST THE CSW FOR RESTART
NO
IS THE COUNT = 0
YES
NO
INCREMENT THE CCW DATA ADDRESS BY 1
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q2-11 (Q2B) LINE END APPENDAGE FOR BSC LINES



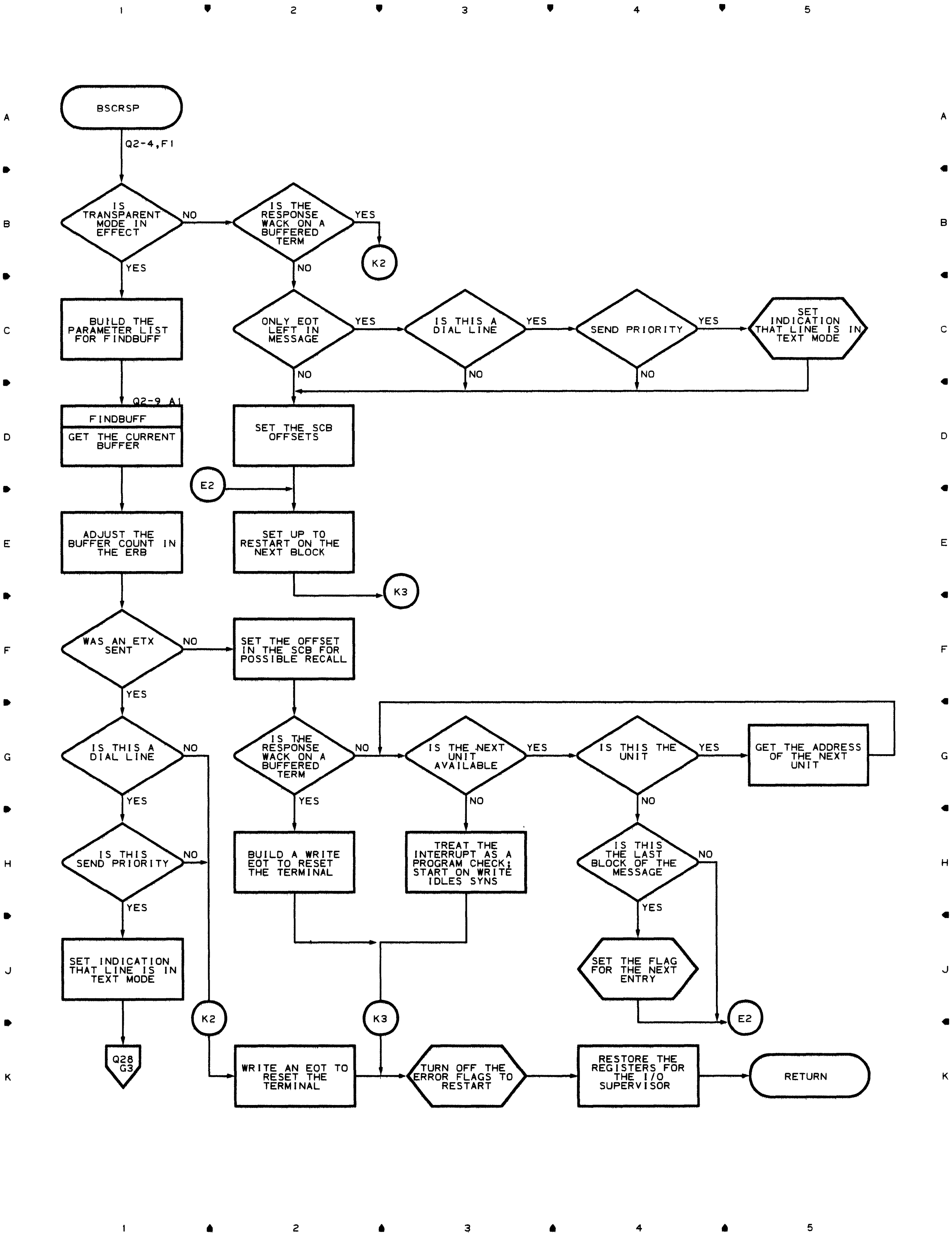
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
A2
A3
SCAN
Q2-6,A2
Q2-8,F1
INITIALIZE THE REGISTERS
IS THIS THE FIRST BLOCK OF THE MESSAGE
YES
NO
GET THE ADDRESS OF THE FIRST BYTE OF DATA
LOGICAL MESSAGE
IS THIS A HEADER
ADJUST HEADER TO TEXT BUFFER
ADJUST CCW TO TEXT CCW
ADJUST THE IDLES COUNT
IS TRANSPARENT MODE IN EFFECT
SET THE LCB AS THE SCAN AREA
LCOUT
GET THE FIRST DATA BYTE FROM THE CURRENT CCW
GET THE SCT ADDRESS; POINT TO STX-ENQ
WAS NON-TRANSPARENT FIRST
C4
IS THIS THE FIRST TIME
INCREMENT THE IDLES COUNT BY 2
IS THE RESPONSE TTD
IS THE RESPONSE STX
IS THE RESPONSE TRANSPARENT TTD
IS THE RESPONSE DLE STX
IS THE RESPONSE ENQ
IS THE RESPONSE EOT
IS THE RESPONSE THE SOH % S SEQUENCE
SET THE 'VALID START' BIT
SET THE STATUS IN THE SCB AS POSSIBLE ABORT
IS THE TRANSPARENT BIT ALREADY SET
E5
SET A FORMAT ERROR - INVALID START
SET THE STATUS IN THE SCB AS POSSIBLE ABORT
SET THE 'ENQ START' AND 'VALID START' BITS
SET THE 'EOT START' AND 'VALID START' BITS
IS THE RESPONSE THE SOH % SEQUENCE
Q2C A1
SET THE 'OLT' AND 'VALID START' BITS
RETURN
IS THIS THE FIRST BLOCK
ADJUST HEADER CCW; ACCOUNT LCOUT; SET NONTRANSPARENT BIT
RETURN
IS THE NON-TRANSPARENT BIT SET
IS THIS THE FIRST TIME
INCREMENT THE IDLES COUNT
SET THE 'VALID START' BIT
RETURN

Chart Q2-12 (Q2C) LINE END APPENDAGE FOR BSC LINES



1
2
3
4
5
Q2C A1
IS THE RESPONSE SOH
YES
SET THE 'VALID START' BIT
NO
IS THE RESPONSE DLE EOT
YES
SET 'VALID START' & 'EOT'; FORCE DISABLE ON NEXT OP
NO
IS THIS A DIAL LINE
NO
SET THE 'ERROR STATUS-INVALID' BIT
YES
IS THIS THE FIRST BLOCK
NO
RETURN
YES
IS THIS A HEADER BUFFER
NO
YES
WAS ENTRY FROM THE PCI APPENDAGE
YES
NO
SET UP TO RESTART ON WRITE ID ACK
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
CHACK
Q2-3,B1 Q2-4,C3
Q2-5,A1,A3
Q2-6,H3,G1
INITIALIZE REGISTERS; GET RESPONSE AREA ADDRESS
IS THE RESPONSE ACK-0
YES
SET THE RESPONSE FLAG FOR ACK-0
NO
IS THE RESPONSE ACK-1
YES
SET THE RESPONSE FLAG FOR ACK-1
NO
IS THE RESPONSE ENQ
YES
SET THE RESPONSE FLAG FOR ENQ
NO
IS THE RESPONSE NAK
YES
SET THE RESPONSE FLAG FOR NAK
NO
IS THE RESPONSE EOT
YES
SET THE RESPONSE FLAG FOR EOT
NO
IS THE RESPONSE WACK
YES
SET THE RESPONSE FLAG FOR WACK
NO
IS THE RESPONSE RVI
YES
SET THE RESPONSE FLAG FOR RVI
NO
IS THE RESPONSE DLE EOT
YES
SET THE RESPONSE FLAG FOR DLE EOT
RETURN
NO
SET THE 'INVALID RESPONSE' FLAG
A
B
C
D
E
F
G
H
J
K

Chart Q2-13 (Q2D) LINE END APPENDAGE FOR BSC LINES



BSCRSP
Q2-4,F1
IS TRANSPARENT MODE IN EFFECT
NO
YES
IS THE RESPONSE WACK ON A BUFFERED TERM
YES
K2
NO
BUILD THE PARAMETER LIST FOR FINDBUFF
ONLY EOT LEFT IN MESSAGE
YES
IS THIS A DIAL LINE
YES
SEND PRIORITY
YES
SET INDICATION THAT LINE IS IN TEXT MODE
NO
NO
NO
Q2-9 A1
FINDBUFF
GET THE CURRENT BUFFER
SET THE SCB OFFSETS
E2
ADJUST THE BUFFER COUNT IN THE ERB
SET UP TO RESTART ON THE NEXT BLOCK
K3
WAS AN ETX SENT
NO
SET THE OFFSET IN THE SCB FOR POSSIBLE RECALL
YES
IS THIS A DIAL LINE
NO
IS THE RESPONSE WACK ON A BUFFERED TERM
NO
IS THE NEXT UNIT AVAILABLE
YES
IS THIS THE UNIT
YES
GET THE ADDRESS OF THE NEXT UNIT
YES
YES
NO
NO
IS THIS SEND PRIORITY
NO
BUILD A WRITE EOT TO RESET THE TERMINAL
TREAT THE INTERRUPT AS A PROGRAM CHECK; START ON WRITE IDLES SYNS
IS THIS THE LAST BLOCK OF THE MESSAGE
NO
YES
YES
SET INDICATION THAT LINE IS IN TEXT MODE
SET THE FLAG FOR THE NEXT ENTRY
K2
K3
E2
Q28 G3
WRITE AN EOT TO RESET THE TERMINAL
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
A
B
C
D
E
F
G
H
J
K
1
2
3
4
5

Chart Q3-1 (Q31) LINE END APPENDAGE FOR START/STOP LINES

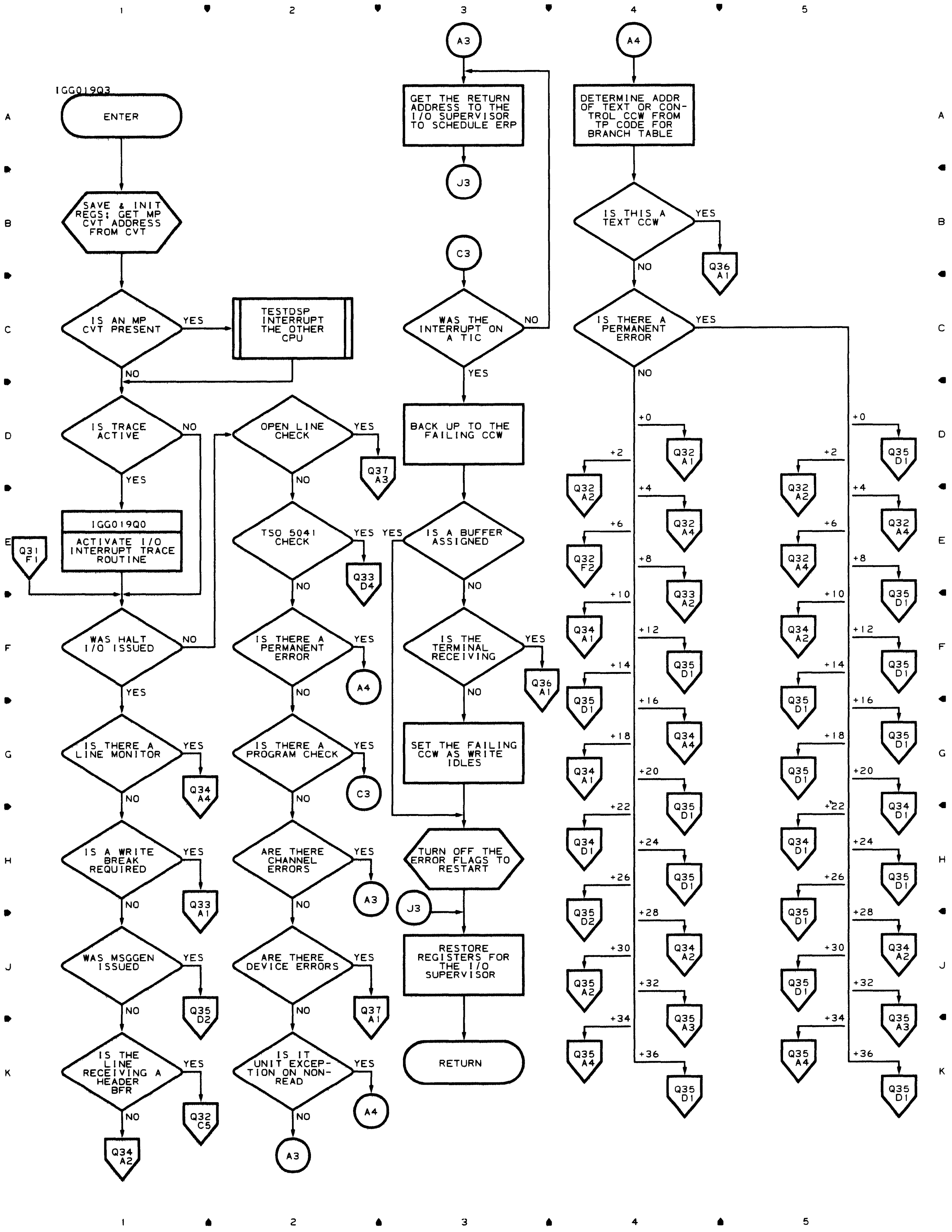
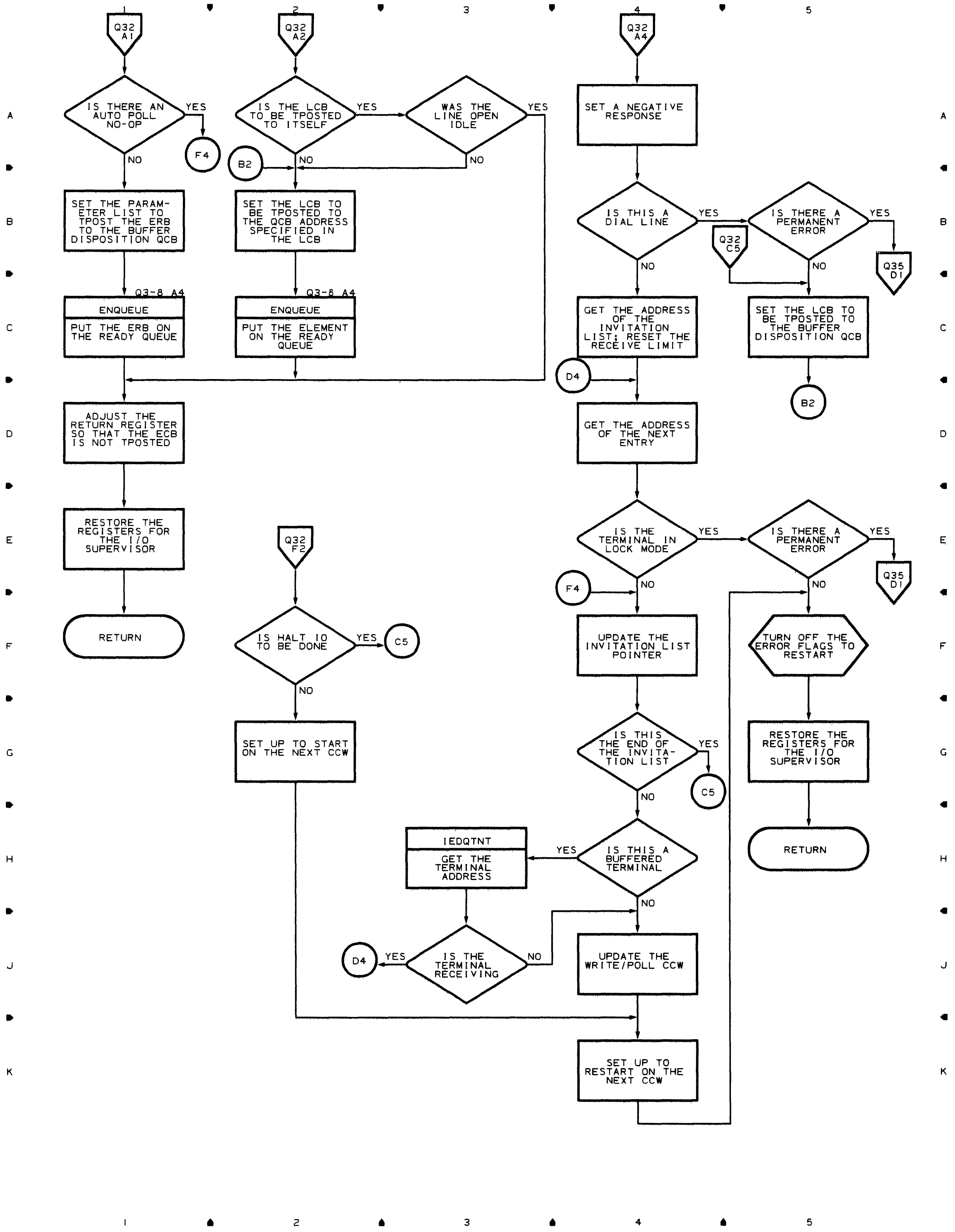


Chart Q3-2 (Q32) LINE END APPENDAGE FOR START/STOP LINES



Q32 A1
IS THERE AN AUTO POLL NO-OP
YES
F4
NO
SET THE PARAMETER LIST TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q3-8 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT TPOSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q32 A2
IS THE LCB TO BE TPOSTED TO ITSELF
YES
WAS THE LINE OPEN IDLE
YES
NO
B2
NO
SET THE LCB TO BE TPOSTED TO THE QCB ADDRESS SPECIFIED IN THE LCB
Q3-8 A4
ENQUEUE
PUT THE ELEMENT ON THE READY QUEUE
Q32 F2
IS HALT IO TO BE DONE
YES
C5
NO
SET UP TO START ON THE NEXT CCW
Q32 A4
SET A NEGATIVE RESPONSE
IS THIS A DIAL LINE
YES
IS THERE A PERMANENT ERROR
YES
Q35 D1
Q32 C5
NO
NO
GET THE ADDRESS OF THE INVITATION LIST; RESET THE RECEIVE LIMIT
SET THE LCB TO BE TPOSTED TO THE BUFFER DISPOSITION QCB
D4
B2
GET THE ADDRESS OF THE NEXT ENTRY
IS THE TERMINAL IN LOCK MODE
YES
IS THERE A PERMANENT ERROR
YES
Q35 D1
F4
NO
NO
UPDATE THE INVITATION LIST POINTER
TURN OFF THE ERROR FLAGS TO RESTART
IS THIS THE END OF THE INVITATION LIST
YES
C5
NO
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
IEDQTNT
GET THE TERMINAL ADDRESS
YES
IS THIS A BUFFERED TERMINAL
NO
D4
YES
IS THE TERMINAL RECEIVING
NO
UPDATE THE WRITE/POLL CCW
SET UP TO RESTART ON THE NEXT CCW
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q3-3 (Q33) LINE END APPENDAGE FOR START/STOP LINES

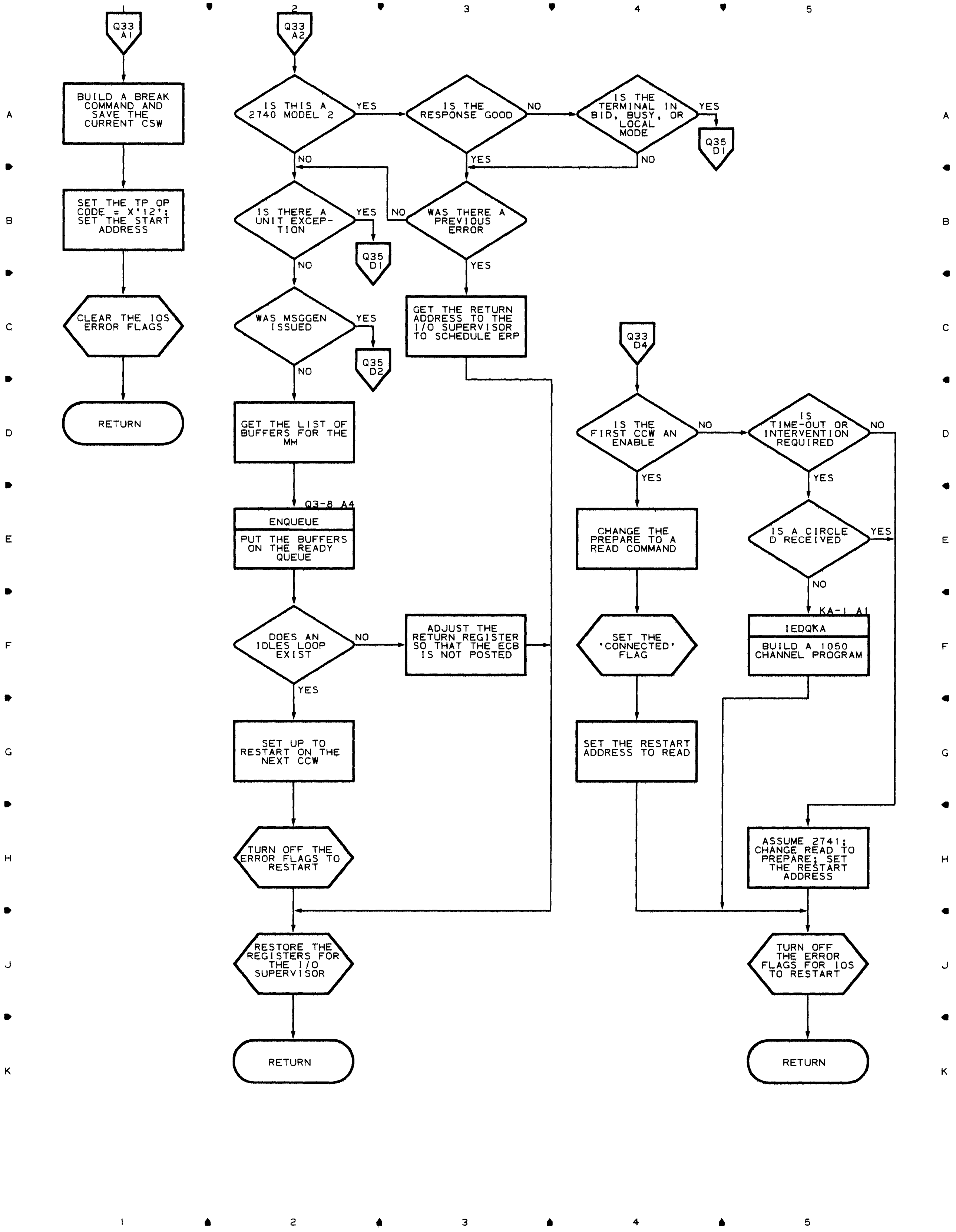

Q33 A1
BUILD A BREAK COMMAND AND SAVE THE CURRENT CSW
SET THE TP OP CODE = X'12'; SET THE START ADDRESS
CLEAR THE IOS ERROR FLAGS
RETURN
Q33 A2
IS THIS A 2740 MODEL 2
YES
NO
IS THERE A UNIT EXCEPTION
YES
Q35 D1
NO
WAS MSGGEN ISSUED
YES
Q35 D2
NO
GET THE LIST OF BUFFERS FOR THE MH
Q3-8 A4
ENQUEUE
PUT THE BUFFERS ON THE READY QUEUE
DOES AN IDLES LOOP EXIST
NO
YES
SET UP TO RESTART ON THE NEXT CCW
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
IS THE RESPONSE GOOD
NO
YES
WAS THERE A PREVIOUS ERROR
NO
YES
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
IS THE TERMINAL IN BID, BUSY, OR LOCAL MODE
YES
Q35 D1
NO
Q33 D4
IS THE FIRST CCW AN ENABLE
NO
YES
CHANGE THE PREPARE TO A READ COMMAND
SET THE 'CONNECTED' FLAG
SET THE RESTART ADDRESS TO READ
IS TIME-OUT OR INTERVENTION REQUIRED
NO
YES
IS A CIRCLE D RECEIVED
YES
NO
KA-1 A1
IEDQKA
BUILD A 1050 CHANNEL PROGRAM
ASSUME 2741; CHANGE READ TO PREPARE; SET THE RESTART ADDRESS
TURN OFF THE ERROR FLAGS FOR IOS TO RESTART
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q3-4 (Q34) LINE END APPENDAGE FOR START/STOP LINES

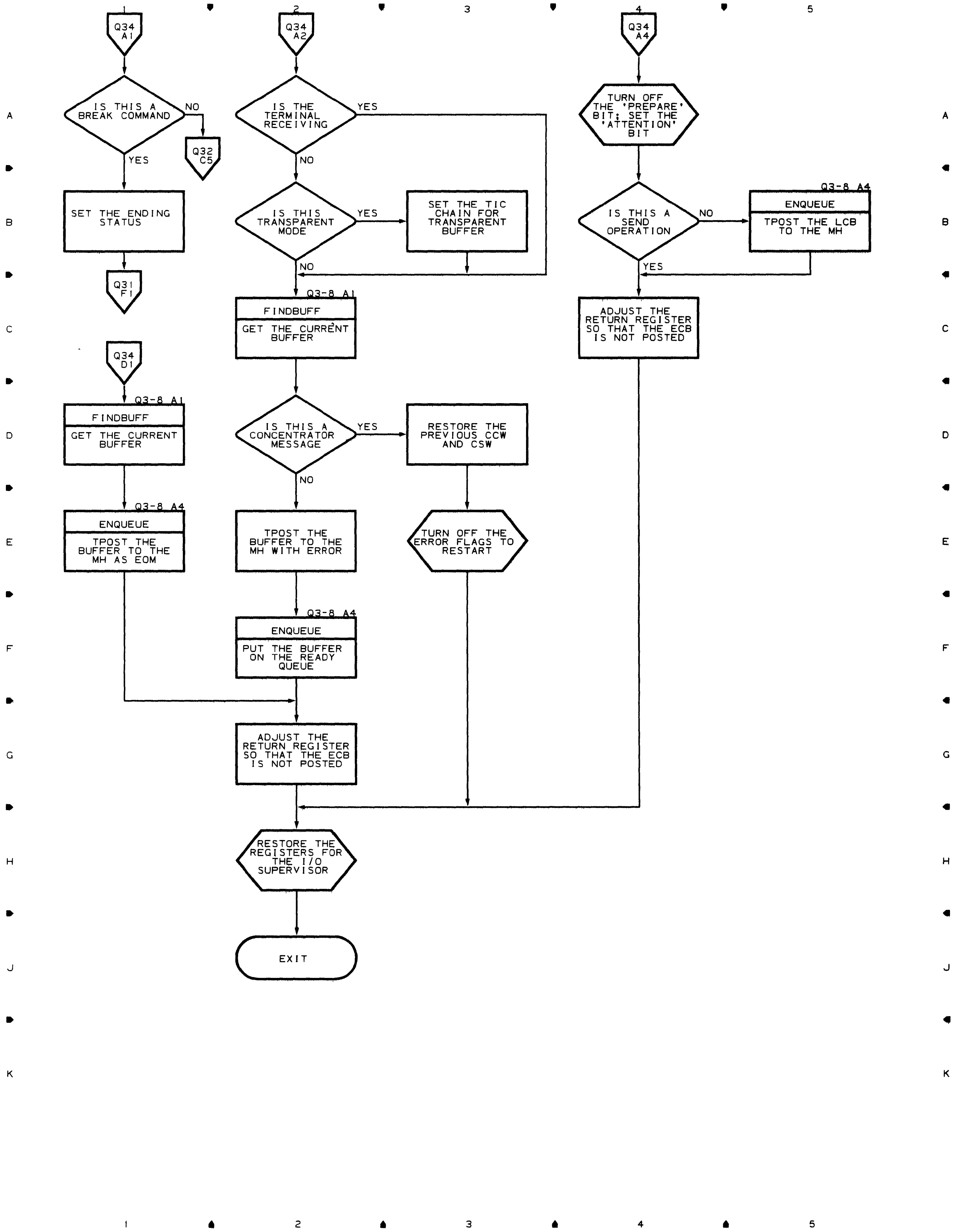
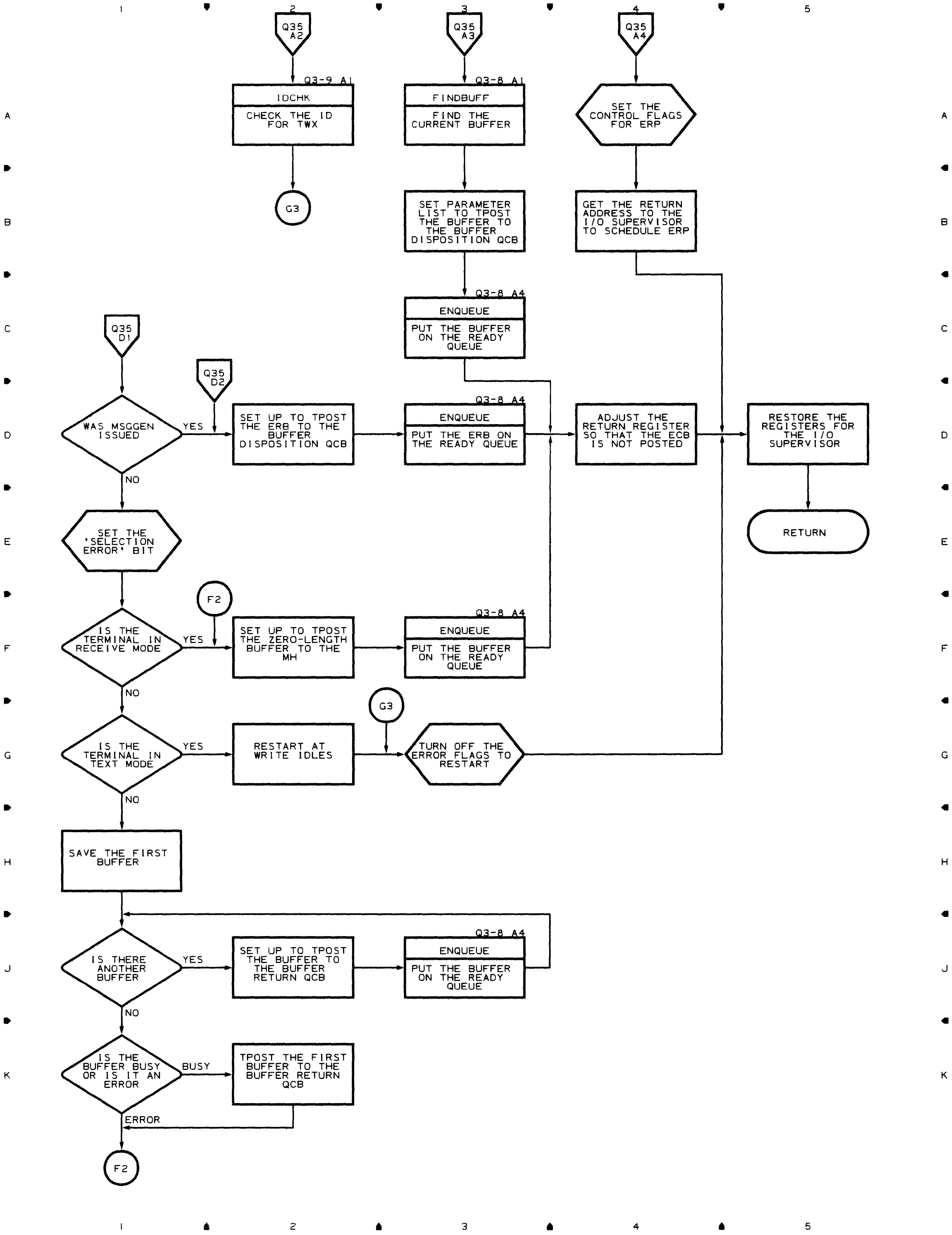
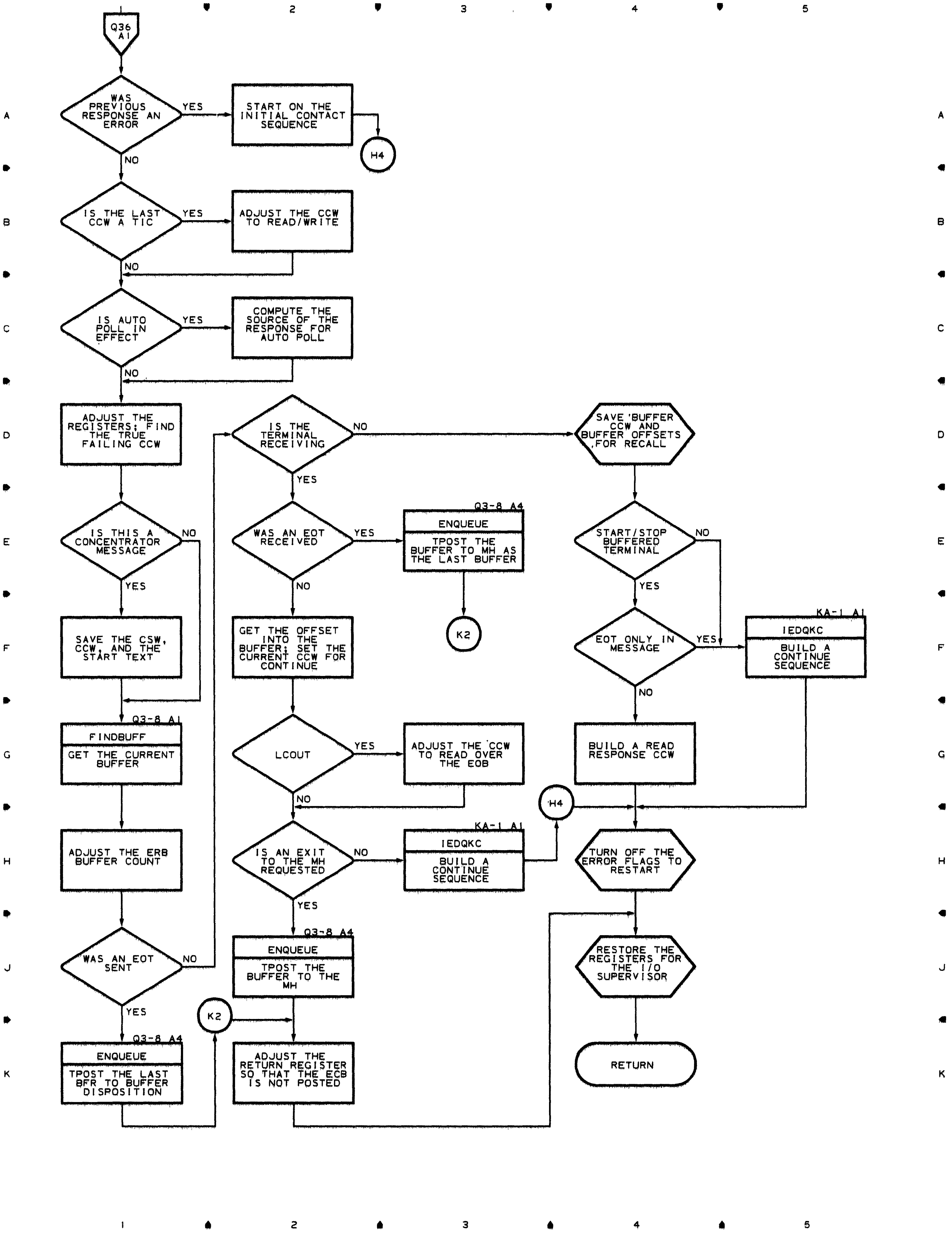

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
Q34 A1
IS THIS A BREAK COMMAND
NO
Q32 C5
YES
SET THE ENDING STATUS
Q31 F1
Q34 D1
Q3-8 A1
FINDBUFF
GET THE CURRENT BUFFER
Q3-8 A4
ENQUEUE
TPOST THE BUFFER TO THE MH AS EOM
Q34 A2
IS THE TERMINAL RECEIVING
YES
NO
IS THIS TRANSPARENT MODE
YES
SET THE TIC CHAIN FOR TRANSPARENT BUFFER
NO
Q3-8 A1
FINDBUFF
GET THE CURRENT BUFFER
IS THIS A CONCENTRATOR MESSAGE
YES
RESTORE THE PREVIOUS CCW AND CSW
NO
TPOST THE BUFFER TO THE MH WITH ERROR
TURN OFF THE ERROR FLAGS TO RESTART
Q3-8 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
EXIT
Q34 A4
TURN OFF THE 'PREPARE' BIT; SET THE 'ATTENTION' BIT
IS THIS A SEND OPERATION
NO
Q3-8 A4
ENQUEUE
TPOST THE LCB TO THE MH
YES
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED

Chart Q3-5 (Q35) LINE END APPENDAGE FOR START/STOP LINES



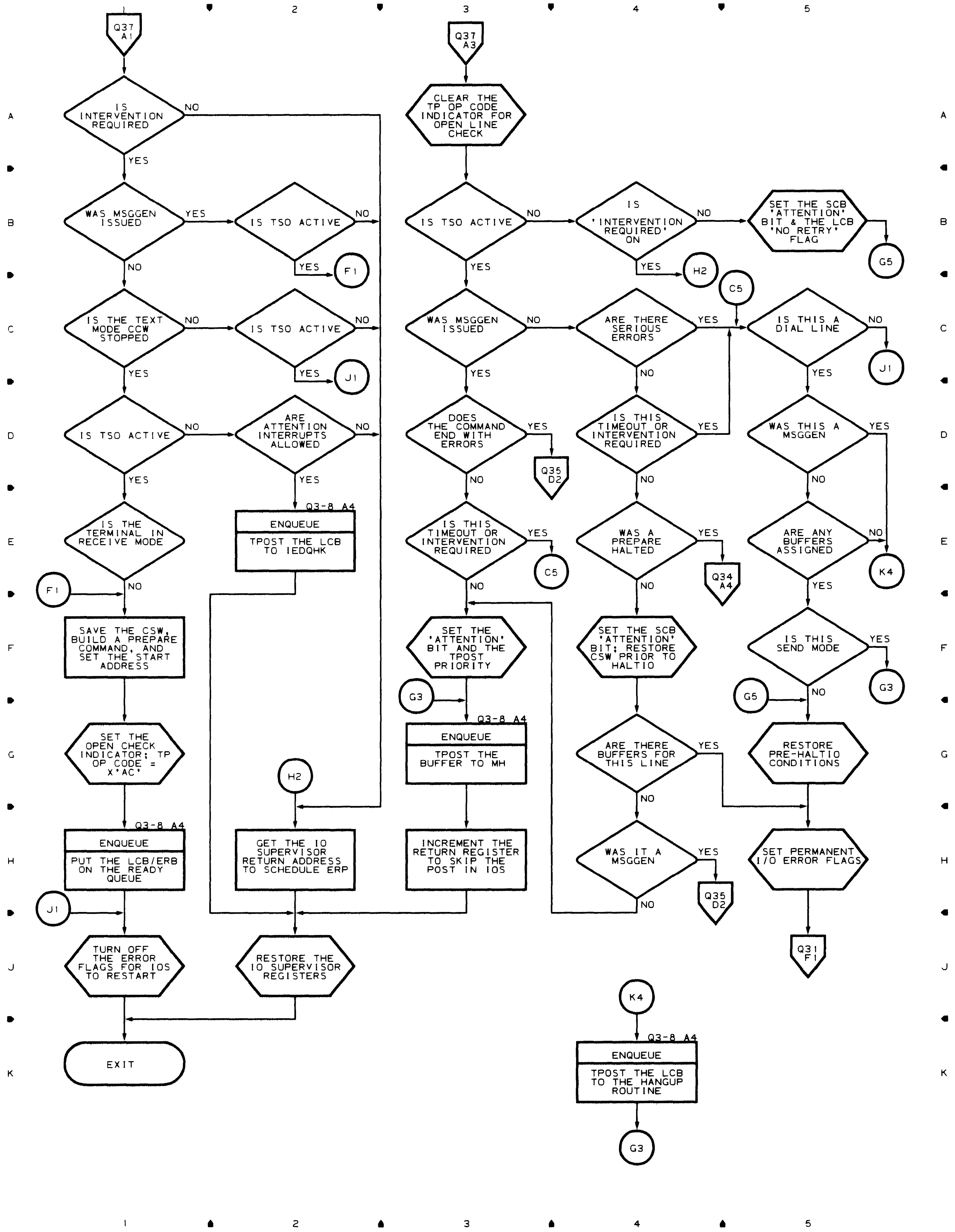
Q35 A2
Q3-9 A1
IDCHK
CHECK THE ID FOR TWX
G3
Q35 A3
Q3-8 A1
FINDBUFF
FIND THE CURRENT BUFFER
SET PARAMETER LIST TO TPOST THE BUFFER TO THE BUFFER DISPOSITION QCB
Q3-8 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
Q35 A4
SET THE CONTROL FLAGS FOR ERP
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
Q35 D1
Q35 D2
WAS MSGGEN ISSUED
YES
SET UP TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q3-8 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
NO
SET THE 'SELECTION ERROR' BIT
F2
IS THE TERMINAL IN RECEIVE MODE
YES
SET UP TO TPOST THE ZERO-LENGTH BUFFER TO THE MH
Q3-8 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
NO
G3
IS THE TERMINAL IN TEXT MODE
YES
RESTART AT WRITE IDLES
TURN OFF THE ERROR FLAGS TO RESTART
NO
SAVE THE FIRST BUFFER
IS THERE ANOTHER BUFFER
YES
SET UP TO TPOST THE BUFFER TO THE BUFFER RETURN QCB
Q3-8 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
NO
IS THE BUFFER BUSY OR IS IT AN ERROR
BUSY
TPOST THE FIRST BUFFER TO THE BUFFER RETURN QCB
ERROR
F2
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q3-6 (Q36) LINE END APPENDAGE FOR START/STOP LINES



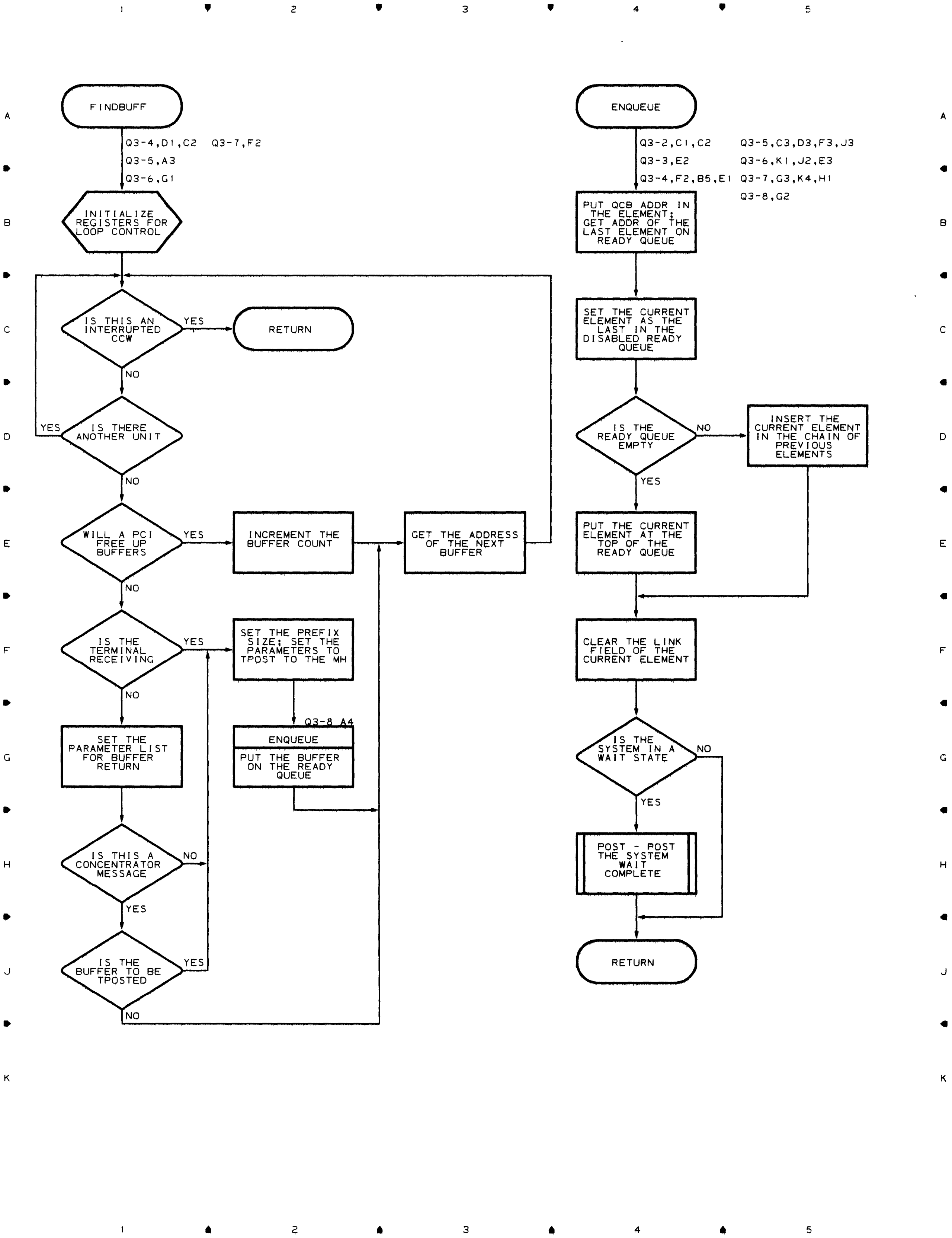
Q36 A1
WAS PREVIOUS RESPONSE AN ERROR
YES
START ON THE INITIAL CONTACT SEQUENCE
H4
NO
IS THE LAST CCW A TIC
YES
ADJUST THE CCW TO READ/WRITE
NO
IS AUTO POLL IN EFFECT
YES
COMPUTE THE SOURCE OF THE RESPONSE FOR AUTO POLL
NO
ADJUST THE REGISTERS; FIND THE TRUE FAILING CCW
IS THIS A CONCENTRATOR MESSAGE
NO
YES
SAVE THE CSW, CCW, AND THE START TEXT
Q3-8 A1
FINDBUFF
GET THE CURRENT BUFFER
ADJUST THE ERB BUFFER COUNT
WAS AN EOT SENT
NO
YES
Q3-8 A4
ENQUEUE
TPOST THE LAST BFR TO BUFFER DISPOSITION
K2
IS THE TERMINAL RECEIVING
NO
YES
WAS AN EOT RECEIVED
YES
Q3-8 A4
ENQUEUE
TPOST THE BUFFER TO MH AS THE LAST BUFFER
K2
NO
GET THE OFFSET INTO THE BUFFER; SET THE CURRENT CCW FOR CONTINUE
LCOUT
YES
ADJUST THE CCW TO READ OVER THE EOB
NO
IS AN EXIT TO THE MH REQUESTED
NO
KA-1 A1
IEDQKC
BUILD A CONTINUE SEQUENCE
H4
YES
Q3-8 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
SAVE 'BUFFER CCW AND BUFFER OFFSETS ,FOR RECALL
START/STOP BUFFERED TERMINAL
NO
YES
EOT ONLY IN MESSAGE
YES
KA-1 A1
IEDQKC
BUILD A CONTINUE SEQUENCE
NO
BUILD A READ RESPONSE CCW
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q3-7 (Q37) LINE END APPENDAGE FOR START/STOP LINES



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
Q37 A1
IS INTERVENTION REQUIRED
NO
YES
WAS MSGGEN ISSUED
YES
IS TSO ACTIVE
NO
YES
F1
NO
IS THE TEXT MODE CCW STOPPED
NO
IS TSO ACTIVE
NO
YES
J1
YES
IS TSO ACTIVE
NO
ARE ATTENTION INTERRUPTS ALLOWED
NO
YES
YES
Q3-8 A4
ENQUEUE
TPOST THE LCB TO IEDQHK
IS THE TERMINAL IN RECEIVE MODE
F1
NO
SAVE THE CSW, BUILD A PREPARE COMMAND, AND SET THE START ADDRESS
SET THE OPEN CHECK INDICATOR; TP OP CODE = X'AC'
Q3-8 A4
ENQUEUE
PUT THE LCB/ERB ON THE READY QUEUE
J1
TURN OFF THE ERROR FLAGS FOR IOS TO RESTART
EXIT
H2
GET THE IO SUPERVISOR RETURN ADDRESS TO SCHEDULE ERP
RESTORE THE IO SUPERVISOR REGISTERS
Q37 A3
CLEAR THE TP OP CODE INDICATOR FOR OPEN LINE CHECK
IS TSO ACTIVE
NO
YES
WAS MSGGEN ISSUED
NO
YES
DOES THE COMMAND END WITH ERRORS
YES
Q35 D2
NO
IS THIS TIMEOUT OR INTERVENTION REQUIRED
YES
C5
NO
SET THE 'ATTENTION' BIT AND THE TPOST PRIORITY
G3
Q3-8 A4
ENQUEUE
TPOST THE BUFFER TO MH
INCREMENT THE RETURN REGISTER TO SKIP THE POST IN IOS
IS 'INTERVENTION REQUIRED' ON
NO
YES
H2
C5
ARE THERE SERIOUS ERRORS
YES
NO
IS THIS TIMEOUT OR INTERVENTION REQUIRED
YES
NO
WAS A PREPARE HALTED
YES
Q34 A4
NO
SET THE SCB 'ATTENTION' BIT; RESTORE CSW PRIOR TO HALTIO
ARE THERE BUFFERS FOR THIS LINE
YES
NO
WAS IT A MSGGEN
YES
Q35 D2
NO
K4
Q3-8 A4
ENQUEUE
TPOST THE LCB TO THE HANGUP ROUTINE
G3
SET THE SCB 'ATTENTION' BIT & THE LCB 'NO RETRY' FLAG
G5
IS THIS A DIAL LINE
NO
J1
YES
WAS THIS A MSGGEN
YES
NO
ARE ANY BUFFERS ASSIGNED
NO
K4
YES
IS THIS SEND MODE
YES
G3
G5
NO
RESTORE PRE-HALTIO CONDITIONS
SET PERMANENT I/O ERROR FLAGS
Q31 F1

Chart Q3-8 (Q38) LINE END APPENDAGE FOR START/STOP LINES



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FINDBUFF
Q3-4,D1,C2 Q3-7,F2
Q3-5,A3
Q3-6,G1
INITIALIZE REGISTERS FOR LOOP CONTROL
IS THIS AN INTERRUPTED CCW
YES
RETURN
NO
IS THERE ANOTHER UNIT
YES
NO
WILL A PCI FREE UP BUFFERS
YES
INCREMENT THE BUFFER COUNT
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE TERMINAL RECEIVING
YES
SET THE PREFIX SIZE; SET THE PARAMETERS TO TPOST TO THE MH
NO
Q3-8 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
SET THE PARAMETER LIST FOR BUFFER RETURN
IS THIS A CONCENTRATOR MESSAGE
NO
YES
IS THE BUFFER TO BE TPOSTED
YES
NO
ENQUEUE
Q3-2,C1,C2 Q3-5,C3,D3,F3,J3
Q3-3,E2 Q3-6,K1,J2,E3
Q3-4,F2,B5,E1 Q3-7,G3,K4,H1
Q3-8,G2
PUT QCB ADDR IN THE ELEMENT; GET ADDR OF THE LAST ELEMENT ON READY QUEUE
SET THE CURRENT ELEMENT AS THE LAST IN THE DISABLED READY QUEUE
IS THE READY QUEUE EMPTY
NO
INSERT THE CURRENT ELEMENT IN THE CHAIN OF PREVIOUS ELEMENTS
YES
PUT THE CURRENT ELEMENT AT THE TOP OF THE READY QUEUE
CLEAR THE LINK FIELD OF THE CURRENT ELEMENT
IS THE SYSTEM IN A WAIT STATE
NO
YES
POST - POST THE SYSTEM WAIT COMPLETE
RETURN

Chart Q3-9 (Q39) LINE END APPENDAGE FOR START/STOP LINES

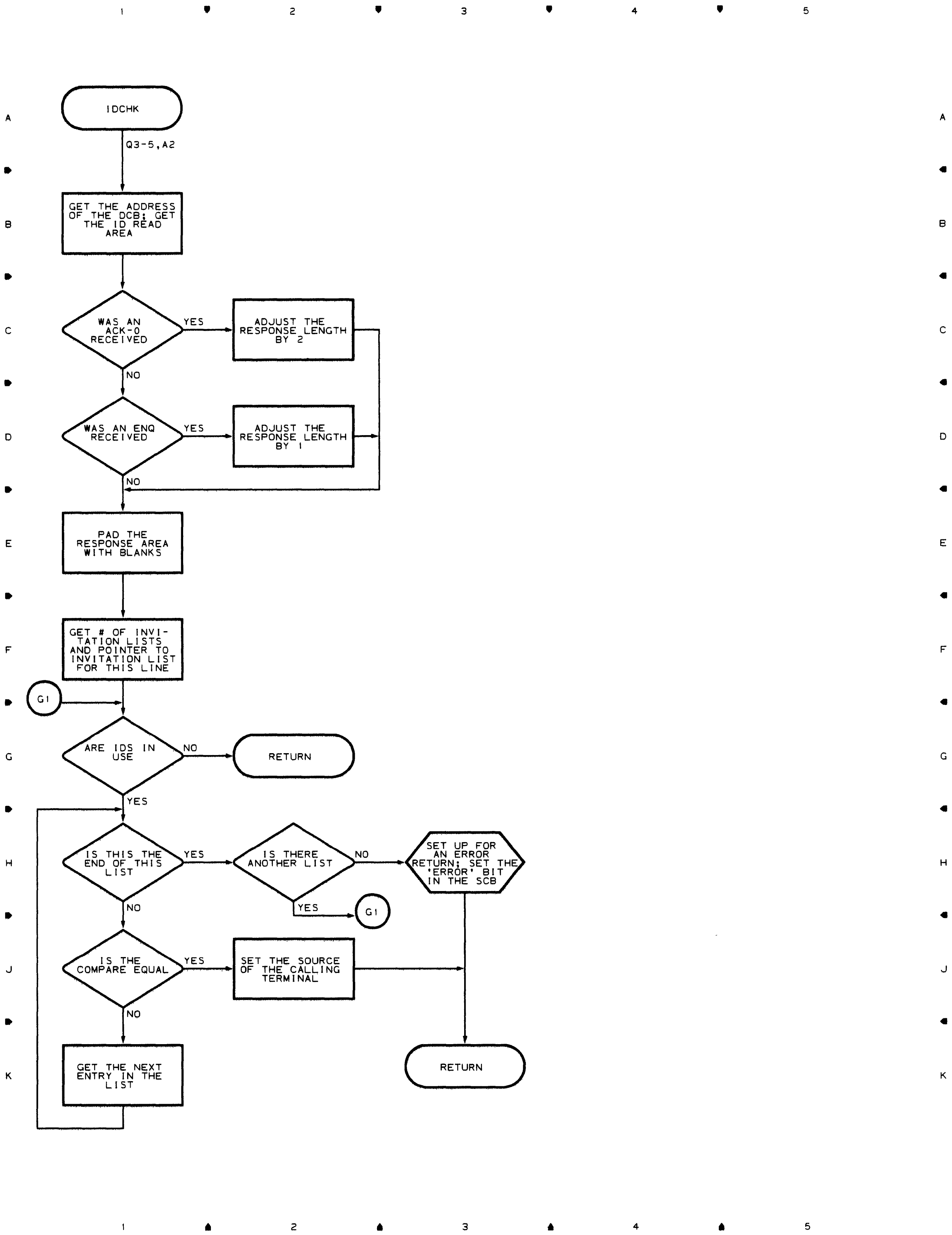

IDCHK
Q3-5,A2
GET THE ADDRESS OF THE DCB; GET THE ID READ AREA
WAS AN ACK-0 RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 2
NO
WAS AN ENQ RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 1
NO
PAD THE RESPONSE AREA WITH BLANKS
GET # OF INVITATION LISTS AND POINTER TO INVITATION LIST FOR THIS LINE
G1
ARE IDS IN USE
NO
RETURN
YES
IS THIS THE END OF THIS LIST
YES
IS THERE ANOTHER LIST
NO
SET UP FOR AN ERROR RETURN; SET THE 'ERROR' BIT IN THE SCB
YES
G1
NO
IS THE COMPARE EQUAL
YES
SET THE SOURCE OF THE CALLING TERMINAL
NO
GET THE NEXT ENTRY IN THE LIST
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q4-1 (Q41) LINE END APPENDAGE FOR LEASED AND START/STOP LINES AND NO TSO

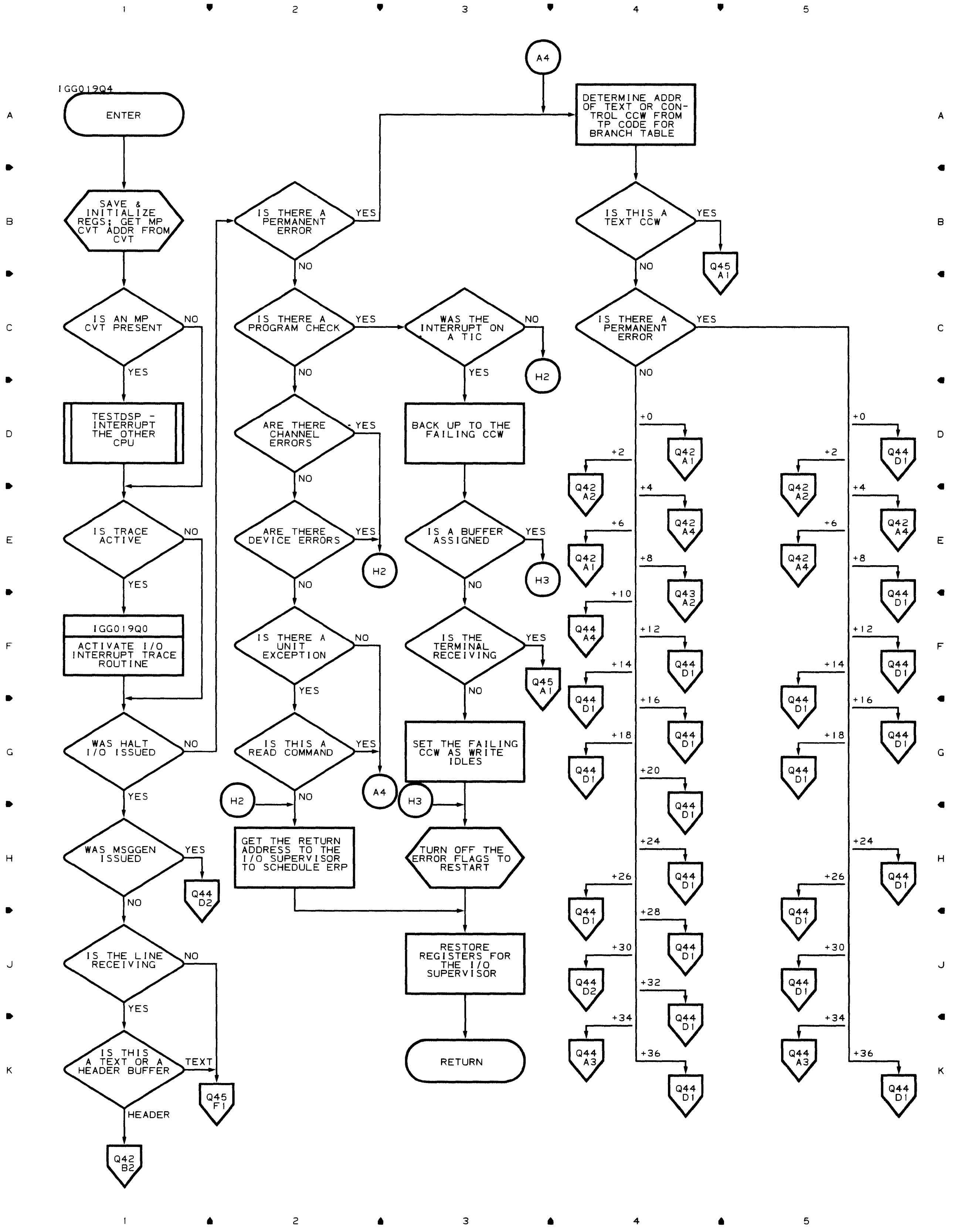


Chart Q4-2 (Q42) LINE END APPENDAGE FOR LEASED AND START/STOP LINES AND NO TSO

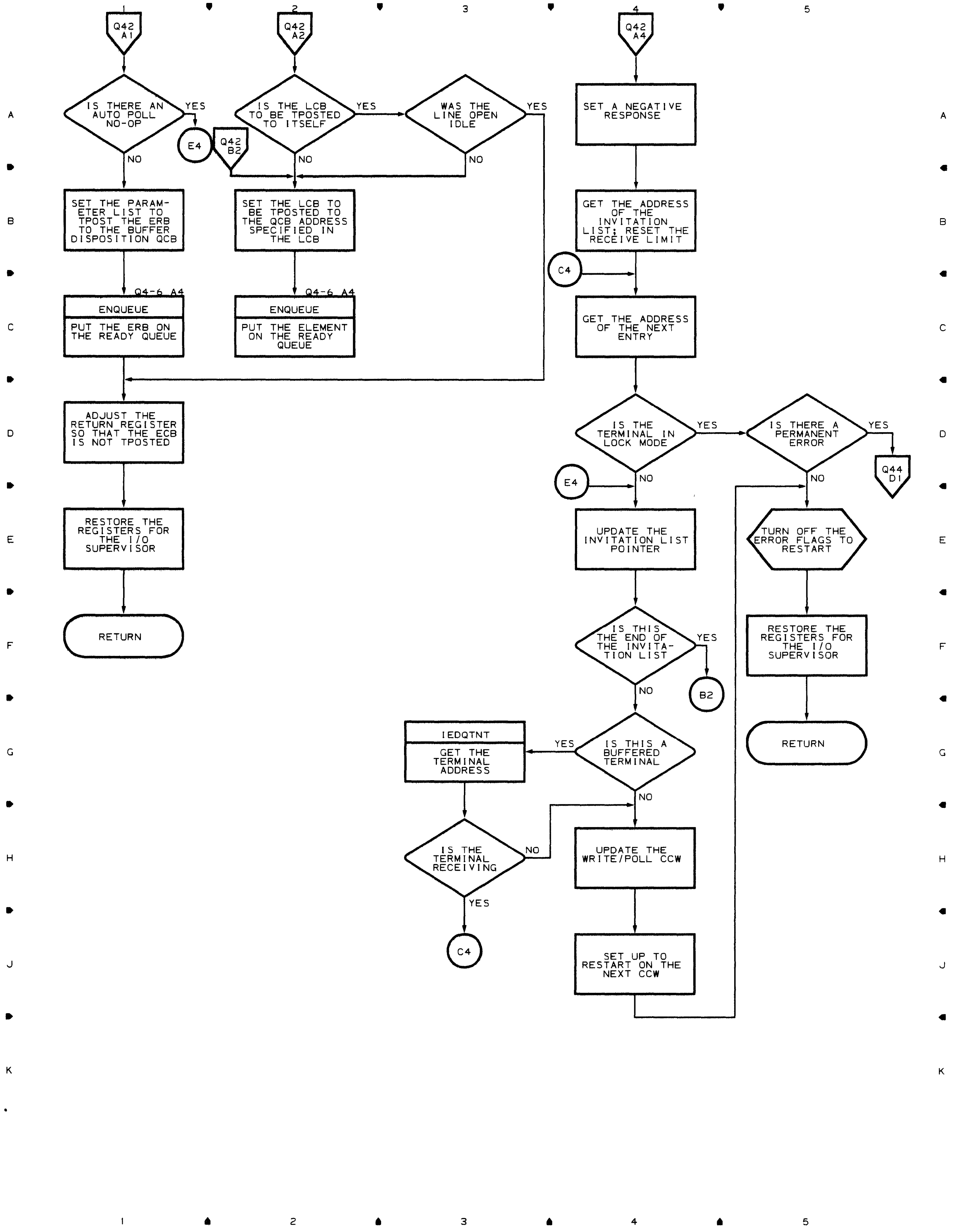

Q42 A1
IS THERE AN AUTO POLL NO-OP
YES
E4
NO
SET THE PARAMETER LIST TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q4-6 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT TPOSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q42 A2
IS THE LCB TO BE TPOSTED TO ITSELF
YES
NO
Q42 B2
SET THE LCB TO BE TPOSTED TO THE QCB ADDRESS SPECIFIED IN THE LCB
Q4-6 A4
ENQUEUE
PUT THE ELEMENT ON THE READY QUEUE
WAS THE LINE OPEN IDLE
YES
NO
Q42 A4
SET A NEGATIVE RESPONSE
GET THE ADDRESS OF THE INVITATION LIST; RESET THE RECEIVE LIMIT
C4
GET THE ADDRESS OF THE NEXT ENTRY
IS THE TERMINAL IN LOCK MODE
YES
E4
NO
UPDATE THE INVITATION LIST POINTER
IS THIS THE END OF THE INVITATION LIST
YES
B2
NO
IS THIS A BUFFERED TERMINAL
YES
NO
IEDQTNT
GET THE TERMINAL ADDRESS
IS THE TERMINAL RECEIVING
NO
YES
C4
UPDATE THE WRITE/POLL CCW
SET UP TO RESTART ON THE NEXT CCW
IS THERE A PERMANENT ERROR
YES
Q44 D1
NO
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q4-3 (Q43) LINE END APPENDAGE FOR LEASED AND START/STOP LINES AND NO TSO

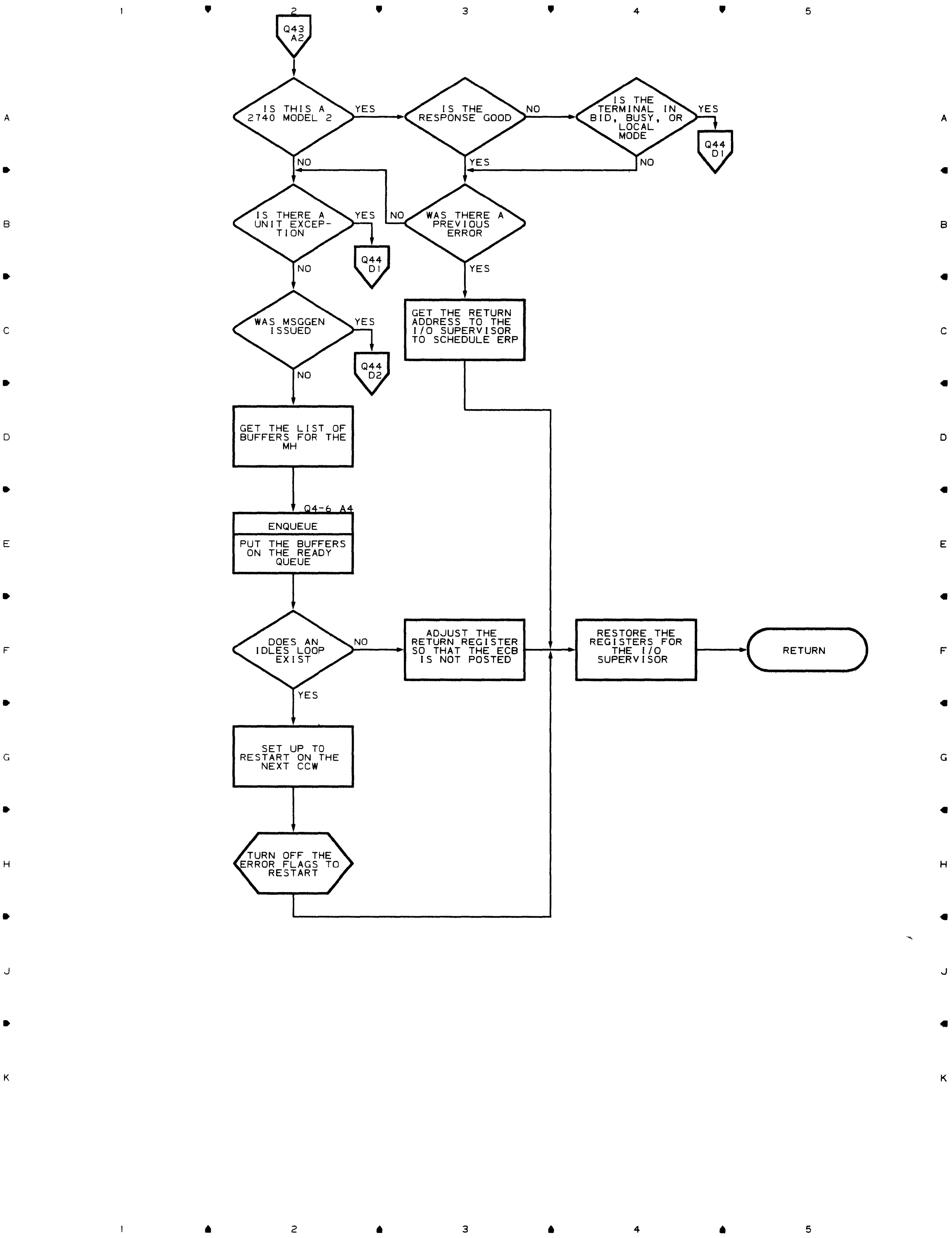
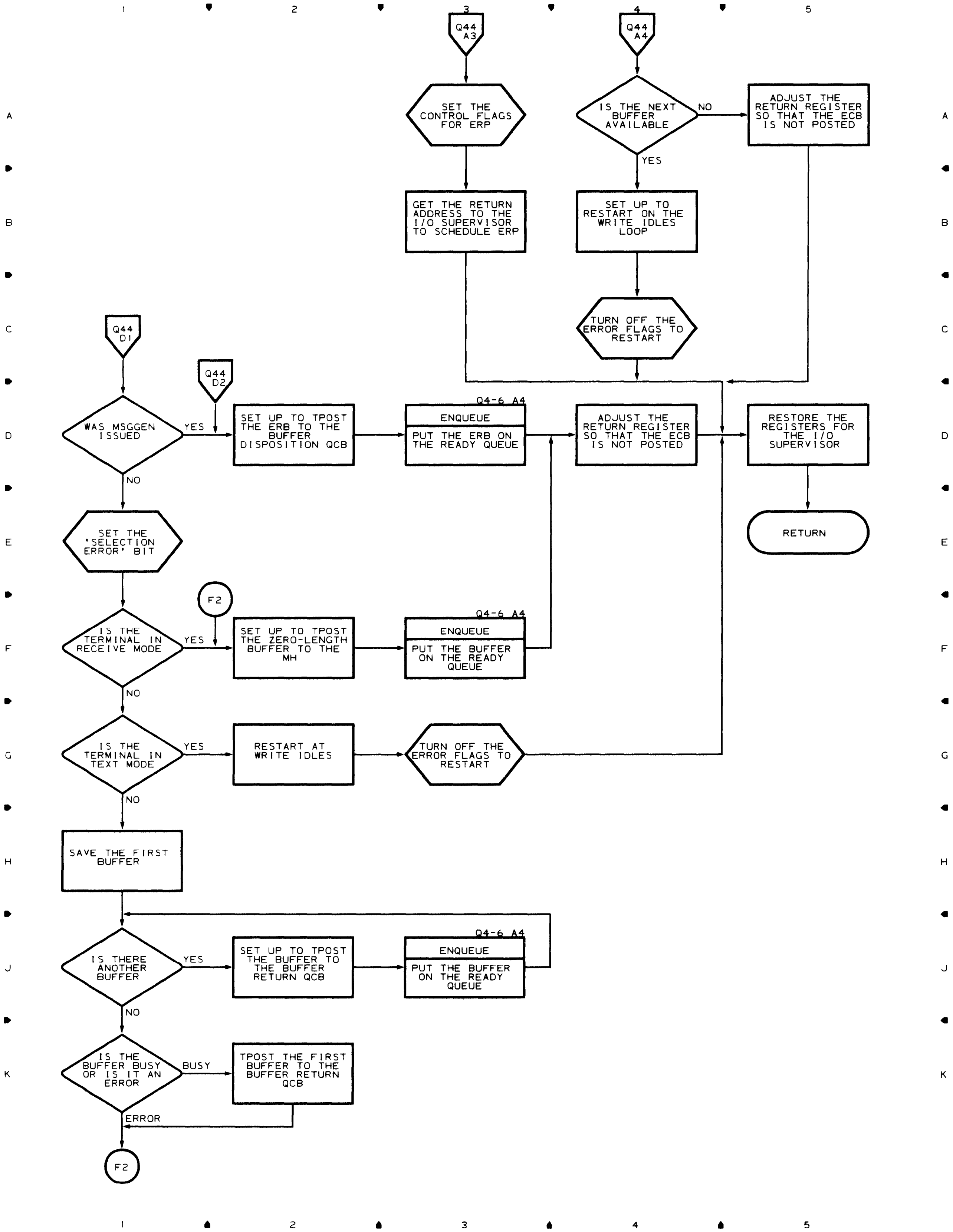

1
2
3
4
5
Q43 A2
IS THIS A 2740 MODEL 2
YES
IS THE RESPONSE GOOD
NO
IS THE TERMINAL IN BID, BUSY, OR LOCAL MODE
YES
Q44 D1
NO
YES
NO
IS THERE A UNIT EXCEPTION
YES
Q44 D1
NO
WAS THERE A PREVIOUS ERROR
YES
NO
WAS MSGGEN ISSUED
YES
Q44 D2
NO
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
GET THE LIST OF BUFFERS FOR THE MH
Q4-6 A4
ENQUEUE
PUT THE BUFFERS ON THE READY QUEUE
DOES AN IDLES LOOP EXIST
NO
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
YES
SET UP TO RESTART ON THE NEXT CCW
TURN OFF THE ERROR FLAGS TO RESTART
A
B
C
D
E
F
G
H
J
K

Chart Q4-4 (Q44) LINE END APPENDAGE FOR LEASED AND START/STOP LINES AND NO TSO



Q44 A3
SET THE CONTROL FLAGS FOR ERP
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
Q44 A4
IS THE NEXT BUFFER AVAILABLE
NO
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
YES
SET UP TO RESTART ON THE WRITE IDLES LOOP
TURN OFF THE ERROR FLAGS TO RESTART
Q44 D1
Q44 D2
WAS MSGGEN ISSUED
YES
SET UP TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q4-6 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
NO
SET THE 'SELECTION ERROR' BIT
F2
IS THE TERMINAL IN RECEIVE MODE
YES
SET UP TO TPOST THE ZERO-LENGTH BUFFER TO THE MH
Q4-6 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
NO
IS THE TERMINAL IN TEXT MODE
YES
RESTART AT WRITE IDLES
TURN OFF THE ERROR FLAGS TO RESTART
NO
SAVE THE FIRST BUFFER
IS THERE ANOTHER BUFFER
YES
SET UP TO TPOST THE BUFFER TO THE BUFFER RETURN QCB
Q4-6 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
NO
IS THE BUFFER BUSY OR IS IT AN ERROR
BUSY
TPOST THE FIRST BUFFER TO THE BUFFER RETURN QCB
ERROR
F2
1 2 3 4 5
A B C D E F G H J K

Chart Q4-5 (Q45) LINE END APPENDAGE FOR LEASED AND START/STOP LINES AND NO TSO

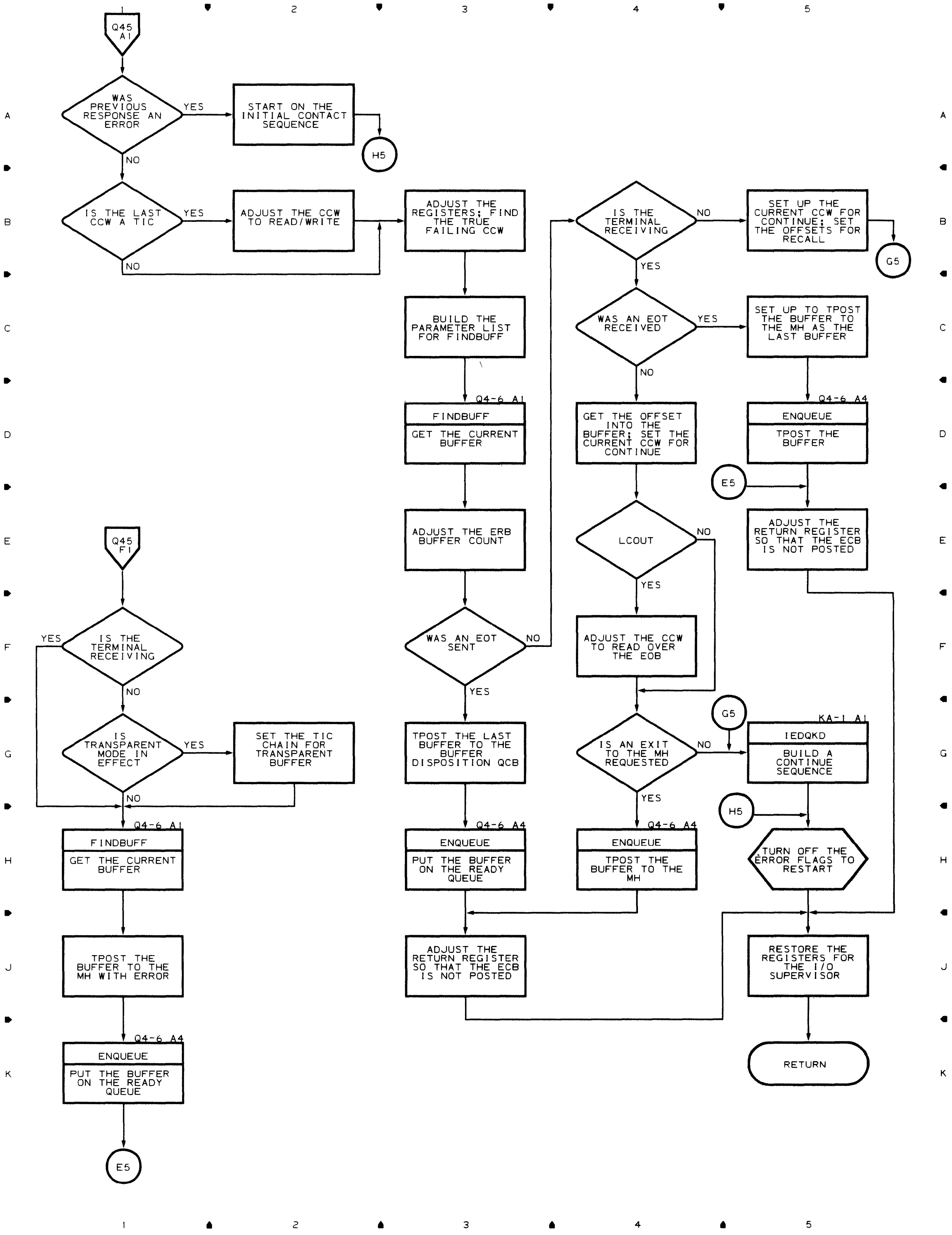

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
Q45 A1
WAS PREVIOUS RESPONSE AN ERROR
YES
START ON THE INITIAL CONTACT SEQUENCE
H5
NO
IS THE LAST CCW A TIC
YES
ADJUST THE CCW TO READ/WRITE
NO
ADJUST THE REGISTERS; FIND THE TRUE FAILING CCW
BUILD THE PARAMETER LIST FOR FINDBUFF
Q4-6 A1
FINDBUFF
GET THE CURRENT BUFFER
ADJUST THE ERB BUFFER COUNT
WAS AN EOT SENT
NO
YES
TPOST THE LAST BUFFER TO THE BUFFER DISPOSITION QCB
Q4-6 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
IS THE TERMINAL RECEIVING
NO
SET UP THE CURRENT CCW FOR CONTINUE; SET THE OFFSETS FOR RECALL
G5
YES
WAS AN EOT RECEIVED
YES
SET UP TO TPOST THE BUFFER TO THE MH AS THE LAST BUFFER
Q4-6 A4
ENQUEUE
TPOST THE BUFFER
NO
GET THE OFFSET INTO THE BUFFER; SET THE CURRENT CCW FOR CONTINUE
E5
LCOUT
NO
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
YES
ADJUST THE CCW TO READ OVER THE EOB
G5
IS AN EXIT TO THE MH REQUESTED
NO
KA-1 A1
IEDQKD
BUILD A CONTINUE SEQUENCE
YES
Q4-6 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
H5
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q45 F1
IS THE TERMINAL RECEIVING
YES
NO
IS TRANSPARENT MODE IN EFFECT
YES
SET THE TIC CHAIN FOR TRANSPARENT BUFFER
NO
Q4-6 A1
FINDBUFF
GET THE CURRENT BUFFER
TPOST THE BUFFER TO THE MH WITH ERROR
Q4-6 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
E5

Chart Q4-6 (Q46) LINE END APPENDAGE FOR LEASED AND START/STOP LINES AND NO TSO

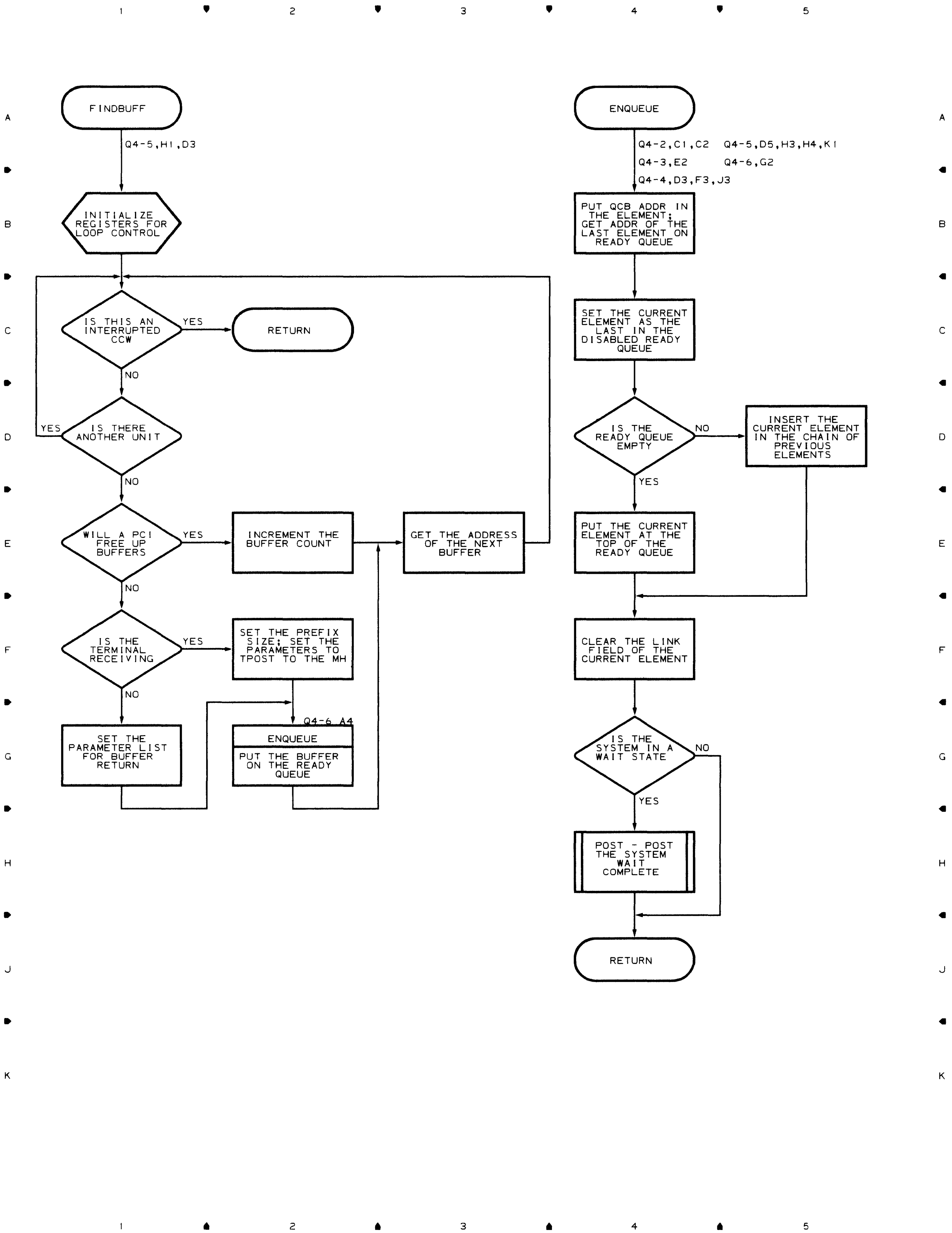

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FINDBUFF
Q4-5,H1,D3
INITIALIZE REGISTERS FOR LOOP CONTROL
IS THIS AN INTERRUPTED CCW
YES
RETURN
NO
IS THERE ANOTHER UNIT
YES
NO
WILL A PCI FREE UP BUFFERS
YES
INCREMENT THE BUFFER COUNT
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE TERMINAL RECEIVING
YES
SET THE PREFIX SIZE; SET THE PARAMETERS TO TPOST TO THE MH
NO
SET THE PARAMETER LIST FOR BUFFER RETURN
Q4-6 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
ENQUEUE
Q4-2,C1,C2 Q4-5,D5,H3,H4,K1
Q4-3,E2 Q4-6,G2
Q4-4,D3,F3,J3
PUT QCB ADDR IN THE ELEMENT; GET ADDR OF THE LAST ELEMENT ON READY QUEUE
SET THE CURRENT ELEMENT AS THE LAST IN THE DISABLED READY QUEUE
IS THE READY QUEUE EMPTY
NO
INSERT THE CURRENT ELEMENT IN THE CHAIN OF PREVIOUS ELEMENTS
YES
PUT THE CURRENT ELEMENT AT THE TOP OF THE READY QUEUE
CLEAR THE LINK FIELD OF THE CURRENT ELEMENT
IS THE SYSTEM IN A WAIT STATE
NO
YES
POST - POST THE SYSTEM WAIT COMPLETE
RETURN

Chart Q5-1 (Q51) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM

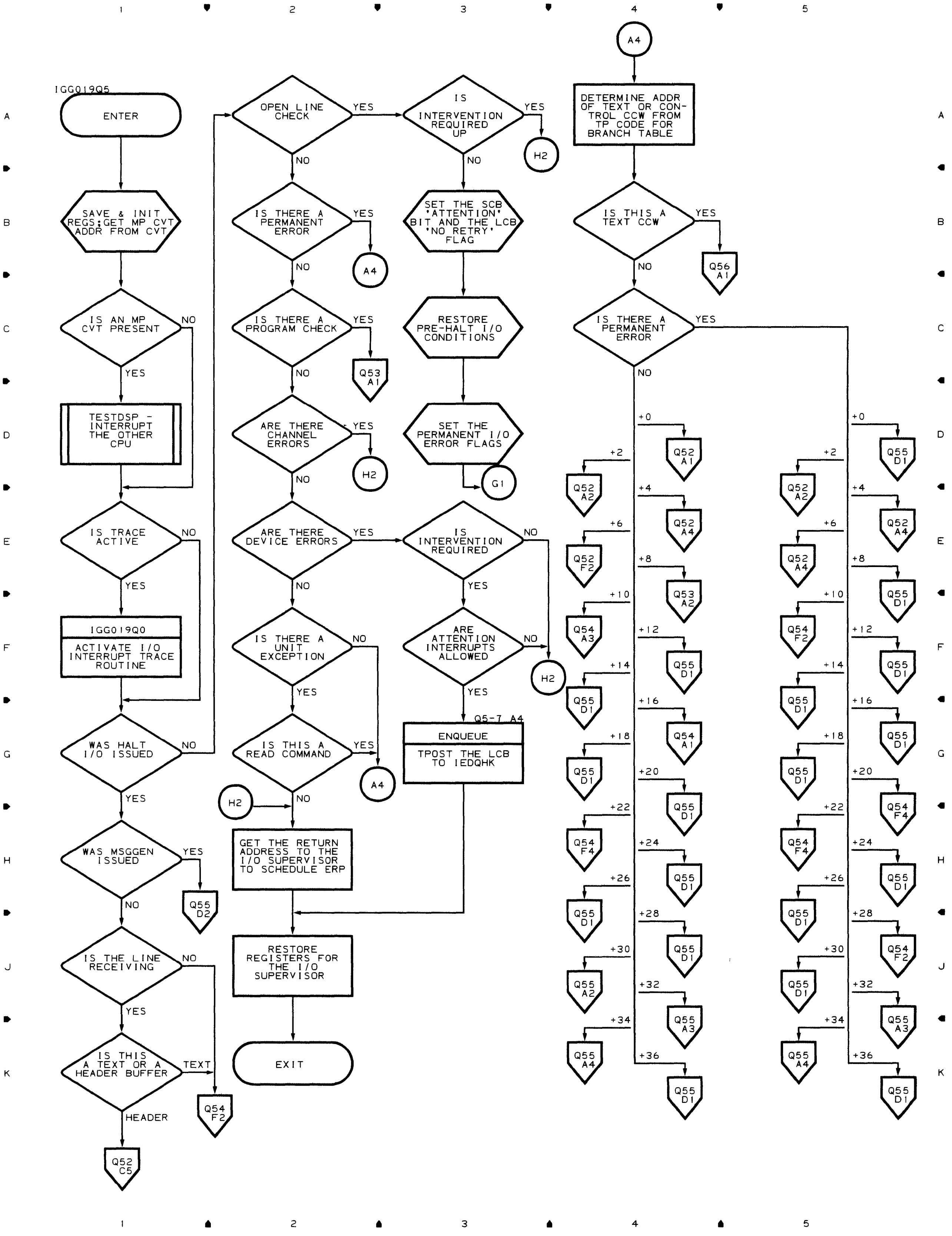


Chart Q5-2 (Q52) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM

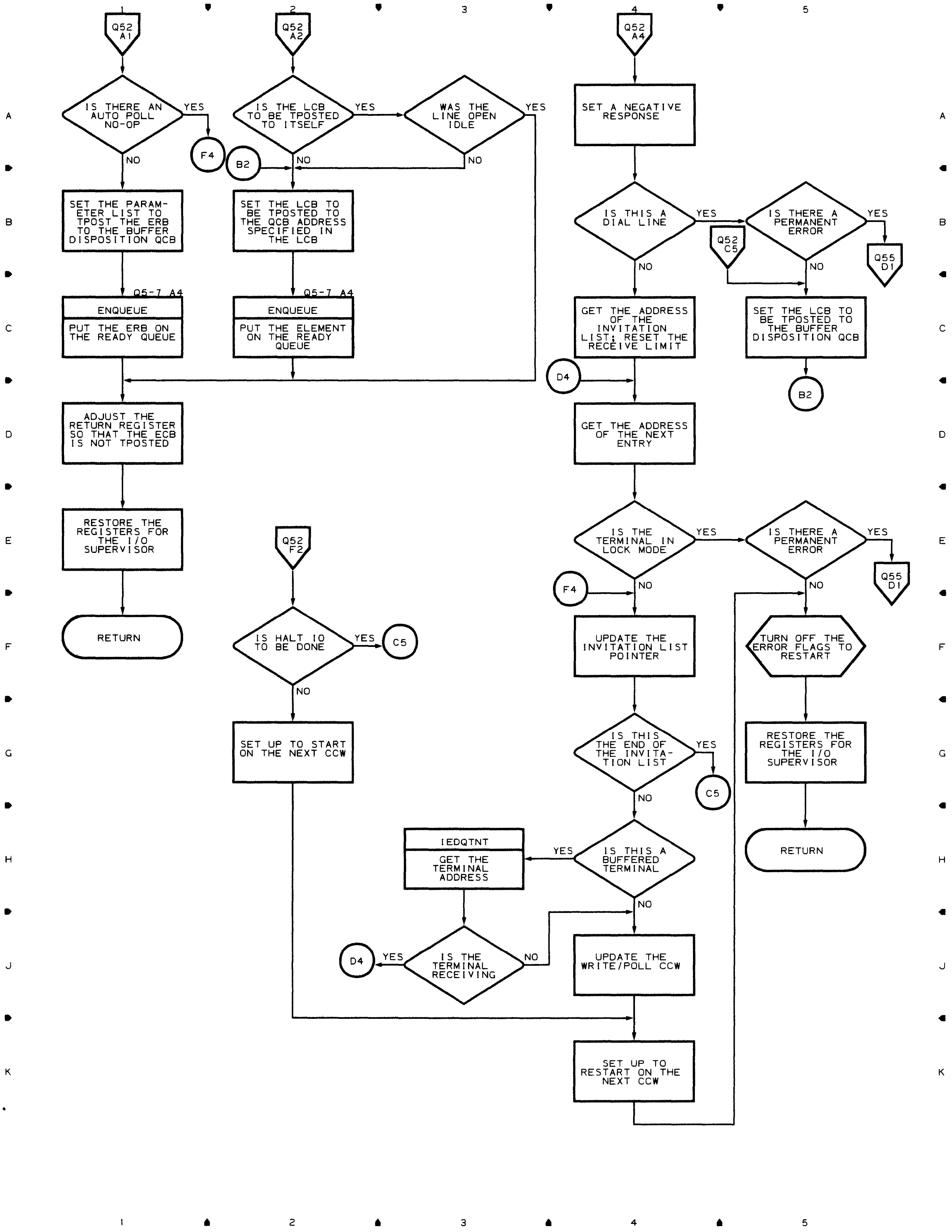

Q52 A1
IS THERE AN AUTO POLL NO-OP
YES
F4
NO
SET THE PARAMETER LIST TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q5-7 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT TPOSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q52 A2
IS THE LCB TO BE TPOSTED TO ITSELF
YES
NO
B2
SET THE LCB TO BE TPOSTED TO THE QCB ADDRESS SPECIFIED IN THE LCB
Q5-7 A4
ENQUEUE
PUT THE ELEMENT ON THE READY QUEUE
WAS THE LINE OPEN IDLE
YES
NO
Q52 F2
IS HALT IO TO BE DONE
YES
C5
NO
SET UP TO START ON THE NEXT CCW
IEDQTNT
GET THE TERMINAL ADDRESS
IS THE TERMINAL RECEIVING
YES
D4
NO
Q52 A4
SET A NEGATIVE RESPONSE
IS THIS A DIAL LINE
YES
NO
Q52 C5
GET THE ADDRESS OF THE INVITATION LIST; RESET THE RECEIVE LIMIT
D4
GET THE ADDRESS OF THE NEXT ENTRY
IS THE TERMINAL IN LOCK MODE
YES
NO
F4
UPDATE THE INVITATION LIST POINTER
IS THIS THE END OF THE INVITATION LIST
YES
C5
NO
IS THIS A BUFFERED TERMINAL
YES
NO
UPDATE THE WRITE/POLL CCW
SET UP TO RESTART ON THE NEXT CCW
IS THERE A PERMANENT ERROR
YES
Q55 D1
NO
SET THE LCB TO BE TPOSTED TO THE BUFFER DISPOSITION QCB
B2
IS THERE A PERMANENT ERROR
YES
Q55 D1
NO
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q5-3 (Q53) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM

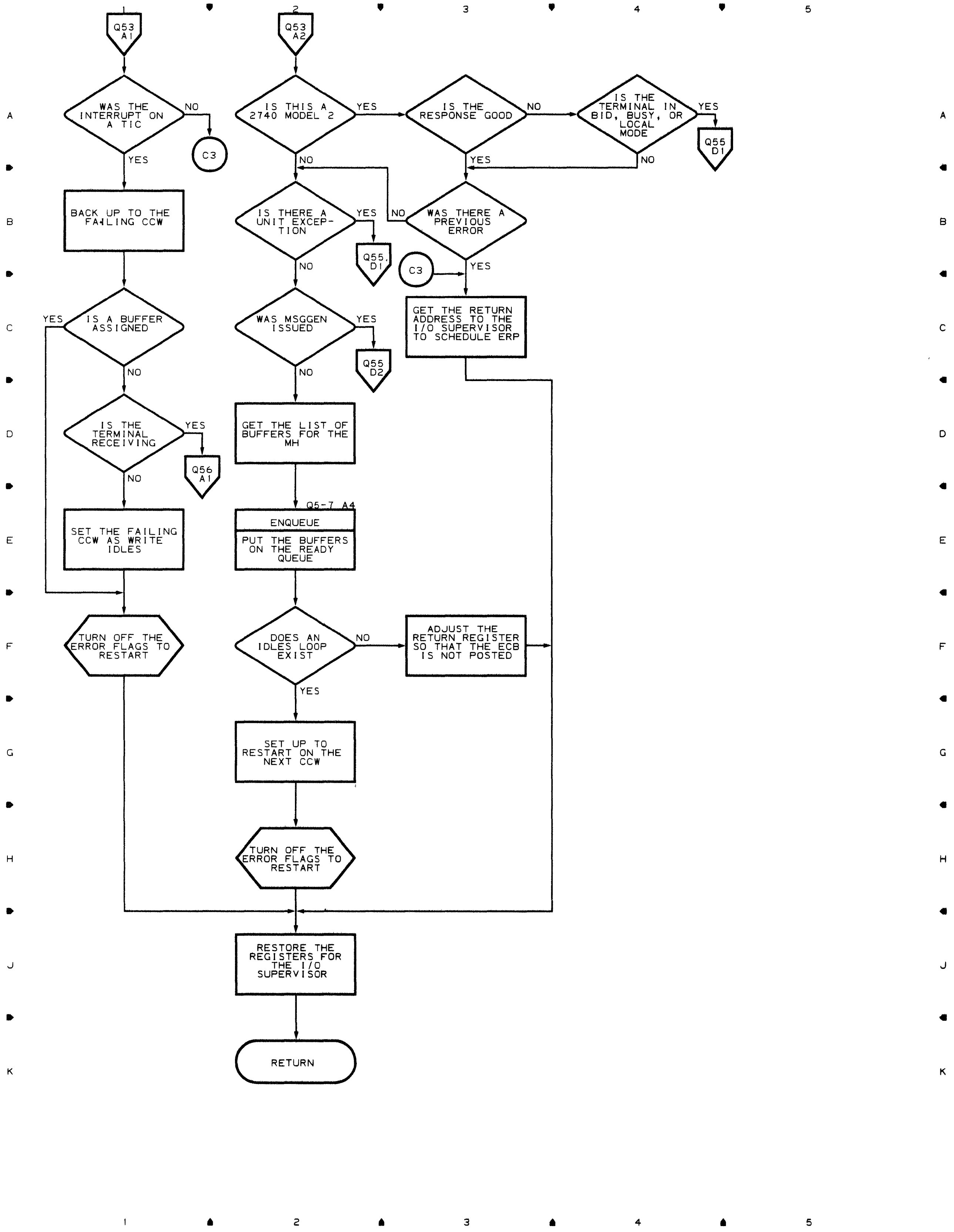

1
2
3
4
5
Q53 A1
WAS THE INTERRUPT ON A TIC
NO
C3
YES
BACK UP TO THE FAILING CCW
IS A BUFFER ASSIGNED
YES
NO
IS THE TERMINAL RECEIVING
YES
Q56 A1
NO
SET THE FAILING CCW AS WRITE IDLES
TURN OFF THE ERROR FLAGS TO RESTART
Q53 A2
IS THIS A 2740 MODEL 2
YES
NO
IS THERE A UNIT EXCEPTION
YES
Q55 D1
NO
WAS MSGGEN ISSUED
YES
Q55 D2
NO
GET THE LIST OF BUFFERS FOR THE MH
Q5-7 A4
ENQUEUE
PUT THE BUFFERS ON THE READY QUEUE
DOES AN IDLES LOOP EXIST
NO
YES
SET UP TO RESTART ON THE NEXT CCW
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
IS THE RESPONSE GOOD
NO
YES
IS THE TERMINAL IN BID, BUSY, OR LOCAL MODE
YES
Q55 D1
NO
WAS THERE A PREVIOUS ERROR
NO
YES
C3
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
A
B
C
D
E
F
G
H
J
K

Chart Q5-4 (Q54) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM

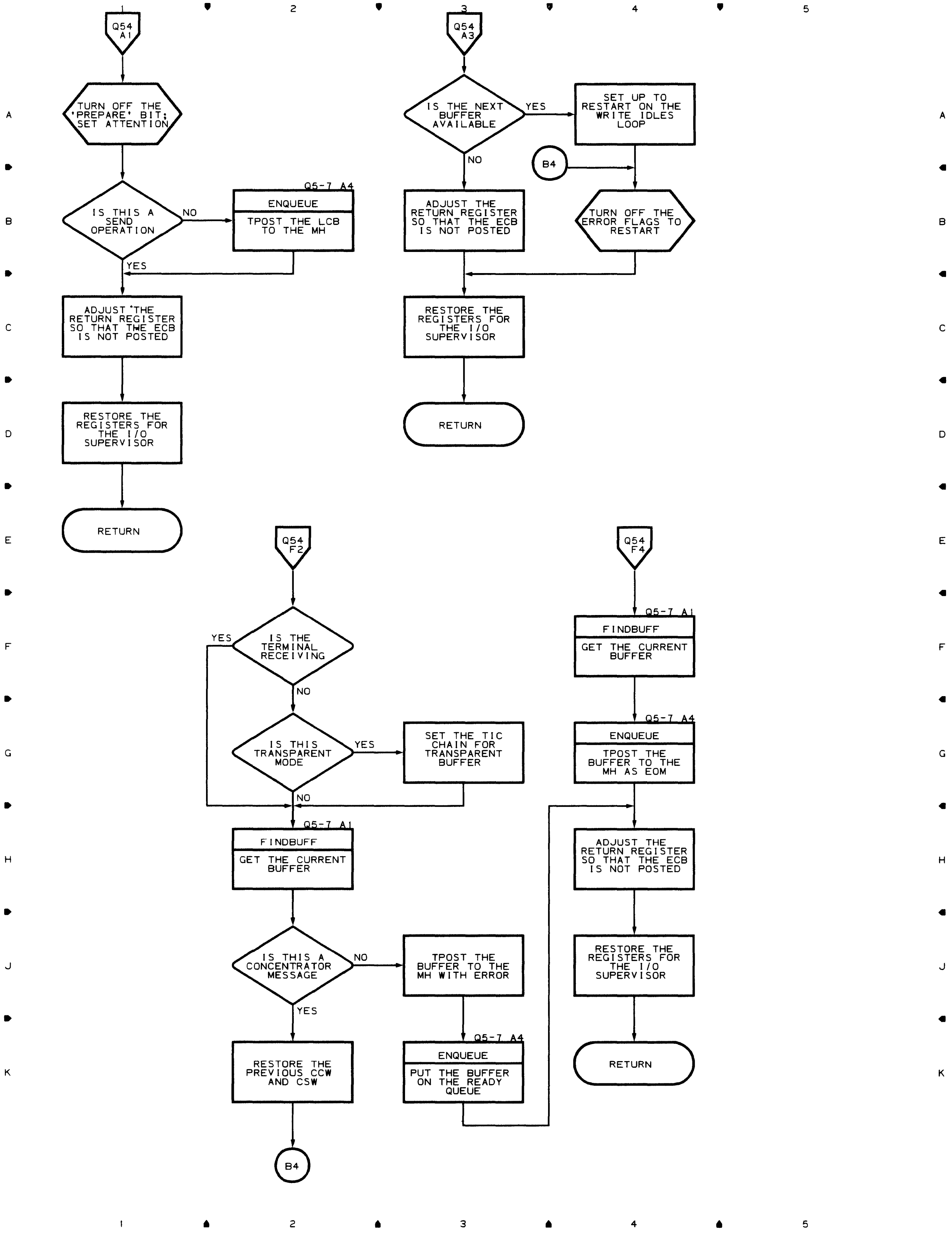

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
Q54 A1
TURN OFF THE 'PREPARE' BIT; SET ATTENTION
IS THIS A SEND OPERATION
NO
YES
Q5-7 A4
ENQUEUE
TPOST THE LCB TO THE MH
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q54 A3
IS THE NEXT BUFFER AVAILABLE
YES
NO
SET UP TO RESTART ON THE WRITE IDLES LOOP
B4
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
Q54 F2
IS THE TERMINAL RECEIVING
YES
NO
IS THIS TRANSPARENT MODE
YES
NO
SET THE TIC CHAIN FOR TRANSPARENT BUFFER
Q5-7 A1
FINDBUFF
GET THE CURRENT BUFFER
IS THIS A CONCENTRATOR MESSAGE
NO
YES
TPOST THE BUFFER TO THE MH WITH ERROR
RESTORE THE PREVIOUS CCW AND CSW
Q5-7 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
B4
Q54 F4
Q5-7 A1
FINDBUFF
GET THE CURRENT BUFFER
Q5-7 A4
ENQUEUE
TPOST THE BUFFER TO THE MH AS EOM
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN

Chart Q5-5 (Q55) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM

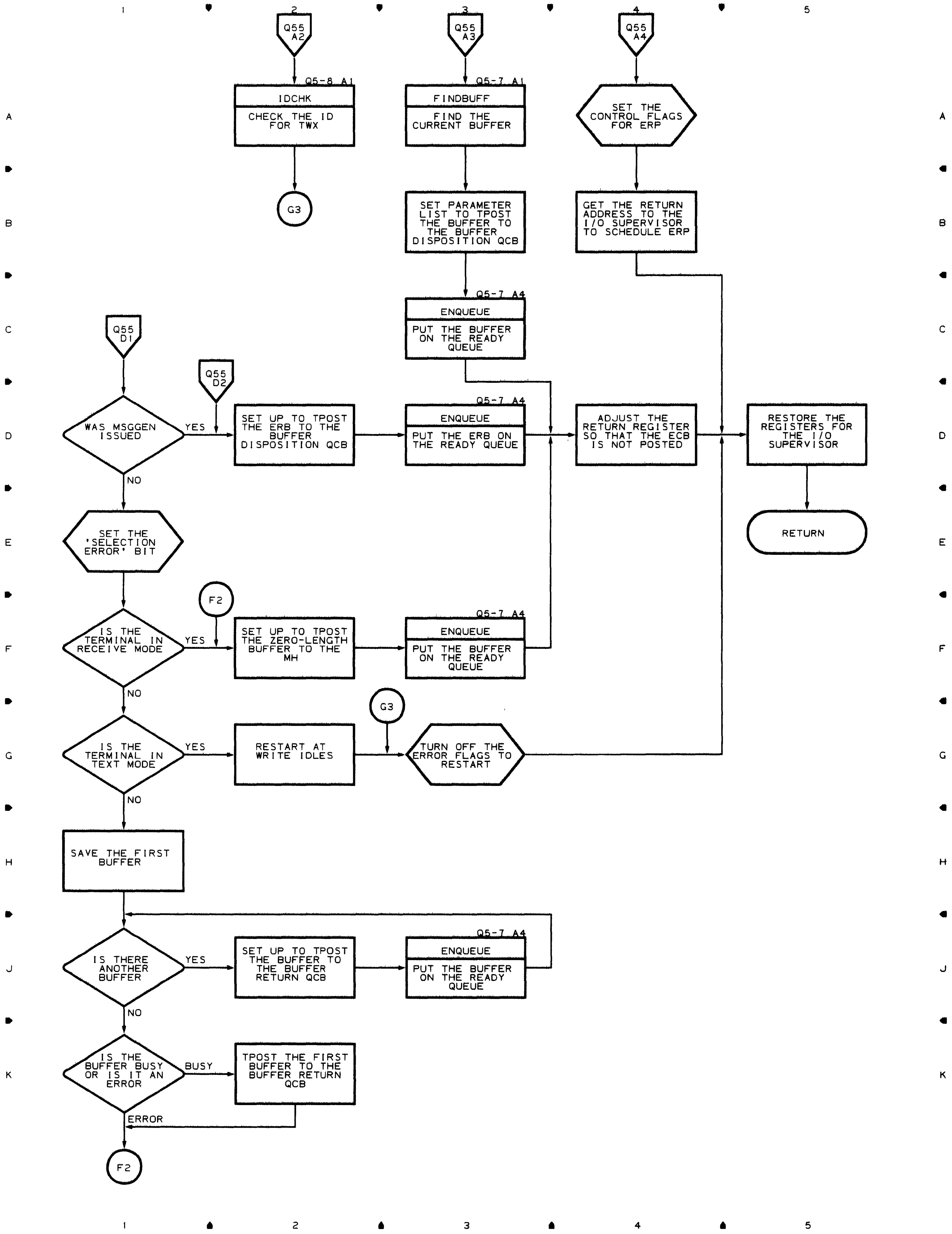
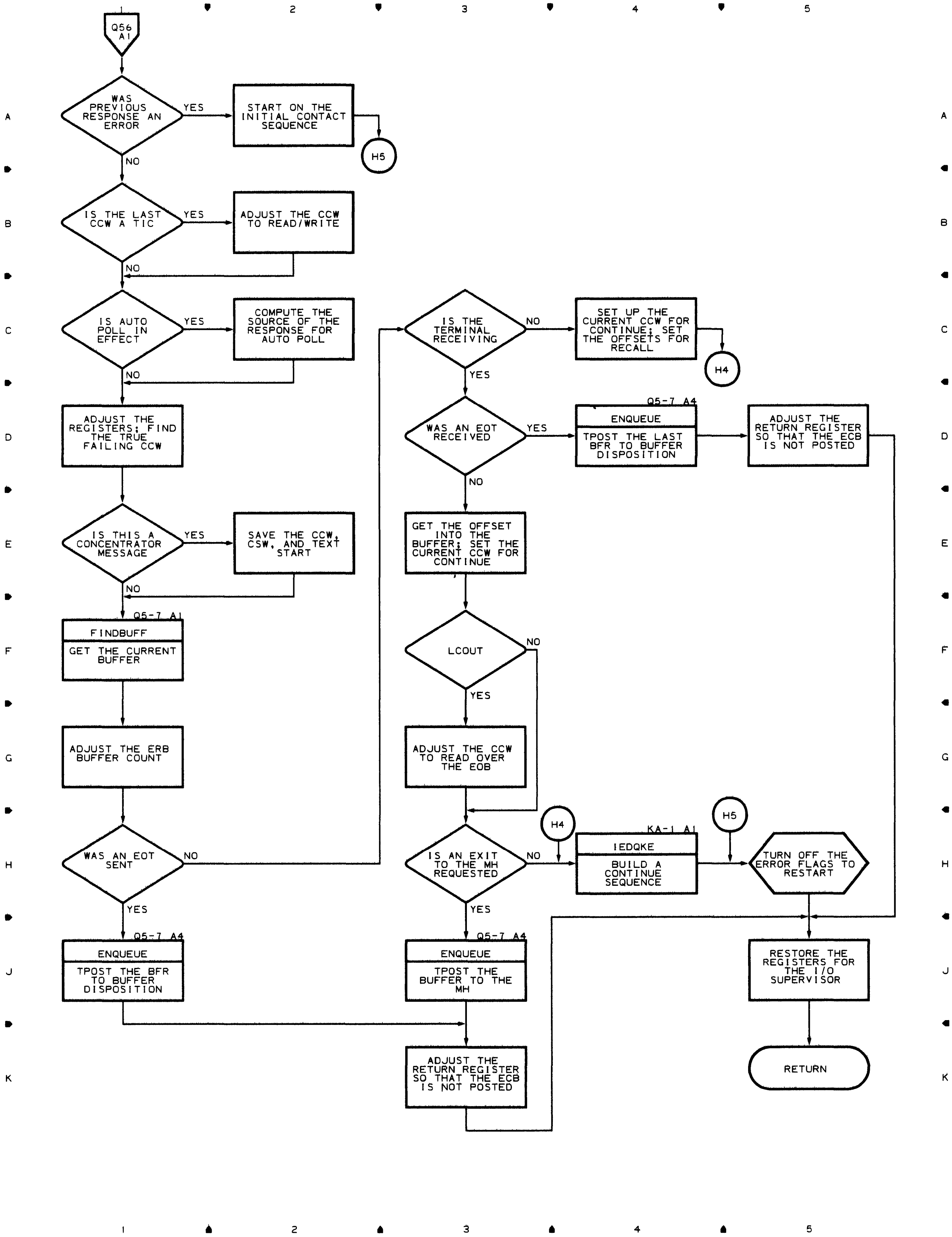

Q55 A2
Q5-8 A1
IDCHK
CHECK THE ID FOR TWX
G3
Q55 A3
Q5-7 A1
FINDBUFF
FIND THE CURRENT BUFFER
SET PARAMETER LIST TO TPOST THE BUFFER TO THE BUFFER DISPOSITION QCB
Q5-7 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
Q55 A4
SET THE CONTROL FLAGS FOR ERP
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
Q55 D1
WAS MSGGEN ISSUED
YES
Q55 D2
SET UP TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
Q5-7 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
NO
SET THE 'SELECTION ERROR' BIT
F2
IS THE TERMINAL IN RECEIVE MODE
YES
SET UP TO TPOST THE ZERO-LENGTH BUFFER TO THE MH
Q5-7 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
NO
G3
IS THE TERMINAL IN TEXT MODE
YES
RESTART AT WRITE IDLES
TURN OFF THE ERROR FLAGS TO RESTART
NO
SAVE THE FIRST BUFFER
IS THERE ANOTHER BUFFER
YES
SET UP TO TPOST THE BUFFER TO THE BUFFER RETURN QCB
Q5-7 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
NO
IS THE BUFFER BUSY OR IS IT AN ERROR
BUSY
TPOST THE FIRST BUFFER TO THE BUFFER RETURN QCB
ERROR
F2
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q5-6 (Q56) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
Q56 A1
WAS PREVIOUS RESPONSE AN ERROR
YES
NO
START ON THE INITIAL CONTACT SEQUENCE
H5
IS THE LAST CCW A TIC
ADJUST THE CCW TO READ/WRITE
IS AUTO POLL IN EFFECT
COMPUTE THE SOURCE OF THE RESPONSE FOR AUTO POLL
ADJUST THE REGISTERS; FIND THE TRUE FAILING CCW
IS THIS A CONCENTRATOR MESSAGE
SAVE THE CCW, CSW, AND TEXT START
Q5-7 A1
FINDBUFF
GET THE CURRENT BUFFER
ADJUST THE ERB BUFFER COUNT
WAS AN EOT SENT
Q5-7 A4
ENQUEUE
TPOST THE BFR TO BUFFER DISPOSITION
IS THE TERMINAL RECEIVING
SET UP THE CURRENT CCW FOR CONTINUE; SET THE OFFSETS FOR RECALL
H4
WAS AN EOT RECEIVED
Q5-7 A4
ENQUEUE
TPOST THE LAST BFR TO BUFFER DISPOSITION
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
GET THE OFFSET INTO THE BUFFER; SET THE CURRENT CCW FOR CONTINUE
LCOUT
ADJUST THE CCW TO READ OVER THE EOB
IS AN EXIT TO THE MH REQUESTED
KA-1 A1
IEDQKE
BUILD A CONTINUE SEQUENCE
TURN OFF THE ERROR FLAGS TO RESTART
Q5-7 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN

Chart Q5-7 (Q57) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM

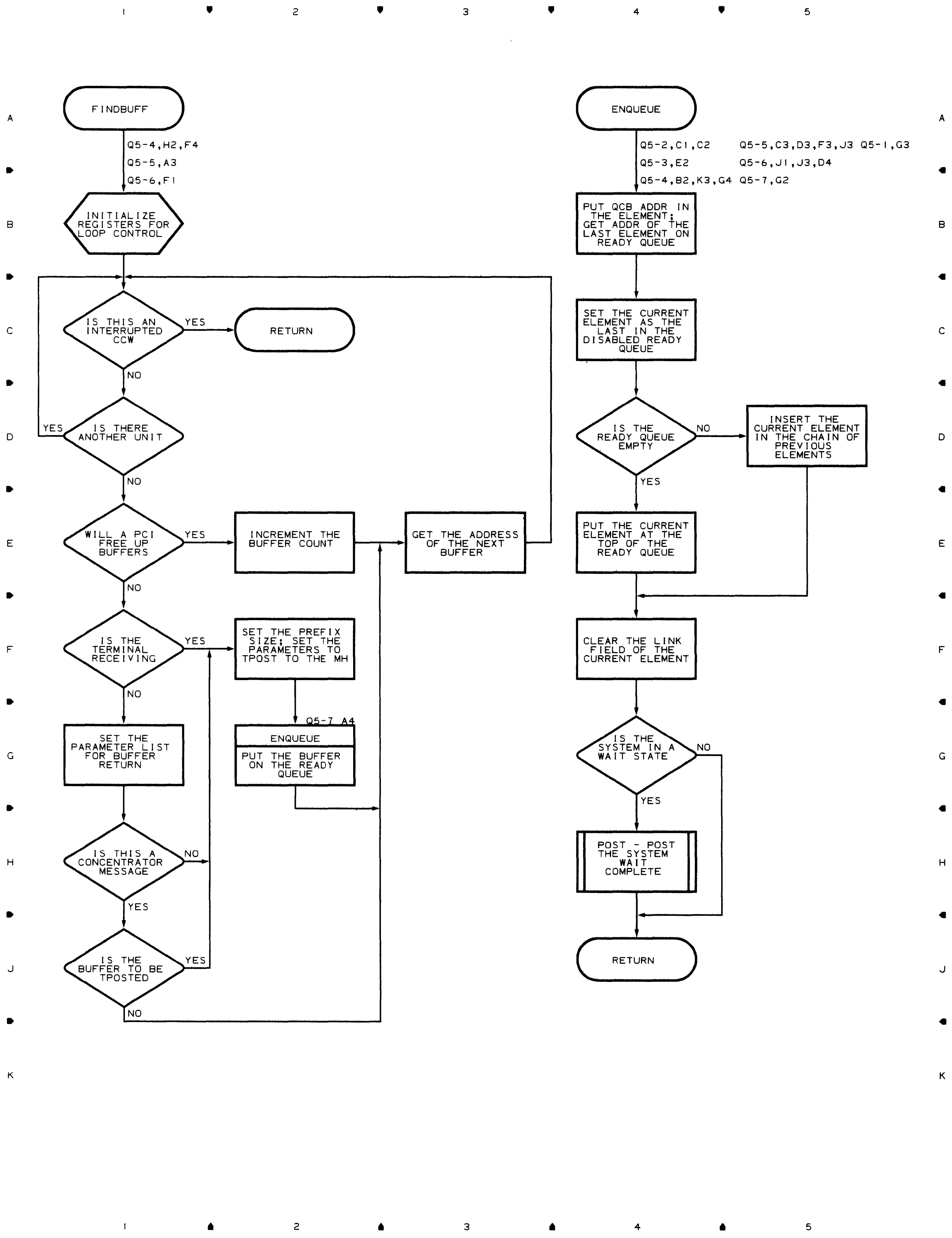
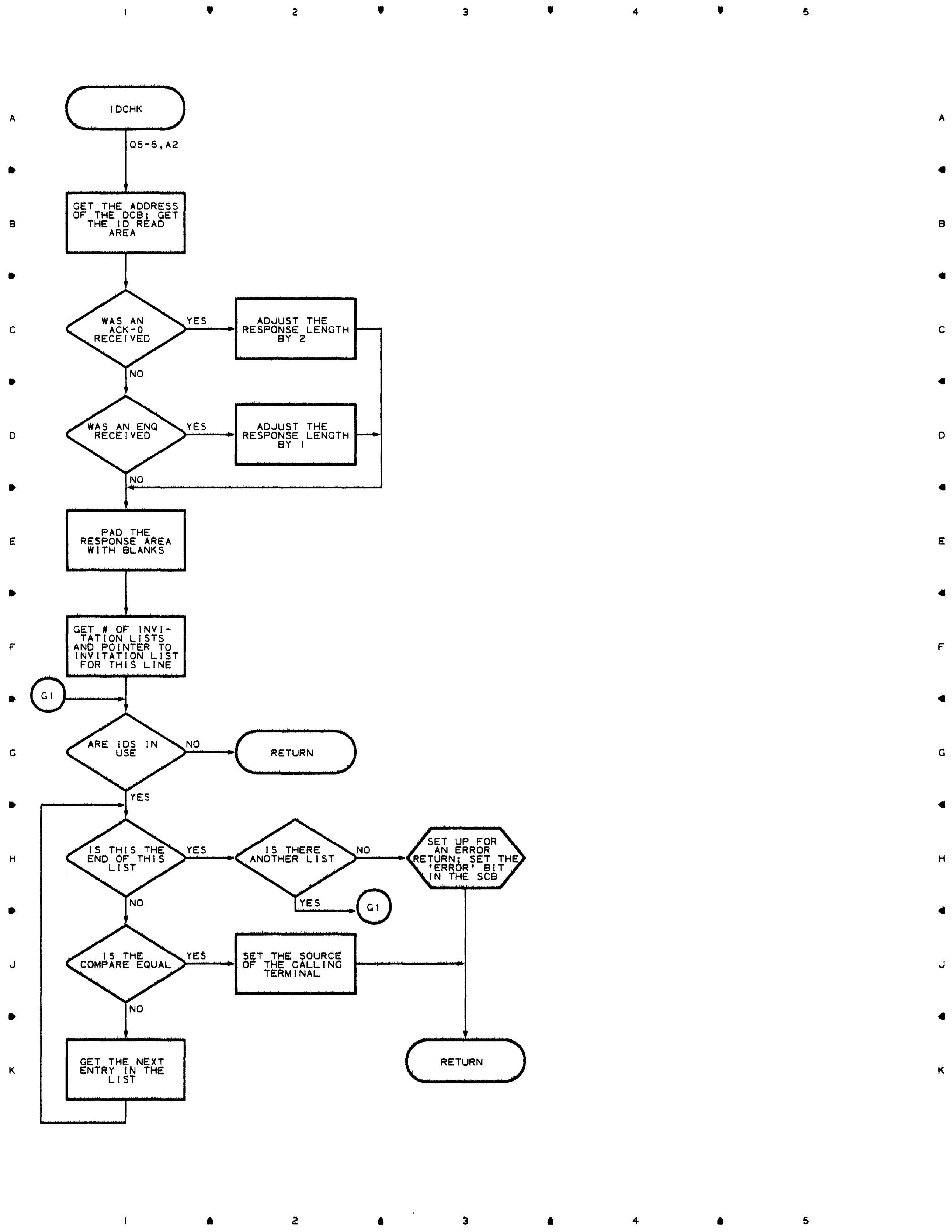

FINDBUFF
Q5-4,H2,F4
Q5-5,A3
Q5-6,F1
INITIALIZE REGISTERS FOR LOOP CONTROL
IS THIS AN INTERRUPTED CCW
YES
RETURN
NO
IS THERE ANOTHER UNIT
YES
NO
WILL A PCI FREE UP BUFFERS
YES
INCREMENT THE BUFFER COUNT
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE TERMINAL RECEIVING
YES
SET THE PREFIX SIZE; SET THE PARAMETERS TO TPOST TO THE MH
NO
SET THE PARAMETER LIST FOR BUFFER RETURN
Q5-7 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
IS THIS A CONCENTRATOR MESSAGE
NO
YES
IS THE BUFFER TO BE TPOSTED
YES
NO
ENQUEUE
Q5-2,C1,C2
Q5-3,E2
Q5-4,B2,K3,G4
Q5-5,C3,D3,F3,J3
Q5-6,J1,J3,D4
Q5-7,G2
Q5-1,G3
PUT QCB ADDR IN THE ELEMENT; GET ADDR OF THE LAST ELEMENT ON READY QUEUE
SET THE CURRENT ELEMENT AS THE LAST IN THE DISABLED READY QUEUE
IS THE READY QUEUE EMPTY
NO
INSERT THE CURRENT ELEMENT IN THE CHAIN OF PREVIOUS ELEMENTS
YES
PUT THE CURRENT ELEMENT AT THE TOP OF THE READY QUEUE
CLEAR THE LINK FIELD OF THE CURRENT ELEMENT
IS THE SYSTEM IN A WAIT STATE
NO
YES
POST - POST THE SYSTEM WAIT COMPLETE
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart Q5-8 (Q58) LINE END APPENDAGE FOR A QTAM-COMPATIBLE SYSTEM



IDCHK
Q5-5,A2
GET THE ADDRESS OF THE DCB; GET THE ID READ AREA
WAS AN ACK-0 RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 2
NO
WAS AN ENQ RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 1
NO
PAD THE RESPONSE AREA WITH BLANKS
GET # OF INVITATION LISTS AND POINTER TO INVITATION LIST FOR THIS LINE
G1
ARE IDS IN USE
NO
RETURN
YES
IS THIS THE END OF THIS LIST
YES
IS THERE ANOTHER LIST
NO
SET UP FOR AN ERROR RETURN; SET THE 'ERROR' BIT IN THE SCB
YES
G1
NO
IS THE COMPARE EQUAL
YES
SET THE SOURCE OF THE CALLING TERMINAL
NO
GET THE NEXT ENTRY IN THE LIST
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart R0-1 (R01) LINE END APPENDAGE

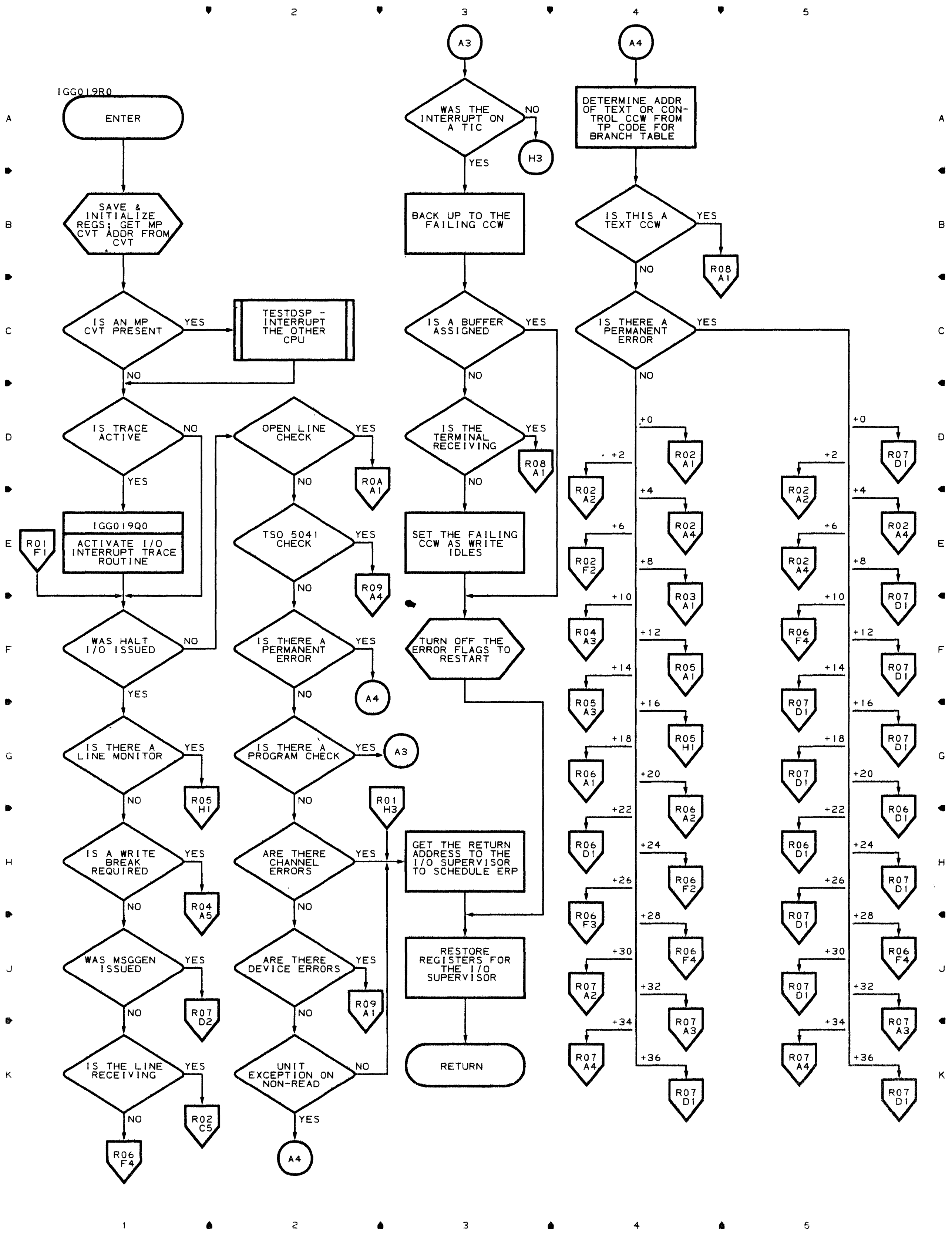
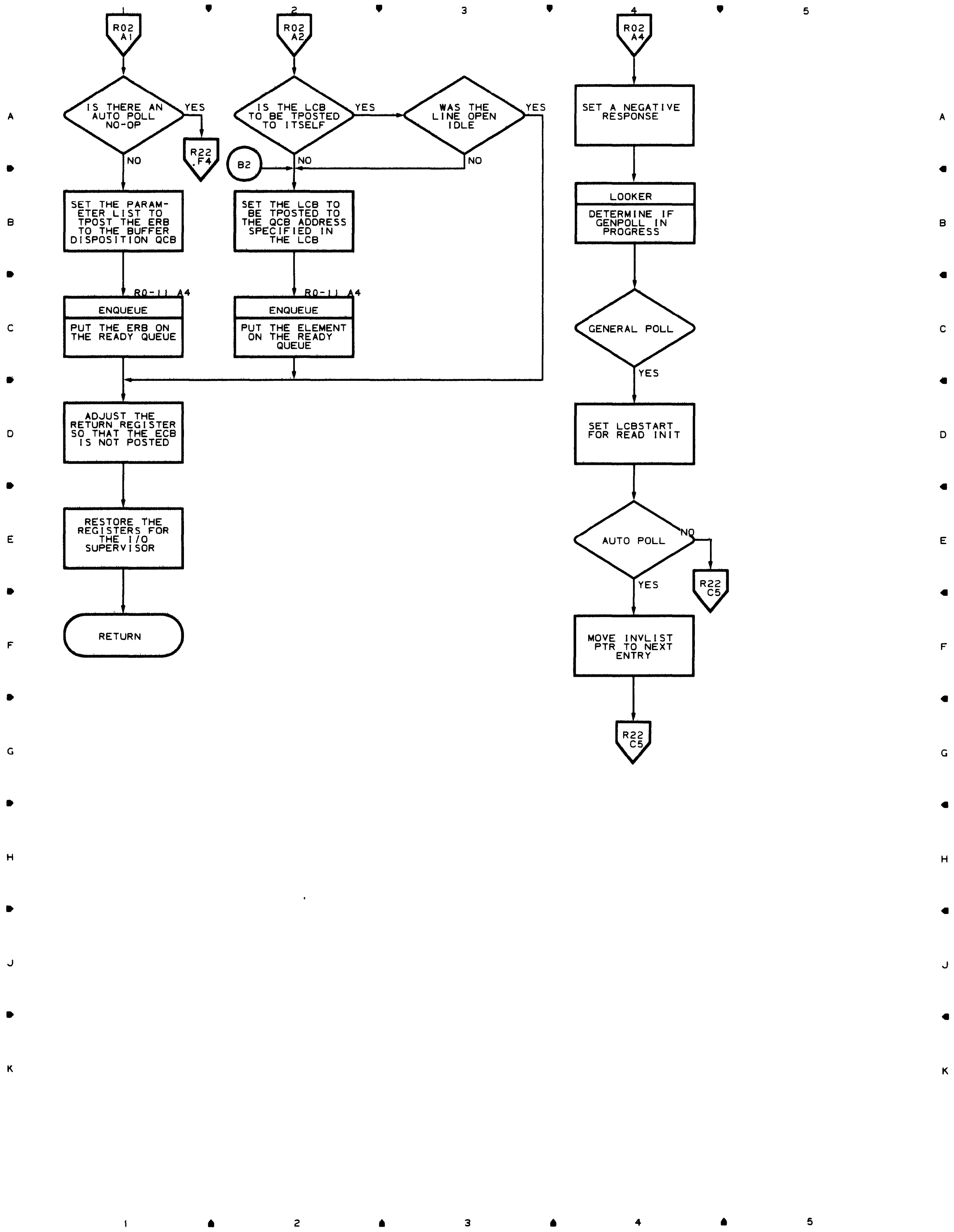


Chart R0-2-1 (R02) LINE END APPENDAGE



R02 A1
IS THERE AN AUTO POLL NO-OP
YES
R22 .F4
NO
SET THE PARAMETER LIST TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
R0-11 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
R02 A2
IS THE LCB TO BE TPOSTED TO ITSELF
YES
NO
B2
SET THE LCB TO BE TPOSTED TO THE QCB ADDRESS SPECIFIED IN THE LCB
R0-11 A4
ENQUEUE
PUT THE ELEMENT ON THE READY QUEUE
WAS THE LINE OPEN IDLE
YES
NO
R02 A4
SET A NEGATIVE RESPONSE
LOOKER
DETERMINE IF GENPOLL IN PROGRESS
GENERAL POLL
YES
SET LCBSTART FOR READ INIT
AUTO POLL
NO
R22 C5
YES
MOVE INVLIST PTR TO NEXT ENTRY
R22 C5
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart R0-2-2 (R02) LINE END APPENDAGE

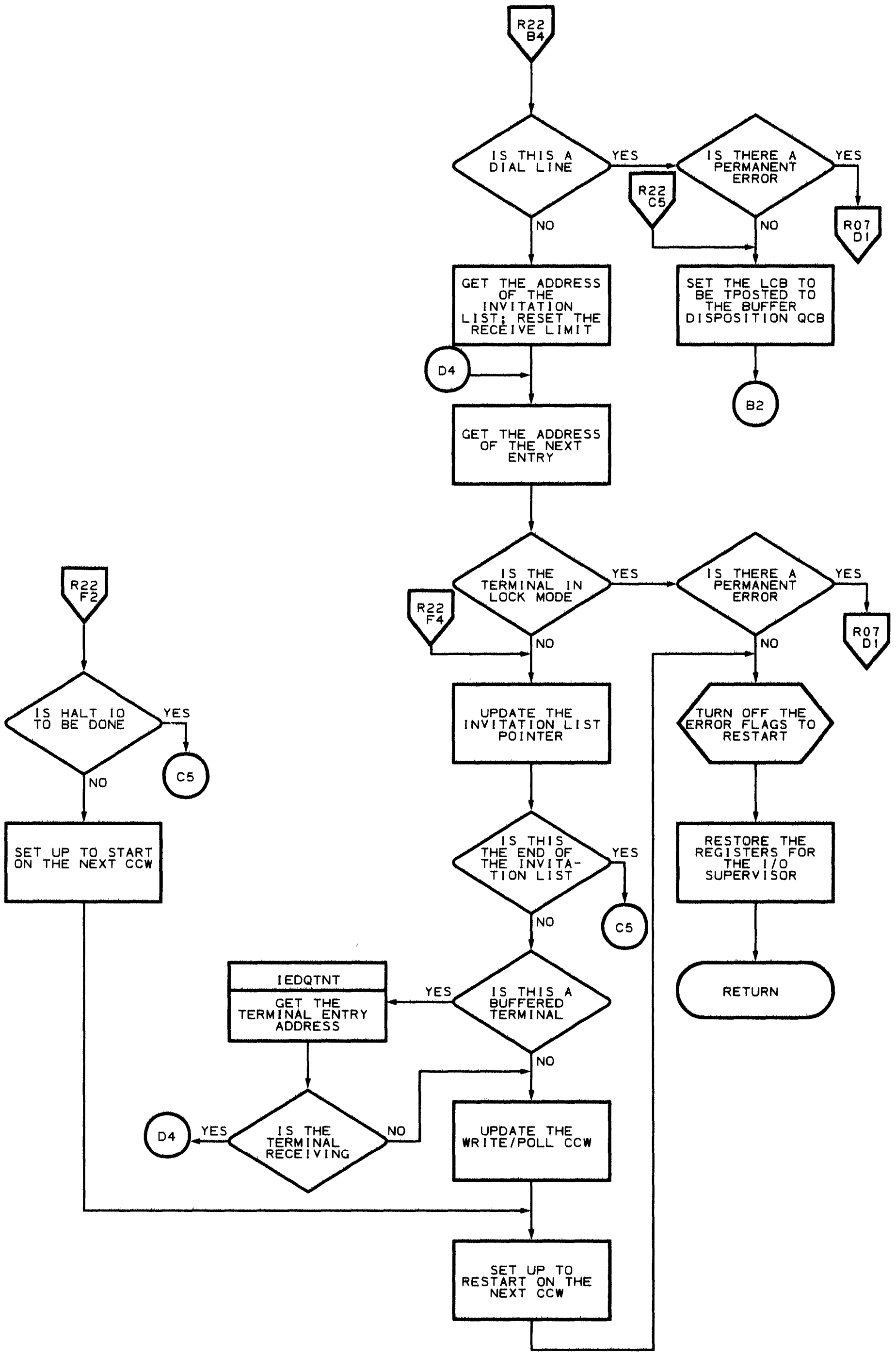

R22 B4
IS THIS A DIAL LINE
YES
NO
IS THERE A PERMANENT ERROR
YES
R07 D1
R22 C5
NO
SET THE LCB TO BE TPOSTED TO THE BUFFER DISPOSITION QCB
B2
GET THE ADDRESS OF THE INVITATION LIST; RESET THE RECEIVE LIMIT
D4
GET THE ADDRESS OF THE NEXT ENTRY
IS THE TERMINAL IN LOCK MODE
YES
IS THERE A PERMANENT ERROR
YES
R07 D1
R22 F4
NO
NO
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
UPDATE THE INVITATION LIST POINTER
IS THIS THE END OF THE INVITA-TION LIST
YES
C5
NO
IS THIS A BUFFERED TERMINAL
YES
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
NO
IS THE TERMINAL RECEIVING
YES
D4
NO
UPDATE THE WRITE/POLL CCW
R22 F2
IS HALT IO TO BE DONE
YES
C5
NO
SET UP TO START ON THE NEXT CCW
SET UP TO RESTART ON THE NEXT CCW

Chart R0-3 (R03) LINE END APPENDAGE

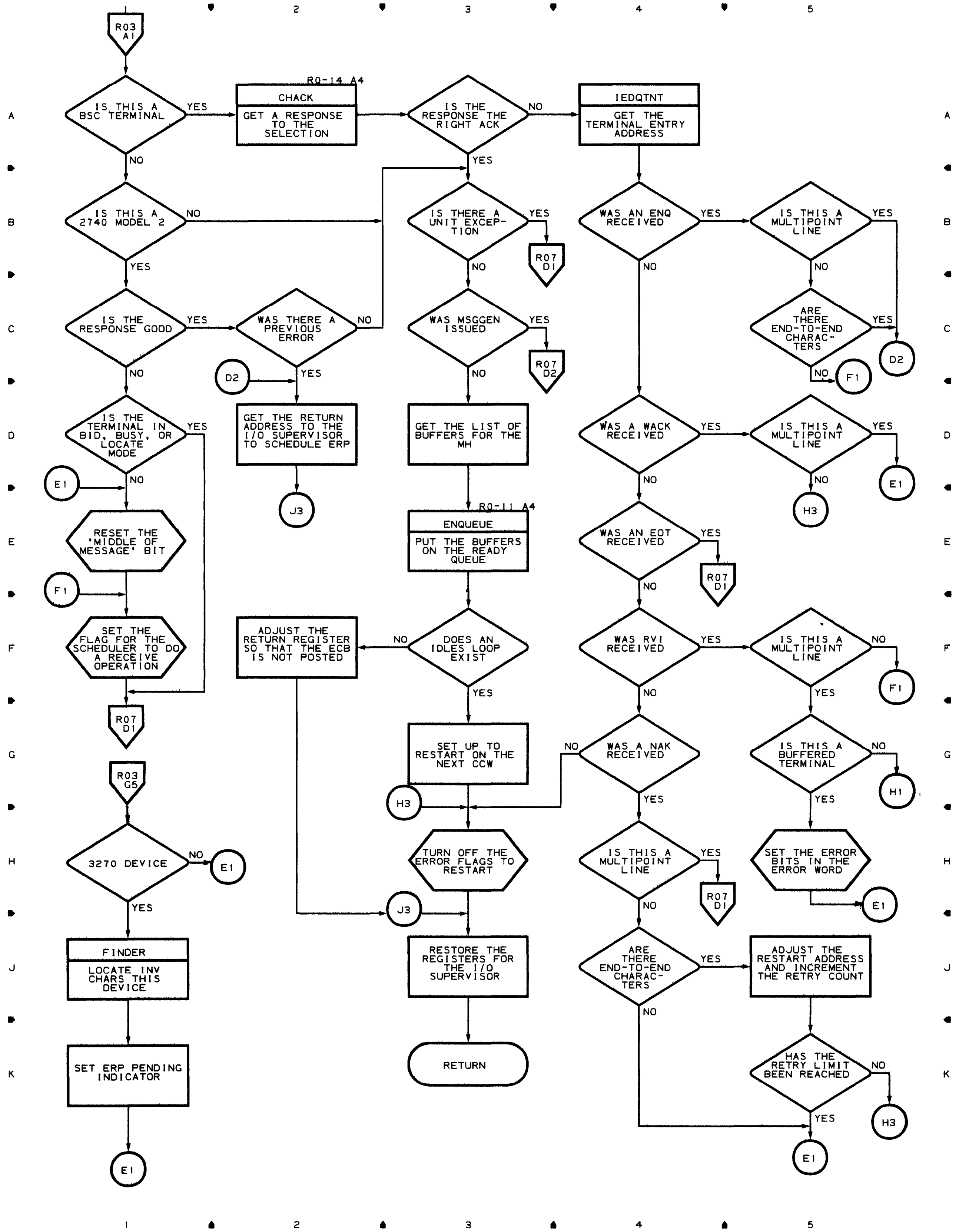

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
R03 A1
IS THIS A BSC TERMINAL
YES
NO
IS THIS A 2740 MODEL 2
NO
YES
IS THE RESPONSE GOOD
YES
NO
IS THE TERMINAL IN BID, BUSY, OR LOCATE MODE
YES
NO
E1
RESET THE 'MIDDLE OF MESSAGE' BIT
F1
SET THE FLAG FOR THE SCHEDULER TO DO A RECEIVE OPERATION
R07 D1
R03 G5
3270 DEVICE
NO
E1
YES
FINDER
LOCATE INV CHARS THIS DEVICE
SET ERP PENDING INDICATOR
E1
R0-14 A4
CHACK
GET A RESPONSE TO THE SELECTION
WAS THERE A PREVIOUS ERROR
NO
YES
D2
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
J3
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
IS THE RESPONSE THE RIGHT ACK
NO
YES
IS THERE A UNIT EXCEPTION
YES
R07 D1
NO
WAS MSGGEN ISSUED
YES
R07 D2
NO
GET THE LIST OF BUFFERS FOR THE MH
R0-11 A4
ENQUEUE
PUT THE BUFFERS ON THE READY QUEUE
DOES AN IDLES LOOP EXIST
NO
YES
SET UP TO RESTART ON THE NEXT CCW
H3
TURN OFF THE ERROR FLAGS TO RESTART
J3
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
IEDQTNT
GET THE TERMINAL ENTRY ADDRESS
WAS AN ENQ RECEIVED
YES
NO
WAS A WACK RECEIVED
YES
NO
WAS AN EOT RECEIVED
YES
R07 D1
NO
WAS RVI RECEIVED
YES
NO
WAS A NAK RECEIVED
NO
YES
IS THIS A MULTIPOINT LINE
YES
R07 D1
NO
ARE THERE END-TO-END CHARAC-TERS
YES
NO
IS THIS A MULTIPOINT LINE
YES
NO
ARE THERE END-TO-END CHARAC-TERS
YES
D2
NO
F1
IS THIS A MULTIPOINT LINE
YES
E1
NO
H3
IS THIS A MULTIPOINT LINE
NO
F1
YES
IS THIS A BUFFERED TERMINAL
NO
H1
YES
SET THE ERROR BITS IN THE ERROR WORD
E1
ADJUST THE RESTART ADDRESS AND INCREMENT THE RETRY COUNT
HAS THE RETRY LIMIT BEEN REACHED
NO
H3
YES
E1

Chart R0-4 (R04) LINE END APPENDAGE

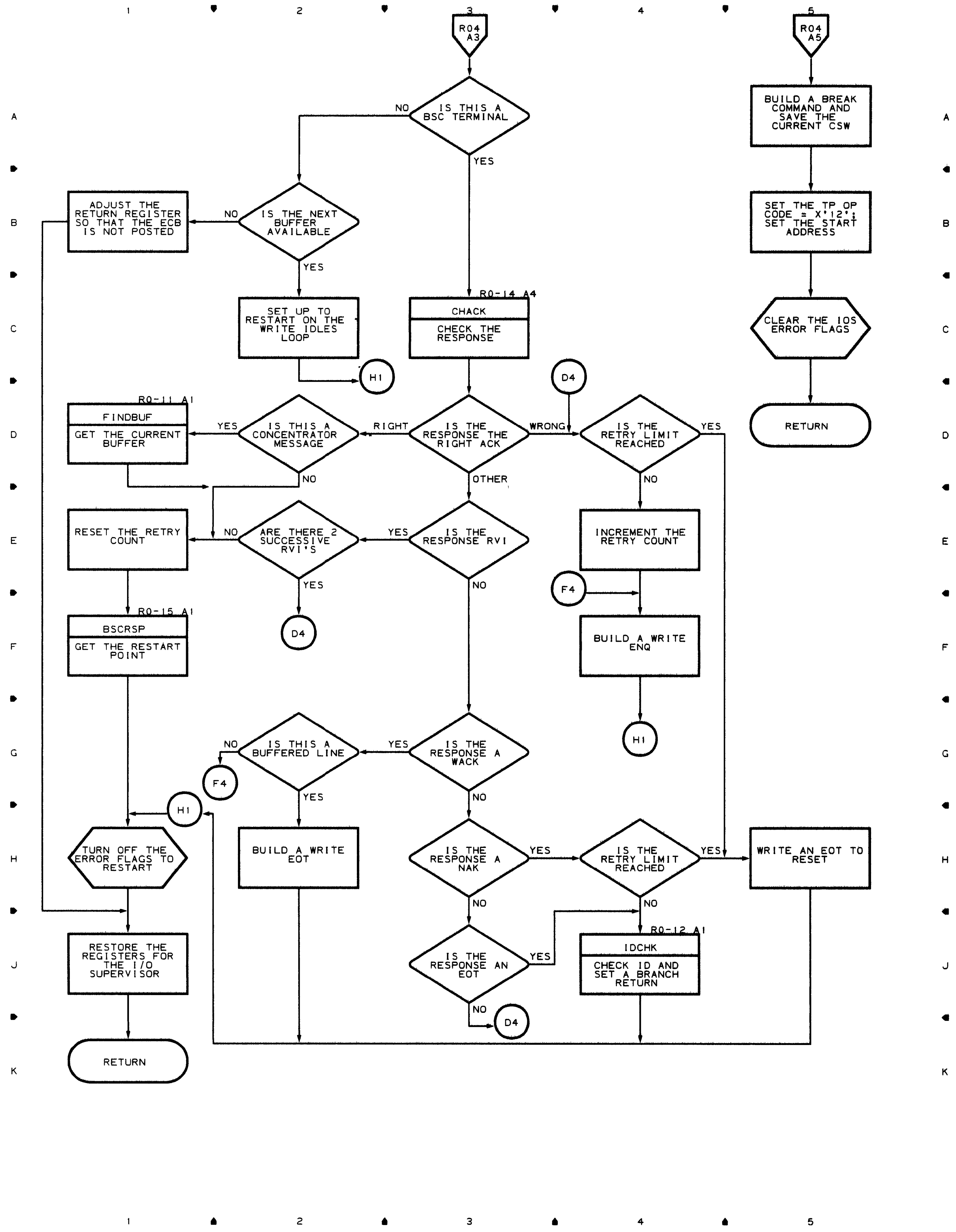

1
2
3
4
5
R04 A3
R04 A5
IS THIS A BSC TERMINAL
NO
YES
BUILD A BREAK COMMAND AND SAVE THE CURRENT CSW
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
IS THE NEXT BUFFER AVAILABLE
SET THE TP OP CODE = X'12'; SET THE START ADDRESS
SET UP TO RESTART ON THE WRITE IDLES LOOP
R0-14 A4
CHACK
CHECK THE RESPONSE
CLEAR THE IOS ERROR FLAGS
H1
D4
R0-11 A1
FINDBUF
GET THE CURRENT BUFFER
IS THIS A CONCENTRATOR MESSAGE
RIGHT
IS THE RESPONSE THE RIGHT ACK
WRONG
IS THE RETRY LIMIT REACHED
RETURN
OTHER
RESET THE RETRY COUNT
ARE THERE 2 SUCCESSIVE RVI'S
IS THE RESPONSE RVI
INCREMENT THE RETRY COUNT
F4
R0-15 A1
BSCRSP
GET THE RESTART POINT
BUILD A WRITE ENQ
IS THIS A BUFFERED LINE
IS THE RESPONSE A WACK
TURN OFF THE ERROR FLAGS TO RESTART
BUILD A WRITE EOT
IS THE RESPONSE A NAK
WRITE AN EOT TO RESET
R0-12 A1
IDCHK
CHECK ID AND SET A BRANCH RETURN
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
IS THE RESPONSE AN EOT
A
B
C
D
E
F
G
H
J
K

Chart R05 (R05) LINE END APPENDAGE

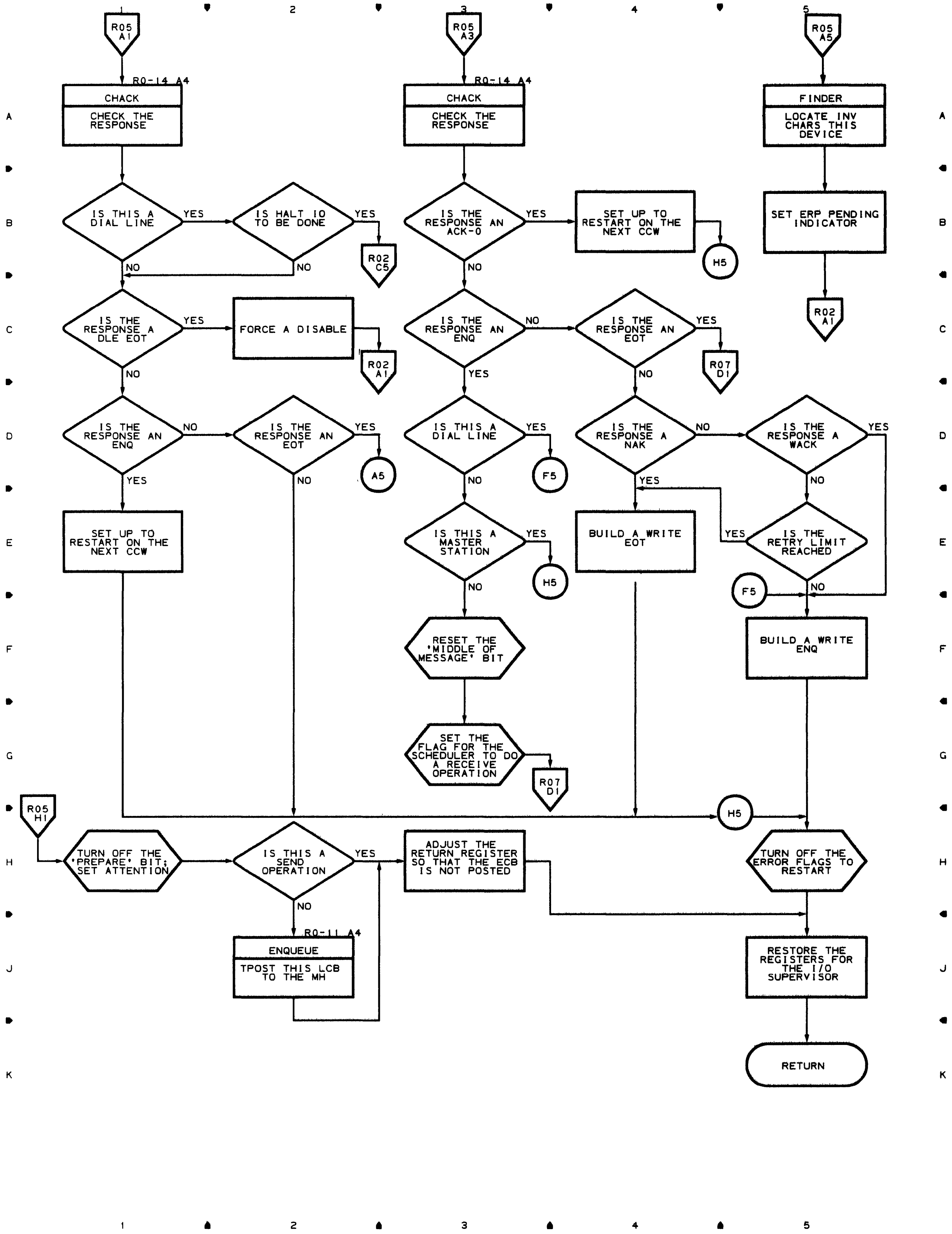


Chart R0-6 (R06) LINE END APPENDAGE

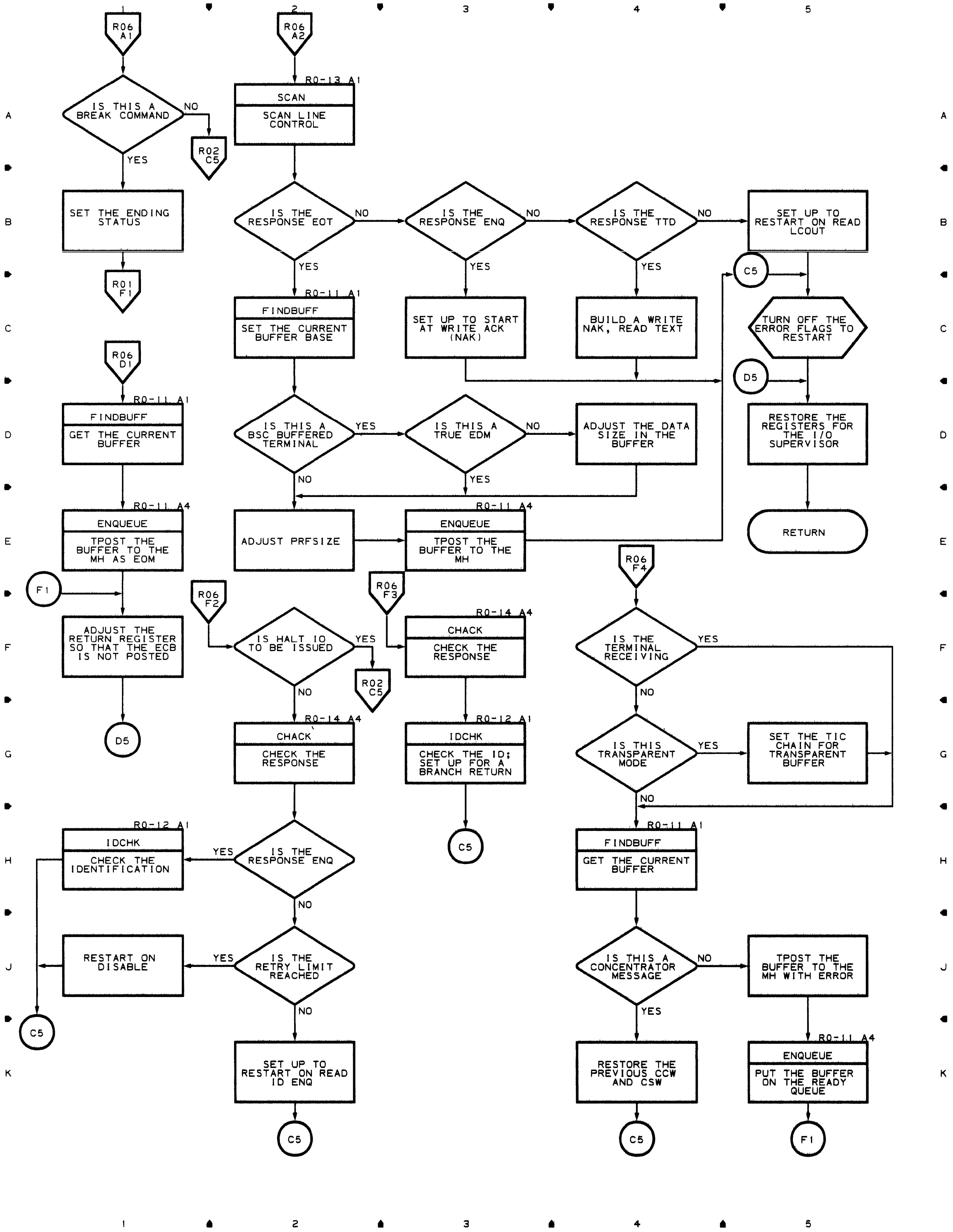

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
R06 A1
IS THIS A BREAK COMMAND
NO
R02 C5
YES
SET THE ENDING STATUS
R01 F1
R06 D1
R0-11 A1
FINDBUFF
GET THE CURRENT BUFFER
R0-11 A4
ENQUEUE
TPOST THE BUFFER TO THE MH AS EOM
F1
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
D5
R0-12 A1
IDCHK
CHECK THE IDENTIFICATION
YES
RESTART ON DISABLE
YES
C5
R06 A2
R0-13 A1
SCAN
SCAN LINE CONTROL
IS THE RESPONSE EOT
NO
YES
R0-11 A1
FINDBUFF
SET THE CURRENT BUFFER BASE
IS THIS A BSC BUFFERED TERMINAL
YES
NO
ADJUST PRFSIZE
R06 F2
IS HALT IO TO BE ISSUED
YES
R02 C5
NO
R0-14 A4
CHACK
CHECK THE RESPONSE
IS THE RESPONSE ENQ
NO
IS THE RETRY LIMIT REACHED
NO
SET UP TO RESTART ON READ ID ENQ
C5
IS THE RESPONSE ENQ
NO
YES
SET UP TO START AT WRITE ACK (NAK)
IS THIS A TRUE EOM
NO
YES
R0-11 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
R06 F3
R0-14 A4
CHACK
CHECK THE RESPONSE
R0-12 A1
IDCHK
CHECK THE ID; SET UP FOR A BRANCH RETURN
C5
IS THE RESPONSE TTD
NO
YES
BUILD A WRITE NAK, READ TEXT
ADJUST THE DATA SIZE IN THE BUFFER
R06 F4
IS THE TERMINAL RECEIVING
YES
NO
IS THIS TRANSPARENT MODE
YES
NO
R0-11 A1
FINDBUFF
GET THE CURRENT BUFFER
IS THIS A CONCENTRATOR MESSAGE
NO
YES
RESTORE THE PREVIOUS CCW AND CSW
C5
SET UP TO RESTART ON READ LCOUT
C5
TURN OFF THE ERROR FLAGS TO RESTART
D5
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
SET THE TIC CHAIN FOR TRANSPARENT BUFFER
TPOST THE BUFFER TO THE MH WITH ERROR
R0-11 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
F1

Chart R0-7 (R07) LINE END APPENDAGE

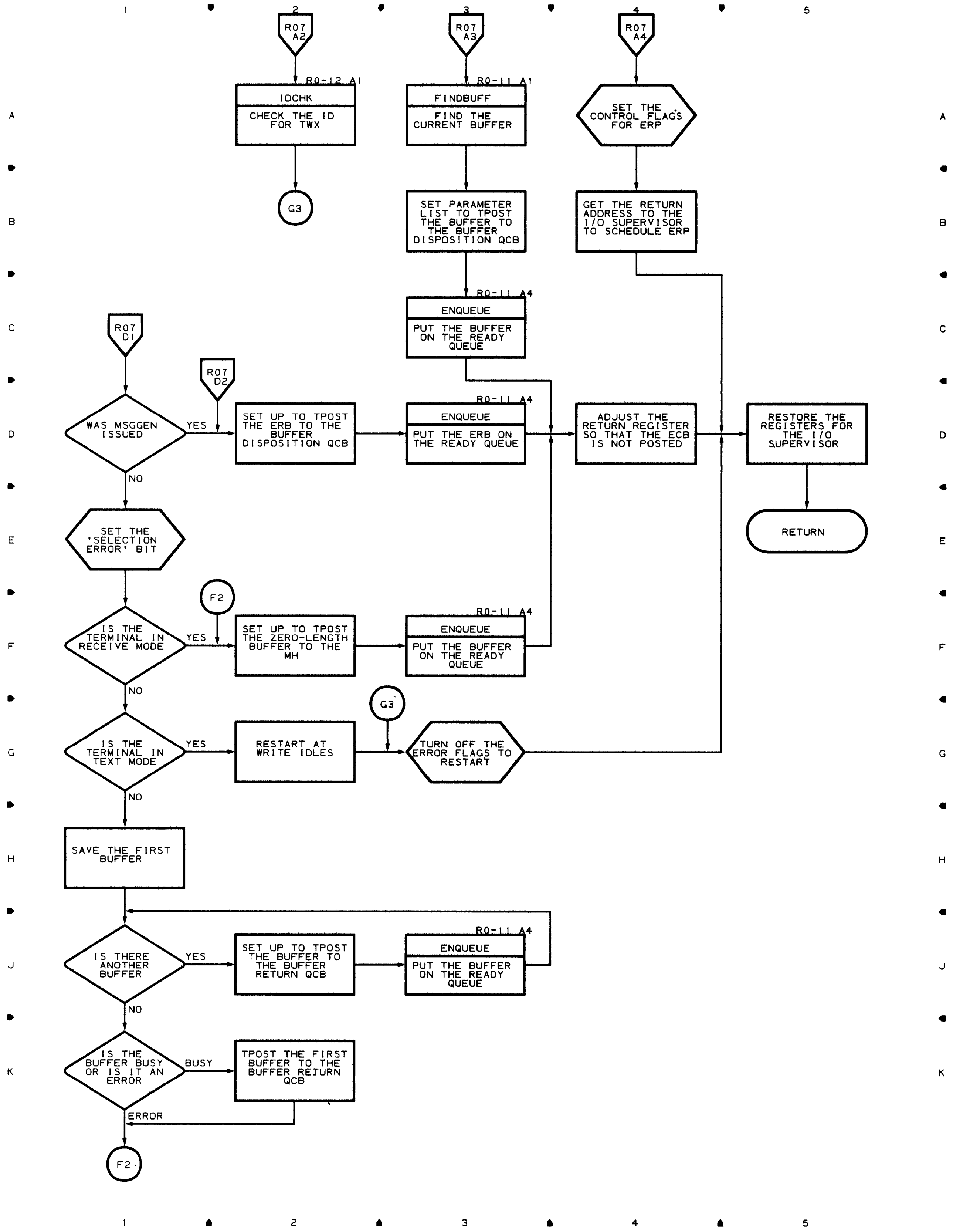
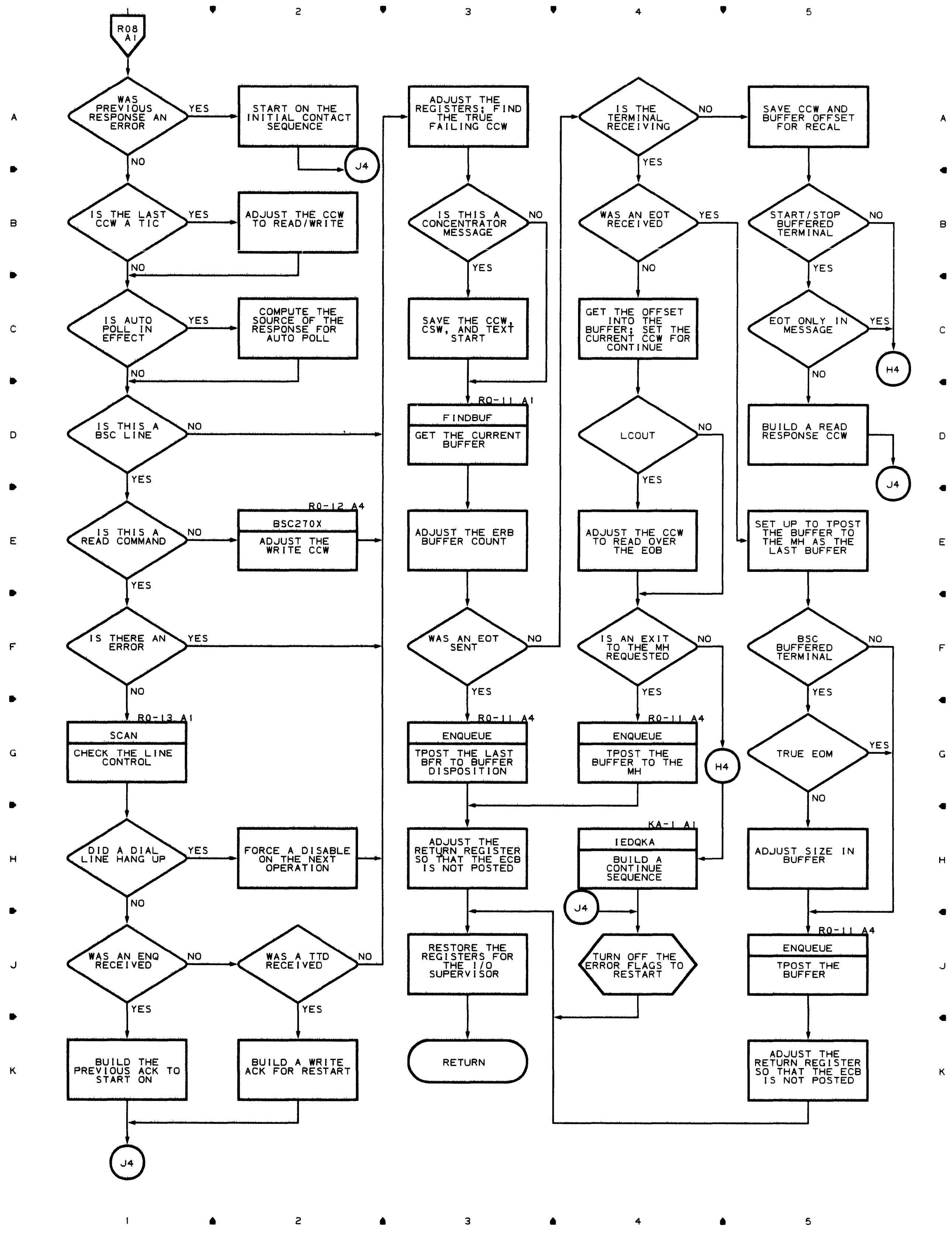

R07 A2
R0-12 A1
IDCHK
CHECK THE ID FOR TWX
G3
R07 A3
R0-11 A1
FINDBUFF
FIND THE CURRENT BUFFER
SET PARAMETER LIST TO TPOST THE BUFFER TO THE BUFFER DISPOSITION QCB
R0-11 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
R07 A4
SET THE CONTROL FLAGS FOR ERP
GET THE RETURN ADDRESS TO THE I/O SUPERVISOR TO SCHEDULE ERP
R07 D1
WAS MSGGEN ISSUED
YES
NO
R07 D2
SET UP TO TPOST THE ERB TO THE BUFFER DISPOSITION QCB
R0-11 A4
ENQUEUE
PUT THE ERB ON THE READY QUEUE
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
SET THE 'SELECTION ERROR' BIT
F2
IS THE TERMINAL IN RECEIVE MODE
YES
NO
SET UP TO TPOST THE ZERO-LENGTH BUFFER TO THE MH
R0-11 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
G3
IS THE TERMINAL IN TEXT MODE
YES
NO
RESTART AT WRITE IDLES
TURN OFF THE ERROR FLAGS TO RESTART
SAVE THE FIRST BUFFER
IS THERE ANOTHER BUFFER
YES
NO
SET UP TO TPOST THE BUFFER TO THE BUFFER RETURN QCB
R0-11 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
IS THE BUFFER BUSY OR IS IT AN ERROR
BUSY
ERROR
TPOST THE FIRST BUFFER TO THE BUFFER RETURN QCB
F2
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart R0-8 (R08) LINE END APPENDAGE



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
R08 A1
WAS PREVIOUS RESPONSE AN ERROR
YES
START ON THE INITIAL CONTACT SEQUENCE
J4
NO
IS THE LAST CCW A TIC
YES
ADJUST THE CCW TO READ/WRITE
NO
IS AUTO POLL IN EFFECT
YES
COMPUTE THE SOURCE OF THE RESPONSE FOR AUTO POLL
NO
IS THIS A BSC LINE
NO
YES
IS THIS A READ COMMAND
NO
R0-12 A4
BSC270X
ADJUST THE WRITE CCW
YES
IS THERE AN ERROR
YES
NO
R0-13 A1
SCAN
CHECK THE LINE CONTROL
DID A DIAL LINE HANG UP
YES
FORCE A DISABLE ON THE NEXT OPERATION
NO
WAS AN ENQ RECEIVED
NO
WAS A TTD RECEIVED
NO
YES
YES
BUILD THE PREVIOUS ACK TO START ON
BUILD A WRITE ACK FOR RESTART
J4
ADJUST THE REGISTERS; FIND THE TRUE FAILING CCW
IS THIS A CONCENTRATOR MESSAGE
NO
YES
SAVE THE CCW, CSW, AND TEXT START
R0-11 A1
FINDBUF
GET THE CURRENT BUFFER
ADJUST THE ERB BUFFER COUNT
WAS AN EOT SENT
NO
YES
R0-11 A4
ENQUEUE
TPOST THE LAST BFR TO BUFFER DISPOSITION
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
IS THE TERMINAL RECEIVING
NO
YES
WAS AN EOT RECEIVED
YES
NO
GET THE OFFSET INTO THE BUFFER; SET THE CURRENT CCW FOR CONTINUE
LCOUT
NO
YES
ADJUST THE CCW TO READ OVER THE EOB
IS AN EXIT TO THE MH REQUESTED
NO
YES
R0-11 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
H4
KA-1 A1
IEDQKA
BUILD A CONTINUE SEQUENCE
J4
TURN OFF THE ERROR FLAGS TO RESTART
SAVE CCW AND BUFFER OFFSET FOR RECAL
START/STOP BUFFERED TERMINAL
NO
YES
EOT ONLY IN MESSAGE
YES
H4
NO
BUILD A READ RESPONSE CCW
J4
SET UP TO TPOST THE BUFFER TO THE MH AS THE LAST BUFFER
BSC BUFFERED TERMINAL
NO
YES
TRUE EOM
YES
NO
ADJUST SIZE IN BUFFER
R0-11 A4
ENQUEUE
TPOST THE BUFFER
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
1
2
3
4
5

Chart R81 (R08) LINE END APPENDAGE

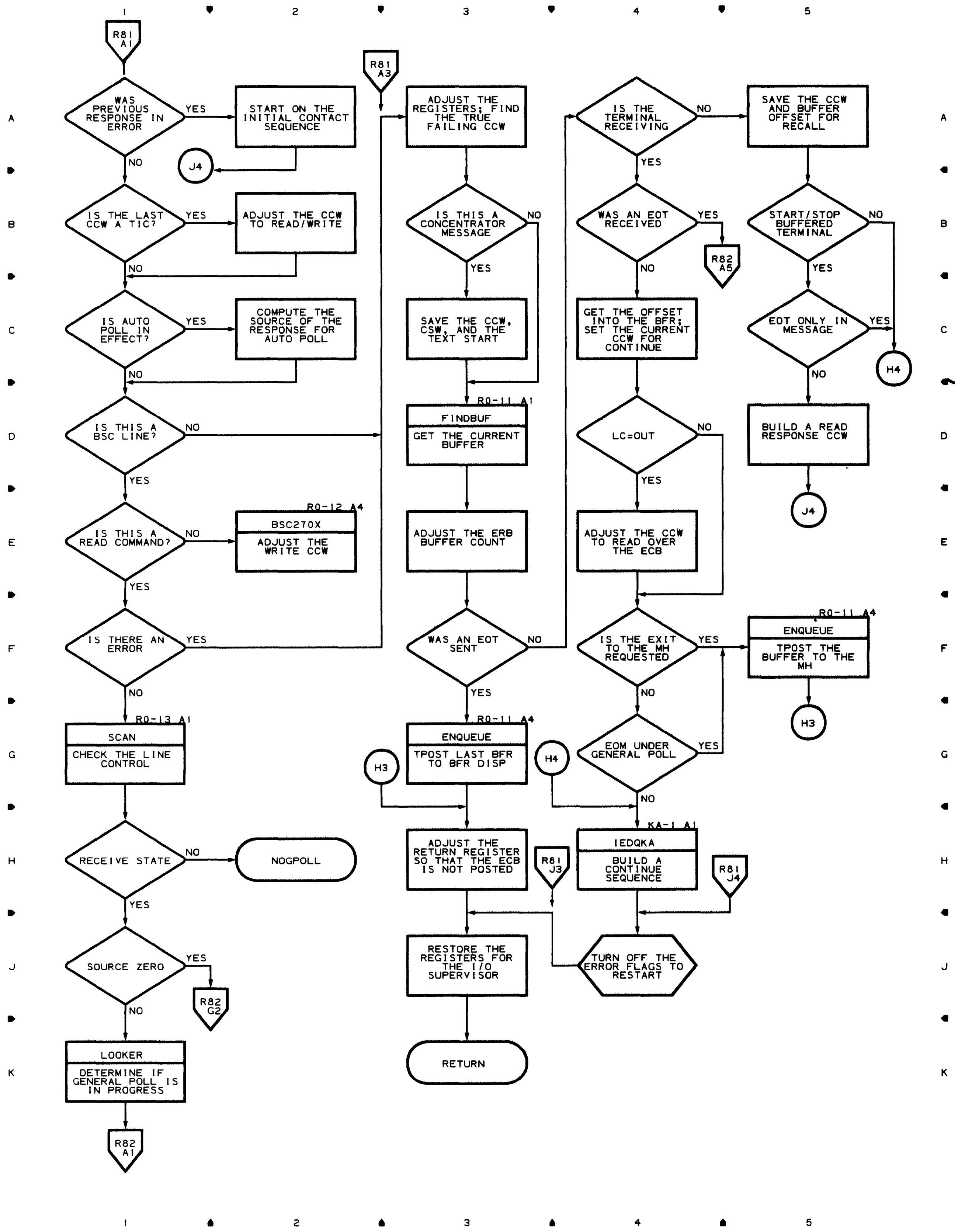

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
R81 A1
WAS PREVIOUS RESPONSE IN ERROR
YES
START ON THE INITIAL CONTACT SEQUENCE
NO
J4
IS THE LAST CCW A TIC?
YES
ADJUST THE CCW TO READ/WRITE
NO
IS AUTO POLL IN EFFECT?
YES
COMPUTE THE SOURCE OF THE RESPONSE FOR AUTO POLL
NO
IS THIS A BSC LINE?
NO
YES
IS THIS A READ COMMAND?
NO
R0-12 A4
BSC270X
ADJUST THE WRITE CCW
YES
IS THERE AN ERROR
YES
NO
R0-13 A1
SCAN
CHECK THE LINE CONTROL
RECEIVE STATE
NO
NOGPOLL
YES
SOURCE ZERO
YES
R82 G2
NO
LOOKER
DETERMINE IF GENERAL POLL IS IN PROGRESS
R82 A1
R81 A3
ADJUST THE REGISTERS; FIND THE TRUE FAILING CCW
IS THIS A CONCENTRATOR MESSAGE
NO
YES
SAVE THE CCW, CSW, AND THE TEXT START
R0-11 A1
FINDBUF
GET THE CURRENT BUFFER
ADJUST THE ERB BUFFER COUNT
WAS AN EOT SENT
NO
YES
R0-11 A4
ENQUEUE
TPOST LAST BFR TO BFR DISP
H3
ADJUST THE RETURN REGISTER SO THAT THE ECB IS NOT POSTED
R81 J3
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
IS THE TERMINAL RECEIVING
NO
YES
WAS AN EOT RECEIVED
YES
R82 A5
NO
GET THE OFFSET INTO THE BFR; SET THE CURRENT CCW FOR CONTINUE
LC=OUT
NO
YES
ADJUST THE CCW TO READ OVER THE ECB
IS THE EXIT TO THE MH REQUESTED
YES
NO
H4
EOM UNDER GENERAL POLL
YES
NO
KA-1 A1
IEDQKA
BUILD A CONTINUE SEQUENCE
R81 J4
TURN OFF THE ERROR FLAGS TO RESTART
SAVE THE CCW AND BUFFER OFFSET FOR RECALL
START/STOP BUFFERED TERMINAL
NO
YES
EOT ONLY IN MESSAGE
YES
H4
NO
BUILD A READ RESPONSE CCW
J4
R0-11 A4
ENQUEUE
TPOST THE BUFFER TO THE MH
H3

Chart R82 (R08) LINE END APPENDAGE

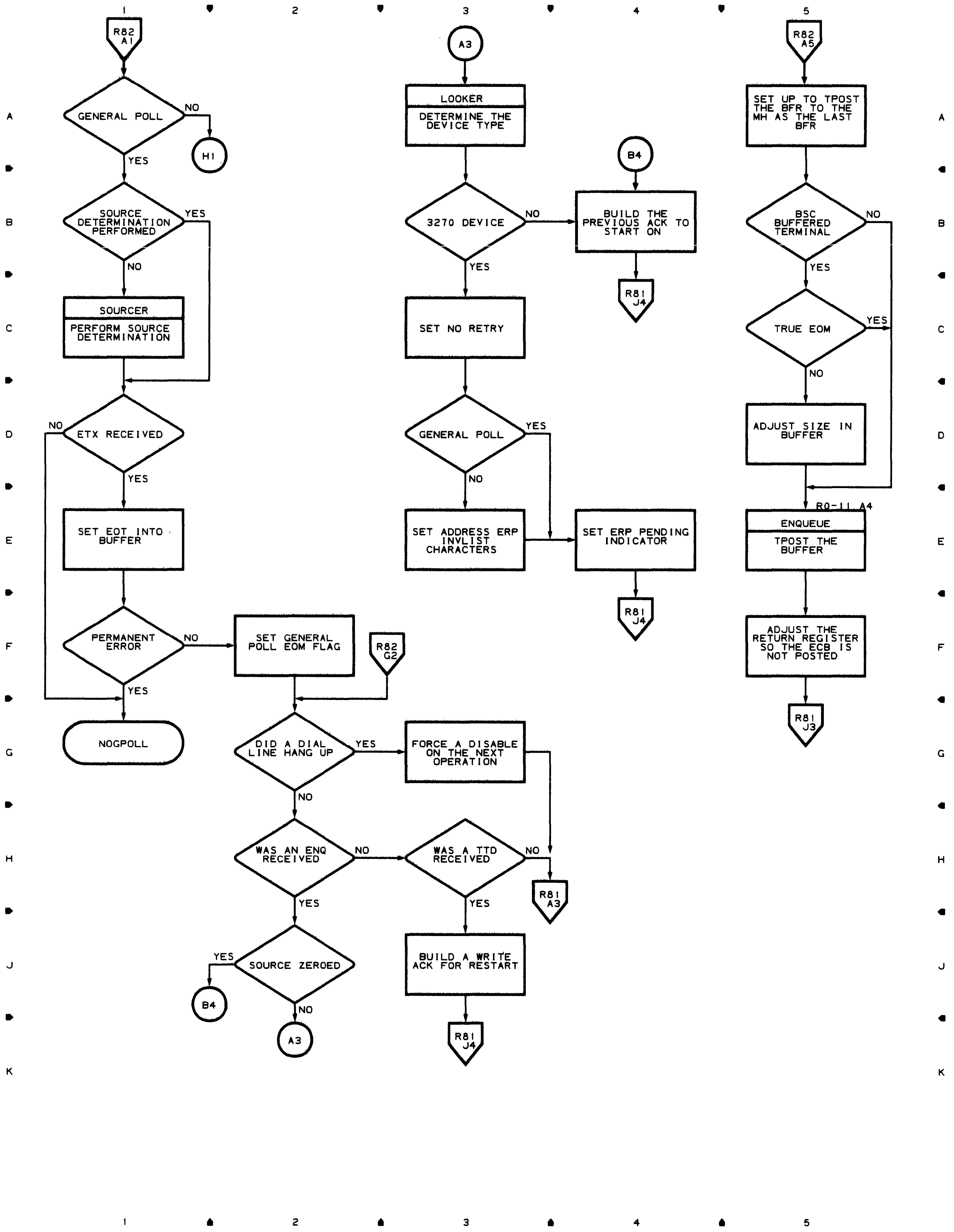


Chart R0-9 (R09) LINE END APPENDAGE

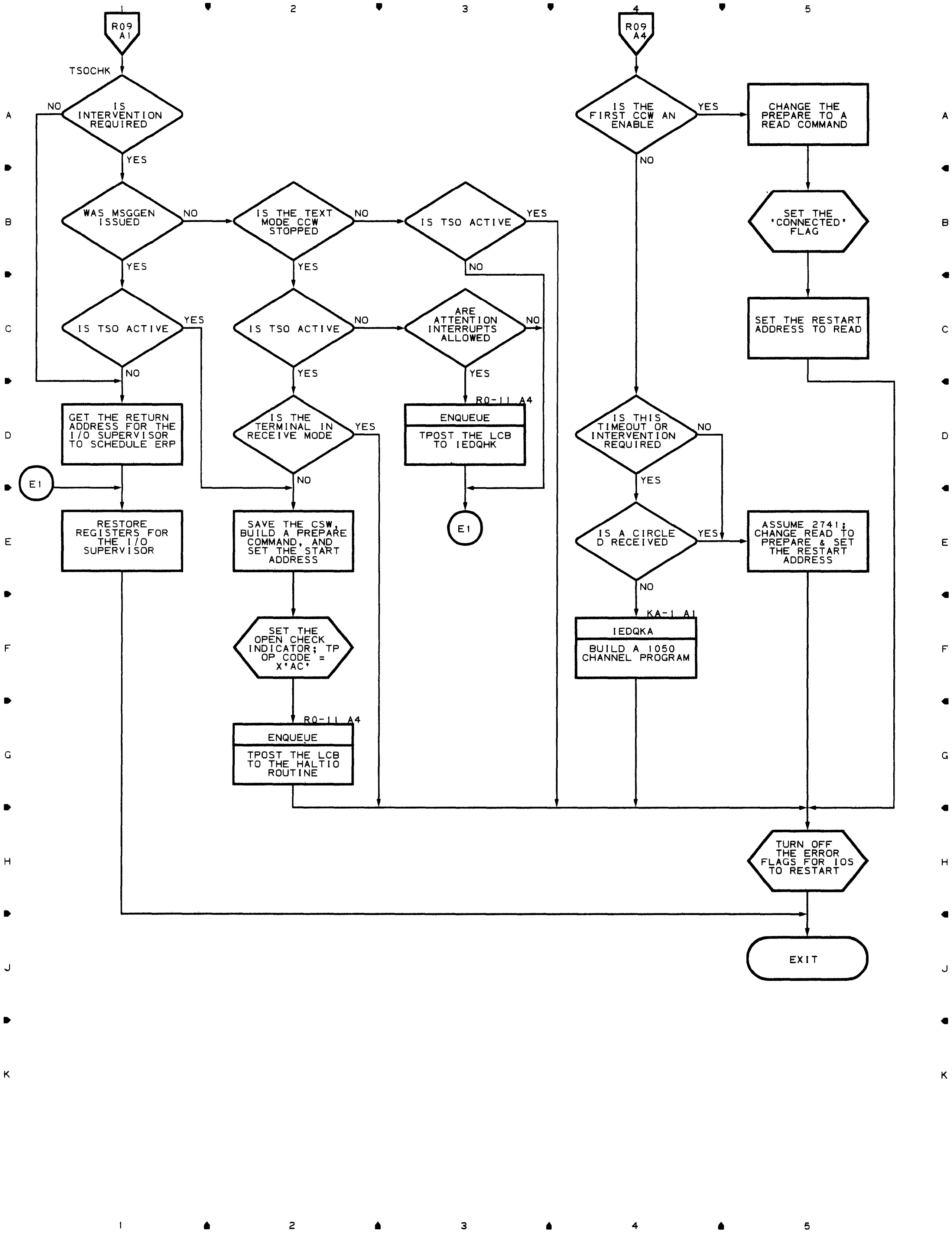

R09 A1
TSOCHK
IS INTERVENTION REQUIRED
NO
YES
WAS MSGGEN ISSUED
NO
YES
IS THE TEXT MODE CCW STOPPED
NO
YES
IS TSO ACTIVE
YES
NO
IS TSO ACTIVE
YES
NO
IS TSO ACTIVE
NO
YES
ARE ATTENTION INTERRUPTS ALLOWED
NO
YES
GET THE RETURN ADDRESS FOR THE I/O SUPERVISOR TO SCHEDULE ERP
IS THE TERMINAL IN RECEIVE MODE
YES
NO
R0-11 A4
ENQUEUE
TPOST THE LCB TO IEDQHK
E1
RESTORE REGISTERS FOR THE I/O SUPERVISOR
SAVE THE CSW, BUILD A PREPARE COMMAND, AND SET THE START ADDRESS
E1
SET THE OPEN CHECK INDICATOR; TP OP CODE = X'AC'
R0-11 A4
ENQUEUE
TPOST THE LCB TO THE HALTIO ROUTINE
R09 A4
IS THE FIRST CCW AN ENABLE
YES
NO
CHANGE THE PREPARE TO A READ COMMAND
SET THE 'CONNECTED' FLAG
SET THE RESTART ADDRESS TO READ
IS THIS TIMEOUT OR INTERVENTION REQUIRED
NO
YES
IS A CIRCLE D RECEIVED
YES
NO
ASSUME 2741; CHANGE READ TO PREPARE & SET THE RESTART ADDRESS
KA-1 A1
IEDQKA
BUILD A 1050 CHANNEL PROGRAM
TURN OFF THE ERROR FLAGS FOR IOS TO RESTART
EXIT
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart R0-10 (R0A) LINE END APPENDAGE

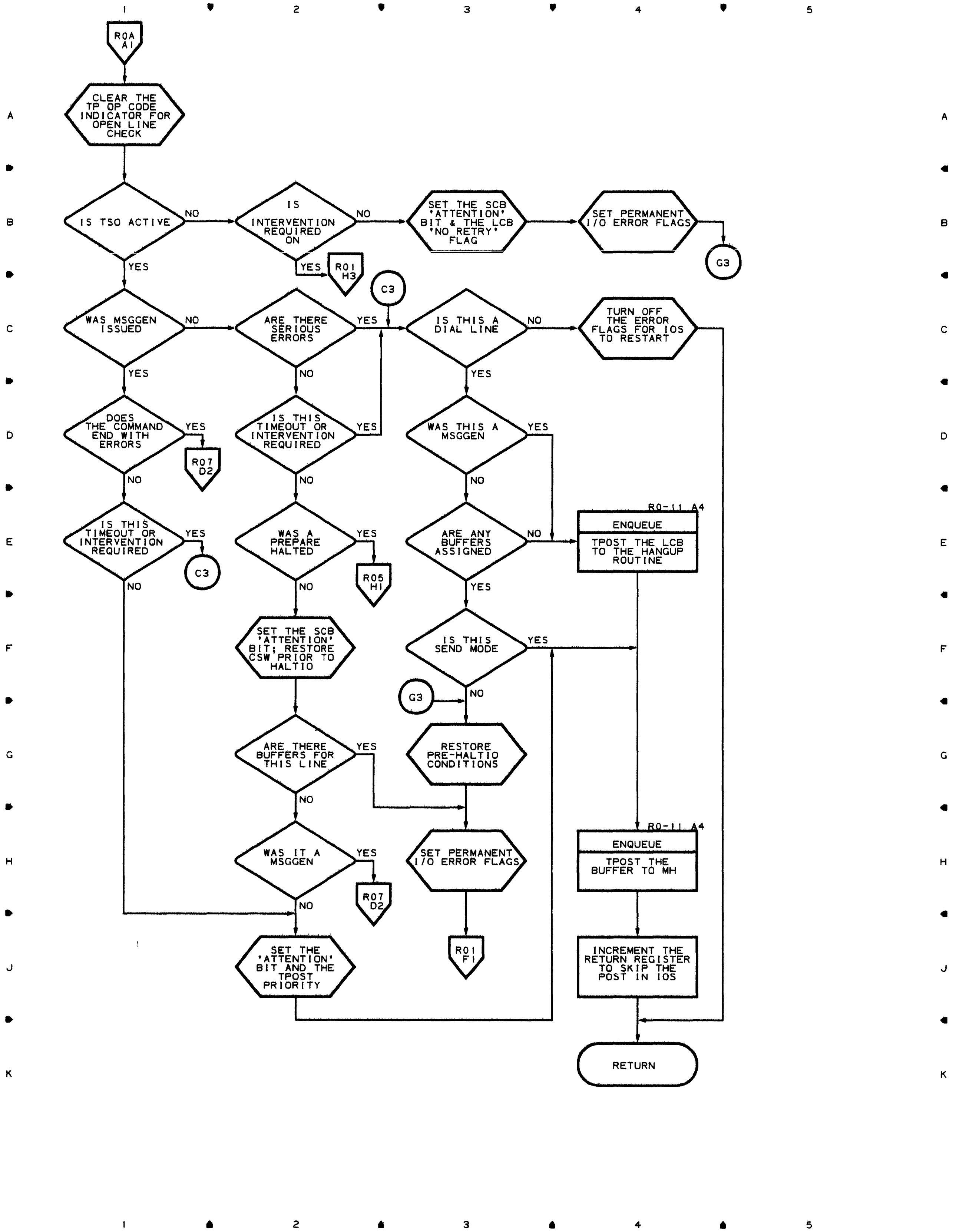

R0A A1
CLEAR THE TP OP CODE INDICATOR FOR OPEN LINE CHECK
IS TSO ACTIVE
NO
YES
IS INTERVENTION REQUIRED ON
NO
YES
R01 H3
SET THE SCB 'ATTENTION' BIT & THE LCB 'NO RETRY' FLAG
SET PERMANENT I/O ERROR FLAGS
G3
C3
WAS MSGGEN ISSUED
NO
YES
ARE THERE SERIOUS ERRORS
YES
NO
IS THIS A DIAL LINE
NO
YES
TURN OFF THE ERROR FLAGS FOR IOS TO RESTART
DOES THE COMMAND END WITH ERRORS
YES
NO
R07 D2
IS THIS TIMEOUT OR INTERVENTION REQUIRED
YES
NO
WAS THIS A MSGGEN
YES
NO
IS THIS TIMEOUT OR INTERVENTION REQUIRED
YES
NO
C3
WAS A PREPARE HALTED
YES
NO
R05 H1
ARE ANY BUFFERS ASSIGNED
NO
YES
R0-11 A4
ENQUEUE
TPOST THE LCB TO THE HANGUP ROUTINE
SET THE SCB 'ATTENTION' BIT; RESTORE CSW PRIOR TO HALTIO
IS THIS SEND MODE
YES
NO
G3
ARE THERE BUFFERS FOR THIS LINE
YES
NO
RESTORE PRE-HALTIO CONDITIONS
R0-11 A4
ENQUEUE
TPOST THE BUFFER TO MH
WAS IT A MSGGEN
YES
NO
R07 D2
SET PERMANENT I/O ERROR FLAGS
SET THE 'ATTENTION' BIT AND THE TPOST PRIORITY
R01 F1
INCREMENT THE RETURN REGISTER TO SKIP THE POST IN IOS
RETURN
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K

Chart R0-11 (R0B) LINE END APPENDAGE

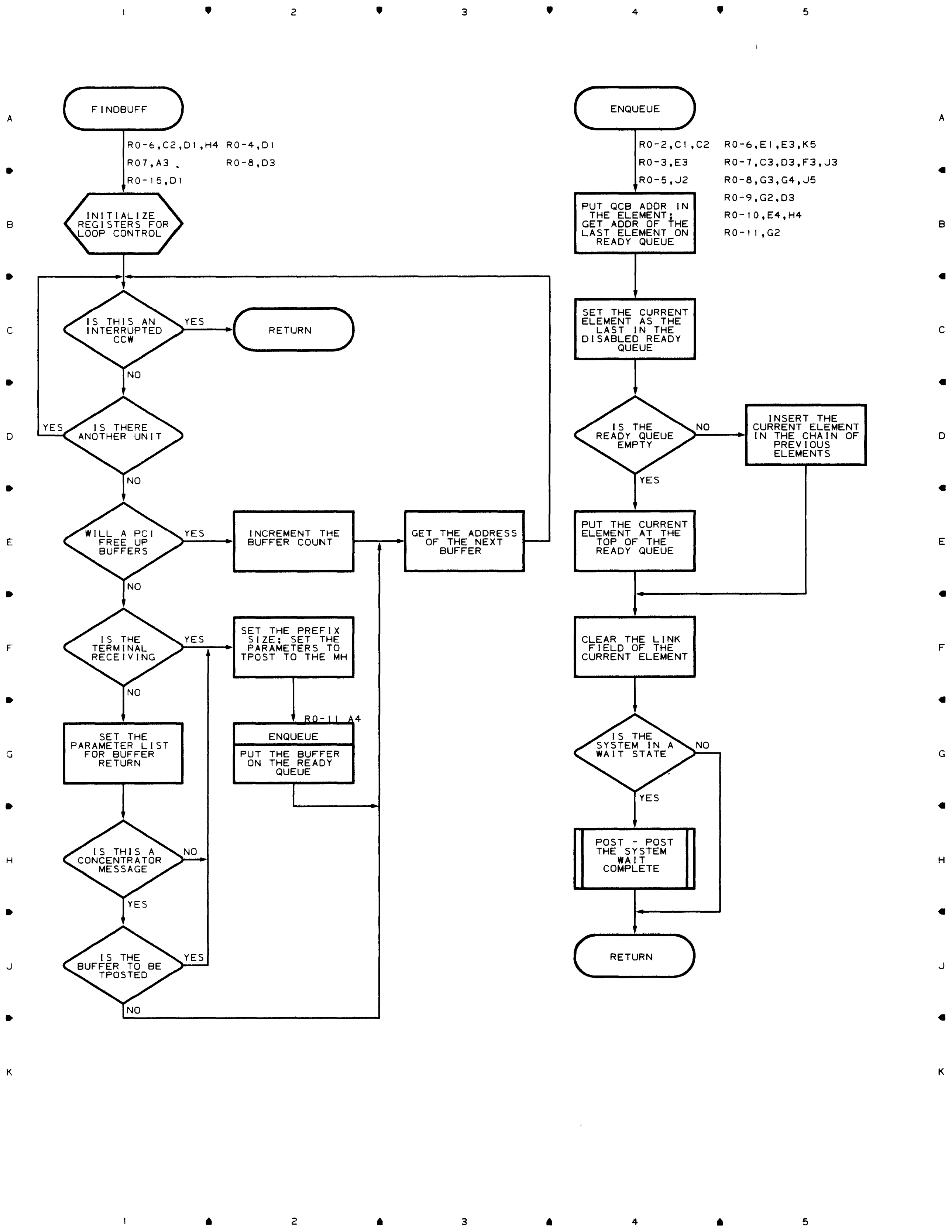

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FINDBUFF
R0-6,C2,D1,H4 R0-4,D1
R07,A3 . R0-8,D3
R0-15,D1
INITIALIZE REGISTERS FOR LOOP CONTROL
IS THIS AN INTERRUPTED CCW
YES
RETURN
NO
IS THERE ANOTHER UNIT
YES
NO
WILL A PCI FREE UP BUFFERS
YES
INCREMENT THE BUFFER COUNT
GET THE ADDRESS OF THE NEXT BUFFER
NO
IS THE TERMINAL RECEIVING
YES
SET THE PREFIX SIZE; SET THE PARAMETERS TO TPOST TO THE MH
R0-11 A4
ENQUEUE
PUT THE BUFFER ON THE READY QUEUE
NO
SET THE PARAMETER LIST FOR BUFFER RETURN
IS THIS A CONCENTRATOR MESSAGE
NO
YES
IS THE BUFFER TO BE TPOSTED
YES
NO
ENQUEUE
R0-2,C1,C2 R0-6,E1,E3,K5
R0-3,E3 R0-7,C3,D3,F3,J3
R0-5,J2 R0-8,G3,G4,J5
R0-9,G2,D3
R0-10,E4,H4
R0-11,G2
PUT QCB ADDR IN THE ELEMENT; GET ADDR OF THE LAST ELEMENT ON READY QUEUE
SET THE CURRENT ELEMENT AS THE LAST IN THE DISABLED READY QUEUE
IS THE READY QUEUE EMPTY
NO
INSERT THE CURRENT ELEMENT IN THE CHAIN OF PREVIOUS ELEMENTS
YES
PUT THE CURRENT ELEMENT AT THE TOP OF THE READY QUEUE
CLEAR THE LINK FIELD OF THE CURRENT ELEMENT
IS THE SYSTEM IN A WAIT STATE
NO
YES
POST - POST THE SYSTEM WAIT COMPLETE
RETURN

Chart R0-12 (R0C) LINE END APPENDAGE

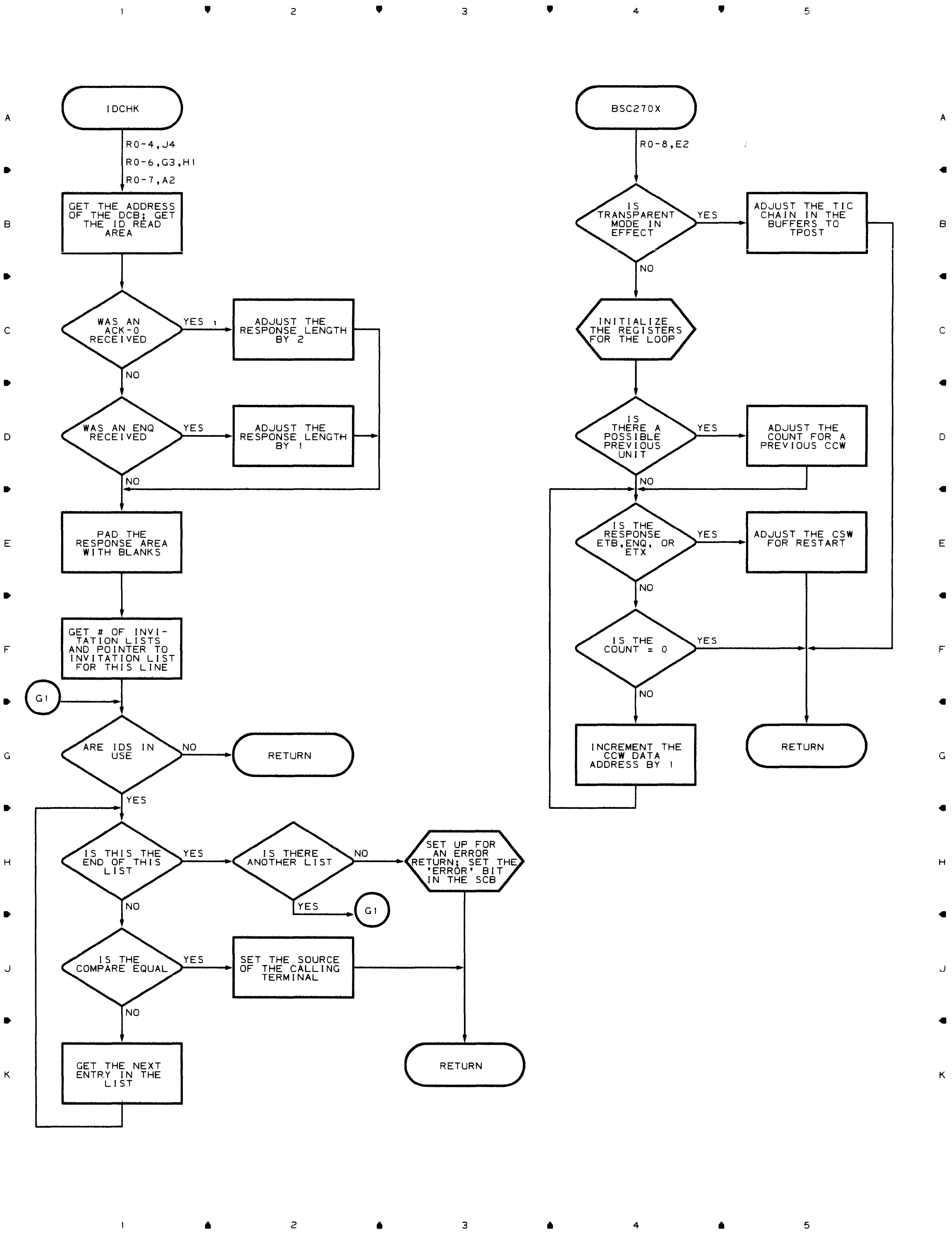

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
IDCHK
R0-4,J4
R0-6,G3,H1
R0-7,A2
GET THE ADDRESS OF THE DCB; GET THE ID READ AREA
WAS AN ACK-0 RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 2
NO
WAS AN ENQ RECEIVED
YES
ADJUST THE RESPONSE LENGTH BY 1
NO
PAD THE RESPONSE AREA WITH BLANKS
GET # OF INVITATION LISTS AND POINTER TO INVITATION LIST FOR THIS LINE
G1
ARE IDS IN USE
NO
RETURN
YES
IS THIS THE END OF THIS LIST
YES
IS THERE ANOTHER LIST
NO
SET UP FOR AN ERROR RETURN; SET THE 'ERROR' BIT IN THE SCB
YES
G1
NO
IS THE COMPARE EQUAL
YES
SET THE SOURCE OF THE CALLING TERMINAL
NO
GET THE NEXT ENTRY IN THE LIST
RETURN
BSC270X
R0-8,E2
IS TRANSPARENT MODE IN EFFECT
YES
ADJUST THE TIC CHAIN IN THE BUFFERS TO TPOST
NO
INITIALIZE THE REGISTERS FOR THE LOOP
IS THERE A POSSIBLE PREVIOUS UNIT
YES
ADJUST THE COUNT FOR A PREVIOUS CCW
NO
IS THE RESPONSE ETB,ENQ, OR ETX
YES
ADJUST THE CSW FOR RESTART
NO
IS THE COUNT = 0
YES
NO
INCREMENT THE CCW DATA ADDRESS BY 1
RETURN

Chart R0-13 (R0D) LINE END APPENDAGE

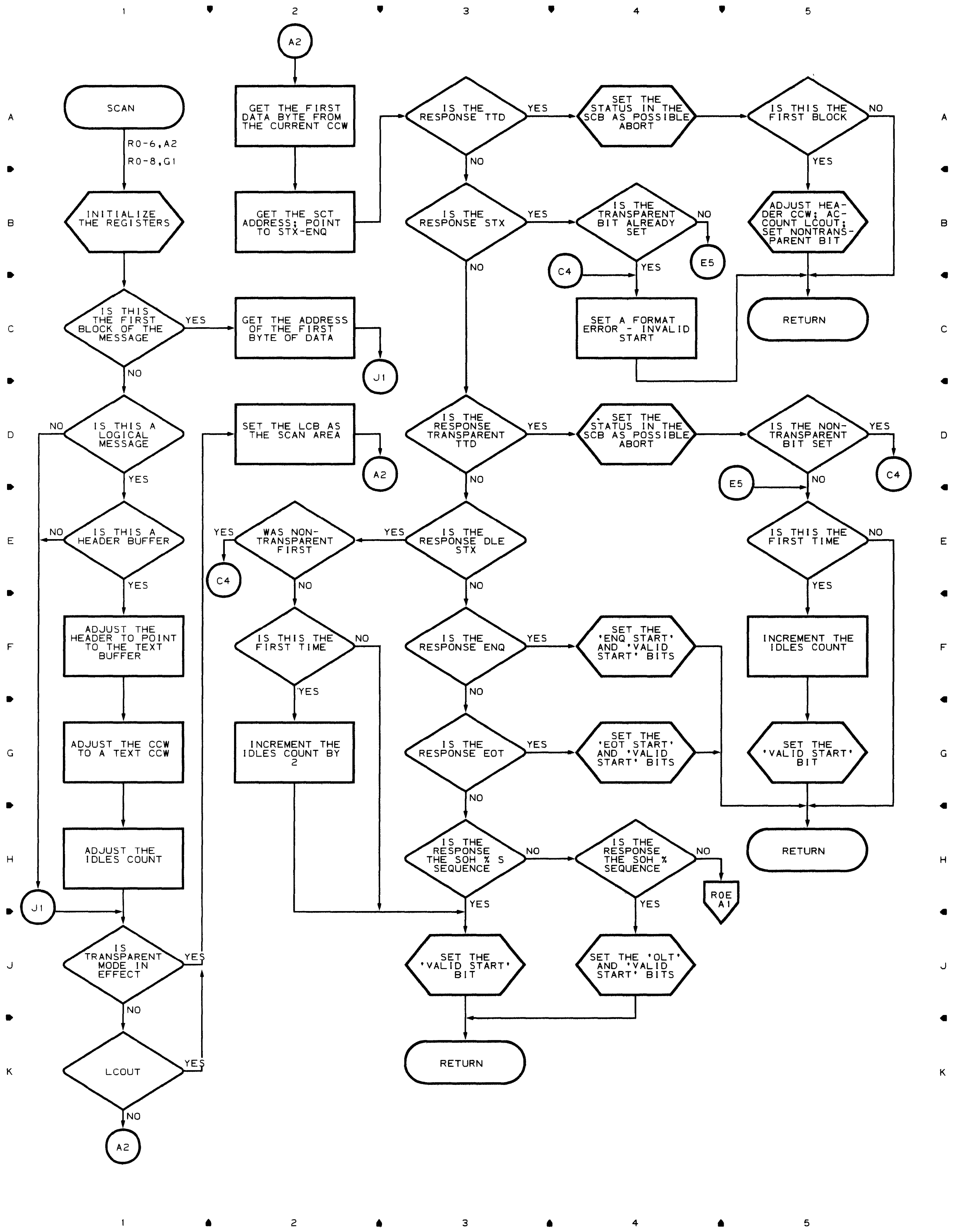
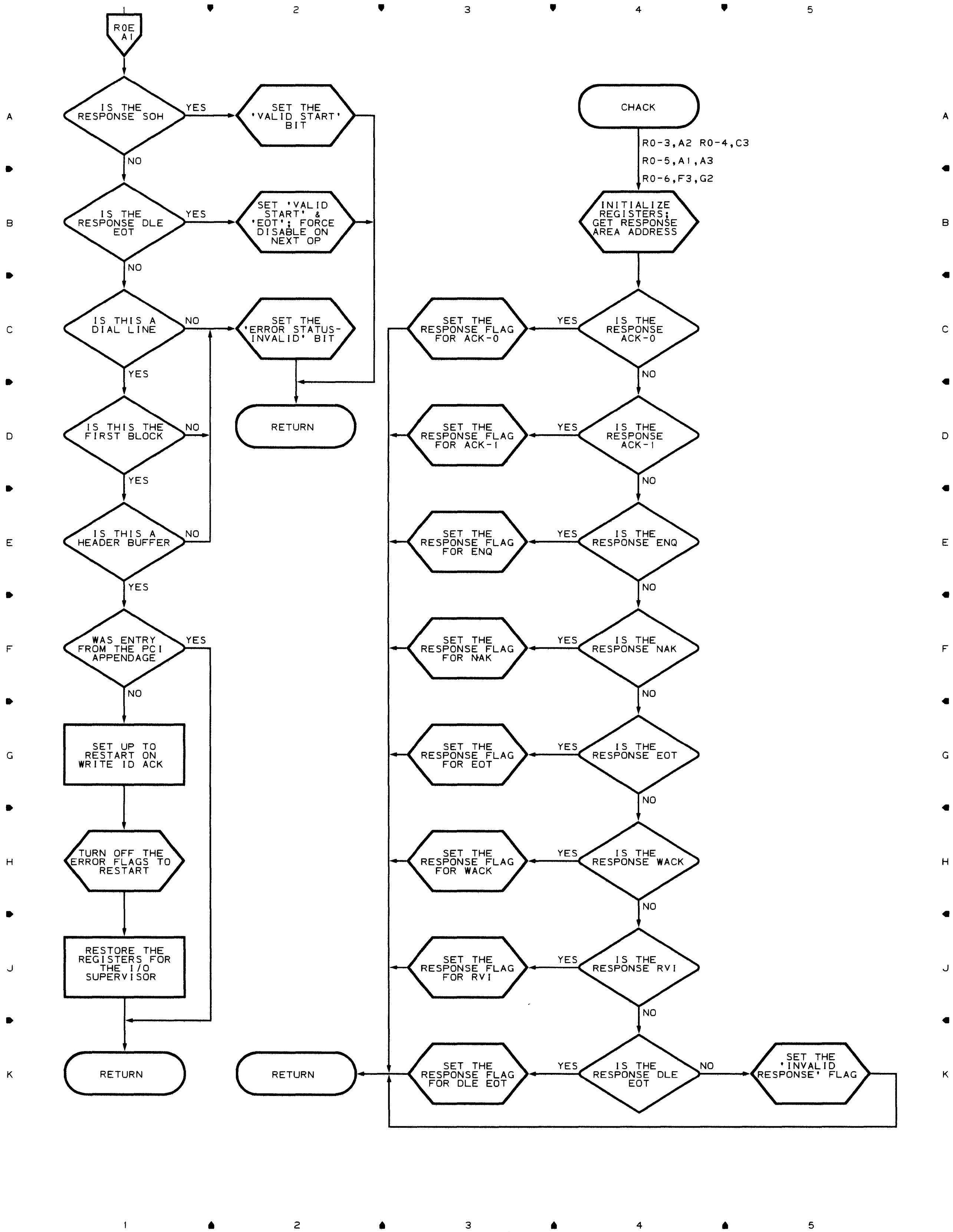
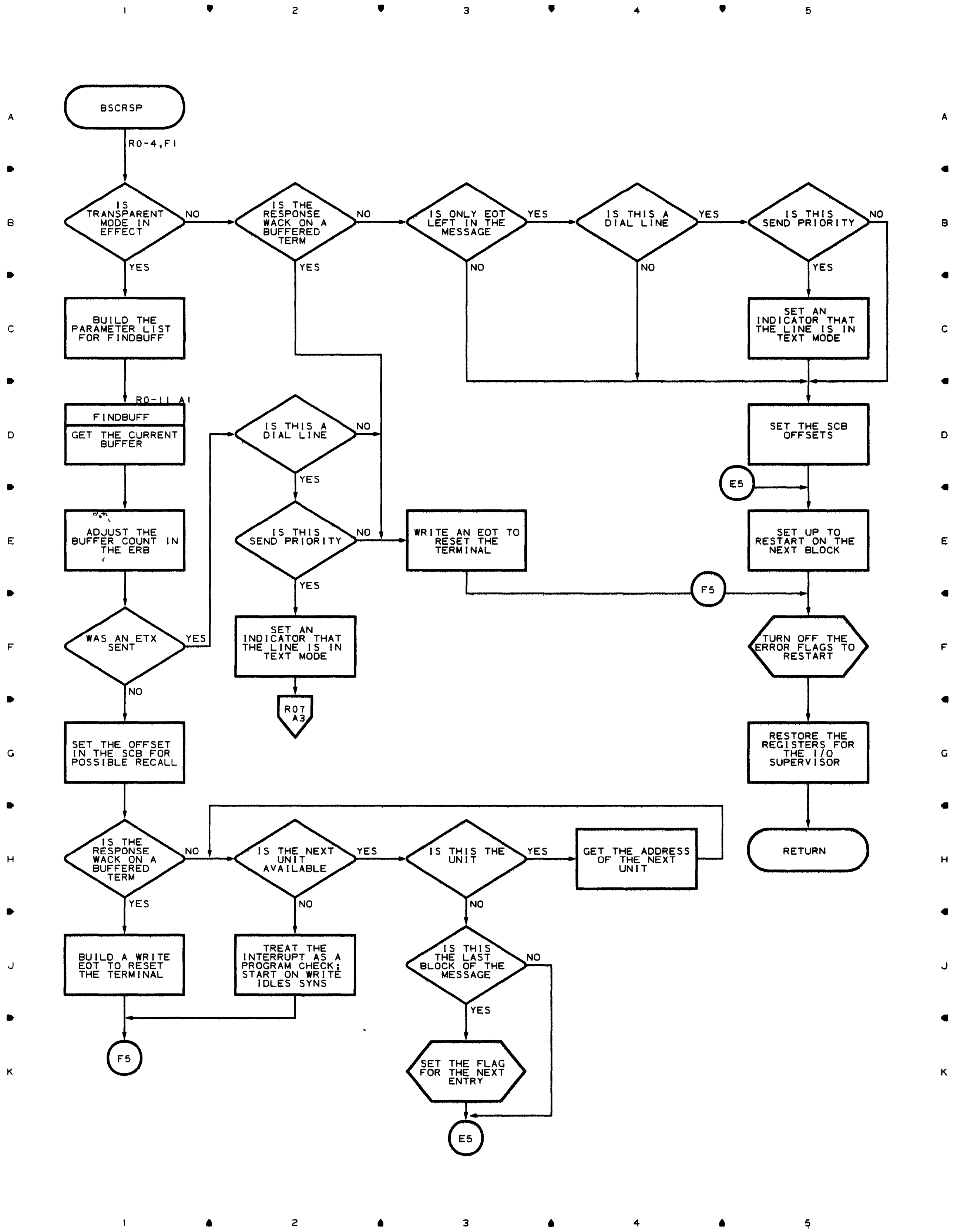

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
SCAN
R0-6,A2
R0-8,G1
INITIALIZE THE REGISTERS
IS THIS THE FIRST BLOCK OF THE MESSAGE
YES
NO
IS THIS A LOGICAL MESSAGE
NO
YES
IS THIS A HEADER BUFFER
NO
YES
ADJUST THE HEADER TO POINT TO THE TEXT BUFFER
ADJUST THE CCW TO A TEXT CCW
ADJUST THE IDLES COUNT
J1
IS TRANSPARENT MODE IN EFFECT
YES
NO
LCOUT
YES
NO
A2
A2
GET THE FIRST DATA BYTE FROM THE CURRENT CCW
GET THE SCT ADDRESS; POINT TO STX-ENQ
GET THE ADDRESS OF THE FIRST BYTE OF DATA
J1
SET THE LCB AS THE SCAN AREA
A2
WAS NON-TRANSPARENT FIRST
YES
C4
NO
IS THIS THE FIRST TIME
NO
YES
INCREMENT THE IDLES COUNT BY 2
IS THE RESPONSE TTD
YES
NO
IS THE RESPONSE STX
YES
NO
IS THE RESPONSE TRANSPARENT TTD
YES
NO
IS THE RESPONSE DLE STX
YES
NO
IS THE RESPONSE ENQ
YES
NO
IS THE RESPONSE EOT
YES
NO
IS THE RESPONSE THE SOH % S SEQUENCE
NO
YES
SET THE 'VALID START' BIT
RETURN
SET THE STATUS IN THE SCB AS POSSIBLE ABORT
IS THE TRANSPARENT BIT ALREADY SET
NO
YES
C4
E5
SET A FORMAT ERROR - INVALID START
SET THE STATUS IN THE SCB AS POSSIBLE ABORT
SET THE 'ENQ START' AND 'VALID START' BITS
SET THE 'EOT START' 'VALID START' BITS
IS THE RESPONSE THE SOH % SEQUENCE
NO
YES
R0E A1
SET THE 'OLT' AND 'VALID START' BITS
IS THIS THE FIRST BLOCK
NO
YES
ADJUST HEADER CCW; ACCOUNT LCOUT; SET NONTRANSPARENT BIT
RETURN
IS THE NON-TRANSPARENT BIT SET
YES
NO
C4
E5
IS THIS THE FIRST TIME
NO
YES
INCREMENT THE IDLES COUNT
SET THE 'VALID START' BIT
RETURN

Chart R0-14 (R0E) LINE END APPENDAGE



1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
R0E A1
IS THE RESPONSE SOH
YES
SET THE 'VALID START' BIT
NO
IS THE RESPONSE DLE EOT
YES
SET 'VALID START' & 'EOT'; FORCE DISABLE ON NEXT OP
NO
IS THIS A DIAL LINE
NO
SET THE 'ERROR STATUS-INVALID' BIT
YES
IS THIS THE FIRST BLOCK
NO
RETURN
YES
IS THIS A HEADER BUFFER
NO
YES
WAS ENTRY FROM THE PCI APPENDAGE
YES
NO
SET UP TO RESTART ON WRITE ID ACK
TURN OFF THE ERROR FLAGS TO RESTART
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
RETURN
RETURN
CHACK
R0-3,A2 R0-4,C3
R0-5,A1,A3
R0-6,F3,G2
INITIALIZE REGISTERS; GET RESPONSE AREA ADDRESS
IS THE RESPONSE ACK-0
YES
SET THE RESPONSE FLAG FOR ACK-0
NO
IS THE RESPONSE ACK-1
YES
SET THE RESPONSE FLAG FOR ACK-1
NO
IS THE RESPONSE ENQ
YES
SET THE RESPONSE FLAG FOR ENQ
NO
IS THE RESPONSE NAK
YES
SET THE RESPONSE FLAG FOR NAK
NO
IS THE RESPONSE EOT
YES
SET THE RESPONSE FLAG FOR EOT
NO
IS THE RESPONSE WACK
YES
SET THE RESPONSE FLAG FOR WACK
NO
IS THE RESPONSE RVI
YES
SET THE RESPONSE FLAG FOR RVI
NO
IS THE RESPONSE DLE EOT
YES
SET THE RESPONSE FLAG FOR DLE EOT
NO
SET THE 'INVALID RESPONSE' FLAG

Chart R0-15 (R0F) LINE END APPENDAGE


1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
BSCRSP
R0-4,F1
IS TRANSPARENT MODE IN EFFECT
NO
YES
IS THE RESPONSE WACK ON A BUFFERED TERM
NO
YES
IS ONLY EOT LEFT IN THE MESSAGE
YES
NO
IS THIS A DIAL LINE
YES
NO
IS THIS SEND PRIORITY
NO
YES
BUILD THE PARAMETER LIST FOR FINDBUFF
SET AN INDICATOR THAT THE LINE IS IN TEXT MODE
R0-11 A1
FINDBUFF
GET THE CURRENT BUFFER
IS THIS A DIAL LINE
NO
YES
SET THE SCB OFFSETS
E5
ADJUST THE BUFFER COUNT IN THE ERB
IS THIS SEND PRIORITY
NO
YES
WRITE AN EOT TO RESET THE TERMINAL
SET UP TO RESTART ON THE NEXT BLOCK
F5
WAS AN ETX SENT
YES
NO
SET AN INDICATOR THAT THE LINE IS IN TEXT MODE
TURN OFF THE ERROR FLAGS TO RESTART
R07 A3
SET THE OFFSET IN THE SCB FOR POSSIBLE RECALL
RESTORE THE REGISTERS FOR THE I/O SUPERVISOR
IS THE RESPONSE WACK ON A BUFFERED TERM
NO
YES
IS THE NEXT UNIT AVAILABLE
YES
NO
IS THIS THE UNIT
YES
NO
GET THE ADDRESS OF THE NEXT UNIT
RETURN
BUILD A WRITE EOT TO RESET THE TERMINAL
TREAT THE INTERRUPT AS A PROGRAM CHECK; START ON WRITE IDLES SYNS
IS THIS THE LAST BLOCK OF THE MESSAGE
NO
YES
F5
SET THE FLAG FOR THE NEXT ENTRY
E5

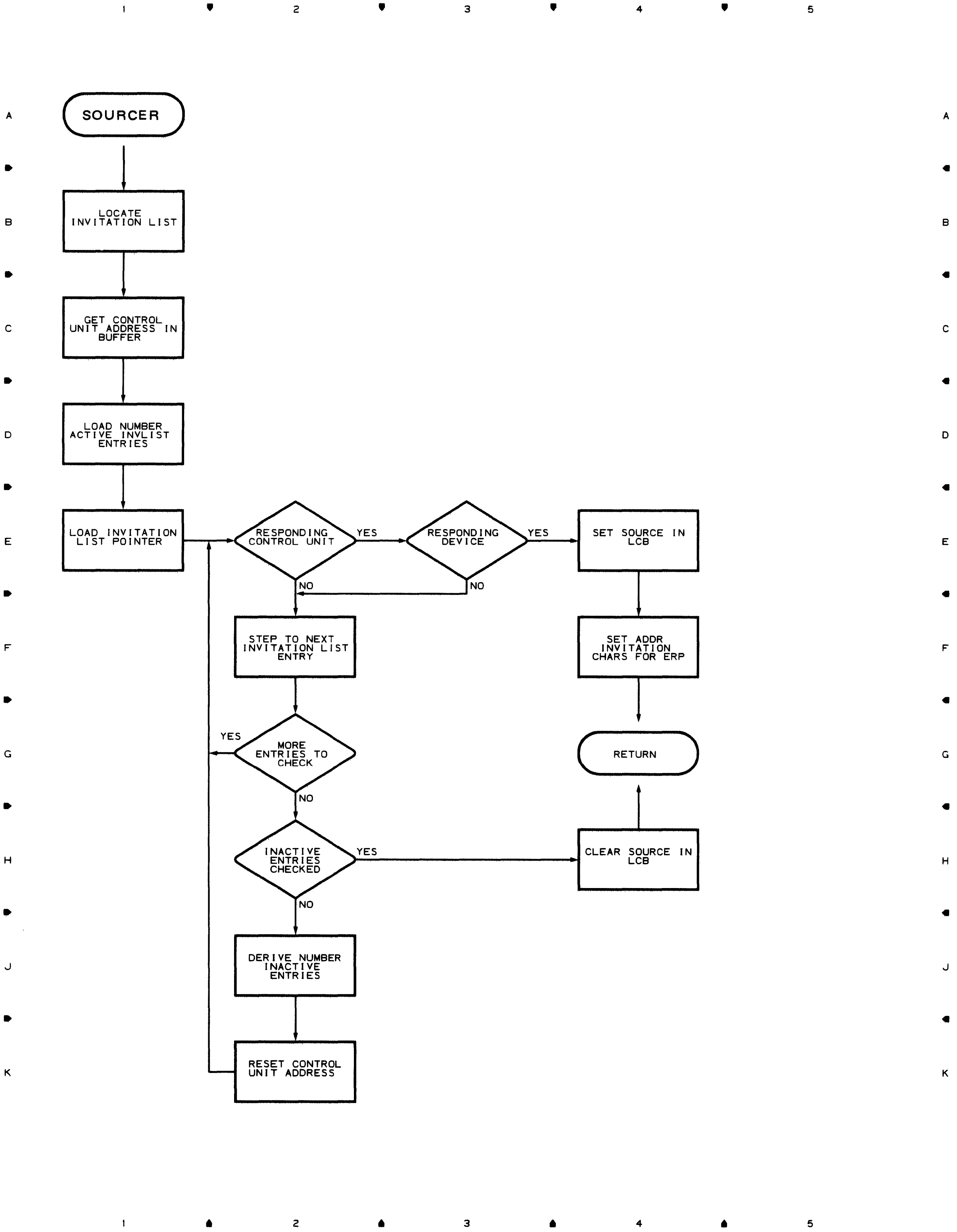

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
SOURCER
LOCATE INVITATION LIST
GET CONTROL UNIT ADDRESS IN BUFFER
LOAD NUMBER ACTIVE INVLIST ENTRIES
LOAD INVITATION LIST POINTER
RESPONDING CONTROL UNIT
YES
RESPONDING DEVICE
YES
SET SOURCE IN LCB
NO
NO
STEP TO NEXT INVITATION LIST ENTRY
SET ADDR INVITATION CHARS FOR ERP
YES
MORE ENTRIES TO CHECK
RETURN
NO
INACTIVE ENTRIES CHECKED
YES
CLEAR SOURCE IN LCB
NO
DERIVE NUMBER INACTIVE ENTRIES
RESET CONTROL UNIT ADDRESS

Chart R0H (R0H) LINE END APPENDAGE

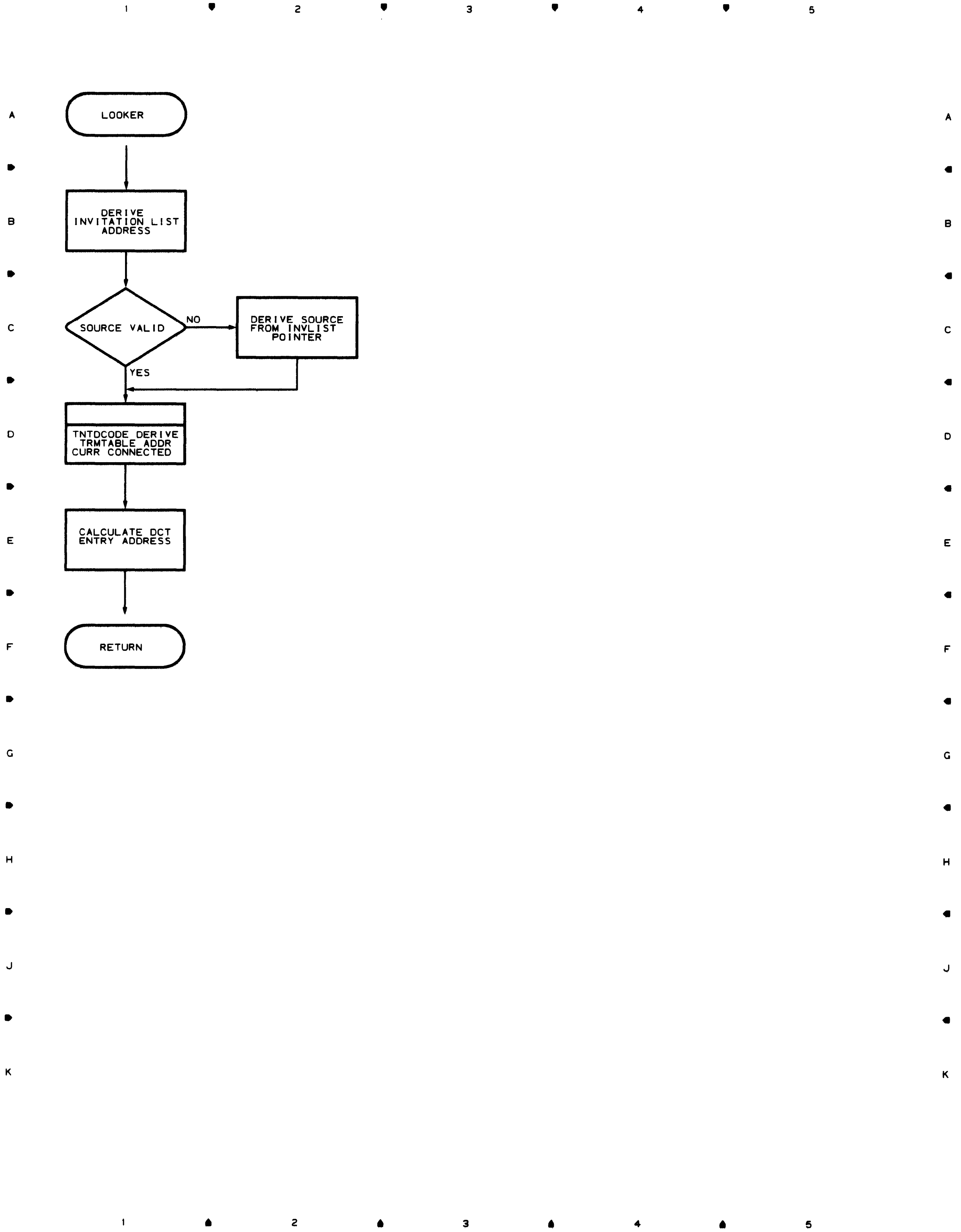
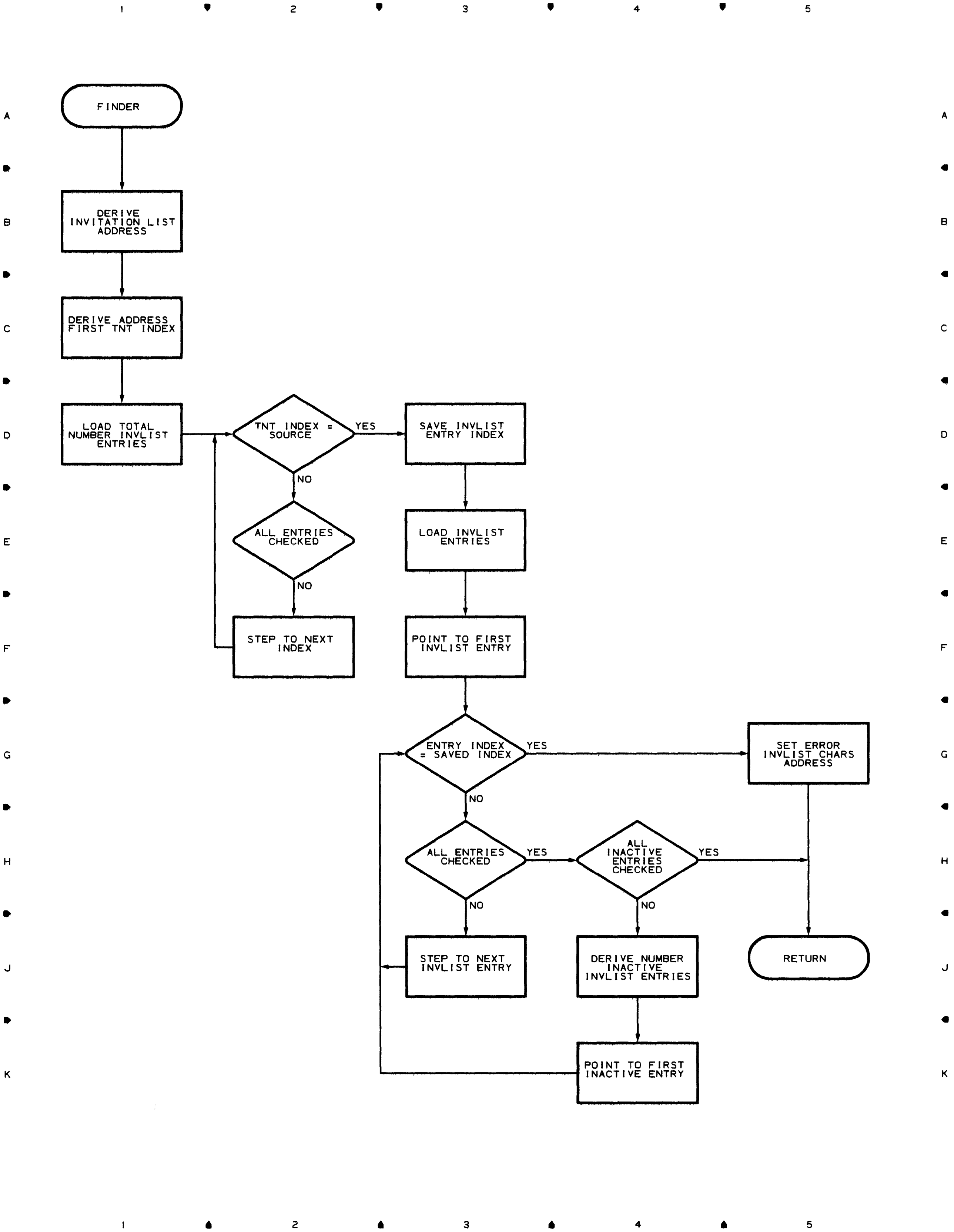
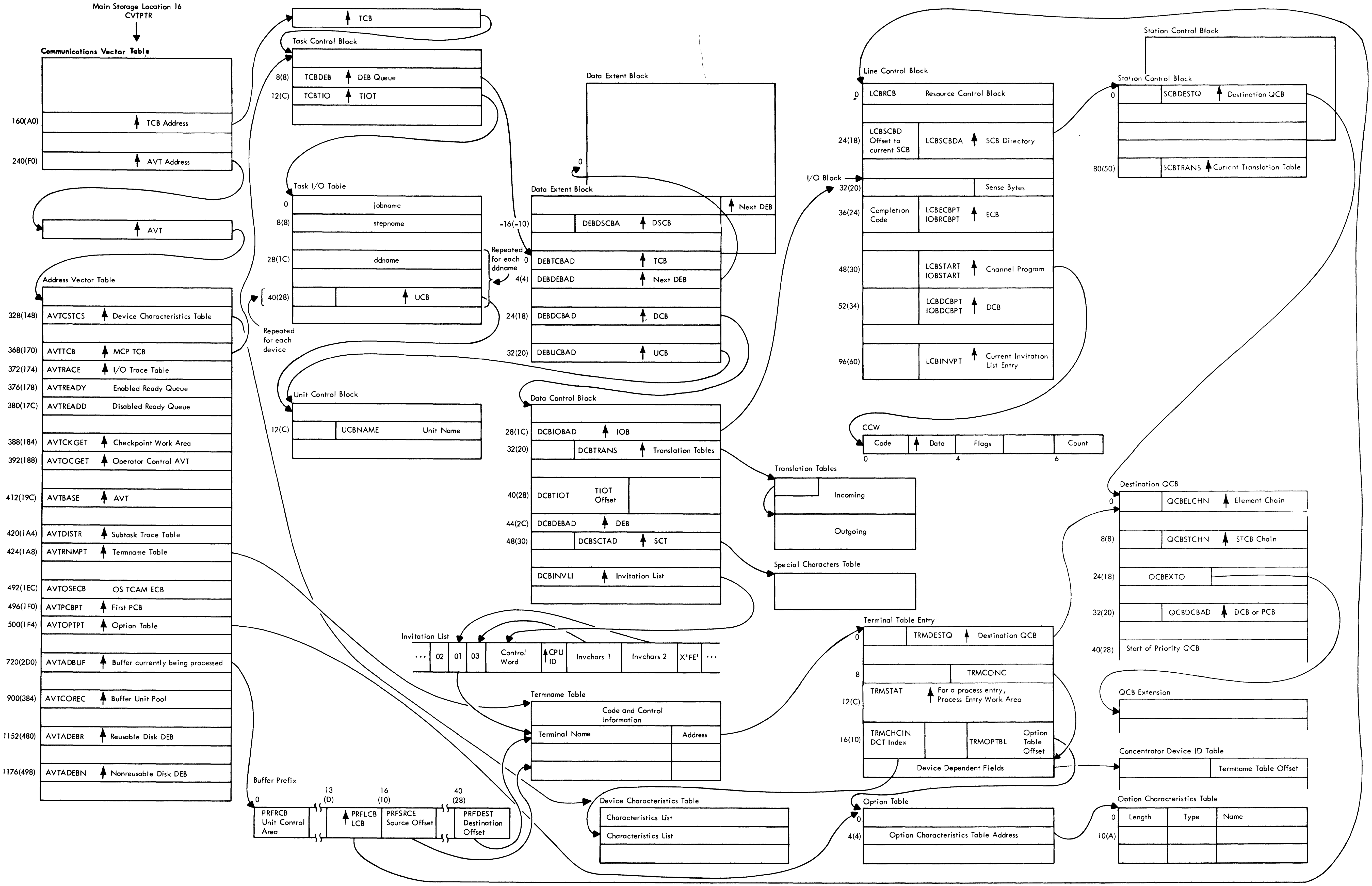

1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
LOOKER
DERIVE INVITATION LIST ADDRESS
SOURCE VALID
NO
YES
DERIVE SOURCE FROM INVLIST POINTER
TNTDCODE DERIVE TRMTABLE ADDR CURR CONNECTED
CALCULATE DCT ENTRY ADDRESS
RETURN

Chart R0I (R0I) LINE END APPENDAGE



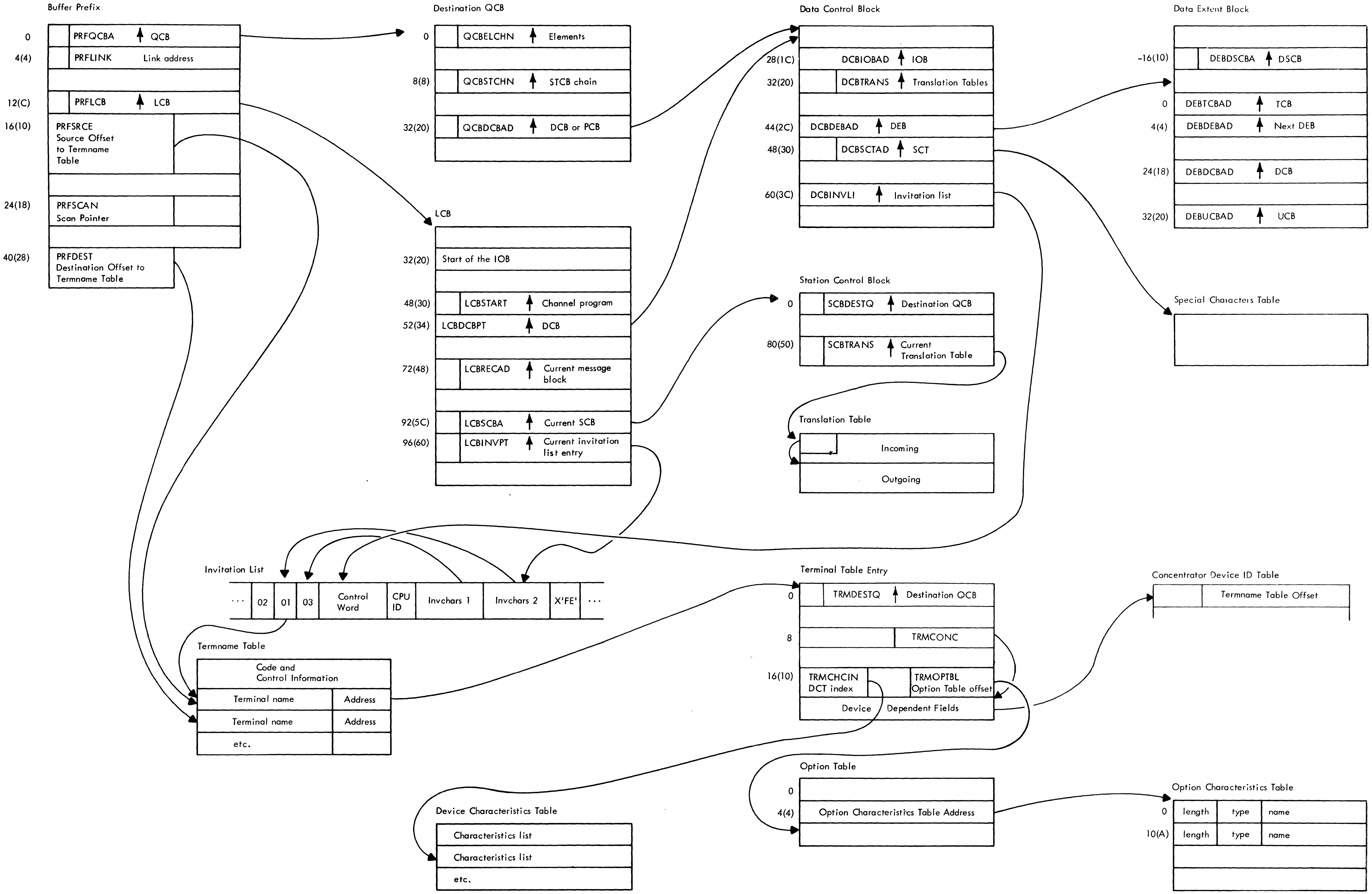
1
2
3
4
5
A
B
C
D
E
F
G
H
J
K
FINDER
DERIVE INVITATION LIST ADDRESS
DERIVE ADDRESS FIRST TNT INDEX
LOAD TOTAL NUMBER INVLIST ENTRIES
TNT INDEX = SOURCE
YES
NO
SAVE INVLIST ENTRY INDEX
ALL ENTRIES CHECKED
NO
LOAD INVLIST ENTRIES
STEP TO NEXT INDEX
POINT TO FIRST INVLIST ENTRY
ENTRY INDEX = SAVED INDEX
YES
NO
SET ERROR INVLIST CHARS ADDRESS
ALL ENTRIES CHECKED
YES
NO
ALL INACTIVE ENTRIES CHECKED
YES
NO
STEP TO NEXT INVLIST ENTRY
DERIVE NUMBER INACTIVE INVLIST ENTRIES
RETURN
POINT TO FIRST INACTIVE ENTRY

TCAM Control Block Linkages


Main Storage Location 16
CVTPTR
Communications Vector Table
160(A0)
TCB Address
240(F0)
AVT Address
AVT
Address Vector Table
328(148) AVTCSTCS Device Characteristics Table
368(170) AVTTCB MCP TCB
372(174) AVTRACE I/O Trace Table
376(178) AVTREADY Enabled Ready Queue
380(17C) AVTREADD Disabled Ready Queue
388(184) AVTCKGET Checkpoint Work Area
392(188) AVTOCGET Operator Control AVT
412(19C) AVTBASE AVT
420(1A4) AVTDISTR Subtask Trace Table
424(1A8) AVTRNMPT Termname Table
492(1EC) AVTOSECB OS TCAM ECB
496(1F0) AVTPCBPT First PCB
500(1F4) AVTOPTPT Option Table
720(2D0) AVTADBUF Buffer currently being processed
900(384) AVTCOREC Buffer Unit Pool
1152(480) AVTADEBR Reusable Disk DEB
1176(498) AVTADEBN Nonreusable Disk DEB
TCB
Task Control Block
8(8) TCBDEB DEB Queue
12(C) TCBTIO TIOT
Task I/O Table
0 jobname
8(8) stepname
28(1C) ddname
40(28) UCB
Repeated for each ddname
Repeated for each device
Unit Control Block
12(C) UCBNAME Unit Name
Data Extent Block
0
Data Extent Block
Next DEB
-16(-10) DEBDSCBA DSCB
0 DEBTCBAD TCB
4(4) DEBDEBAD Next DEB
24(18) DEBDCBAD DCB
32(20) DEBUCBAD UCB
Data Control Block
28(1C) DCBIOBAD IOB
32(20) DCBTRANS Translation Tables
40(28) DCBTIOT TIOT Offset
44(2C) DCBDEBAD DEB
48(30) DCBSCTAD SCT
DCBINVLI Invitation List
Invitation List
... 02 01 03 Control Word CPU ID Invchars 1 Invchars 2 X'FE' ...
Termname Table
Code and Control Information
Terminal Name
Address
Buffer Prefix
0 13 (D) 16 (10) 40 (28)
PRFRCB Unit Control Area
PRFLCB LCB
PRFSRCE Source Offset
PRFDEST Destination Offset
Device Characteristics Table
Characteristics List
Characteristics List
Line Control Block
0 LCBRCB Resource Control Block
24(18) LCBSCBD Offset to current SCB LCBSCBDA SCB Directory
I/O Block
32(20) Sense Bytes
36(24) Completion Code LCBECBPT IOBRCBPT ECB
48(30) LCBSTART IOBSTART Channel Program
52(34) LCBDCBPT IOBDCBPT DCB
96(60) LCBINVPT Current Invitation List Entry
CCW
Code Data Flags Count
0 4 6
Translation Tables
Incoming
Outgoing
Special Characters Table
Terminal Table Entry
0 TRMDESTQ Destination QCB
8 TRMCONC
12(C) TRMSTAT For a process entry, Process Entry Work Area
16(10) TRMCHCIN DCT Index TRMOPTBL Option Table Offset
Device Dependent Fields
Option Table
0
4(4) Option Characteristics Table Address
Station Control Block
Station Control Block
0 SCBDESTQ Destination QCB
80(50) SCBTRANS Current Translation Table
Destination QCB
0 QCBELCHN Element Chain
8(8) QCBSTCHN STCB Chain
24(18) QCBEXTO
32(20) QCBDCBAD DCB or PCB
40(28) Start of Priority QCB
QCB Extension
Concentrator Device ID Table
Termname Table Offset
Option Characteristics Table
0 Length Type Name
10(A)

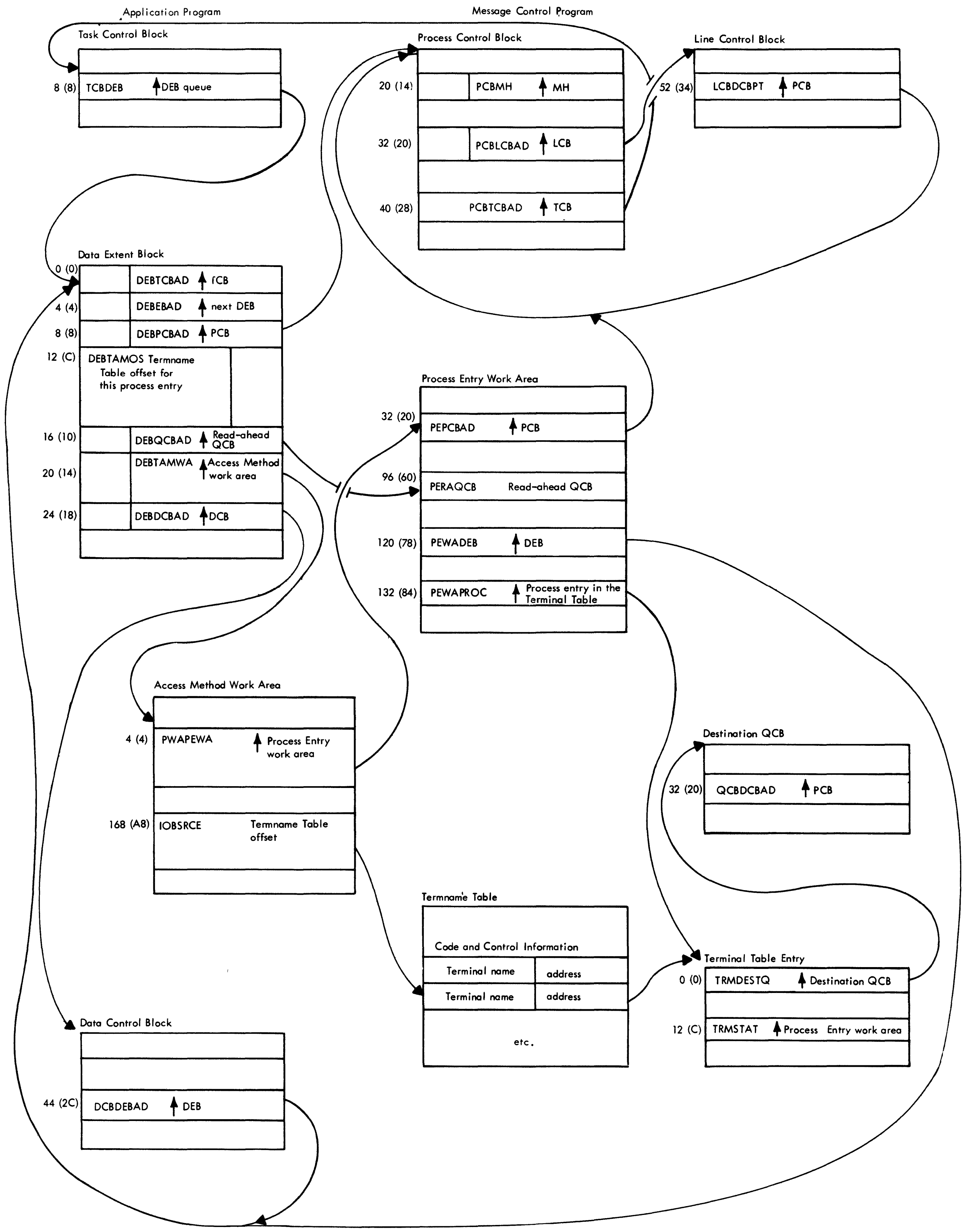
Insert foldout page 395 at end of book.

Linkages from a TCAM Buffer Prefix



Insert foldout page 397 at end of book.

Linkage among Storage Areas in the MCP and an Application Program



Insert foldout page 399 at end of book.

Address Vector Table

The TCAM address vector table (AVT) is assembled at the beginning of a Message Control Program. The basic AVT occupies bytes 0-1055 and is assembled when ENVIRON=TSO on the INTRO macro. If main-storage-only queuing is specified (DISK=NO,ENVIRON=TCAM or MIXED), the AVT occupies bytes 0—1079. When disk queuing is used, the AVT occupies bytes 0—1225.

When either the Disk Message Queues Open or the Line Group Open routine loads the TCAM Dispatcher, the routine also places in the CVT a pointer to a field that contains the address of the AVT. The fields in the AVT are initialized both during the assembly of the INTRO macro and at MCP initialization time.

The AVT provides work areas in which TCAM routines can store variables. The AVT also contains constant areas shared by more than one macro expansion or TCAM subroutine. The AVT contains five save areas—one for the MCP, one for each level of control in the MCP, and one for disabled code. For efficient internal control, the AVT also contains module addresses, special elements, control bytes and bits, and the two ready queues.

The DSECT names of the AVT fields are shown in the following layout. A more detailed description of the fields and the data they might contain follows the DSECT layout.

IEDQAVTD

0 (0)	**AVTSAVE1**	Message Control Program Save Area
+72 (48)	**AVTSAVE2**	Dispatcher Save Area
+144 (90)	**AVTSAVE3**	Subtask Save Area
+216 (D8)	**AVTSAVE4**	First Level Subroutine Save Area
+288 (120)	**AVTSAVEX**	Disabled Save Area
+320 (140)	**AVTDLQ**	DLQ=Termname
+328 (148)	**AVTCSTCS**	Address of the First Entry in the Device Characteristics Table
+332 (14C)	**AVTDPARM**	Disabled Parameter List
+336 (150)	**AVTDOUBX**	Disabled Doubleword Scratch Area
+344 (158)	**AVTDOUBL**	Enabled Scratch Area
+352 (160)	**AVTCTLCH**	Operator Control Characters
+360 (168)	**AVTPASWD**	Password
+368 (170)	**AVTTCB**	Address of the Message Control Program's TCB; Set by OPEN
+372 (174)	**AVTRACE**	Trace Table Address
+376 (178)	**AVTREADY**	Enabled Ready Queue
+380 (17C)	**AVTREADD**	Disabled FIFO Ready Queue
+388 (184)	**AVTCKGET**	Checkpoint Work Area Address
+392 (188)	**AVTOCGET**	Operator Control Work Area Address

Offset	Field	Description
+396 (18C)	AVTEXA2S	Executed Instructions to Save the User's Registers
+402 (192)	AVTEXS2A	Executed Instructions to Save the User's Registers
408 (198)	AVTPARM	Address of Parameters
412 (19C)	AVTBASE	Address of the AVT
416 (1A0)	AVTPARM3	Address of Additional Optional Parameters
420 (1A4)	AVTDISTR	Address of the Dispatcher Subtask Trace Table
424 (1A8)	AVTRNMPT	Address of the Termname Table
428 (1AC)	AVTRDYA	Address of User Exit in the READY Macro Expansion
432 (1B0)	AVTBSCAN	Line End Appendage BSC Message Scan
436 (1B4)	AVTRARTN	Address of Routine to Update Line I/O Trace Table
440 (1B8)	AVTPOST	Tpost Parameter List Used by Operator Control
448 (1C0)	AVTSPLPT	Start Parameter List Pointer; Set by INTRO
452 (1C4)	AVTCIB	CIB=Integer
453 (1C5)	AVTNCKPR	CKREQS=Integer
454 (1C6)	AVTNOLBF	LNUNITS=Integer
456 (1C8)	AVTAS	Address of the Hold/Release Terminal Routine
460 (1CC)	AVTCKTCB	Address of the Checkpoint TCB
464 (1D0)	AVTOCTCB	Address of the Operator Control TCB
468 (1D4)	AVTOLTCB	Address of the On-Line Test TCB
472 (1D8)	AVTCWTCB	Address of the FE Common Write TCB

476 (1DC)		**AVTCWECA** FE Common Write ECB
480 (1E0)		**AVTCKECA** Checkpoint ECB
484 (1E4)		**AVTOLECA** On-Line Test ECB
488 (1E8)		**AVTOPECA** Operator Control ECB
492 (1EC)		**AVTOSECB** ECB Used by the Dispatcher to Cause TCAM Task to be in the Wait State
496 (1F0)		**AVTPCBPT** Address of the First Process Control Block
500 (1F4)		**AVTOPTPT** Address of the Option Table
504 (1F8)		**AVTKA02** Address of the I/O Generator in the Activate Subtask
508 (1FC)		**AVTREXIT** TREXIT=Name
512 (200)		**AVTCRSRF** CROSSRF=Integer
516 (204)		**AVTCOMPT** Address of Communications Parameter List
520 (208)		**AVTUI** Address of the User Interface Routine
524 (20C)		**AVTE8** Address of the Application Program Binary Search
528 (210)	**AVTOLIST** OLTEST-Integer	**AVTHG02** Address of the Routine to Remove a Checkpoint Element from the Time Delay QCB
532 (214)		**AVTAL** Address of the Scan at Offset Routine
536 (218)		**AVTGD** Address of the Buffer Association Routine
540 (21C)		**AVTGT** Address of the Transparent CCW Builder Routine (IEDQGT)
544 (220)		**AVTAX** Address of the Buffer Scan Routine

548 (224)		**AVTEA** Address of the TCAM Dispatcher
552 (228)		**AVTHA** Address of the Receive Scheduler
556 (22C)	**AVTSCOPT** Scheduler Option Field	**AVTHD** Address of the Send Scheduler
560 (230)		**AVTEW** Address of the Get Scheduler
564 (234)		**AVTEC** Address of the Put Scheduler
568 (238)		**AVTEZ** Address of the Get FIFO Scheduler
572 (23C)		**AVTBZ** Address of the Log Scheduler
576 (240)		**AVTR1** Address of the Dial Scheduler
580 (244)		**AVTHB** Address of the Buffered Scheduler
584 (248)		**AVTE7** Address of the Retrieve Scheduler
588 (24C)		Address of the Local Receive Scheduler
592 (250)		**AVTCSCH** Address of the Concentrator Send Scheduler
596 (254)		Reserved
600 (258)		Reserved
604 (25C)		**AVTCMBSS** Address of the COMMBUF Send Scheduler
608 (260)		Reserved
612 (264)		Reserved
616 (268)		**AVTABEND** BALR 1,0

620 (26C)	B IEDSVC13	
624 (270)	**AVTDMECB**	Dummy Line I/O ECB
628 (274)	**AVTA3TL**	Address of the Translate List for the Dynamic Translation Routine (IEDQA3)
632 (278)	**AVTTONE**	WTTONE=Integer; Address of World Trade Tone Characters
636 (27C)	**AVTNX**	Address of the Operator Awareness Message Routing Routine
640 (280)	**AVTIOT**	Address of Line I/O Trace Table Handler
644 (284)	**AVTHI**	Address of System Delay QCB
648 (288)	**AVTHK**	Address of the Stopline QCB
652 (28C)	**AVTCKRMV**	Request for Removal of Checkpoint Routine Element from Time Delay Queue
668 (29C)	**AVTCKELE**	Checkpoint Request Element, Start of Checkpoint QCB
676 (2A4)	**AVTSCBSZ**	SCB Size
677 (2A5)	**AVTCKQAD**	Address of the Checkpoint QCB
680 (2A8)	**AVTCKELF**	Checkpoint Request Element Flags
681 (2A9)	**AVTCPRCD**	CPRCDS=Integer
682 (2AA)	**AVTCKELV**	CPINTVL=Time Interval
684 (2AC)	**AVTCKTIM**	Time of Day Interrupt
686 (2AE)		Index to QCB Address
287 (2AF)	**AVTOPERL**	OPEN Error Locator
688 (2B0)	**AVTOPXCL**	ID of OPEN Module with Error
690 (2B2)	**AVTOPERT**	OPEN Error Type
691 (2B3)	**AVTCKBYT**	Status at Checkpoint and Time Delay
692 (2B4)	**AVTOPETR**	INTRO Return Code
	AVTHG01	Address of Time Delay Subroutine
696 (2B8)	**AVTCKLNK**	Link Field On the Time Queue
700 (2BC)	**AVTDELEM**	Dummy Last Element
704 (2C0)	**AVTDELAD**	Address of the Dummy Last Element

708 (2C4)	**AVTCCELE**	Incident Checkpoint Request Element
716 (2CC)	**AVTCLRHI**	Mask for Clearing Left Two Bytes of a Register
718 (2CE)	**AVTHFF**	Half Word of X'FFFF'
720 (2D0)	**AVTADBUF**	Address of Buffer
724 (2D4)	**AVT2260L**	Address of 2260 Local Receive Scheduler
728 (2D8)	**AVTSYSER**	System Error Flags
729 (2D9)	**AVTMSGS**	List of Optional VCONs
732 (2DC)	**AVTCBQCB**	Address of the COMMBUF Master QCB
736 (2E0)	**AVTSUPPT**	Address of the Start-up Message QCB
740 (2E4)	**AVTTSOPT**	Address of the Time Sharing Input QCB
744 (2E8)	**AVTOCQPT**	Address of the Application Program Open/Close Routine
748 (2EC)	**AVTDELYB**	Time Delay Subtask QCB
764 (2FC)	**AVTREFTM**	Reference Time
766 (2FE)	**AVTINOUT**	Dummy INEND/OUTEND Parameter List
768 (300)	**AVTIMQPS**	SVC 102 Parameter
776 (308)	**AVTTIMQ**	Time Delay Queue
780 (30C)	**AVTBFREB**	Buffer Request QCB
792 (318)	**AVTBFRTB**	Buffer Return QCB
804 (324)	**AVTCKPTB**	Checkpoint QCB
816 (330)	**AVTOPCOB**	Operator Control QCB
828 (33C)	**AVTOLTQB**	On-Line Test QCB

840 (348)	AVTACTIB	Activate QCB
852 (354)	AVTCLOSB	Closedown QCB
864 (360)	AVTCPRMB	QCB to Remove an Element from the Time Delay QCB
876 (36C)	AVTDSIOB	Disk I/O QCB
888 (378)	AVTCPBCB	CPB Cleanup QCB
900 (384)	AVTCOREC	Buffer Unit Pool Address
904 (388)	AVTCADDR	Main Storage Queue Count
908 (38C)	AVTFZERO	Fullword of All Zeros
912 (390)	AVTCAREA	FE Common Write Interface Area – Address of the Patch Module
916 (394)	AVTCWPM1	FE Common Write Interface Area – First Parameter Pointer
920 (398)	AVTCWEC1	FE Common Write Interface Area – First ECB
924 (39C)	AVTCWFL1	FE Common Write – Flag Byte 1
925 (39D)	AVTCWFL2	FE Common Write – Flag Byte 2
926 (39E)	AVTCWTS1	FE Common Write – Flag Byte 3
927 (39F)	AVTCWTS2	FE Common Write – Flag Byte 4
928 (3A0)	AVTCWPM2	FE Common Write Interface Area – Second Parameter Pointer
932 (3A4)	AVTCWEC2	FE Common Write Interface Area – Second ECB
936 (3A8)	AVTAFE10	Address of FE STCB Trace Dump Routine
940 (3AC)	AVTAFE20	Address of FE I/O Trace Dump Routine
944 (3B0)	AVTAFE30	Address of FE Buffer Dump Routine
948 (3B4)	AVTCWINT	FE Common Write Interface Area – Patch Area

1012 (3F4) **AVTGETMN** GETMAIN Parameter List			
		1022 (3FE) **AVTHA2** Constant = 2	
1024 (400) **AVTHA3** Constant = 3		1026 (402) **AVTHA4** Constant = 4	
1028 (404) **AVTHA7** Constant = 7		1030 (406) **AVTHA16** Constant = 16	
1032 (408) **AVTKEYLE** KEYLEN on the Message Queues		1034 (40A) **AVTLNCNT** Number of Lines Opened	
1036 (40C) **AVTOPCNT** Number of Lines Taken by Operator Control		1038 (40E) **AVTOPCON** Termname Table Offset to the Primary Operator Control Terminal	
1040 (410) **AVTAVFCT** Number of Buffers in the Buffer Units Pool		1042 (412) **AVTSMCNT** Number of Lines Serviced by the Start-up Message Subtask	
1044 (414) **AVTINTLV** Number of Seconds of a System Delay INTVAL=Integer		1046 (416) **AVTDLQX** Offset in Termname Table of the Dead Letter Queue	
1048 (418) **AVTDUMBR** Dummy Line Trace Table Update		1050 (41A) **AVTBIT1** Flag Bits	1051 (41B) **AVTBIT2** Flag Bits
1052 (41C) **AVTBIT3** Flag Bits	1053 (41D) **AVTCKRST** RESTART=Integer	1054 (41E) **AVTDSKCT** Number of Buffers on CPBs	
1056 (420) **AVTHM02** Address of the Destination Scheduler			
1060 (424) **AVTCMIN** MSMIN=Integer			
1064 (428) **AVTCMAX** MSMAX=Integer			
1068 (42C) **AVTTOTNC** Number of Records in the Entire Message Queues Data Set (MSUNITS=Integer)			
1072 (430) **AVTNCPBQ** Queue of Buffers and ERBs Waiting to be Processed			
1080 (438) **AVTFL** Address of the Disk EXCP Driver Routine			
1084 (43C) **AVTIA** Address of the REUS part of the Reusability — Copy Subtask			
1088 (440) **AVTCOPY** Copy Subtask QCB Pointer			

1092 (444)	**AVTDKAPQ** Queue of CPBs to be Processed by CPB Cleanup (Disabled)
1100 (44C)	**AVTDKENQ** Queue of CPBs to be Processed by CPB Cleanup (Enabled)
1108 (454)	**AVTNOBFQ** Queue of CPBs without Buffers
1116 (45C)	**AVTREUSQ** Reserved
1124 (464)	**AVTINCPQ** Queue of CPBs Requesting I/O be Done by EXCP Driver
1132 (46C)	**AVTFCPB** Address of the CPB Free Pool
1136 (470)	**AVTCPBPT** Address of the CPB Free Pool to be Freed by Disk Close
1140 (474)	**AVTIOBR** Address of a Series of IOBs, One for Each Extent of the Reusable Disk Queue
1144 (478)	**AVTIOBN** Address of a Series of IOBs, One for Each Extent of the Nonreusable Disk Queue
1148 (47C)	**AVTLODPT** Absolute Disk Record Number Indicating Time to Activate the REUS part of the Reusability – Copy Subtask
1152 (480)	**AVTADEBR** Address of the DEBEOEA Field in the DEB for the Reusable Disk Message Queues Data Set
1156 (484)	**AVTNOVOR** Number of Extents in the Reusable Disk Message Queues Data Set
1160 (488)	**AVTRCTRR** Number of Records Per Track On the Reusable Disk Message Queues Data Set
1164 (48C)	**AVTTRCYR** Number of Tracks Per Cylinder On the Reusable Disk Message Queues Data Set
1168 (490)	**AVTTOTNR** Number of Records in the Entire Reusable Disk Message Queues Data Set
1172 (494)	**AVTVOLRR** Product of the Number of Extents Times the Number of Records Per Track On the Reusable Disk Message Queues Data Set
1176 (498)	**AVTADEBN** Address of the DEBEOEA Field in the DEB for the Nonreusable Disk Message Queues Data Set
1180 (49C)	**AVTNOVON** Number of Extents in the Nonreusable Disk Message Queues Data Set

1184 (4A0)	**AVTRCTRN** Number of Records Per Track On the Nonreusable Disk Message Queues Data Set
1188 (4A4)	**AVTTRCYN** Number of Tracks Per Cylinder On the Nonreusable Disk Message Queues Data Set
1192 (4A8)	**AVTTOTNN** Number of Records in the Entire Nonreusable Disk Message Queues Data Set
1196 (4AC)	**AVTVOLRN** Product of the Number of Extents Times the Number of Records Per Track On the Nonreusable Disk Message Queues Data Set
1200 (4B0)	**AVTHRESN** Absolute Record Number (Threshold) to Cause Closedown Due to the Filling of the Nonreusable Disk Message Queues Data Set
1204 (4B4)	**AVTNADDR** Index to Nonreusable Disk Relative Record Number of the Next Record to be Assigned
1208 (4B8)	**AVTRADDR** Index to Reusable Disk Relative Record Number of the Next Record to be Assigned
1212 (4BC)	**AVTHRESE** Nonreusable Threshold Closedown Element
1223 (4C7)	**AVTHRESS** Status Completion Code
1224 (4C8)	**AVTCPBNO** CPB=Integer
1226 (4CA)	Reserved

Offset		*Name*	*Bytes*	*Description*
0	(0)	AVTSAVE1	72	Message Control Program save area
72	(48)	AVTSAVE2	72	Dispatcher save area
144	(90)	AVTSAVE3	72	Subtask save area
216	(D8)	AVTSAVE4	72	First-level subroutine save area
288	(120)	AVTSAVEX	40	Disabled save area
320	(140)	AVTDLQ	8	At assembly time, set by the DLQ=termname operand of the INTRO macro. After the termname table is sorted, this value is moved to AVTDLQX and this field (AVTDLQ) is overlaid and used as part of the disabled save area.
328	(148)	AVTCSTCS	4	Address of the first entry in the device characteristics table
332	(14C)	AVTDPARM	4	Disabled parameter list (used with AVTDOUBX)
336	(150)	AVTDOUBX	8	Disabled doubleword scratch area
344	(158)	AVTDOUBL	8	Enabled doubleword scratch area
352	(160)	AVTCTLCH	8	Operator Control characters
360	(168)	AVTPASWD	8	Message Control Program password
368	(170)	AVTTCB	4	Address of the Message Control Program TCB—set by the first open routine
372	(174)	AVTRACE	4	Line I/O interrupt trace table address
The following are the ready queues for the TCAM Dispatcher:				
376	(178)	AVTREADY	4	Enabled ready queue—points to the first item in the chain of elements that is to be processed by the TCAM Dispatcher
380	(17C)	AVTREADD	8	Disabled FIFO ready queue—controls the chain of elements tposted from disabled routines. The first word points to the first element; the second word points to the last element on the chain.
388	(184)	AVTCKGET	4	Address of the checkpoint work area; set after a successful GETMAIN is completed by the Checkpoint Open routine
392	(188)	AVTOCGET	4	Address of the operator control work area
396	(18C)	AVTEXA2S	6	Instructions to be executed to save the user's registers
402	(192)	AVTEXS2A	6	Continuation of the instructions to be executed to save the user's registers
408	(198)	AVTPARM	4	Address of the parameters to be processed
412	(19C)	AVTBASE	4	Address of the AVT
416	(1A0)	AVTPARM3	4	Address of additional optional parameters
420	(1A4)	AVTDISTR	4	Address of the Dispatcher's subtask trace table
424	(1A8)	AVTRNMPT	4	Address of the termname table
428	(1AC)	AVTRDYA	4	User exit address in the READY macro expansion
432	(1B0)	AVTBSCAN	4	Line End Appendage address for BSC message scan
436	(1B4)	AVTRART	4	Address of the routine to update the line I/O interrupt trace table
440	(1B8)	AVTPOST	8	Tpost parameter list used by Operator Control
448	(1C0)	AVTSPLPT	4	Start parameter list address—set by the INTRO macro expansion

Offset	*Name*	*Bytes*	*Description*
452 (1C4)	AVTCIB	1	The maximum number of command input blocks that can be utilized at any one time in the TCAM system—set by the CIB=integer operand of the INTRO macro
453 (1C5)	AVTNCKPR	1	The maximum decimal number of destination queues in use at any time for application programs that use a CKREQ macro—set by the CKREQS=integer operand of the INTRO macro
454 (1C6)	AVTNOLBF	2	Specifies the number of buffer units that may be used to build buffers for messages—set by the LNUNITS=integer operand of the INTRO macro
456 (1C8)	AVTAS	4	Address of the Hold/Release Terminal routine
The following are the addresses of the TCBs of the attached tasks:			
460 (1CC)	AVTCKTCB	4	Address of the Checkpoint TCB
464 (1D0)	AVTOCTCB	4	Address of the Operator ControlTCB
468 (1D4)	AVTOLTCB	4	Address of the On-Line Test TCB
472 (1D8)	AVTCWTCB	4	Address of the FE Common Write TCB
The following are the event control blocks (ECBs) for the attached tasks:			
476 (1DC)	AVTCWECA	4	FE Common Write ECB
480 (1E0)	AVTCKECA	4	Checkpoint ECB
484 (1E4)	AVTOLECA	4	On-Line Test ECB
488 (1E8)	AVTOPECA	4	Operator Control ECB
492 (1EC)	AVTOSECB	4	ECB used by the Dispatcher to cause the TCAM task to be in the WAIT state
496 (1F0)	AVTPCBPT	4	Address of the first process control block (PCB)
500 (1F4)	AVTOPTPT	4	Address of the option table
504 (1F8)	AVTKA02	4	Address of the I/O Generator routine in the Activate subtask
508 (1FC)	AVTREXIT	4	Address of a user-written routine to be given control when all entries in the TCAM I/O interrupt trace table have been filled—set by the TREXIT=name operand of the INTRO macro
512 (200)	AVTCRSRF	4	Specifies the number of entries in the cross-reference table—set by the CROSSRF=integer operand of the INTRO macro; replaced by the address of the cross-reference table.
516 (204)	AVTCOMPT	4	Address of the communications parameter list
520 (208)	AVTUI	4	Address of the User Interface routine
524 (20C)	AVTE8	4	Address of the Application Program Binary Search routine
528 (210)	AVTHG02	4	Address of the routine to remove a checkpoint element from the time delay QCB
528 (210)	AVTOLTST	1	Set by the OLTEST=integer operand of the INTRO macro
532 (214)	AVTAL	4	Address of the Scan at Offset routine
536 (218)	AVTGD	4	Address of the Buffer Association routine
540 (21C)	AVTGT	4	Address of the Transparent Transmission CCW Building routine (IEDQGT)

Offset	*Name*	*Bytes*	*Description*
544 (220)	AVTAX	4	Address of the Buffer Scan routine
548 (224)	AVTEA	4	Address of the TCAM Dispatcher
552 (228)	AVTHA	4	Address of the Receive Scheduler
556 (22C)	AVTHD	4	Address of the Send Scheduler
556 (22C)	AVTSCOPT	1	Scheduler option field

Bit Definitions:

Name	*Bit*	*Value*	*Description*
AVTCMBUF	3	X'10'	Common buffer transmission
AVTCONC	4	X'08'	Concentrator mixed
AVTCONCO	5	X'04'	Concentrator only
AVTN2741	6	X'02'	No 2741 and no TSO
AVTNDIAL	7	X'01'	No dial

Offset	*Name*	*Bytes*	*Description*
560 (230)	AVTEW	4	Address of the Get Scheduler
564 (234)	AVTEC	4	Address of the Put Scheduler
568 (238)	AVTEZ	4	Address of the Get FIFO Scheduler
572 (23C)	AVTBZ	4	Address of the Log Scheduler
576 (240)	AVTR1	4	Address of the Dial Scheduler
580 (244)	AVTHB	4	Address of the Buffered Scheduler
584 (248)	AVTE7	4	Address of the Retrieve Scheduler
588 (24C)		4	Address of the Local Receive Scheduler
592 (250)	AVTCSCH	4	Address of the Concentrator Send Scheduler

The following are the special elements used in TCAM:

Offset	*Name*	*Bytes*	*Description*
596 (254)		8	Reserved
604 (256)	AVTCMBSS	4	Address of the COMMBUF Send Scheduler
608 (260)		8	Reserved
616 (268)		2	Reserved
618 (26A)	AVTABEND	6	This field contains the following code for an 045 abend: BALR 1,0 B IEDSVC13
624 (270)	AVTDMECB	4	Address of the dummy line I/O ECB
628 (274)	AVTA3TL	4	Address of the translate list for the Dynamic Translation routine
632 (278)	AVTTONE	4	Contains either a zero or the address of a field consisting of 2 halfwords; the first contains the WTTONE integer from the INTRO macro and the second a X'FF' representing the number of characters specified by WTTONE.
636 (27C)	AVTNX	4	Address of Operator Awareness Message Routing routine
640 (280)	AVTIOT	4	Address of Line I/O Trace Table routine
644 (284)	AVTHI	4	Address of system delay QCB
648 (288)	AVTHK	4	Address of stopline QCB

Offset	*Name*	*Bytes*	*Description*
652 (28C)	AVTCKRMV	16	Request for removal of the checkpoint request element from the time delay queue
668 (29C)	AVTCKELE	8	Checkpoint request element—the Time Delay or Reusability subtasks tpost this element to start the checkpoint routines
676 (2A4)	AVTSCBSZ	1	Specifies the number of bytes in the SCB including the save area for the user's registers.
677 (2A5)	AVTCKQAD	3	Address of checkpoint QCB
680 (2A8)	AVTCKELF	1	Checkpoint request element flag bits

Bit definitions:

Name	*Bit*	*Value*	*Description*
AVTCRDYN	0	X'80'	Checkpoint requested by the READY macro expansion
AVTCMCPN	1	X'40'	Checkpoint requested by the MCPCLOSE macro
	2	X'20'	Unused
AVTCINCN	3	X'10'	Checkpoint requested by the No Incident Records routine
AVTCCLCN	4	X'08'	Closedown completion bit
AVTCPIPN	5	X'04'	Checkpoint in progress bit
AVTCRTLN	6	X'02'	Checkpoint requested
AVTWARM	7	X'01'	Warm restart

Offset	*Name*	*Bytes*	*Description*
681 (2A9)	AVTCPRCD	1	The number of environment checkpoint records to be retained in the checkpoint data set at any one time—set by the CPRCDS=integer operand of the INTRO macro
682 (2AA)	AVTCKELV	2	The number of seconds between environment checkpoints—set by the CPINTVL=integer operand of the INTRO macro
684 (2AC)	AVTCKTIM	2	Time-of-day interrupt
686 (2AE)		1	Index to the QCB address
687 (2AF)	AVTOPERL	1	Open error location
688 (2B0)	AVTOPXCL	2	Module ID of the routine that has an error
690 (2B2)	AVTOPERT	1	Specifies the type of open error that occurred
691 (2B3)	AVTCKBYT	1	Specifies the checkpoint and time delay status
692 (2B4)	AVTOPETR	1	INTRO return code
692 (2B4)	AVTHG01	4	Address of the Time Delay subroutine
696 (2B8)	AVTCKLNK	4	Link field on the time queue
700 (2BC)	AVTDELEM	4	Dummy last element—used as the last element in any QCB's (or the ready queue's) element chain
704 (2C0)	AVTDELAD	4	Address of the dummy last element
708 (2C4)	AVTCCELE	8	Incident checkpoint request element—tposted by the Operator Control task to request an incident checkpoint
716 (2CC)	AVTCLRHI	2	Mask used with the next halfword to clear the left two bytes of a register

Offset		*Name*	*Bytes*	*Description*		
718	(2CE)	AVTFF	2	Halfword equal to X'FFFF'		
720	(2D0)	AVTADBUF	4	Address of the buffer currently being processed		
724	(2D4)	AVT2260L	4	Address of the 2260 Local Receive Scheduler		
728	(2D8)	AVTSYSER	1	System error flag byte—set by the operands of the INTRO macro as follows:		
			Name	*Bit*	*Value*	*Description*
			AVTCMINN	0	X'80'	The number of main-storage queue units less than that specified by MSMIN=integer
			AVTCMAXN	1	X'40'	The number of main-storage queue units more than that specified by MSMAX=integer
				2-7		Reserved
729	(2D9)	AVTMSGS	3	Address of a list of optional VCONs		

The following is a list of pointers to QCBs:

732	(2DC)	AVTCBQCB	4	Address of the COMMBUF master QCB
736	(2E0)	AVTSUPPT	4	Address of the start-up message QCB
740	(2E4)	AVTTSOPT	4	Address of the time sharing input QCB
744	(2E8)	AVTOCQPT	4	Address of the application program Open/Close subtask

The following is a list of required QCBs:

748	(2EC)	AVTDELYB	20	Time Delay subtask QCB
764	(2FC)	AVTREFTM	2	Represents the reference time, current time of day, plus or minus 6 hours
766	(2FE)	AVTINOUT	2	Dummy INEND/OUTEND parameter list
768	(300)	AVTIMQPS	8	SVC 102 parameter—to tpost the time QCB to itself at the interrupt
776	(308)	AVTTIMQ	4	Time delay queue
780	(30C)	AVTBFREB	12	Buffer request QCB
792	(318)	AVTBFRTB	12	Buffer request QCB
804	(324)	AVTCKPTB	12	Checkpoint QCB
816	(330)	AVTOPCOB	12	Operator Control QCB
828	(33C)	AVTOLTQB	12	On-Line Test QCB
840	(348)	AVTACTIB	12	Activate QCB
852	(354)	AVTCLOSB	12	Closedown completion QCB
864	(360)	AVTCPRMB	12	QCB to remove an element from the time delay QCB
876	(36C)	AVTDSIOB	12	Disk I/O QCB
888	(378)	AVTCPBCB	12	CPB cleanup QCB
900	(384)	AVTCOREC	4	Buffer unit pool address
904	(388)	AVTCADDR	4	Main-storage queue count
908	(38C)	AVTFZERO	4	Fullword of zeros

The following is the FE Common Write task interface area:

912	(390)	AVTCAREA	4	Address of the Patch module for this task

Offset	*Name*	*Bytes*	*Description*
916 (394)	AVTCWPM1	4	First parameter list pointer for this task
920 (398)	AVTCWEC1	4	First ECB for this task
924 (39C)	AVTCWFL1	1	First flag byte for this task

Bit definitions:

Name	*Bit*	*Value*	*Description*
AVTCOMWN	0	X'80'	Specifies that the FE Common Write task is attached; set by the COMWRTE=YES operand of the INTRO macro

Offset	*Name*	*Bytes*	*Description*
925 (39D)	AVTCWFL2	1	Second flag byte for this task

Name	*Bit*	*Value*	*Description*
AVTCWACT	0	X'80'	Specifies that the FE Common Write task is active; set by the COMWRTE=YES operand of the INTRO macro

Offset	*Name*	*Bytes*	*Description*
926 (39E)	AVTCWTS1	1	Third flag byte for this task
927 (39F)	AVTCWTS2	1	Fourth flag byte for this task
928 (3A0)	AVTCWPM2	4	Second parameter pointer for this task
932 (3A4)	AVTCWEC2	4	Second ECB for this task
936 (3A8)	AVTAFE10	4	Address of the FE STCB Trace Dump routine—IEDQFE10
940 (3AC)	AVTAFE20	4	Address of the FE I/O Trace Dump routine—IEDQFE20
944 (3B0)	AVTAFE30	4	Address of the FE Buffer Dump routine—IEDQFE30
948 (3B4)		64	Patch area for this task
1012 (3F4)	AVTGETMN	10	GETMAIN parameter list
1022 (3FE)	AVTHA2	2	Constant = 2
1024 (400)	AVTHA3	2	Constant = 3
1026 (402)	AVTHA4	2	Constant = 4
1028 (404)	AVTHA7	2	Constant = 7
1030 (406)	AVTHA16	2	Constant = 16
1032 (408)	AVTKEYLE	2	Specifies the size in bytes of a buffer unit—set by the KEYLEN=integer operand of the INTRO macro
1034 (40A)	AVTLNCNT	2	Number of lines opened—set by the Line Group Open routine—checked by the Time Delay subtask
1036 (40C)	AVTOPCNT	2	Number of lines taken by the Operator Control task—set by the System Delay subtask and the Operator Control task
1038 (40E)	AVTOPCON	2	Termname table offset to the entry for the primary Operator Control terminal—set by the PRIMARY=termname operand of the INTRO macro
1040 (410)	AVTAVFCT	2	Number of buffers in the buffer unit pool—this value is equal to the sum of the LNUNITS=integer and the MSUNITS=integer operands of the INTRO macro
1042 (412)	AVTSMCNT	2	Number of lines serviced by the Start-up Message subtask

Offset	*Name*	*Bytes*	*Description*
1044 (414)	AVTINTLV	2	Number of seconds of a system delay—set by Operator Control or by the INTVAL=integer operand of the INTRO macro; checked by the Time Delay subtask
1046 (416)	AVTDLQX	2	Termname table offset of the dead-letter queue—moved from the AVTDLQ field of the AVT after the termname table is sorted at execution time
1048 (418)	AVTDUMBR	2	Dummy line I/O interrupt trace table update
1050 (41A)	AVTBIT1	1	Flag bits

Bit Definitions:

Name	*Bit*	*Value*	*Description*
AVTAPLKN	0	X'80'	Prevents the Disk End Appendage from adding a CPB to the disabled disk end QCB for CPB Clean-up
AVTAPLKF	0 Off	X'7F'	Mask to permit the Disk End Appendage to add a CPB to the disabled disk end QCB for CPB Cleanup
AVTTSON	1	X'40'	Specifies that the TCAM environment has TSO or is mixed—set by the ENVIRON=TSO or MIXED operand of the INTRO macro
AVTAQTAN	2	X'20'	Specifies that the system environment has TCAM or is mixed—set by the ENVIRON=TCAM or MIXED operand of INTRO
AVTDLAYN	3	X'10'	Specifies that a system delay is in effect—set by the Operator Control task
AVTDLAYF	3 Off	X'EF'	Mask to specify that a system delay is not in effect—bit 3 is turned off by the Time Delay subtask
AVTREADN	4	X'08'	Specifies that the READY macro expansion has been executed—set by the READY macro expansion; checked by the open routines
AVTCLOSN	5	X'04'	Closedown indicator: 0—closedown not requested 1—closedown requested
AVTQUCKN	6	X'02'	Type of closedown: 0—Flush closedown 1—Quick closedown
AVTDISKN	7	X'01'	Specifies that none of the message queues data sets are disk queued

Offset	*Name*	*Bytes*	*Description*
1051 (41B)	AVTBIT2	1	Flag bits

Bit definitions:

Name	*Bit*	*Value*	*Description*
AVTRUFTN	0	X'80'	Reserved
AVTRUF	0 Off	X'7F'	Mask for the "Reusability first time" switch turned off by Reusability

Offset	*Name*	*Bytes*	*Description*		
		AVTREUSN	1	X‘40’	Specifies that Reusability is running—set by Reusability; checked by CPB Cleanup
		AVTREUSF	1 Off	X‘BF’	Mask to specify that Reusability is not running—turned off by Reusability
		AVTCOPYN	2	X‘20’	Specifies that the Reusability–Copy function is requesting control
			3	X‘10’	Specifies that TOPMSG=NO is set in the INTRO macro
		AVTSTRTN	4	X‘08’	Restart is in progress
		AVTSTRTF	4 Off	X‘F7’	Mask to specify that restart is not in progress
		AVTOPEIN	5	X‘04’	Initial load done indicator
			6,7	X‘03’	Specifies the line type as nonswitched Start/Stop only— set by the Activate routine or the Line End Appendage
			6	X‘02’	Specifies the line type as Start/Stop, switched or nonswitched—set by the Activate routine or the Line End Appendage
			7	X‘01’	Specifies the line type as binary synchronous—set by the Activate routine or Line End Appendage
			All Off		Specifies the line type as both BSC and Start/Stop, switched and nonswitched, all possible line combinations—set by the Activate routine or Line End Appendage
1052 (41C)	AVTBIT3	1	Flag bits		
		Bit Definitions:			
		Name	*Bit*	*Value*	*Description*
		AVTSTAN	7	X‘01’	Specifies that either a cold or warm restart is to be performed following a normal quick close or a flush close—set by STARTUP=C or STARTUP=W operand of the INTRO macro
		AVTSTACN	6	X‘02’	Specifies that a cold start is to be performed following a normal quick or a flush close and that a continuation restart is to be performed following system failure—set by the STARTUP=C operand of the INTRO macro
		AVTSTAWN	6 Off	X‘FD’	Mask to specify that a warm restart is to be performed following a normal quick or a flush close and that a continuation restart is to be performed following system failure—set by the STARTUP=W operand of the INTRO macro
		AVTSTAIN	5	X‘04’	Specifies that the status of each invitation list is to be included in the checkpoint record—set by the STARTUP=I operand of the INTRO macro

Offset	Name	Bytes	Description		
		AVTSTAYN	4	X'08'	Specifies that no continuation restart is to be performed following a normal quick close, a flush close, or system failure—set by the STARTUP=Y operand of the INTRO macro
		AVTOLTBN	3	X'10'	Specifies that the maximum size in the OLTEST=keyword operand in the INTRO macro (the maximum number of on-line tests that can be performed) has been reached—set, checked, and reset by TOTE
		AVTTSAB	2	X'20'	Specifies that TSO has abended—set by the Time Sharing Abend module; checked by the TSINPUT and TSOUTPUT routines; reset by the Start Time Sharing routine
		AVTRFULN	1	X'40'	Reusable disk zone full—set by Reusability
		AVTRFULF	0,2, 3 Off	X'BF'	Mask to specify that reusable disk is ready to receive—checked by Receive Scheduler and Line End Appendage; turned off by Reusability
		AVTRECVN	0	X'80'	Main-storage queue is full—set by Destination Scheduler when the number of main-storage queue units > or = the number specified in the MSMAX= operand of the INTRO macro; turned off by Disk I/O; checked by the Receive Scheduler and Line End Appendage
1053 (41D)	AVTCKRST	1	Specifies which checkpoint record the TCAM restart facility should use in attempting to restructure the MCP environment as it existed at the time of closedown or system failure—set by the RESTART=integer operand of the INTRO macro		
1054 (41E)	AVTDSKCT	2	Specifies the number of buffers on CPBs		

****

This is the end of the basic AVT when
ENVIRON=TSO

****

Offset	Name	Bytes	Description
1056 (420)	AVTMH02	4	Address of the Destination Scheduler
1060 (424)	AVTCMIN	4	Specifies the percentage of the number of units in the message queues data set below which the data set is not crowded—set by the MSMIN=integer operand of the INTRO macro
1064 (428)	AVTCMAX	4	Specifies the percentage of the number of units in the message queues data set above which means that the data set is nearly full—set by the MSMAX=integer operand of the INTRO macro
1068 (42C)	AVTTOTNC	4	Number of records in the entire message queues data set—set by the MSUNITS=integer operand of the INTRO macro
1072 (430)	AVTNCPBQ	8	Queue of buffers and ERBs waiting to be processed

Offset	*Name*	*Bytes*	*Description*
			**
			This is the end of the AVT when main-storage queuing only is specified (DISK=NO, ENVIRON=TCAM or MIXED)
			**
1080 (438)	AVTFL	4	Address of the Disk EXCP Driver routine
1084 (43C)	AVTIA	4	Address of the REUS part of the Reusability–Copy subtask
1088 (440)	AVTCOPY	4	Address of the Copy subtask QCB
1092 (444)	AVTDKAPQ	8	Queue of the CPBs to be processed by CPB Cleanup (disabled)
1100 (44C)	AVTDKENQ	8	Queue of CPBs to be processed by CPB Cleanup (enabled)
1108 (454)	AVTNOBFQ	8	Queue of CPBs without buffers—used by CPB Cleanup
1116 (45C)	AVTREUSQ	8	Reserved
1124 (464)	AVTINCPQ	8	Queue of CPBs requesting that I/O be done by EXCP Driver
1132 (46C)	AVTFCPB	4	Queue of inactive CPBs—the CPB free pool
1136 (470)	AVTCPBPT	4	Address of the CPB free pool to be freed by the Disk Close routine—AVTFCPB is initially set to this same value
1140 (474)	AVTIOBR	4	Address of a series of IOBs, one for each extent of the reusable disk queue
1144 (478)	AVTIOBN	4	Address of a list of IOBs, one for each extent of the nonreusable disk queue
1148 (47C)	AVTLODPT	4	Absolute disk record number that indicates when the REUS part of the Reusability–Copy subtask is to activated—the initial value is 3/8 of the total number of records on the reusable disk message queues data set

The next 6-word area is initiated by the OPEN for the reusable disk message queues data set for use by the MBBCCHHR Converter routine.

1152 (480)	AVTADEBR	4	Address of the DEBEOEA field in the DEB for the reusable disk message queues data set
1156 (484)	AVTNOVOR	4	Number of extents in the reusable disk message queues data set
1160 (488)	AVTRCTRR	4	Number of records per track on the reusable disk message queues data set
1164 (48C)	AVTTRCYR	4	Number of tracks per cylinder on the reusable disk message queues data set
1168 (490)	AVTTOTNR	4	Number of records in the entire reusable disk message queues data set
1172 (494)	AVTVOLRR	4	Product of the number of extents times the number of records per track on the reusable disk message queues data set

The next 7-word area is initialized by the OPEN for the nonreusable disk message queues data set for use by the MBBCCHHR Converter routine.

1176 (498)	AVTADEBN	4	Address of the DEBEOEA field in the DEB for the nonreusable disk message queues data set
1180 (49C)	AVTNOVON	4	Number of extents in the nonreusable disk message queues data set

Offset	*Name*	*Bytes*	*Description*
1184 (4A0)	AVTRCTRN	4	Number of records per track on the nonreusable disk message queues data set
1188 (4A4)	AVTTRCYN	4	Number of tracks per cylinder on the nonreusable disk message queues data set
1192 (4A8)	AVTTOTNN	4	Number of records in the entire nonreusable disk message queues data set
1196 (4AC)	AVTVOLRN	4	Product of the number of extents times the number of records per track on the nonreusable disk message queues data set
1200 (4B0)	AVTHRESN	4	The absolute record number that is the threshold to cause closedown due to the filling of the nonreusable disk message queues data set
1204 (4B4)	AVTNADDR	4	Index to nonreusable disk relative record number—next available location*
1208 (4B8)	AVTRADDR	4	Index to reusable disk relative record number—next available location*
1212 (4BC)	AVTHRESE	12	Nonreusable threshold closedown element
1223 (4C7)	AVTHRESS	1	Completion code—used to indicate status X'FF' — an unused element X'F0' — the element has been tposted X'00' — Closedown indication X'04' — Closedown indication
1224 (4C8)	AVTCPBNO	2	Specifies the value coded in the CPB=integer operand of the INTRO macro

***Note:** *This field contains a number which, when adjusted, (by adding 3 and dividing 4) yields the absolute relative record number.*

Access Method Work Area

The access method work area (IEDQWRKA) is a variable-length table that provides intermediate storage fields, pointers to control blocks, switches, and space for a work area. When a DCB in an application program is being opened, the GET/PUT and READ/WRITE Open Executor (IGG01946) allocates main storage for and initializes the access method work area.

The Open Executor puts the address of the work area in the DEBTAMWA field of the data extent block (DEB) for the application program. The address of the DEB is in the DCBDEBAD field in the associated data control block (DCB) in the application program. The DEB address is also in the PEWADEB field of the process entry work area in the MCP so that routines in the MCP can refer to the access method work area by first examining the DEB.

The access method work area is variable in length depending upon whether or not the user specified a SYNAD exit routine. If the user does not specify a SYNAD exit routine, the fullword field GWASTAT/PWASTAT is set to zero (0). If, however, the user does specify such a routine, the field GWASTAT/PWASTAT contains the address of the status indicators. The status indicators are in a 14-byte field that is added to the end of the access method work area when required by a SYNAD routine request. There are two status indicators for the SYNAD routine. The first is bit zero of the second byte of the 14-byte area. When this bit is set to 1, the command issued is rejected because work units are out of sequence. The second status indicator is bit 1 of the thirteenth byte. When this bit is set to 1, an incorrect length has been specified, thus creating a work area overflow.

The format of the access method work area is illustrated below; descriptions of the fields follow.

IEDQWRKA

Offset	Field	Description
0 (0)	PWASAVE GWASAVE	Address of User's Register Save Area
4 (4)	PWAPEWA	Address of the Process Entry Work Area
8 (8)	GWAPEB	Address of a Part-Empty Buffer
12 (C)	PWASTART GWAMOVE	Address of the First Byte of Data in the Work Area Address of the Next Byte in a Buffer to be Moved
16 (10)	PWACKPT GWACKPT	Address of the User's Checkpoint Routine
20 (14)	GWAPEWA	Address of the Next Empty Byte in the User's Work Area
24 (18)	PWAECB GWAECB	PUT/WRITE ECB GET/READ ECB
28 (1C)	PWAELEM GWAELEM	Special AQCTL Element
48 (30)	PWALIST GWALIST MOVEAD	AQCTL Parameter List Address of the Field to be Moved
52 (34)	TARGETAD	Address of Where the Data is to be Moved
56 (38)	PFLAG	End-of-List Indicator
57 (39)	LENGTHAD	Address of the Length of the Field
60 (3C)	PWASAVA GWASAVA	PUT/WRITE Save Area GET/READ Save Area
132 (84)	PWAFLG	PUT/WRITE Reader Needed EOM Processed GET/READ
134 (86)		Reserved
136 (88)	IOBPSAVE	Address of a Partly Empty Buffer Unit
140 (8C)	GWASTAT PWASTAT	Address of GET/READ Status Indicators Address of PUT/WRITE Status Indicators
144 (90)	PWASOWA GWASOWA	Size of the User's Work Area
146 (92)	PWACTL	Work Area Contents Descriptor Byte
147 (93)	GWARDEL	Record Delimiter
148 (94)	GWABUFL	Size of an MCP Buffer
150 (96)	PWAOPTCD GWAOPTCD	General Switches
151 (97)	PWARECFM GWARECFM	General Switches

152 (98) **GWALRECL** Size of a Logical Work Unit	154 (9A) **PWAOFF** Termname Table Offset for Message Destination
156 (9C) **CTLADDR** Address of the Work Area Control Byte	
160 (A0) **GWASCAN** Size of Field to be Scanned	162 (A2) **BUFCNT** Empty-Buffer Counter
164 (A4) **IOBUSZE** Count of Data in a Logical Buffer	166 (A6) **IOBPSZE** Prefix Size Work Area
168 (A8) **IOBSRCE** Termname Table Offset	170 (AA) Reserved
172 (AC) **GWARTVE** Message Retrieval Work Area	
180 (B4) **GWADTSA** General Switches	181 (B5) Reserved

Note: *When there are two field names for one field, those field names beginning with P are used when the user is coding in PUT mode, and those field names beginning with G are used when the user is coding in GET mode.*

Offset		*Name*	*Bytes*	*Description*
0	(0)	PWASAVE	4	Address of the user register save area
0	(0)	GWASAVE	4	Address of the user register save area
4	(4)	PWAPEWA	4	Address of the process entry work area
8	(8)	GWAPEB	4	Address of a partially empty buffer—the one being used
12	(C)	PWASTART	4	Address of the first byte of data in the user work area
12	(C)	GWAMOVE	4	Address of next byte to be moved in a buffer
16	(10)	PWACKPT	4	Reserved
16	(10)	GWACKPT	4	Reserved
20	(14)	GWAPEWA	4	Address of next empty byte in user work area
24	(18)	PWAECB	4	PUT/WRITE ECB
24	(18)	GWAECB	4	GET/READ ECB
28	(1C)	PWAELEM	20	Special AQCTL element
28	(1C)	GWAELEM	20	Special AQCTL element
48	(30)	PWALIST	4	AQCTL parameter list
48	(30)	GWALIST	4	AQCTL parameter list
48	(30)	MOVEAD	4	Address of the field to be moved
52	(34)	TARGETAD	4	Address of the area into which data is to be moved
56	(38)	PFLAG	1	Indicator of end of parameter list
57	(39)	LENGTHAD	3	Address of the length field of the parameter list
60	(3C)	PWASAVA	72	PUT/WRITE save area
60	(3C)	GWASAVA	72	READ/CHECK save area

Offset	*Name*	*Bytes*	*Description*		
132 (84)	PWAFLG	2	X'20' header needed (PUT/WRITE)		
	PWAFLG+1		X'80' EOM processed (GET/READ)		
136 (88)	IOBPSAVE	4	Address of partially empty buffer unit		
140 (8C)	GWASTAT	4	Address of status indicators		
140 (8C)	PWASTAT	4	Address of status indicators		
144 (90)	PWASOWA	2	Size of user work area		
144 (90)	GWASOWA	2	Size of user work area		
146 (92)	PWACTL	1	Work area contents descriptor byte— contains a value indicating whether the message in the work area is the first, intermediate, or last segment of the message. The following are the bit settings:		
			Bits	*Value*	*Meaning*
			0,1,2, 3,7	X'F1'	first segment (header)
			0,1,2, 3,6	X'F2'	last segment (EOM)
			0,1,2, 3,6,7	X'F3'	entire message
			1	X'40'	intermediate segment
147 (93)	GWARDEL	1	End of record for GET/PUT—copied from the process entry		
148 (94)	GWABUFL	2	Size of MCP buffer		
150 (96)	PWAOPTCD	1	General switch; bit settings are:		
		Name	*Bits*	*Value*	*Meaning*
		FIRSTPUT	7	X'01'	first-time switch for locate mode
150 (96)	GWAOPTCD	1	General switch; bit settings are:		
		Name	*Bits*	*Value*	*Meaning*
		TNMEFLG	0	X'80'	OPTCD=W (source terminal field)
		MSGFLG	1	X'40'	OPTCB=U (message rather than record format)
		CTLBYTE	2	X'20'	OPTCD=C (control byte flag)
		EODADFLG	3	X'10'	EODAD exit flag mask
		RECDEL	4	X'08'	First-time RECDEL flag
		RTVFLG	5	X'04'	Retrieve mode switch mask
		PARTBUF	6	X'02'	Partially empty buffer left on main-storage QCB
		SYNADFLG	7	X'01'	DCBOPTCD bit which, if set, effects exit to SYNAD routine
151 (97)	PWARECFM	1	PUT/WRITE; no bits set		
151 (97)	GWARECFM	1	GET/READ; bit settings are:		
		Name	*Bits*	*Value*	*Meaning*
		RETFLG	5	X'04'	Retrieve mode may be entered
		INCWA	7	X'01'	Incomplete work area
152 (98)	GWALRECL	2	Size of logical work unit		

Offset	*Name*	*Bytes*	*Description*
154 (9A)	PWAOFF	2	PUT Scheduler—termname table offset for message destination
156 (9C)	CTLADDR	4	Address of work area control byte; address of PWACTL within the work area
160 (A0)	GWASCAN	2	Size of field to be scanned
162 (A2)	BUFCNT	2	Empty-buffer counter
164 (A4)	IOBUSZE	2	Count of data in a logical buffer—number of bytes in a buffer unit
166 (A6)	IOBPSZE	2	Number of bytes in a buffer—prefix size work area
168 (A8)	IOBSRCE	2	Termname table offset
170 (AA)		2	Reserved
172 (AC)	GWARTVE	8	Message retrieval work area
180 (B4)	GWADTSA	1	General switch; bit settings are:
		Name	*Bits* *Value* *Meaning*
		DELETE	1 X‘40’ DELETE=YES
181 (B5)		3	Reserved

(This page left blank intentionally)

Buffer Prefix

First buffer of a message:

Offset

0	1	4	5		8	12 (C)	13 (D)	16 (10)	18 (12)	20 (14)	21 (15)	24 (18)	26 (1A)	29 (1D)	32 (20)	35 (23)	38 (26)	40 (28)
Key **PRFKEY** or CCW OP Code	QCB address **PRFQCBA** or Next address to be transferred	Priority **PRFPRI** or CCW flags	Link field **PRFLINK** Unused	CCW count	Link to next unit and TIC CCW	Number of units in this buffer	LCB address	Source offset in the Termname Table	Size of data in this buffer	Status byte	Pointer to additional records on disk **PRFXTRA** or to the current record in main storage	Scan pointer offset	Pointer to next buffer of this message if not last buffer **PRFNTXT** or text queue-back chain if last buffer	Pointer to the first unit of the current buffer	Pointer to the first buffer of the next message	Queue-back chain of the first buffers of messages	Input sequence number	Destination offset in the Termname Table
PRFOPCDE	**PRFIOADR**	**PRFFLAGS**		**PRFCOUNT**	**PRFTIC**	**PRFNBUNT**	**PRFLCB**	**PRFSRCE**	**PRFSIZE**	**PRFSTAT1**	**PRFCORE**	**PRFSCAN**	**PRFTQBCK**	**PRFCRCD**	**PRFNHDR**	**PRFHQBCK**	**PRFISEQ**	**PRFDEST**

RCB

First or 30-byte Buffer Prefix

The first 12 bytes are not placed on the queue for the message queues data set.

0	12 (C)	42 (2A)
Unit control area	First buffer prefix **PRFSUNIT**	Start of the message header or data **PRFSHDR**

Subsequent buffer of a message:

Offset

0	1	4	5		8	12(C)	13 (D)	16 (10)	18 (12)	20 (14)	21 (15)	24 (18)	26 (1A)	29 (1D)	32 (20)
Key **PRFKEY** or CCW OP Code	QCB address **PRFQCBA** or Next address to be transferred	Priority **PRFPRI** or CCW flags	Link field **PRFLINK** Unused	CCW count	Link to next unit and TIC CCW	Number of units in this buffer	LCB address	Source offset in the Termname Table	Size of data in this buffer	Status byte	Pointer to additional records on disk **PRFXTRA** or to the current record in main storage	Scan pointer offset	Pointer to next buffer of this message if not last buffer **PRFNTXT** or text queue-back chain if last buffer	Pointer to the first unit of the current buffer	Pointer to the first buffer of the current message
PRFOPCDE	**PRFIOADR**	**PRFFLAGS**		**PRFCOUNT**	**PRFTIC**	**PRFNBUNT**	**PRFLCB**	**PRFSRCE**	**PRFSIZE**	**PRFSTAT1**	**PRFCORE**	**PRFSCAN**	**PRFTQBCK**	**PRFCRCK**	**PRFCHDR**

RCB

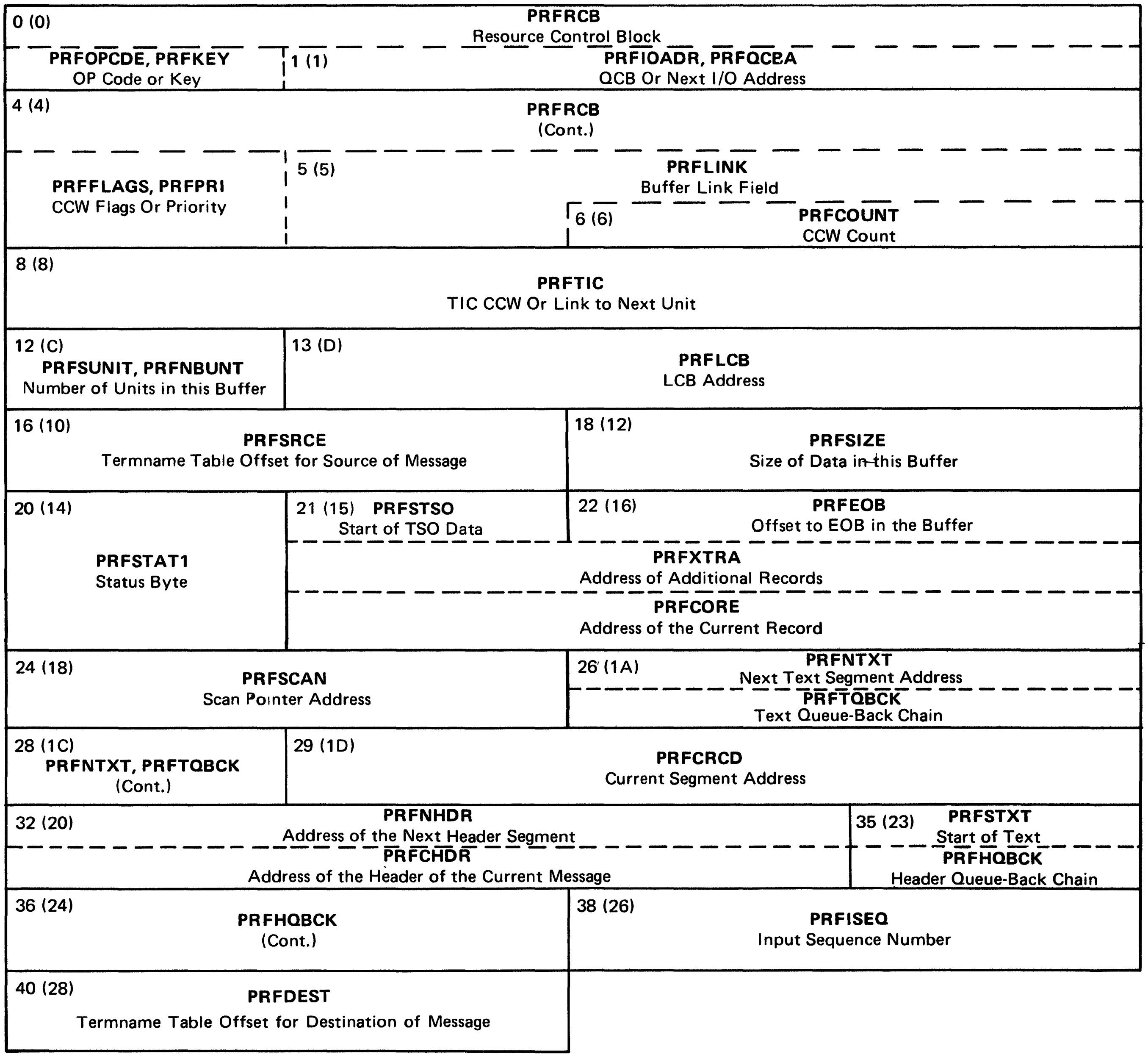
Subsequent or 23-byte Buffer Prefix

The first 12 bytes are not placed on the queue for the message queues data set.

0	12 (C)	35 (23)
Unit control area	Subsequent buffer prefix **PRFSUNIT**	Continuation of message header or start or continuation of message data **PRFSTXT**

Insert foldout page 429 at end of book.

IEDQPRF



0 (0) **PRFRCB** Resource Control Block

PRFOPCDE, PRFKEY OP Code or Key | 1 (1) **PRFIOADR, PRFQCBA** QCB Or Next I/O Address

4 (4) **PRFRCB** (Cont.)

PRFFLAGS, PRFPRI CCW Flags Or Priority | 5 (5) **PRFLINK** Buffer Link Field | 6 (6) **PRFCOUNT** CCW Count

8 (8) **PRFTIC** TIC CCW Or Link to Next Unit

12 (C) **PRFSUNIT, PRFNBUNT** Number of Units in this Buffer | 13 (D) **PRFLCB** LCB Address

16 (10) **PRFSRCE** Termname Table Offset for Source of Message | 18 (12) **PRFSIZE** Size of Data in-this Buffer

20 (14) **PRFSTAT1** Status Byte | 21 (15) **PRFSTSO** Start of TSO Data | 22 (16) **PRFEOB** Offset to EOB in the Buffer

PRFXTRA Address of Additional Records

PRFCORE Address of the Current Record

24 (18) **PRFSCAN** Scan Pointer Address | 26 (1A) **PRFNTXT** Next Text Segment Address | **PRFTQBCK** Text Queue-Back Chain

28 (1C) **PRFNTXT, PRFTQBCK** (Cont.) | 29 (1D) **PRFCRCD** Current Segment Address

32 (20) **PRFNHDR** Address of the Next Header Segment | **PRFCHDR** Address of the Header of the Current Message | 35 (23) **PRFSTXT** Start of Text | **PRFHQBCK** Header Queue-Back Chain

36 (24) **PRFHQBCK** (Cont.) | 38 (26) **PRFISEQ** Input Sequence Number

40 (28) **PRFDEST** Termname Table Offset for Destination of Message

0	(0)	PRFRCB	8	Resource control block
0	(0)	PRFOPCDE	1	CCW operation code, when I/O is being performed
0	(0)	PRFKEY	1	Element key of the buffer
1	(1)	PRFIOADR	3	Next data byte (address) to be transferred (Read), when I/O is being performed
1	(1)	PRFQCBA	3	QCB address, when the buffer is an element
4	(4)	PRFFLAGS	1	CCW flags, when I/O is being performed
4	(4)	PRFPRI	1	Priority of the buffer, when it is an element
5	(5)	PRFLINK	3	Link field of the buffer, unused when a CCW
6	(6)	PRFCOUNT	2	CCW (Read/Write) count

Offset		*Name*	*Bytes*	*Description*		
8	(8)	PRFTIC	4	TIC CCW & link to the next unit. The high-order byte contains one of the following:		
			Name	*Bits*	*Value*	*Meaning*
			PRFEOMSG	0	X‘80’	Logical end of message
			PRFBFMM	1	X‘40’	Header converted to text
12	(C)	PRFSUNIT		Start of the logical unit		
12	(C)	PRFNBUNT	1	Number of units in this buffer		
13	(D)	PRFLCB	3	Pointer to the LCB		
16	(10)	PRFSRCE	2	Termname table offset for the source of the message		
18	(12)	PRFSIZE	2	Size of data in this buffer		
20	(14)	PRFSTAT1	1	Status byte:		
			Name	*Bits*	*Value*	*Meaning*
			PRFCNCLN	0	X‘80’	Cancel message has been executed
			PRFCNCLF	All On	X‘FF’	Mask to specify that the message is not canceled
			PRFERMGN	1	X‘40’	Error message is in this buffer
			PRFERMGF	1 Off	X‘BF’	Mask to specify that this is not an error message buffer
			PRFITCPN	2	X‘20’	Message is being held
			PRFITCPF	2 Off	X‘DF’	Mask to specify that the message is not being held
			PRFTSBUF	3	X‘10’	This is a TSO buffer
			PRFDUPLN	4	X‘08’	Duplicate-header buffer
			PRFDUPLF	4 Off	X‘F7’	Mask to specify an original buffer
			PRFEOFN	5	X‘04’	SETEOF was executed
			PRFEOFF	5 Off	X‘FB’	Mask to specify that SETEOF was not executed
			PRFLOCK = PRFEOFN			LOCK executed this message
			PRFNLSTN	6	X‘02’	Not the last buffer of a message
			PRFNLSTF	6 Off	X‘FD’	Mask to specify the last buffer of a message
			PRENHDRN	7	X‘01’	Not the first buffer of a message
			PRFNHDRF	7 Off	X‘FE’	Mask to specify the first buffer of a message
21	(15)	PRFSTSO		Start of time sharing data		
21	(15)	PRFXTRA	3	Pointer to the additional records		
21	(15)	PRFCORE	3	Pointer to the current record		
22	(16)	PRFEOB	2	Offset to EOB in the buffer (receive)		
24	(18)	PRFSCAN	2	Scan pointer address		
26	(1A)	PRFNTXT	3	Pointer to the next text segment		
26	(1A)	PRFTQBCK	3	Queue-back chain of text segments		
29	(1D)	PRFCRCD	3	Pointer to the current segment		
32	(20)	PRFNHDR	3	Pointer to the next header segment		
32	(20)	PRFCHDR	3	Pointer to the header of the current message		
35	(23)	PRFSTXT		Start of text data in a subsequent buffer		
35	(23)	PRFHQBCK	3	Queue-back chain of header segments		
38	(26)	PRFISEQ	2	Input sequence number		
40	(28)	PRFDEST	2	Termname table offset for the destination of the message		

Channel Program Block

The channel program block (IEDQCPB) contains the disk channel program and other information pertinent to the disk I/O involved. Within the channel program the CPB contains pointers to its associated unit and to the next CPB as well as the actual number of the unit being processed and its MBBCCHHR equivalent. The address of the first CPB is in the AVTCPBPT field of the address vector table. The same address is in the AVTFCPB field of the AVT at INTRO execution time, but this field changes during the execution of the channel program as it always points to the first CPB in the LIFO CPB queue.

In disk queuing, CPBs are used to read to or write from the destination queues. If disk queuing is used, the pool of CPBs is created by a nonresident routine called by the INTRO macro expansion. The user specifies the number of CPBs to be built to handle the message queues buffers in the CPB=integer operand of the INTRO macro instruction. Each CPB is built in main storage and is an allocated a work area equal in size to one buffer unit (including the 12-byte unit control area). Initially this unit is contiguous with the CPB, but as processing continues, the unit may be from the buffer unit pool. The CPBXREA field points to the associated unit, which is actually the disk data area.

The format of the channel program block is illustrated below; descriptions of the fields follow.

IEDQCPB

0 (0) **CPBHEADF** Seek Head CCW		
CPBSEEK OP Code	1 (1) **CPBHEAD** Head ID Address	
4 (4) **CPBHEADF** (Cont.)		
CPBSEKFL Seek Flag	5 (5) Reserved	6 (6) **CPBSEKCT** Seek Count
8 (8) **CPBSETAF** Set Sector CCW		
CPBSET OP Code	9 (9) **CPBSETA** Set Sector ID Address	
12 (C) **CPBSETAF** (Cont.)		
CPBSETFL Set Sector Flag	13 (D) Reserved	14 (E) **CPBSETCT** Set Sector Count
16 (10) **CPBSRECF** Search ID Equal CCW		
CPBSRCH OP Code	17 (11) **CPBSREC** Record ID Address	
20 (14) **CPBSRECF** (Cont.)		
CPBSRHFL Search Flag	21 (15) Reserved	22 (16) **CPBSRHCT** Search Count
24 (18) **CPBTICSF** TIC to Search CCW		
CPBTIC1 OP Code	25 (19) **CPBTICS** Search CCW Address	
28 (1C) **CPBTICSF** (Cont.)		
CPBSECTR Sector ID	29 (1D) **CPBUNUSD** Reserved	
32 (20) **CPBAREAF** Read/Write CCW		

CPBRDWR OP Code	33 (21) CPBAREA I/O Area Address		
36 (24) CPBAREAF (Cont.)			
CPBRWFL Read/Write Flag	37 (25) Reserved	38 (26) CPBCOUNT Number of Bytes to Read or Write	
40 (28) CPBXREAF Second Read/Write CCW			
CPBXDWR OP Code	41 (29) CPBXREA I/O Area Address		
44 (2C) CPBXREAF (Cont.)			
CPBXWFL Read/Write Flag	45 (2D) Reserved	46 (2E) CPBXOUNT Number of Bytes to Read or Write	
48 (30) CPBNEXTF TIC to Next CPB CCW			
CPBTIC2 OP Code	49 (31) CPBNEXT Next CPB Address		
52 (34) CPBADDR Index to Absolute Record Number			
CPBFLAG Flag Byte			
56 (38) CPBABSAD MBBCCHHR Value			
64 (40) CPBINWKA Work Area Data Count	65 (41) CPBTOUNT Data to be Moved Count	66 (42) CPBWKACT Work Area Start Address	67 (43) CPBNUMB Current CPB Number
68 (44) CPBAERBF ERB Address			
CPBUNTCT Unit Data Count	69 (45) CPBAERB ERB Address		

Offset		*Name*	*Bytes*	*Description*
0	(0)	CPBHEADF		Start of the Seek Head CCW
0	(0)	CPBHEADF		Start of the Seek Head CCW
0	(0)	CPBSEEK	1	Seek Head op code
1	(1)	CPBHEAD	3	Pointer to the head ID
4	(4)	CPBSEKFL	1	Seek CCW flag, command chaining
6	(6)	CPBSEKCT	2	Seek count of 6
8	(8)	CPBSETAF		Start of the Set Sector CCW
8	(8)	CPBSET	1	Set Sector op code
9	(9)	CPBSETA	3	Pointer to sector ID byte
12	(C)	CPBSETFL	1	Set Sector flag byte
13	(D)		1	Reserved
14	(E)	CPBSETCT	2	Set Sector count of 1
16	(10)	CPBSRCH	1	Search ID Equal op code
17	(11)	CPBSREC	3	Pointer to the record ID
20	(14)	CPBSRHFL	1	Search CCW flag
21	(15)		1	Reserved
22	(16)	CPBSRHCT	3	Search count of 5
24	(18)	CPBTICSF		Start of the TIC to Search CCW
24	(18)	CPBTIC1	1	TIC op code
25	(19)	CPBTICS	3	Address of the Search CCW
28	(1C)	CPBSECTR	1	Set sector ID
29	(1D)	CPBUNUSA	3	Reserved
32	(20)	CPBAREAF		Start of the Read/Write CCW
32	(20)	CPBRDWR	1	Read/Write op code
33	(21)	CPBAREA	3	Address of the I/O area
36	(24)	CPBRWFL	1	Read/Write flag
37	(25)		1	Reserved
38	(26)	CPBCOUNT	2	Number of bytes to be read or written
40	(28)	CPBXREAF		Start of the second Read/Write CCW
40	(28)	CPBXDWR	1	Read/Write op code
41	(29)	CPBXREA	3	Address of the I/O area
44	(2C)	CPBXWFL	1	Read/Write flag
45	(2S)		1	Reserved
46	(2E)	CPBXOUNT	2	Number of bytes to be read or written
48	(30)	CPBNEXTF		Start of the TIC to next CPB CCW
48	(30)	CPBTIC2	1	TIC op code
49	(31)	CPBNEXT	3	Pointer to the next CPB
52	(34)	CPBFLAG	1	Flag byte

Offset		*Name*	*Bytes*	*Description*
52	(34)	CPBADDR	4	Index to absolute record number—this field contains a number which when adjusted (by adding 3 and dividing by 4), yields the absolute record number.
55	(37)	CPBQTYPE	1	The low-order two bits of the number determine the queue type as follows: B'11'—Reusable disk queuing B'10'—Reserved B'01'—Nonreusable disk queuing B'00'—Main-storage queuing
56	(38)	CPBABSAD	8	MBBCCHHR value
64	(40)	CPBINWKA	1	Count of the data in the work area
64	(40)		4	LCB address, if the CPB is for IGG019RP
65	(41)	CPBTOUNT	1	Count of the data to be moved into a unit
66	(42)	CPBWKACT	1	Where to start in the work area
67	(43)	CPBNUMB	1	Sequential number of the current CPB
68	(44)	CPBAERBF	4	Address of the ERB, or the work area unit address (for IGG019RP)
68	(44)	CPBUNTCT	1	Count of data already in the unit
69	(45)	CPBAERB	3	

The following are the CCW bit definitions:

	Name	*Bits*	*Value*	*Meaning*
CCW Flags:				
	CPBCDC	0	X'80'	Data chaining
	CPBCCC	1	X'40'	Command chaining
	CPBSLIC	2	X'20'	Suppress incorrect length
	CPBSKIPC	3	X'10'	Skip data
CCW Commands:				
	CPBWRITB	7	X'01'	Write Data or Key and Data bit
	CPBNOPC	6,7	X'03'	NO OP command
	CPBWRC	5,7	X'05'	Write Data command
	CPBRDC	5,6	X'06'	Read Data command
	CPBKEYB	4	X'08'	Key bit
	CPBTICC	4	X'08'	TIC command
	CPBWRKC	4,5,7	X'0D'	Write Key and Data command
	CPBRDKC	4,5,6	X'0E'	Read Key and Data command
	CPBSEEKC	3,4,6,7	X'1B'	Seek Head command
	CPBSETC	2,6,7	X'23'	Set Sector command

Checkpoint Disk Records

Checkpoint Control Record: The checkpoint control record is written on disk from the area starting at CKPCNTLR in the checkpoint work area each time that an environment checkpoint record is written.

Offset	0	1	2	3	4	5
	Flag byte CKPFLAGS	Index to the current environment record CKPTTRCT	Number of incident records CKPINCNT	Number of available incident records CKPINCNO	TTR of the last CKREQ record on first CKREQ records track CKPCRRNO	TTR of the first CKREQ record CKPTTRCR

Offset	8	9	12 (C)	14 (E)	15 (F)
	TTR of last incident record on first incident records track CKPINRNO	TTR of the first incident record CKPTTRIN	Number of bytes in an environment record segment CKPBPERR	Value of the INTRO operand CKREQS CKPCKRQS	Value of the INTRO operand CPRCDS CKPCPRCD

Offset	16 (10)	17 (11)	18 (12)	20 (14)	21 (15)
	Number of incident records per track CKPIPERT	Number of CKREQ records per track CKPCPERT	Length of a CKREQ record CKPCKRLN	Number of environment record segments per track CKPRPERT	TTR of the last incident record written CKPTTRLI

Offset	24 (18)	26 (1A)	28 (1C)	30 (1E)
	Length of an incident record CKPINCLN	Data on track preceding current environment segment CKPSECLT	Data on track preceding current incident record CKPSECLI	Data on track preceding first CKREQ record CKPSECCR

Offset	32 (20)	34 (22)	37 (25)
	Data on track preceding first incident record CKPSECIN	3 byte TTR for first environment record. All other TTRs follow this CKPTTRT1	Data on track preceding first segment of environment record CKPSECT1

Offset	*Name*	*Bytes*	*Description*	*Initialized By*	*Altered By*
0 (0)	CKPFLAGS	1	Flag byte:		
			X‘80’—normal closedown	Set by IGG01943 Turned off by IGG01944	
			X‘10’—Open CKREQ		IGG01944
			X‘20’—Open incident		IGG01944
			X‘40’—Open environment		IGG01943
			X‘08’—No environment records are available		IGG01943, IGG01941
			X‘04’—Value of start-up parameter that indicates whether invitation lists are to be checkpointed	Set by IGG01949	IGG01943
			X‘02’—OS synchronous checkpoint	IGG01949	IGG01949

Offset		Name	Bytes	Description	Initialized By	Altered By
				X'01'—Operator control incident records are present		
1	(1)	CKPTTRCT	1	Index to the current environment checkpoint record	IGG01942 initializes this field to 1	IEDQNP changes this field after each environment checkpoint
2	(2)	CKPINCNT	1	Total number of incident records in the data set	IGG01949	
3	(3)	CKPINCNO	1	Number of incident records that are available for use	Cold start (IGG01949) Warm start (IGG01941)	IEDQNG,IEDQNH, IEDQNI,IEDQNJ, IEDQNO
4	(4)	CKPCRRNO	1	TTR of the last CKREQ record on the first track that contains CKREQ records	IGG01942	
5	(5)	CKPTTRCR	3	TTR of the first CKREQ record	IGG01942	
8	(8)	CKPINRNO	1	TTR of the last incident record on the first track that contains incident records	IGG01942	
9	(9)	CKPTTRIN	3	TTR of the first incident record	IGG01942	
12	(C)	CKPBPERR	2	Number of bytes in each environment record segment	IGG01949	
14	(E)	CKPRKRQS	1	Value of CKREQS (from INTRO) for the last start-up—used at restart time instead of the corresponding value in the AVT	Cold start IGG01942 Warm start (IGG01944)	
15	(F)	CKPCPRCD	1	Value of CPRCDS (from INTRO) for last start-up—used at restart time instead of the corresponding value in the AVT	Cold start (IGG01942) Warm start (IGG01943)	
16	(10)	CKPIPERT	1	Number of incident records per track	IGG01949	

Offset		*Name*	*Bytes*	*Description*	*Initialized By*	*Altered By*
1.7	(11)	CKPPRQNO	1	Maximum number of priority QCBs used by an OS synchronous process entry	IGG01949	
		CKPCPERT		Number of CKREQ records per track (overlays CKPPRQNO)	IGG01949	
18	(12)	CKPCKRLN	2	Length of a CKREQ record, depends on the number of option fields	IGG01949	
20	(14)	CKPRPERT	1	Number of environment record segments per track	IGG01949	
21	(15)	CKPTTRLI	3	TTR of the last incident record written	IGG01941	IEDQNP
24	(18)	CKPINCLN	2	Length of an incident record	IGG01949	
26	(1A)	CKPSECLT	2	Data on track preceding the environment record	IEDQNP	
28	(1C)	CKPSECLI	2	Data on track preceding the current incident record	IEDQNP	
30	(1E)	CKPSECCR	2	Data on track preceding the first CKREQ record	IEDQNP	
32	(20)	CKPSECIN	2	Data on track preceding the first incident record	IEDQNP	
34	(22)	CKPTTRT1	3	3-byte TTR for the first environment record. All other TTRs follow this.	IEDQNP	
37	(25)	CKPSECT1	2	Data on track preceding the first segment of the environment record	IEDQNP	

There are as many three-byte TTR fields for environment checkpoint records as there are records indicated in CKPCPRCD.

Environment Checkpoint Record Segment: Main storage in which to build an environment checkpoint record segment is obtained by the Environment Checkpoint routine (IEDQNK) each time that an environment checkpoint is requested. The format and length of an environment checkpoint vary according to option table and terminal table entries. The environment record contains one section of data, with the associated option fields, for each single, group, line, and process entry of the terminal table.

Offset		*Name*	*Bytes*	*Description*	*Initialized By*
0	(0)	CDRDATE	4	Date of the checkpoint	IEDQNP
4	(4)	CDRTIME	4	Time that the record is written	IEDQNP
8	(8)	CDRKEY	1	Key byte: X'1C'—last segment of an environment checkpoint record X'20'—a segment that is not the last segment of an environment checkpoint record	IEDQNK
9	(9)	CDRTTRLI	3	TTR of the last incident record written	IEDQNP
12	(C)	CDRDATA		This is the point at which the checkpointed fields from the terminal table start. Only single, group, line, and process entries are checkpointed, and different fields are included under different conditions. These conditions are stated as each item is described. Each entry is checkpointed as follows:	
			1	Terminal entry status byte (from TRMSTATE) included only for a single, group, or line entry	IEDQNK
			2	Input sequence number (from TRMINSEQ) included only for a single, group, line, or process entry that is disk queued	IEDQNK
			2	Output sequence number (from TRMOUTSQ) included only for a single, group, line, or process entry that is disk queued	IEDQNK
			n	Option fields for the terminal table entry	IEDQNK
			2	Count of messages for this destination (from QCBMSGCT in the QCB referred to by TRMDESTQ) included for any single, group, line, or process entry that has not had its QCB checkpointed	IEDQNK

Offset	*Name*	*Bytes*	*Description*	*Initialized By*
		3	Queue-back message chain pointer (from QCBQBACK) included for any single, group, line, or process entry that has not had its QCB checkpointed	IEDQNK
		21	Disk pointers from QCBDNHDR through QCBLFEFO in a priority-level QCB that is attached to this destination QCB; there is one of these 21-byte entries for each priority-level QCB attached to a destination QCB that is being checkpointed	IEDQNK
		3	LCBSTAT1, LCBSTAT2, DCBINTVL for any single, group, or line entry	IEDQNK
		n	Invitation list for any single, line, or group entry that has not had its destination QCB checkpointed; QCBDCBAD points to the DCB, and DCBINVLI points to the invitation list; the length of the list is equal to the number of entries times the length of each entry plus eight control bytes	IEDQNK
12+n		2	The following information is at the end of the environment record. Termname table offset to the primary operator control terminal (from the AVT field AVTOPCON)	IEDQNK
		2	Number of seconds in a system system delay (from the AVT field AVTINTLV)	IEDQNK
		1	TCAM status byte (from the AVT field AVTBIT1)	IEDQNK
		1	TCAM status byte (from the AVT field AVTBIT2)	IEDQNK
		4	Nonreusable disk relative record address (from the AVT field AVTNADDR)	IEDQNK
		4	Reusable disk relative record address (from the AVT field AVTRADDR)	IEDQNK
		4	Value of AVTLODPT	IEDQNK

In summary, the general format of an environment checkpoint record is as follows:

Offset

0	4	8	9	12							
Date	Time	Key X'20'	TTR of last incident rcd used	Terminal Table data for the first entry	Option fields for the first entry	QCB data for the first entry	LCB data for the first entry	DCB data for the first entry	Invitation List for the first entry	Terminal Table data for the second entry	Option fields for the second entry

Second segment offset

0	4	8	9	12			
Date	Time	Key X'1C'	TTR of last incident rcd used	QCB data for the second entry	LCB data for the second entry	DCB data for the second entry	Invitation List for the second entry

Terminal Table data for the last entry	Option fields for the last entry	QCB data for the last entry	LCB data for the last entry	DCB data for the last entry	Invitation List for the last entry	AVT fields

Incident Checkpoint Record for the CHECKPT Macro: The Build Incident Record for the MH routine (IEDQNG) issues a GETMAIN for main storage in which to build this incident checkpoint record and places the address of this area at CKPLDRB in the checkpoint work area. If the CHECKPT macro is issued in the incoming group of MH, the terminal that sent the current buffer is checkpointed. If the CHECKPT macro is issued in the outgoing group of MH, the terminal that is to receive the current message is checkpointed. The length of this record depends on which option table fields are used for the terminal being checkpointed. The Build Incident Record for the MH routine uses the LCB field LCBTTCIN, the offset to the current termname table entry, as input to the Termname Table code (IEDQTNT) to get the correct terminal entry address. The terminal entry field TRMOPTBL is an offset to the beginning of the option table fields for this terminal. The routine adds the option table offset from the terminal entry to the option table address (from AVTOPTPT in the AVT) to refer to the beginning of the option table data for this terminal and uses the individual option entry offsets in the terminal entry to refer to the specific option data entries for this terminal. The second word of the option table contains the address of the option characteristics table, each entry of which corresponds in consecutive order to each option entry offset in a terminal entry. If the Build Incident Record for the MH routine finds that a halfword option entry offset in the terminal entry does not contain X'FF', the routine gets the address of the option data by adding the halfword option entry offset to the beginning of the option data for this terminal to get the beginning of this data field, gets the length of this option data field for the corresponding option characteristics table entry, and moves the option data to the next available location in the incident checkpoint record.

Offset		*Name*	*Bytes*	*Description*	*Initialized By*
0	(0)	CDRDATE	4	Date of the checkpoint	IEDQNB
4	(4)	CDRTIME	4	Time that the record is written	IEDQNP
8	(8)	CDRKEY	1	Key byte: D'00'—CHECKPT record	IEDQNG
9	(9)	CDRSTAT	1	The terminal status (from TRMSTATE)	IEDQNG

Offset	*Name*	*Bytes*	*Description*	*Initialized By*
10 (A)	CDROFFS	2	The offset to the terminal that is currently connected on the line of the LCB that is the request element (from LCBTTCIN)	IEDQNG
12 (C)	CDRSEQIN	2	Input sequence number	IEDQNG
14 (E)	CDRSEQOU	2	Output sequence number	IEDQNG
16 (10)	CDROPTN		Beginning of the option fields defined for the terminal referred to by the offset in bytes 10-11. The manner in which IEDQNG checkpoints these option fields is described in the write-up that precedes this record layout.	IEDQNG

In summary, the general format of an incident checkpoint record for the CHECKPT macro is as follows:

0	4	8	9	10 (A)	12 (C)
Date	Time	Key D'00'	Terminal status	Terminal offset	Sequence numbers
16 (10) Option data fields					

Incident Checkpoint for Operator Control: The Build Incident Checkpoint for Operator Control routine (IEDQNJ) issues a GETMAIN for main storage in which to build this incident checkpoint record and places the address of this area at CKPLDRB in the checkpoint work area. This routine initializes this checkpoint record from the operator control checkpoint element pointed to by OPCCOPCE in the Operator Control AVT.

Offset	*Name*	*Bytes*	*Description*	*Initialized By*
0 (0)	CDRDATE	4	Date of the checkpoint	IEDQNP
4 (4)	CDRTIME	4	Time that the record is written	IEDQNP
8 (8)	CDRKEY	1	Key byte: D'16'—Operator Control record	IEDQNJ
9 (9)	CDRTTRLI	3	Flag bits: Bit 22—ON—Last segment Off—Intermediate segment Bit 23—On—Continuation segment Off—First or only segment	IEDQNJ
12 (C)		36	Operator Control checkpoint element pointed to by OPCCOPCE in the Operator Control AVT	IEDQNJ

In summary, the format of an incident checkpoint record for operator control is as follows:

	0	4	8	9
	Date	Time	Key D'16'	Flag
12 (C)	Operator Control Checkpoint Element			

Incident Checkpoint for the TCHNG Macro: The Build Incident Checkpoint for TCHNG routine (IEDQNH) issues a GETMAIN for main storage in which to build this incident checkpoint record and places the address of this area at CKPLDRB in the checkpoint work area. The checkpoint of the option data fields is handled exactly as explained in the Incident Checkpoint for the CHECKPT Macro discussion.

Offset		*Name*	*Bytes*	*Description*	*Initialized By*
0	(0)	CDRDATE	4	Date of the checkpoint	IEDQNP
4	(4)	CDRTIME	4	Time that the record is written	IEDQNP
8	(8)	CDRKEY	1	Key byte: D'04'—TCHNG record	IEDQNH
9	(9)	CDRSTAT	1	Terminal entry status byte (from TRMSTATE)	IEDQNH
10	(A)	CDROFFS	2	Offset to the termname table entry for the terminal being checkpointed (from bytes 12-13 of the checkpoint request element)	IEDQNH
12	(C)	CDROPTN		Beginning of the option fields defined for the terminal referred to by the offset in bytes 10-11.	IEDQNH

In summary, the general format of an incident checkpoint for TCHNG record is as follows:

0	4	8	9	10 (A)	12 (C)
Date	Time	Key D'04'	Terminal status byte	Terminal offset	Option data fields

CKREQ Checkpoint Record: The Build CKREQ Disk Record routine (IEDQNM) issues a GETMAIN macro for main storage in which to build this CKREQ checkpoint record and places a pointer to this area in the CKPLDRB field of the checkpoint work area. The format and length of this checkpoint record depends upon the number of priority QCBs associated with the destination QCB that is being checkpointed; there is one 21-byte area of QCB disk pointers for each priority level. The checkpoint of the option data fields is handled exactly as explained in the Incident Checkpoint for the CHECKPT Macro discussion. The CKREQ record DSECT is IEDQCDRD.

Offset		*Name*	*Bytes*	*Description*	*Initialized By*
0	(0)	CDRCKFLG	1	Flag bits:	
				Bit 0— On—CLREQ is not complete Off—CKREQ is complete	IEDQNM
1	(1)		3	Link address of the checkpoint disk I/O queue (from CKPIOQF and CKPIOQL in the checkpoint work area)	IEDQNM
4	(4)	CDRCKIN	2	Input sequence number (from TRMINSEQ in the terminal entry that is referred to by the offset at CDRCKOFF)	IEDQNM
6	(6)	CDRCKOUT	2	Output sequence number (from TRMOUTSQ in the terminal entry that is referred to by the offset at CDRCKOFF)	IEDQNM
8	(8)	CDRKEY	1	Key byte:	IEDQNM
				X'18'—CKREQ record	
9	(9)		1	Reserved	
10	(A)	CDRCKOFF	2	Termname Table offset (from DEBTAMOS in the associated DEB)	IEDQNM
12	(C)	CDRCKMSG	2	QCB message count (from QCBMSGCT in the destination QCB)	IEDQNM
14	(E)	CDRCKQBC	3	Queue-back chain pointer (from QCBQBACK in the destination QCB)	IEDQNM
17	(11)	CDRCKQCB	21	Priority QCB disk pointers (from the first 21 bytes of the priority level QCB):	IEDQNM
				QCBDNDHR—disk record number for the next first unit of a message received	
				QCBFHDLZ—disk record number of the first unit of the first message in the last zone used for this queue	
				QCBFHDTZ—disk record number of the first unit of the first message for this queue in the current zone	

Offset	*Name*	*Bytes*	*Description*	*Initialized By*
			QCBINTFF—disk record number in the link field of the message on the read-ahead queue	
			QCBINTLF—disk record number of the message on the read-ahead queue	
			QCBFFEFO—disk record number of the first message received in FEFO order	
			QCBLFEFO—disk record number of the last message received in FEFO order	
17 +(21 x n) where *n* is the number of priority level QCBs	CDRCKOPT		Beginning of the option fields defined for the terminal referred to by CDRCKOFF	IEDQNM

In summary, the general format of a CKREQ checkpoint record is as follows:

Offset

0	1	4	6
Flag CDRCKFLG	Link address	Input sequence number CDRCKIN	Output sequence number CDRCKOUT

Offset

8	9	10 (A)	12 (C)	14 (E)
Key X'18' CDRKEY	Reserved	Terminal name offset CDRCKOFF	QCB message count CDRCKMSG	Queue-back chain pointer CDRCKQBC

Offset

17 (11)			17 + (21 x n)
Priority QCB disk pointers for the first priority level CDRCKQCB		Priority QCB disk pointers for the last priority level	Option data fields

Checkpoint Elements

Environment Checkpoint Request Element:

Defined at AVTCKELE in the AVT

Four words long

Key field—always B'01110000'

Source flag:

B'10000000'—requested by READY

B'01000000'—requested by MCPCLOSE

B'00010000'—requested by the Checkpoint-No Incident Records routine

B'00100000'—requested by other routines

Offset			
0	Key X'70'	QCB address	
4	Link address		
8	Source flag	Reserved	Checkpoint time interval
12 (C)	Time of interrupt		Reserved

MH Checkpoint Request Element:

Defined as the LCB

Key field—always B'00000000'

Offset		
0	Key X'00'	Checkpoint QCB address
4	Link address	
60 (5A)	Terminal name offset	

Application Program Checkpoint Request Element:

Defined at PCBWRKA in the PCB—one for each application program

Four words long

Key field—depends on the macro

B'01100000'—requested by CKREQ

B'00010000'—requested by TCHNG

CKREQ

Offset		
0	Key X'60'	Checkpoint QCB address
4	Link address	
8	ECB	
12 (C)	DEB chain address	

TCHNG

Offset		
0	Key X'10'	Checkpoint QCB address
4	Link address	
8	ECB	
12 (C)	Terminal name offset	Reserved

Operator Control Checkpoint Request Element:

Offset		
0	Key X'40'	Checkpoint QCB address
4	Link address	

Defined at AVTCCELE in the AVT

Two words long

Key field—B'01000000'—requested by VARY, MODIFY, RELEASE, HOLD, ICHNG, MRELEASE, or RELEASEM

Checkpoint QCB:

Offset		
0	Key X'02'	Request element chain
4	ECB	
8	Address of the STCB code offset	

Defined at AVTCKPTB in the AVT

Three words long

Third word always points to the key field of this QCB. The key field is the offset to the checkpoint STCB

Key field—B'00000010'—tells the TCAM Dispatcher to post the ECB in the second word and to take the element off the top of the ready queue and chain it to the request element chain, which is in the first word of the QCB.

Checkpoint Work Area

The checkpoint work area is a local constants and variables area that is used by all of the checkpoint routines. This work area contains the checkpoint data set control record, as well as pointers to the other checkpoint records. The checkpoint work area is allocated by a GETMAIN in the Checkpoint Open routine (IGG01941), which also places the address of the work area in the AVTCKGET field of the AVT. During a cold start-up, the constant fields in the work area are initialized by the Checkpoint Open routine, the Checkpoint Disk Initialization routine (IGG01942), and the Checkpoint Disk Allocation routine (IGG01949). The variable fields in the checkpoint work area are initialized and changed as required by the checkpoint routines.

IEDQCKPD

Offset	Field	Description
0 (0)	**CKPSAVE1**	Save Area for the Load Module
72 (48)	**CKPIOB**	IOB for Checkpoint Disk I/O
	CKPIOFL1	
	CKPIOFL2	
	CKPIOSN0	
	CKPIOSN1	
76 (4C)	**CKPIOECB**	ECB Address
80 (50)	**CKPIOFL3**	
81 (51)	**CKPIOCSW**	Channel Status Word
88 (58)	**CKPIOSIO**	Condition Codes
89 (59)	**CKPIOCPA**	Channel Program Address
92 (5C)		Reserved
93 (5D)	**CKPIODCB**	DCB Address
96 (60)		Reserved
97 (61)	**CKPIORST**	Restart Address
100 (64)	**CKPIOBCI**	Block Count Increment
102 (66)	**CKPIORC**	Error Count
104 (68)	**CKPIOM**	M Seek Address
105 (69)	**CKPIOBB**	BB Seek Address
107 (6B)	**CKPIOCC**	CC Seek Address
		Continued
109 (6D)	**CKPIOHH**	HH Seek Address
111 (6F)	**CKPIOR**	R Seek Address
112 (70)	**CKPECB**	ECB Posted by IOS
116 (74)	**CKPEXCP**	Address of the Current Record Being Written
120 (78)	**CKPCNVRT**	Label Used for the CVD Instruction
	CKPECBL	ECB List for WAIT
	CKPEPLOC	EPLOC for the LOAD Macro
128 (80)	**CKPIOQF**	Address of the First Record On the Checkpoint Disk I/O Queue
132 (84)	**CKPIOQL**	Address of the Last Record On the Checkpoint Disk I/O Queue
136 (88)	**CKPLREB**	Address of the Last Request Element for Which a Record Was Built

140 (8C)	CKPLDRB	Address of the Last Record Built
144 (90)	CKPSECTR	Sector ID for Set Sector Command
	CKPCTTRB	Beginning of the CKREQ-TTR Table
148 (94)	CKPCPARM	Parameters for the Convert Routine
156 (9C)	CKPPARM2	Parameter for the Sector Convert Routine
160 (A0)	CKPCRLEN	Control Record Length
161 (A1)	CKPSWCH1	Switch for the Checkpoint QCB
162 (A2)	CKPSWCH2	Switch for the Checkpointed Invitation List
163 (A3)	CKPERRCT	Count Or Read Errors Found by IGG01943
164 (A4)		Reserved
168 (A8)	CKPCCWS CKPSEEKC	Channel Program Sect Cylinder
176 (B0)	CKPSETSC	Set Sector or No OP
184 (B8)	CKPSCHID	Search ID Command
192 (C0)	CKPTIC	TIC Command
197 (C5)	CKPTTRLT	TTR of the Last Segment Written
200 (C8)	CKPRW	Read/Write
208 (D0)	CKPGETML	GETMAIN Parameter List
218 (DA)	CKPWKALN	Checkpoint Work Area Length
220 (DC)	CKPMSG CKPMSGLN	Message Buffer Length
222 (DE)		Reserved

224 (E0) **CKPMSGTX** Message Text **CKPSAVE2** Save Area

261 (105) **CKPMSGTP** Type of Checkpoint Record

262 (106) **CKPRCDSR** No Segments in One Environment Checkpoint

264 (108)

266 (10A) **CKPTRKSA** Number of Tracks Available

267 (10B) **CKPMSGL** Message Length **CKPMSGPN** Process Entry Name

268 (10C) **CKPMSGPN** (Cont.)

281 (119) **CKPMSGGL** GETMAIN Length That Was Not Satisfied

284 (11C) **CKPMSGGL** (Cont.)

CKPTRMAD Terminal Entry Address

288 (120) **CKPCNTLR** Beginning of the Checkpoint Control Record

Temporary Use of the Checkpoint Work Area During Checkpoint Open:

116 (74) **CKPCYLNO** Cylinder Number		118 (76) **CKPHEDNO** Head Number
120 (78) **CKPRCDNO** Record Number	121 (79) **CKPKEYLN** Key Length	122 (7A) **CKPDATLN** Data Length
124 (7C) **CKPCTTRC** Current Entry in the CKREQ-TTR Table		
128 (80) **CKPDATIM** Date and Time of the Last Environment Checkpoint		
136 (88) **CKPIPERE** Number of Incident Or CKREQ Records in One Environment Record Segment		138 (8A) Reserved
140 (8C) **CKPCTTRA** Address of the TTR of the Environment Record to be Used for Restart		

Offset		*Name*	*Bytes*	*Description*
0	(0)	CKPSAVE1	72	Save area for the load module
72	(48)	CKPIOB	40	IOB for the checkpoint disk I/O operations
72	(48)	CKPIOFL1	1	I/O error flags
73	(49)	CKPIOFL2	1	I/O error flags
74	(4A)	CKPIOSN0	1	
75	(4B)	CKPIOSN1	1	
76	(4C)	CKPIOECB	4	ECB address
80	(50)	CKPIOFL3	1	I/O error flags
81	(51)	CKPIOCSW	7	Channel status word
88	(58)	CKPIOSIO	1	Start I/O condition codes
89	(59)	CKPIOCPA	3	Channel program address
92	(5C)		1	Reserved
93	(5D)	CKPIODCB	3	DCB address
96	(60)		1	Reserved
97	(61)	CKPIORST	3	Restart address
100	(64)	CKPIOBCI	2	Block count increment
102	(66)	CKPIORC	2	Error count
104	(68)	CKPIOM	1	M seek address
105	(69)	CKPIOBB	2	BB seek address
107	(6B)	CKPIOCC	2	CC seek address
109	(6D)	CKPIOHH	2	HH seek address
111	(6F)	CKPIOR	1	R seek address
112	(70)	CKPECB	4	ECB posted by the I/O Supervisor
116	(74)	CKPEXCP	4	Address of the current record being written
116	(74)	CKPCYLNO	2	During checkpoint open, the cylinder number
118	(76)	CKPHEDNO	2	During checkpoint open, the head number
120	(78)	CKPCNVRT	8	Label used for the CVD instruction
120	(78)	CKPECBL	8	ECB list for WAIT
120	(78)	CKPEPLOC	8	EPLOC for the LOAD macro
120	(78)	CKPRCDNO	1	During checkpoint open, the record number
121	(79)	CKPKEYLN	1	During checkpoint open, the key length
122	(7A)	CKPDATLN	2	During checkpoint open, the data length
124	(7C)	CKPCTTRC	4	Address of the current entry in the CKREQ-TTR table—used for restart open
128	(80)	CKPIOQF	4	Address of the first record on the checkpoint disk I/O queue
128	(80)	CKPDATIM	8	Date and time of the last environment checkpoint, used during checkpoint open
132	(84)	CKPIOQL	4	Address of the last record on the checkpoint disk I/O queue
136	(88)	CKPLREB	4	Address of the last request element for which a checkpoint record was built

Offset		Name	Bytes	Description
136	(88)	CKPIPERE	2	During checkpoint open, the number of incident or CKREQ checkpoints in one environment record segment
140	(8C)	CKPLDRB	4	Address of the last disk record built
140	(8C)	CKPCTTRA	4	During checkpoint open, the address of the TTR of the environment record being used for restart
144	(90)	CKPCTTRB	4	Address of the beginning of the CKREQ-TTR table
148	(94)	CKPCPARM	8	Parameters for the Convert routine: the address of the DEB and the address for the conversion result
156	(9C)	CKPPARM2	4	Parameter for Sector Convert routine (UCB type and address)
160	A0)	CKPCRLEN	1	Length of the control record
161	(A1)	CKPSWCH1	1	Switch used for comparing a QCB to see if it has been checkpointed
162	(A2)	CKPSWCH2	1	Switch used for comparing an invitation list to determine whether it has been checkpointed
163	(A3)	CKPERRCT	1	Count of the read errors found by IGG01943
164	(A4)		4	Reserved
168	(A8)	CKPCCWS	32	Channel program
168	(A8)	CKPSEEKC	8	Seek Cylinder command
184	(B8)	CKPSCHID	8	Search ID command
192	(C0)	CKPTIC	8	TIC command
197	(C5)	CKPTTRLT	3	TTR of the last environment segment written
200	(C8)	CKPRW	8	Read/Write command
		CKPREAD		Read Data CCW
		CKPWRITE		Write Data CCW
		CKPWCKD		Write Count, Key, and Data CCW
208	(C0)	CKPGETML	10	GETMAIN parameter list
218	(DA)	CKPWKALN	2	Length of the checkpoint work area
220	(DC)	CKPMSG		Message buffer used for WTO
220	(DC)	CKPMSGLN	2	Length of the message buffer
222	(DE)		2	Reserved
224	(E0)	CKPMSGTX	37	Message text
224	(E0)	CKPSAVE2	15	Temporary storage area
261	(105)	CKPMSGTP	20	Type of checkpoint record
262	(106)	CKPRCDSR	2	Number of segments in one environment checkpoint
264	(108)	CKPTRKLN	2	Reserved
266	(10A)	CKPTRKSA	2	Number of tracks available in the checkpoint data set
267	(10B)	CKPMSGPN	4	Process entry name
281	(119)	CKPMSGGL	4	GETMAIN length that could not be satisfied
284	(11C)	CKPTRMAD	4	Terminal entry address
288	(120)	CKPCNTLR	1	Beginning of the checkpoint control record

Command Input Buffer

The command input buffer (IEZCIB) is a variable-length communication parameter list that is used by Operator Control to process a command. The buffer describes the command sent from the console. The CIB shows the command code, the identification of the console that issued the command, and the actual data in the command.

When the INTRO macro instruction is expanded at TCAM execution time, the INTRO macro generates linkage to a module that issues an EXTRACT macro. The FIELDS= parameter specified on the EXTRACT macro is FIELDS=COMM, which calls for the communication parameter list. AVTCOMPT is specified as the answer area address on the EXTRACT macro. The operating system places the address of the communication parameter list (command input buffer) in the AVTCOMPT field of the address vector table.

When a command is entered, SVC 34 performs a GETMAIN for the area required by the command input buffer, and the buffer is initialized at that time.

The format of the command input buffer is illustrated below; descriptions of the fields follow.

IEZCIB

0 (0)	**CIBNEXT**	Address of the Next CIB in the Queue
4 (4)	**CIBVERB**	Code Byte
5 (5)	**CIBLEN**	Buffer Length
6 (6)		Reserved
12 (C)	**CIBCONID**	Console ID
13 (D)		Reserved
14 (E)	**CIBDATLN**	CIB Data Length
16 (10)	**CIBDATA**	CIB Data

Offset		*Name*	*Byte*		*Description*	
0	(0)	CIBNEXT	4		Address of the next CIB in the queue (0 for last)	
4	(4)	CIBVERB	1		Bit settings for this field are:	
			Name	*Bits*	*Value*	*Meaning*
			CIBSTART	5	X‘04’	START command code
			CIBMODFY	1,5	X‘44’	MODIFY command code
			CIBSTOP	1,2,5	X‘64’	STOP command code
			CIBVARY	2,4	X‘28’	VARY command code
			CIBHALT	2,3,4,5	X‘3C’	HALT command code
			CIBDISPL	1,2,4	X‘68’	DISPLAY command code
			CIBHOLD	1,2,4,5	X‘6C’	HOLD command code
			CIBRELSE	1,2,3	X‘70’	RELEASE command code
5	(5)	CIBLEN	1		Length of the butfer (including control fields) in doublewords	
6	(6)		6		Reserved	
12	(C)	CIBCONID	1		Identifier of the console issuing the command	
13	(D)		1		Reserved	
14	(E)	CIBDATLN	2		Length of data in the CIB	
16	(10)	CIBDATA	n		Beginning of the data from the command operand:	

START data—contains the fourth positional parameter, *parmvalue*

MODIFY data—contains the residual operand image following the comma, terminating the first positional parameter

STOP data—none, CIB generated only to give the console ID to the recipient task

VARY data—contains the operand field for the command issued

HALT data—contains the operand field for the command issued

DISPLAY data—contains the operand field for the command issued

HOLD data—contains the operand field for the command issued
RELEASE data—contains the operand field for the commandissued

Concentrator Data Ready Queue

A concentrator data ready queue (DRQ) controls message concentration for output to a concentrator. There is one DRQ for every concentrator defined in the TCAM system. A DRQ is the same size as a master destination QCB.

A data ready queue has three primary fields: a pointer to the element chain, a link address, and a pointer to the STCB chain. The element chain consists of send scheduler STCBs from destination queues that have data ready to be sent to a concentrator. The link and the STCB chain fields are the same as for a master QCB.

The address of the DRQ for a concentrator is in the TRMDESTQ field of the terminal entry for the concentrator.

The DSECT names of the DRQ fields are shown in the following layout. Descriptions of the fields follow the layout.

IEDQDRQ

0 (0) **DRQFLAG1** Flag Byte	1 (1) **DRQELCHN** Element Chain		
4 (4) **DRQPRI** Priority	5 (5) **DRQLINK** DRQ Link Field		
8 (8) **DRQSTVTO** Index into the Subtask Vector Table	9 (9) **DRQSTCHN** STCB Chain		
12 (C) **DRQSTPRI** STCB Priority	13 (D) **DRQSLINK** Pointer to the Next STCB		
16 (10) **DRQBUFCT** Total Buffer Count	17 (11) **DRQERBCT** ERB Buffer Count	18 (12) **DRQTDO** Time Delay Queue Offset	19 (13) **DRQSTAT** QCB Status Byte
20 (14) **DRQSCBOF** SCB Offset	21 (15) **DRQCURQ** Pointer to the Current QCB		
24 (18) **DRQFLAG3** DRQ Flag Byte	25 (19) **DRQRESV** Reserved	26 (1A) **DRQCTBCT** CTB Count	27 (1B) **DRQCTBMX** Maximum CTBs per Concentrator
28 (1C) **DRQPRLVL** Highest-Priority Level Message	29 (1D) **DRQTDLNK** Link Field for the Time Delay Queue **DRQPRVLK** Link Field Pointer		
32 (20) **DRQRELLN** Relative Line Number	33 (21) **DRQDCBAD** DCB Address		
36 (24) **DRQFLAG2** DRQ Status Byte	37 (25) **DRQQBACK** Queue-Back Chain Pointer		

Offset	*Name*	*Bytes*	*Description*		
0 (0)	DRQFLAG1	1	DRQ flag byte		
		Bit Definitions:			
		Name	*Bit*	*Value*	*Meaning*
		DRQDRQQ	5	X'04'	Indicates a DRQ
		DRQFQCB	6	X'02'	Indicates a QCB
		DRQHELD	7	X'01'	Stop sending—reuse
1 (1)	DRQELCHN	3	Element chain of Send Scheduler STCBs		
4 (4)	DRQPRI	1	Priority of the DRQ		
5 (5)	DRQLINK	3	Link field of the DRQ		
8 (8)	DRQSTVTO	1	Index to the appropriate entry in the subtask vector table		
		Bit Definitions:			
		Name	*Bit*	*Value*	*Meaning*
		DRQCSVTO	2	X'20'	DRQ VTO (offset)
9 (9)	DRQSTCHN	3	STCB chain pointer		
12(C)	DRQSTPRI	1	Priority of the STCB		
13(D)	DRQSLINK	3	Pointer to the next STCB in the chain		
16(10)	DRQBUFCT	1	Total buffer count		
17(11)	DRQERBCT	1	ERB buffer count		
18(12)	DRQTDO	1	Time delay queue offset		
19(13)	DRQSTAT	1	Status of this QCB		
		Bit Definitions:			
		Name	*Bit*	*Value*	*Meaning*
		DRQEOM	0	X'80'	End of message is sent
		DRQTRMHO	1	X'40'	Terminal was held
		DRQBUFRD	2	X'20'	Buffered terminal
		DRQSEND	3	X'10'	Sending to a buffered terminal
		DRQRECEV	4	X'08'	Receiving from a buffered terminal
		DRQSCHDL	5	X'04'	Put in the time delay queue when inactive
		DRQCLOCK	6	X'02'	ON=CLOCK, OFF=INTERVAL
		DRQTIME	7	X'01'	Delay is greater than 12 hours
20 (14)	DRQSCBOF	1	Offset to the proper SCB		
21 (15)	DRQCURQ	3	Pointer to the current QCB		
24 (18)	DRQFLAG3	1	DRQ flag byte		
		Bit Definitions:			
		Name	*Bit*	*Value*	*Meaning*
		DRQFSPCI	1	X'80'	First PCI for a concentrated message
		DRQERBAV	2	X'40'	Enabled code requested the ERB

Offset		*Name*	*Byte*	*Description*
25	(19)	DRQRESV	1	Reserved
26	(1A)	DRQCTBCT	1	CTB count
27	(1B)	DRQCTBMX	1	Maximum number of CTBs per concentrated message
28	(1C)	DRQPRLVL	1	Highest-priority level message
29	(1D)	DRQPRVLK	3	Pointer to the link field
29	(1D)	DRQTDLNK	3	Link field for the time delay queue
32	(20)	DRQRELLN	1	Relative line number
33	(21)	DRQDCBAD	3	DCB address
36	(24)	DRQFLAG2	1	DRQ status byte

Bit Definitions:

Name	*Bit*	*Value*	*Meaning*
DRQTSSES	0	X‘80’	TSO session is in progress
DRQRSRV	3	X‘10’	Reuse serviced bit
DRQTERMQ	4	X‘08’	Queuing is by terminal
DRQSDFFO	5	X‘04’	Currently sending a FEFO message
DRQPROC	6	X‘02’	This QCB is for a process entry
DRQCKPT	7	X‘01’	Checkpoint flag

37	(25)	DRQQBACK	3	Queue-back chain pointer

(This page left blank·intentionally)

Common Buffer Data Area Prefix

The common buffer data area prefix is a three-word prefix to each data area used for transmitting broadcast data. Storage is allocated for each data area prefix at INTRO time. The data area prefix is partially initialized at INTRO time, the rest when the COMMBUF macro is executed.

The format of the common buffer data area prefix is shown in the following layout. Descriptions of the fields follow the illustration.

IEDCBDA

<table>
<tr><td>0 (0)
CBDAINDX
Index Byte</td><td colspan="2">1 (1)
CBDAADDR
Pionter to Data in the Area</td></tr>
<tr><td colspan="2">4 (4)
CBDALEN
Length of Data in this Area</td><td>6 (6)
CBDAUSCT
Number of LCBs Using this Area</td></tr>
<tr><td>8 (8)
CBDAFLG1
Flag Byte</td><td>9 (9)
CBDAFLG2
Flag Byte</td><td>10 (A)
Reserved</td></tr>
</table>

Offset		*Name*	*Bytes*	*Description*
0	(0)	CBDAINDX	1	Index byte
1	(1)	CBDAADDR	3	Pointer to data in this area
4	(4)	CBDALEN	2	Length of data in this area
6	(6)	CBDAUSCT	2	Number of LCBs using this area
8	(8)	CBDAFLG1	1	Flag byte 1

Bit Definitions:

Name	*Bit*	*Value*	*Meaning*
CBDAINUS	7	X'01'	data in use

9	(9)	CBDAFLG2	1	Flag byte 1
10	(A)		2	Reserved

(This page left blank intentionally)

Common Buffer Master QCB

The common buffer master QCB (CMB) is a fixed-length, control block of 32 bytes. This control block regulates the use of common buffer data areas and STCBs used in transmitting broadcast data. Storage is allocated for the QCB at INTRO time. The QCB is also initialized at INTRO time.

Note: *There is no priority QCB for common buffer transmission. The QCB is truncated at the displacement 32 (X'20').*

The format of the common buffer master QCB is shown in the following layout. Descriptions of the fields follow the layout.

IEDCMB

0 (0) **CMBFLAG1** Flag Byte	1 (1) **CMBSTCB** Address of First STCB
4 (4) **CMBPRI** Priority	5 (5) **CMBLINK** Pionter to Next STCB in Chain
8 (8) **CMBSTVTO** Index to Entry in Subtask Vector Table	9 (9) **CMBRETRN** Reserved
12 (C) **CMBDAREA** Number of Data Areas	14 (E) **CMBASTCB** Number of Available STCBs
16 (10) **CMBFINDX** Index to First Data Area Prefix	17 (11) **CMBFIRST** Address of First Data Area Prefix
20 (14) **CMBLINDX** Index to Last Data Area Prefix	21 (18) **CMBLAST** Address of Last Data Area Prefix
24 (18) **CMBNINDX** Index to Next Data Area Prefix	25 (19) **CMBNEXT** Address of Next Data Area Prefix
28 (1C) Reserved	30 (1E) **CMBSIZE** Data Area Size

Offset		*Name*	*Bytes*	*Description*
0	(0)	CMBFLAG1	1	Flag byte
1	(1)	CMBSTCB	3	Address of first STCB
4	(4)	CMBPRI	1	Priority
5	(5)	CMBLINK	3	Pointer to next STCB in chain
8	(8)	CMBSTVTO	1	Index to the entry in the subtask vector table
9	(9)	CMBRETRN	3	Reserved
12	(C)	CMBDAREA	2	Number of data areas
14	(E)	CMBASTCB	2	Number of available STCBs
16	(10)	CMBFINDX	1	Index to the first data area prefix

Offset		*Name*	*Byte*	*Description*
17	(11)	CMBFIRST	3	Address of the first data area prefix
20	(14)	CMBLINDX	1	Index to the last data area prefix
21	(15)	CMBLAST	3	Address of the last data area prefix
24	(18)	CMBINDX	1	Index to the next data area prefix
25	(19)	CMBNEXT	3	Address of the next data area prefix
28	(1C)		2	Reserved
30	(1E)	CMBSIZE	2	Data area size

Concentrator Device ID Table

There is one device ID table for each concentrator defined in the TCAM system. Each table consists of a control area that contains information about the entire table, an entry for the concentrator, and one entry for each of the attached terminals. Each entry contains the length of the ID, the device ID, and the termname table offset for this concentrator or terminal. A device ID table is used to find the proper terminal entry for a terminal that is attached to the concentrator.

A device ID of X'FF' represents either a concentrator or an attached terminal for which DVCID=NONE is coded. A X'FE' denotes the end of the table.

The following is the format of the control area and of the concentrator entry for a concentrator device ID table.

IEDQDVCT

0 (0) **DVCNO** Number of Entries in the Table **DVCIDLTH** Length of Device ID Characters	1 (1) **DVCRSV** Reserved **DVCCHAR** ID Characters	2 (2) **DVCENLTH** Entry Length	3 (3) **DVCSTAT** Status Byte
4 (4) **DVCECW** End of the Control Word X'01'	5 (5) X 'FF'	6 (6) Termname Table Offset	

The device ID entries for the terminals attached to the concentrator follow the concentrator entry. Each device ID entry has one of the two following formats.

If DVCID=NONE is coded:

DVCID = NONE

X '01'	X 'FF'	Termname Table Offset

V-17-C

If DVCID=CHAR is coded:

DVCID = CHAR

Length of the Device ID	Device ID Characters	Termname Table Offset

V-17-D

The following is the assembled DSECT format of this table.

Offset	*Name*	*Bytes*	*Description*			
0 (0)	DVCNO	1	For the control area, the number of entries in the table			
0 (0)	DVCIDLTH	1	For a concentrator entry, the length of the device ID characters			
1 (1)	DVCRSV	1	For the control area, reserved			
1 (1)	DVCCHAR	1+	For a concentrator entry, the ID characters (1 or more bytes)			
2 (2)	DVCENLTH	1	For the control area, the length of an entry			
3 (3)	DVCSTAT	1	For the control area, a status byte			
		Name	*Bit*	*Value*	*Meaning*	
		DVCSORTD	1	X‘40’	The table is sorted	
4 (4)	DVCECW	The end of the control word				
	DVCEND	X‘FE’—the end of the table				

Data Control Block

The data control block (DCB) is a storage area through which information needed for the access routines to store and retrieve data is communicated. The format of a TCAM DCB is determined by the character of the data set it represents. There are five types of data control blocks used in TCAM message control programs and application programs. They are:

line groups
message queues
checkpoint
message logging
application program

The TCAM DCB is divided into three segments—prefix, foundation, and extension. The contents of the foundation segment changes during processing. Storage is allocated for the DCB at assembly time, and it is initialized partially at assembly time and partially at execution time according to the parameters specified on the DD card. Before open time, the first doubleword of the foundation segment, at a displacement of 40 (X'28) from the beginning of the DCB, contains the *ddname* of the data set to be opened. After the data set is opened, the same doubleword contains the address of the data extent block. This address is used to set up linkages in the TCAM execution.

The address of the TCAM data control block is in the DEBDCBAD field of the data extent block. The same address is also in the QCBDCBAD field of the destination queue control block.

The format of a data control block is illustrated below; descriptions of the fields follow.

Data Control Block DSECT (IHADCB)

Data Set Interface

Line Group

20 (14)	**DCBUFOU** **DCBUFIN**	Number of Buffers
21 (15)	**DCBMH**	MH Address for this Line Group
24 (18)	**DCBINTVL**	Invitation Delay Interval
25 (19)	**DCBPCI**	PCI Byte
26 (1A)	**DCBDSORG**	Data Set Organization
28 (1C)	**DCBBFUMA**	Maximum Buffer Count for Transfer
	DCBIOBAD	IOB Base Address
32 (20)	**DCBCPRI**	Priority
33 (21)	**DCBTRANS**	Translation Table Address
36 (24)	**DCBEIOBX**	Extended IOB Index
	DCBEXLST	Exit List Address

Message Queues

20 (14)		Reserved
26 (1A)	**DCBDSORG**	Data Set Organization
28 (1C)		Reserved
	DCBIOBAD	Before Open, AVT Address
32 (20)	**DCBTHRES**	Disk Threshold Value
33 (21)		Reserved
36 (24)		Reserved
	DCBEXLST	Exit List Address

Checkpoint

20 (14)		Reserved
26 (1A)	**DCBDSORG**	Data Set Organization
28 (1C)		Reserved
	DCBIOBAD	Before Open, AVT Address
32 (20)		Reserved
36 (24)		Reserved
	DCBEXLST	Exit List Address

Data Set Interface (Cont.)

Message Logging

20 (14)	Reserved
32 (20)	**DCBEODAD** DECB Pointer
36 (24)	Reserved

Application Program

20 (14)	Reserved		
24 (18)	**DCBBUFL** Buffer Length	26 (1A)	**DCBDSORG** Data Set Organization
28 (1C)	Reserved		
32 (20)	**DCBEODAD** End-of-File Routine Address		
36 (24)	**DCBRECFM** Record Format		**DCBEXLST** Exit List Address

Foundation

Before OPEN

40 (28) **DCBDDNAM** Data Set Name		
48 (30) **DCBOFLGS** Open Flags	49 (31) **DCBIFLG** IOS Error Flags	50 (32) **DCBMACR** Macro Instruction Reference

After OPEN

40 (28) **DCBTIOT** DD Offset	42 (2A) **DCBMACRF** Macro Instruction Reference
44 (2C) **DCBIFLGS** IOS Error Flags	**DCBDEBAD** DEB Address
48 (30) **DCBOFLGS** Open Flags	

Extension

Line Group

48 (30) Reserved	**DCBSCTAD** Special Characters Table Address	
52 (34) **DCBILCT** Count of Invitation Lists	53 (35) **DCBUNTCT** Unit Count	54 (36) **DCBBUFSI** Buffer Size
56 (38) **DCBRESER** Reserve Bytes Counts		
60 (3C) **DCBINVLI** Invitation List Address		
DCBINVLI		
DCBINVLI		
DCBINVLI		

Message Queues/Checkpoint

48 (30)	Reserved
52 (34) **DCBOPTCD** Code Byte	53 (35) Reserved

Extension (Cont.)

Message Logging

48 (30) Reserved	**DCBREAD, DCBWRITE** READ or WRITE Module Address
52 (34) Reserved	
72 (48) **DCBNCP** Count of Write Operations	73 (49) Reserved

Application Program

48 (30) Reserved	**DCBREAD, DCBWRITE** **DCBGET, DCBPUT** READ/WRITE or GET/PUT Module Address
52 (34) **DCBOPTCD** Code Byte	**DCBCHECK** CHECK Module Address
56 (38) **DCBSYNAD** Synchronizing Routine Address	
60 (3C) Reserved	62 (3E) **DCBBLKSI** Maximum Block Size
64 (40) Reserved	82 (52) **DCBLRECL** Logical Record Length
84 (54) **DCBCNTRL, DCBNOTE, DCBPOINT** CNTRL or NOTE/POINT Routine Address	

Offset	*Name*	*Bytes*	*Description*
Line Group Interface			
20(14)	DCBBUFIN/ DCBBUFOU	1	Bits 0-3: Number of buffers assigned initially for receiving operations, for each line in line group Bits 4-7: Number of buffers assigned initially for sending operations, for each line in the line group
21(15)	DCBMH	3	Address of the message handler for this line group
24(18)	DCBINTVL	1	Number of seconds on invitation delay
25(19)	DCBPCI	1	Program-controlled interruption (PCI) handling byte:

Bit	*Value*	*Meaning*
0	X'80'	PCI=(X,)
1	X'40'	PCI=(,X)
2	X'20'	PCI=(A,)
3	X'10'	PCI=(,A)
4	X'08'	PCI=(N,)
5	X__04'	PCI=(,N)
6	X'02'	PCI=(R,)
7	X'01'	PCI=(,R)

Offset	*Name*	*Bytes*	*Description*
26 (1A)	DCBDSORG	2	Data set organization: Byte 0=0 Byte 1 (Code)=TX X'40'
28 (1C)	DCBBFUMA	1	Maximum number of buffers to be used for data transfer for each line in this group
28 (1C)	DCBIOBAD	4	Before open: address of AVT. After open: base for addressing IOBs (BASE=address of first IOB minus length of one LCB)
32 (20)	DCBCPRI	1	Relative priority to be given to sending and receiving operations

Bits	*Value*	*Meaning*
0-4		Reserved bits
5	X'04'	R—Receiving has priority
6	X'02'	E—Receiving and sending have equal priori
7	X'01'	S—Sending has priority

Offset	*Name*	*Bytes*	*Description*
33 (21)	DCBTRANS	3	Address of the translation table

Table	*Code*
IEDQ10	1030
IEDQ11	1050
IEDQ12	105F
IEDQ13	1060
IEDQ14	2260
IEDQ15	2265
IEDQ16	2740
IEDQ17	274F

Offset	*Name*	*Bytes*	*Description*	
			IEDQ18	ITA2
			IEDQ19	ZSC3
			IEDQ20	TTYA
			IEDQ21	TTYB
			IEDQ22	TTYC
			IEDQ23	6BIT
			IEDQ24	ASCI
			IEDQ25	EBCD
			IEDQ26	BC41
			IEDQ27	EB41
			IEDQ28	CR41
			user table	user table name
36 (24)	DCBEIOBX	1	Extended IOB index (size of an LCB)	
36 (24)	DCBEXLST	4	Address of the exit list	

Direct Access Storage Device Message Queue Interface, Checkpoint Data Set Interface, Message Logging Interface, Application Program Interface

Offset	*Name*	*Bytes*	*Description*
20 (14)		4	Reserved
24 (18)	DCBBUFL	2	Length of the buffer
26 (1A)	DCBDSORG	2	Data set organization: Byte 0=0 Byte 1 (Code)=TQ X'20'
28 (1C)		1	Reserved
28 (1C)	DCBIOBAD	4	Before open: address of the AVT
32 (20)	DCBTHRES	1	Percentage of the nonreusable disk, message queue records to be used before a flush closedown of the system is initiated
32 (20)	DCBEODAD	4	Message logging—work area used as a DECB pointer; Application program—address of user end-of-file routine
36 (24)	DCBRECFM	1	Record format
36 (24)	DCBEXLST	4	Address of the exit list

Foundation Segment—Before Open

Offset	*Name*	*Bytes*	*Description*		
40 (28)	DCBDDNAM	8	Data set name		
48 (30)	DCBOFLGS	1	Flags used by OPEN:		
			Bit	*Value*	*Meaning*
			0,1,2, 4,5,6		Reserved
			3	X'10'	Open has been successfully completed
			7	X'01'	DCB is being processed by I/O support rou
49 (31)	DCBIFLG	1	Used by IOS for error conditions		
50 (32)	DCBMACR	2	Macro instruction reference:		
			Bit	*Value*	*Meaning*
		Byte 1			
			0,2,3, 4,5,6,7		Reserved

Offset		Name	Bytes	Description		
				1	X‘40’	GET
			Byte 2			
				0,2,3, 4,5,6,7		Reserved
				1	X‘40’	PUT
Foundation Segment—After Open						
40	(28)	DCBTIOT	2	Offset of the DD entry from beginning of the TIOT		
42	(2A)	DCBMACRF	2	Same as DCBMACR before OPEN		
44	(2C)	DCBIFLGS	1	Same as DCBIFLG before OPEN		
45	(2D)	DCBDEBAD	3	Address of DEB		
48	(30)	DCBOFLGS	1	Same as DCBOFLGS before OPEN		
Line Group Extension						
49	(31)	DCBSCTAD	3	Address of special characters table		
52	(34)	DCBILCT	1	Count of invitation lists		
53	(35)	DCBUNTCT	1	Before open: numerical value of the SCT. After open: count of units for one buffer		
54	(36)	DCBBUFSI	2	Size of all buffers used for this line group		
56	(38)	DCBRESER	4	4 one-byte values (zero default value)		
		Byte 1		Number of bytes reserved in the buffer receiving the first incoming segment of a message		
		Byte 2		Number of bytes reserved in all buffers except the one containing the first segment of a message		
		Bytes 3-4		Reserved		
60	(3C)	DCBINVLI	4n	4-byte address for each (*n*) invitation list		
				Bits	*Value*	*Meaning*
			Byte 1	0,1, 3,5, 6,7,		Reserved
				2	Off	[A,]
				4	Off	[,A]
				2	On	[B,]
				4	On	[,B]
			Bytes 2-4			Reserved
Message Queues/Checkpoint Extension						
49	(31)		3	Reserved		
52	(34)	DCBOPTCD	1	Code byte:		
				Bit	*Value*	*Meaning*
				2	X‘20’	Checkpoint
				6	X‘02’	Nonreusable disk queues
				7	X‘01’	Reusable disk queues

Offset	*Name*	*Bytes*	*Description*		
53 (35)		7	Reserved		
Message Logging Extension					
48 (30)	DCBREAD, DCBWRITE	4	Address of the READ or WRITE module		
52 (34)		20	Reserved		
72 (48)	DCBNCP	1	Number of Write operations that can be performed		
73 (49)		15	Reserved		
Application Program Extension					
48 (30)	DCBREAD, DCBWRITE DCBGET, DCBPUT	4	Address of the READ or WRITE module Address of the GET or PUT module		
52 (34)	DCBOPTCD	1	Option codes		
52 (34)	DCBCHECK	4	Address of the CHECK module		
56 (38)	DCBSYNAD	4	Address of the user synchronizing routine		
60 (3C)	DCBFLAG1	1	TCAM flag byte		
			Bits	*Value*	*Meaning*
			0	X'80'	STOP=QUICK Specified by user
			1	X'40'	STOP=FLUSH Specified by user
61 (3D)		1	Reserved		
62 (3E)	DCBBLKSI	2	Maximum block size		
64 (40)		18	Reserved		
82 (52)	DCBLRECL	2	Logical record length or block size		
84 (54)	DCBCNTRL, DCBNOTE, DCBPOINT		Address of the CNTRL or the NOTE/POINT module		

(This page left blank intentionally)

Data Event Control Block

The data event control block (DECB) is created when a READ or WRITE macro instruction is expanded. It contains information about the input or output operation that is requested by the macro instruction.

The format for the data event control block is illustrated below; descriptions of the fields follow the illustration.

DECB

0 (0) **DECSDECB** Event Control Block	
4 (4) **DECTYPE** Reserved	6 (6) **DECLNGTH** Length of Data or of Key and Data
8 (8) **DECDCBAD** DCB Address	
12 (C) **DECAREA** Read/Write Area Address	
16 (10) **DECIOBPT** Reserved	

Offset		*Name*	*Bytes*	*Description*
0	(0)	DECSDECB	4	Event control block
4	(4)	DECTYPE	2	Reserved
6	(6)	DECLNGTH	2	Length of key and data (if there is a key); length of work area for an application program
8	(8)	DECDCBAD	4	Address of the DCB to which this I/O request is related
12	(C)	DECAREA	4	Address of the Read/Write area; address of work area for an application program
16	(10)	DECIOBPT	4	Reserved

(This page left blank intentionally)

Data Extent Block

The data extent block (DEB) is a fixed-length control block with a 36-byte prefix. The DEB describes the extents of the data set with which the DEB is associated. The DEB contains such addresses as the DCB, the UCB, and the TCB. The number of extents associated with the data set is also in the DEB. For line groups, the DEB contains the number of lines in a line group and with which line number the data set is used. For a message queue, the DEB contains the number of extents of the data set and their size. The data extent block prefix contains the addresses of the data set appendages (the PCI Appendage, the Channel End Appendage, and others).

The address of the DEBTCBAD field of the data extent block is in the DCBDEBAD field of the data control block. The address of the beginning of the DEB prefix is at a displacement of –36(–X'24') from the address of the DEBTCBAD field. Storage is allocated for the DEB and it is initialized at open time.

Note: *The displacements on this control block do not agree with the TDEBD macro, which has the relative zero displacement at DEBEOEA. The disk message queues routines use the TDEBD macro offsets. The AVTADEBN and AVTADEBR fields of the TCAM AVT contain the address of the DEBEOEA field of the DEB.*

The format of the DEB prefix and the data extent block itself is illustrated below; descriptions of the fields follow.

IEDQDEB

-36 (-24) **DEBEOEA** Address of the End-of-Extent Appendage	
-32 (-20) **DEBSIOA** Address of the Start I/O Appendage	
-28 (-1C) **DEBPCIA** Address of the PCI Appendage	
-24 (-18) **DEBCEA** Address of the Channel End Appendage	
-20 (-14) **DEBXCEA** Address of the Abnormal End Appendage	
-16 (-10) **DEBWKARA** I/O Support Work Area	-15 (-F) **DEBDSCBA** Address of the DSCB
-8 (-8) **DEBDCBMK** DCB Modification Mask	
-4 (-4) **DEBLNGTH** Length of the DEB in Double Words	
0 (0) **DEBNMSUB** Number of OPEN Subroutines	**DEBTCBAD** Address of the TCB
4 (4) **DEBAMLNG** Length of Access Method Section	**DEBDEBAD** Address of the Next DEB
8 (8) **DEBOFLGS** Data Set Flags	**DEBIRBAD** Address of the IRB
12 (C) **DEBOPATB** Type of I/O	**DEBSYSPG** Address of the First IOB in the System Purge Chain
16 (10) **DEBNMEXT** Number of Extents	**DEBUSRPG** Address of the First IOB in the User Purge Chain
20 (14) **DEBPRIOR** Zero	**DEBECBAD** Address of the Parameter List to Find the Purge ECB
24 (18) **DEBPROTG** Protection Key DEB ID	**DEBDCBAD** Address of the DCB
28 (1C) **DEBEXSCL** Extent Scale	**DEBAPPAD** Address of the I/O Appendage Vector Table
32 (20) **DEBDVMOD** Device Modifier	**DEBUCBAD** Address of the UCB

Offset		Name	Bytes	Description
-36	(-24)	DEBEOEA	4	Address of End-Of-Extent Appendage
-32	(-20)	DEBSIOA	4	Address of Start I/O Appendage
-28	(-1C)	DEBPCIA	4	Address of PCI Appendage
-24	(-18)	DEBCEA	4	Address of Channel End Appendage
-20	(-14)	DEBXCEA	4	Address of Abnormal and Normal Line End Appendage
-16	(-10)	DEBWKARA	1	I/O support work area
-15	(-F)	DEBDSCBA	7	Address of DSCB
-8	(-8)	DEBDCMK	4	DCB modification mask
-4	(-4)	DEBLNGTH	4	Length of the DEB in doublewords
0	(0)	DEBNMSUB	1	Number of OPEN subroutines
0	(0)	DEBTCBAD	4	Address of the TCB
4	(4)	DEBAMLNG		Length access method section
4	(4)	DEBDEBAD	4	Address of the next DEB
8	(8)	DEBOFLGS	1	Data set flags
8	(8)	DEBIRDAD	4	Address of the IRB
12	(C)	DEBOPATB	1	Type of I/O
12	(C)	DEBSYSPG	4	Address of the first IOB in the system purge chain
16	(10)	DEBNMEXT	1	Number of extents
16	(10)	DEBUSRPG	4	Address of the first IOB in the user purge chain
20	(14)	DEBPRIOR	1	Zero
20	(14)	DEBECBA	4	Address of the parameter list to find the purge ECB
24	(18)	DEBPROTG		Protection key DEB ID
24	(18)	DEBDCBAD	4	Address of the DCB
28	(1C)	DEBEXSCL	1	Extent scale
28	(1C)	DEBAPPAD	4	Address of the I/O Appendage vector table
32	(20)	DEBDVMOD	1	Device modifier
32	(20)	DEBUCBAD	4	Address of the UCB

(This page left blank intentionally)

Data Extent Block for Application Programs

There is a special application program data extent block (DEB) that has the same DSECT name, IEDQDEB, as the regular TCAM DEB. The format of this special DEB and descriptions of the fields follow.

IEDQDEB – Application Program

0 (0) **DEBTAMID** TCAM DEB Identifier	1 (1) **DEBTCBAD** Address of the TCB for this DEB
4 (4) Reserved	5 (5) **DEBDEBAD** Address of the Next DEB
8 (8) Reserved	9 (9) **DEBPCBAD** Address of the Process Control Block
12 (C) **DEBTAMOS** Process Entry Termname Table Offset	14 (E) **DEBSOWA** Size of Locate Mode Work Area
16 (10) **DEBTAMPP** Post Pending Flag Byte	17 (11) **DEBQCBAD** Address of Read-Ahead QCB
20 (14) Reserved	21 (15) **DEBTAMWA** Address of TCAM Access Method Work Area
24 (18) Reserved	25 (19) **DEBDCBAD** Address of the DCB for this DEB
28 (1C) Reserved	29 (1D) **DEBLCMWA** Address of Locate Mode Work Area
32 (20) **DEBEND** End of DEB **DEBSIZE** Size of DEB	

Offset		*Name*	*Bytes*	*Description*
0	(0)	DEBTAMID	1	TCAM DEB identifier; if bits 0 and 1 are on, this is a TCAM DEB
1	(1)	DEBTCBAD	3	Address of the TCB for this DEB
4	(4)		1	Reserved
5	(5)	DEBDEBAD	3	Address of the next DEB in the same task
8	(8)		1	Reserved
9	(9)	DEBPCBAD	3	Address of the process control block for this task
12	(C)	DEBTAMOS	2	Offset to the termname table entry for the corresponding process entry
14	(E)	DEBSOWA	2	Size of the locate mode work area
16	(10)	DEBTAMPP	1	Post-pending flag byte
17	(11)	DEBQCBAD	3	Address of the read-ahead QCB
20	(14)		1	Reserved
21	(15)	DEBTAMWA	3	Address of the TCAM access method work area
24	(18)		1	Reserved
25	(19)	DEBDCBAD	3	Address of the DCB for this DEB
28	(1C)		1	Reserved
29	(1D)	DEBLCMWA	3	Address of the locate mode work area
32	(20)	DEBEND	1	End of the DEB indicator
32	(20)	DEBSIZE	1	Size of the DEB in bytes

Device Characteristics Table

The device characteristics table (DCT) is a variable-length table that contains one four-byte entry for each type of terminal or station defined in the TCAM system. The DCT is generated by the specification of the TERMINAL macro instructions. At assembly time, each entry is allocated and initialized to describe the characteristics of the particular type of terminal or group of terminals; a single four-byte entry is generated for all terminals that have identical characteristics.

The address of the device characteristics table is assembled in the AVTCSTCS field of the address vector table. The one-byte index (TRMCHCIN) in a terminal entry in the terminal table provides the offset to the specific device characteristics table entry for a station.

Bits are set in the DCT entry to indicate the type of station. Combinations of these bit settings may be coded where applicable. The specific values for a DCT entry are outlined below.

Offset	*Name*	*Value*	*Description*
0 (0)			Reserved
1 (1)	CINHIBIT	X‘80’	Terminal can use Read Inhibit CCWs
	CBREAK	X‘40’	Terminal has the Reverse Break feature
	CATTEN	X‘20’	Terminal has the Attention feature
	C50 41	X‘10’	2741 and 1050 Interrupt Feature supported
	C2741	X‘08’	2741 on this line
	C3270	X‘04’	3270 device
	CSRDEU	X‘02’	Stand-alone device
	CUMASK	X‘01’	Control unit (2848 or 3270)
2 (2)	CBISYNC	X‘80’	BSC station
	CBRDCST	X‘40’	Terminal is for broadcast data
	CTWX	X‘20’	TWX 3335 terminal
	CSTNCTL	X‘10’	Terminal has the Station Control feature
	CXMITCTL	X‘08’	Terminal has the Transmit Control feature
	CCONTENT	X‘04’	Contention device
	CLOCAL	X‘02’	Local device
	CAUDIO	X‘01’	Audio device
3 (3)	CWTTA	X‘40’	World Trade Telegraph
	CENDCTL	X‘20’	Terminal has end-to-end controls (2780)
	CCHECK	X‘10’	Terminal has the Checking feature
	CCONTIN	X‘04’	Terminal is capable of a Continue operation
	CNOIDLES	X‘02’	Terminal has no idles defined (2260 Remote)
	C2760	X‘01’	2760

(This page left blank intentionally)

Disk Data Area

The disk record is composed of count, key, and data. The count field is set at disk initialization time. When a unit is used as a disk buffer, the data portion of the disk record comes from the first six bytes of the unit, and the key portion of the disk record (which contains the text of the message itself) comes from that portion of the unit following the 12-byte unit prefix. The disk data area is the first six bytes of the unit prefix. When the unit is a disk buffer or is going through the channel, the address of the disk data area is in the Read or Write Data CCW in the channel program block. The address of the disk data area is usually also in the CPBXREA field of the channel program block.

Storage is allocated for the disk data area at IEDQXA execution time. At that same time, the disk data area is initialized to zeros. The actual data in the disk data area is placed there either by Destination Scheduler (IEDQHM) or by Reusability–Copy (IGG019RP).

The first six bytes of the IEDQDATA DSECT defines the data portion of the disk record (the disk data area). The last two bytes of the DSECT are bytes 7 and 8 of the unit prefix and are used only in main storage (they are not written to disk and are, therefore, not part of the disk data area).

The format of the IEDQDATA DSECT is illustrated below; descriptions of the fields follow.

IEDQDATA

0 (0) **DATFLAGS** Flag Byte	1 (1) **DATFEFO** FEFO Pointer
4 (4) **DATCOUNT** Text Byte Count **DATSEQOT** Output Sequence Number	6 (6) **DATSCAN** Scan Pointer Save Area

Offset		*Name*	*Bytes*	*Description*		
0	(0)	DATFLAGS	1	Flag byte:		
			Name	*Bit*	*Value*	*Meaning*
			DATNPRFX	0	X‘80’	No prefix is in the record
				1 Off	X‘7F’	Mask to specify that a prefix is in the record
			DATSENT	1	X‘40’	Message has been serviced
				1 Off	X‘BF’	Mask to specify that the message has not been serviced
			DATCNCLD	2	X‘20’	Message is canceled
				2 Off	X‘DF’	Mask to specify that the message is not canceled
			DATLOSTN	3 Off	X‘EF’	Mask to specify that a message is lost from the main-storage queue
1	(1)	DATFEFO	3	FEFO pointer to the next message to be completely received for this destination		
4	(4)	DATCOUNT	2	For text records only, the number of bytes of significant text in this record key field, or zero if not the last text record		
4	(4)	DATSEQOT	2	For header records only, the sequence-out number		
6	(6)	DATSCAN	2	Saves the scan pointer (number of reserve characters remaining) while building a buffer from this unit; not used in a main-storage disk message queue data set and not part of the disk data area		

Element Request Block

The element request block (ERB) is a fixed-length table of fourteen bytes located at a displacement of X'4C' from the beginning of the line control block. TCAM uses the ERB to request buffers for transmissions of data. The beginning of the element request block is at a displacement of +44 (X'2C') from the beginning of the input/output block within the LCB. The address of the IOB is in the DCBIOBAD field of the data control block.

Storage is allocated for the element request block at open time. The ERB is initialized at various times depending upon its function. When it is being used to request buffers, the ERB may be initialized by the Send Scheduler, Receive Scheduler, Get Scheduler, or the Put Scheduler. When it is being used to get recalled buffers, the ERB may be initialized by Buffer Disposition, EOB Check, or the Buffered Terminal Scheduler.

When TCAM uses an element request block (ERB) to request buffers for a line, it tposts an ERB to the appropriate QCB to obtain filled buffers for a send operation or empty buffers for a receive operation. The QCB pointer refers to the queue control block to which the ERB is tposted. The link address points to the next element on the queue that contains the ERB. The status field indicates the status of the ERB (for example, that it has been tposted for a buffer, or that it is available, etc.). The chain field contains a pointer to the first buffer in a chain of buffers to be used in the operation. If the buffer unit pool is empty (all buffer units are in use), the ERB is placed in a chain of ERBs waiting for buffers and remains there until a buffer is returned and assigned to it. The two count fields indicate the number of buffers requested for an operation. Two fields are necessary because a disabled routine may need to increment the count and an enabled routine to decrement the count.

The format of the element request block and descriptions of the fields are included in the discussion of the line control block.

(This page left blank intentionally)

Invitation List

The INVLIST=(name of list,...) operand of a DCB macro specifies the names of the invitation lists for the lines of the line group represented by the DCB. There is one invitation list for each line in a line group, and the DCB contains a pointer to the control word of each of its invitation lists. An INVLIST macro specifies the actual entries in each invitation list.

An invitation list contains the invitation (polling) characters for terminals that may generate messages to the CPU on the same line. The order in which the invitation characters of the terminals are listed determines the order in which the terminals on the line are polled.

Invitation lists may contain both active and inactive entries. Active entries are those invited to enter a message on each pass through the list; an X'FE' follows the last active entry. An inactive entry is one that is not currently being invited to enter messages. Inactive entries in the list are located after the X'FE' indicator. The methods of establishing and altering the status of the entries in the invitation list are discussed in the section *Establishing Contact* in the *System/360 OS TCAM Programmer's Guide,* Order No. GC30-2024.

The general format of an invitation list is eight bytes of control information, followed by an invitation list entry for each active terminal on the line, followed by an end-of-list indicator (X'FE'), followed by an entry for each inactive terminal on the line.

An invitation list with *n* active entries has the following format:

-2n		-4	-2	0	+4	+8					
Reln	•••	Rel 2	Rel 1	Control Word	CPU ID	Invchars	1	Invchars	2	n	X 'FE'

Rel1-Reln are the two-byte relative positions in the termname table for the entries represented by the invitation characters. There is one two-byte field for each entry in the invitation list, in reverse order.

Control Word is a field defining the status of the invitation list. (See format below.)

CPU ID , for dial terminals, is the address of a field that contains the ID sequence assigned to the computer. The field referred to contains a length byte, which specifies the number of bytes in the ID sequence, followed by the ID sequence itself. For buffered terminals, the CPU ID field in an invitation list has the following format:

Offset	+4	+5	+6	+7
	Active Count	UCB Status	Reserved	Terminal Count

Active Count is the number of active terminals on the line to which TCAM is currently sending. This field is initialized to zero at line open time.

UCB Status is set to X'01' at line open time if the UCB for the line indicates Auto Poll. Otherwise, this field contains X'00'.

Terminal Count is the total number of terminals on this line. This field is initialized at line open time.

Invchars are the invitation or polling characters to be used for the terminal. The one-byte index following *Invchars* points to the corresponding relative position field that precedes the control word.

X'FE' is the end-of-list indicator, which is used to separate active and inactive entries. An EOT character precedes the X'FE' as an end-of-transmission character in an invitation list for BSC Auto Poll terminals.

The control word of an invitation list has the following format:

Offset	0	+1	+2	+3
	Total Entries	Active Entries	Width	Status

Total entries indicates the number of active and inactive entries in the list (if this byte is equal to zero, the list is for an output-only line; there is no message traffic from the terminals).

Active entries indicates the number of entries currently being invited. If byte 1 is equal to zero, all the entries in the list are inactive.

Width indicates the size of each entry in the list (the size includes the one-byte index that follows the invitation characters).

Status indicates whether the list is active or inactive and whether it is being autopolled.

Status bits	*Meaning*
0	ON—EOT= was specified on the INVLIST macro OFF—EOT= was not specified on the INVLIST macro
1	ON—Offsets to the termname table entries have been sorted OFF—Offsets to the termname table entries have not been sorted
2	Contention bit
3-4	Reserved
5	Indicates whether the list has been processed by Checkpoint/Restart (flip/flop)
6	ON—Active list OFF—Inactive list
7	ON—List is being autopolled OFF—Programmed poll is in effect

The invitation list entries have the same format whether the terminals are under control of the Auto Poll facility, the programmed poll facility, or some other scheme, such as contention. The width of each entry is indicated in byte 2 of the control word.

The format of each entry in an invitation list is:

Invitation Characters	K

The invitation characters (polling characters) are in the hexadecimal form of the transmission code. *K* is the one-byte index field used to indicate the relative position of the entry in the list and to find the two-byte pointer to the corresponding entry in the termname table.

(This page left blank intentionally)

Input/Output Control Block (IOBLOCKS)

The input/output control block is a map of the major control blocks used in I/O operations to the test device. It contains the VCB address, LCB address, and termname table address for the test device, the TECB address, the IOB, DCB, and ECB.

IOBLOCKS

0 (0)	**ECBOLT**	Event Control Block
4 (4)	**OLTUCBA**	Unit Control Block Address
6 (6)	**OLTTNOFF**	Terminal Name Table Entry Offset
8 (8)	**OLTTCRLN**	TCAM Relative Line No.
9 (9)	**OLTLCBA**	LCB Address
12 (C)	**OLTTNTA**	Terminal Name Table Entry Address
16 (10)	**OLTRLTNT**	Address of Real TNT Entry
20 (14)	**OLTEABLN**	Extended Area Length
22 (16)	**OLTFLG1**	Test Device Flag Byte
23 (17)		Reserved
24 (18)	**OLTDTBUF**	Data Blocking Field Response Buffer Address
28 (1C)	**OLTDTCNT**	Data Blocking Field Response Buffer Size
30 (1E)	**OLTFLAG2**	Data Blocking Flags
31 (1F)		
32 (20)		Reserved
36 (24)	**TECBADDR**	Address of Test Event Control Block
40 (28)	**IOBFLG1**	First Flag Byte
41 (29)	**IOBFLG2**	Second Flag Byte
42 (2A)	**IOBSNS**	First Two sense Byte
44 (2C)	**IOBECBAD**	ECB Address
44 (2C)	**IOBECBCD**	ECB Code
45 (2D)		
48 (30)	**IOBCSW**	Channel Status Word
48 (30)	**IOBFLG3**	Third Flag Byte
49 (31)	**IOBCSW1**	Last 7 Bytes of Last CSW

IOBLOCKS

56 (38)	IOBCCWAD	CCW Address
56 (38)	IOBSIOCD	Start I/O Code
57 (39)		Reserved
60 (3C)	IOBDCBAD	DCB Address
64 (40)	IOBREPM	Reposition Modifier
65 (41)	IOBRSTAD	Restart Address
68 (44)	IOBINCR	Block Count Increment
70 (46)	IOBERRCT	Error Counts
72 (48)	IOBUCBX	UCB Index
73 (49)	IOBWORK	Work Area
76 (4C)	IOBFLG4	TOTE and Appendage Flags
77 (4D)	IOBCSWS	CSW Save Area
84 (54)	ORG	
	DCBDCDEP	Device Dependent Field
104 (68)	DCBBUFNO	No. of Buffers in Data Set
105 (69)	DCBBUFCB	Buffer Pool Control BUFCB Address
108 (6C)	DCBBUFL	Buffer Length
110 (6E)	DCBDSORG	Data Set Organization
112 (70)	DCBIOBAD	I/O Block Address
116 (74)	DCBBGFEK	Buff. Techn., Alignm.
117 (75)	DCBEODAD	End of Data Set Routine
120 (78)	DCBRECFM	Record Format
121 (79)	DCBEXLST	Exit List
124 (7C)	DCBTIOT	DD Offset
126 (7E)	DCBMACRF	Macro Instruction Reference
128 (80)	DCBIFLGS	I/O Supervisor Flags
129 (81)	DCBDEBAD	Data Extent Block Address

IOBLOCKS

132 (84) **DCBOFLGS** Open Flags	133 (85) Reserved
136 (88) **DCBOPTCD** Option Codes	137 (89) Reserved
138 (90) **DCBEOEA** End of Extent Appendage	140 (92) **DCBPCIA** Program Controlled Interrupt Appendage
142 (94) **DCBSIOA** Start I/O Appendage	144 (96) **DCBCENDA** Channel End Appendage
146 (98) **DCBXENDA** Abnormal End Appendage	148 (9A) Reserved

Offset		*Name*	*Bytes*	*Description*		
0	(0)	ECBOLT	4	Event Control Block		
4	(4)	OLTUCBA	2	Unit Control Block address		
6	(6)	OLTTNOFF	2	Terminal Name Table Entry offset		
8	(8)	OLTTCRLN	1	TCAM Relative Line No.		
9	(9)	OLTLCBA	3	LCB address		
12	(C)	OLTTNTA	4	Terminal Name Table Entry address		
16	(10)	OLTRLTNT	4	Address of Real TNT Entry		
20	(14)	OLTEABLN	2	Extended area length		
22	(16)	OLTFLG1	1	Test Device Flag Byte		
			Name	*Bits*	*Value*	*Meaning*
			OLTTNTAS	0	X'80'	TOTE TNT Entry assigned
23	(17)		1	Reserved		
24	(18)	OLTDTBUF	4	Data Blocking Field Response Buff address		
28	(1C)	OLTDTCNT	2	Data Blocking Field Response Buffer Size		
30	(1E)	OLTFLAG2	1	Data Blocking Flags		
			Name	*Bits*	*Value*	*Meaning*
			OLTPTIMD	0	X'80'	Post WAITIO immediately
			OLTMVDAT	1	X'40'	Move response to buffer
			OLTLV3IO	2	X'20'	Last EXIO to device was level 3
			OLTSIOAC	3	X'10'	EXIO to device is outstanding
31	(1F)		5	Reserved		
36	(24)	TECBADDR	4	Address of Test Event Control Block		
40	(28)	IOBFLG1	1	First Flag Byte		
41	(29)	IOBFLG2	1	Second Flag Byte		
42	(2A)	IOBSENS	2	First two sense bytes		
44	(2C)	IOBECBAD		ECB address		
44	(2C)	ICBECBCD	1	ECB code		
45	(2D)		3			
48	(30)	IOBCSW		Channel Status Word		
48	(30)	IOBFLG3	1	Third Flag Byte		
49	(31)	IOBCSW1	7	Last 7 bytes of last CSW		
56	(38)	IOBCCWAD		CCW address		
56	(38)	IOBSIOCD	1	Start I/O Code		
57	(39)		3			
60	(3C)	IOBDCBAD	4	DCB address		
64	(40)	IOBREPM	1	Reposition Modifier		

Offset		*Name*	*Bytes*	*Description*		
65	(41)	IOBRSTAD	3	Restart address		
68	(44)	IOBINCR	2	Block Count Increment		
70	(46)	IOBERRCT	2	Error Counts		
72	(48)	IOBUCBX	1	UCB Index		
73	(49)	IOBWORK	3	Work area		
76	(4C)	IOBFLG4	1	Flags for TOTE and its appendix		
			Name	*Bits*	*Value*	*Meaning*
			IOBATTN	0	X'80'	Attention Interrupt expected
			IOBCSWV	1	X'40'	CSW Save Area Valid
			IOBSEC	2	X'20'	Secondary IOB
			IOBPPI	3	X'10'	Primary IOB
			IOBATNE	4	X'08'	Error on CE/DE before ATTN
			IOBCSWNV	5	X'04'	CSW Area 2 invalid
77	(4D)	IOBCSWS	7	CSW Save Area		
84	(54)	ORG	44			
84	(54)	DCB				
84	(54)	DCBDCDEP	20	Device dependent field		
104	(68)	DCBBUFNO	1	Number of buffers in data set		
105	(69)	DCBBUFCB	3	Buffer Pool Control Block address		
108	(6C)	DCBBUFL	2	Buffer lentgh		
110	(6E)	DCBDSORG	2	Data set organization		
112	(70)	DCBIOBAD	4	I/O Block address		
116	(74)	DCBGFEK	1	Buffer technique, alignment		
117	(75)	DCBEODAD	3	End of data set routine		
120	(78)	DCBRECFM	1	Record format		
121	(79)	DCBEXLST	3	Exit List		
124	(7C)	DCBTIOT	2	DD offset		
126	(7E)	DCBMACRF	2	Macro instruction reference		
128	(80)	DCBIFLGS	1	I/O supervisor flags		
129	(81)	DCBDEBAD	3	Data extent block address		
132	(84)	DCBOFLGS	1	Open flags		
133	(85)		3	Reserved		
136	(88)	DCBOPTCD	1	Option codes		
137	(89)		7	Reserved		

Offset		Name	Bytes	Description
138	(90)	DCBEOEA	2	End of extent appendage
140	(92)	DCBPCIA	2	Program controlled interrupt appendage
142	(94)	DCBSIOA	2	Start I/O appendage
144	(96)	DCBCENDA	2	Channel end appendage
146	(98)	DCBXENDA	2	Abnormal end appendage
148	(9A)		2	Reserved

(This page left blank.intentionally)

Line Control Block

The line control block (IEDQLCB) is a fixed-length table that contains the information that must be maintained on a line or line group basis. There is one line control block for each line. The LCB maintains such information as pointers to the channel program, the corresponding DCB, the last serviced PCI, and the chain of waiting QCBs. The LCB also contains the buffer chain, the subtask chain, and the I/O status. When the LCB is functioning as a QCB, the line control block contains the address of the first STCB. Within the LCB, at a displacement of 76 (X‘4C’), is the element request block. (For further information on the ERB see the discussion of the element request block.) The I/O block is also in the LCB at a displacement of X‘20’ from the beginning.

To find the address of a specific LCB for a line group from a DCB, the TCAM modules first multiply the relative line number for this line times the value in DCBEIOBX and add the result to the value in DCBIOBAD. The result is the address of the IOB for this LCB. The LCB begins at –X‘20’ from the IOB address.

Storage is allocated and the line control block is initialized at open time for the DCB for the line group.

The format of the line control block is illustrated below; descriptions of the fields follow.

IEDQLCB

0 (0) **LCBKEY** Element Key of Buffer **LCBRCB** Resource Control Block	1 (1) **LCBQCBA** Address of the QCB		
4 (4) **LCBPRI** Priority of Buffer	5 (5) **LCBLINK** Link Field of Buffer		
8 (8) **LCBRSKEY** Receive Scheduler Key	9 (9) **LCBSTCBA** Address of the First STCB When LCB is a QCB		
12 (C) **LCBRSPRI** Receive Scheduler Priority	13 (D) **LCBRSLNK** Address of the Next Item in the Chain		
16 (10) **LCBEOLTD** End of List Time Delay		18 (12) **LCBTDL** Time Delay Queue Offset	19 (13) **LCBTSOB** TSO Status Bits
20 (14) **LCBCHAIN** Disposition Status Bits	21 (15) **LCBINSRC** In-source Chain		
24 (18) **LCBNTXT** Save Area for PRFNTXT	25 (19) **LCBSCBDA** Address of the SCB Directory **LCBLNENT** TNT Offset to Line Entry		
28 (1C) **LCBISZE** Count of Idles Reserved	29 (1D) **LCBFSBFR** First Buffer Assigned to this LCB **LCBLSBFR** Last Buffer Assigned to this LCB		
32 (20) **LCBFLAG1** IOS Flags 1	33 (21) **LCBFLAG2** IOS Flags 2	34 (22) **LCBSENS0** Sense Byte 0	35 (23) **LCBSENS1** Sense Byte 1
36 (24) **LCBECBCC** Completion Code	37 (25) **LCBECBPT** Address of the ECB		
40 (28) **LCBFLAG3** IOS Flags 3	41 (29) **LCBCSW** Last Channel Status Word		
48 (30) **LCBSIOCC** SIO Condition Code	49 (31) **LCBSTART** Address of the Channel Program		
52 (34) **LCBDCBPT** Address of the DCB			
56 (38) **LCBRESTR** Error Message Data **LCBRCQCB** QCB to Which to Post the Recalled Buffer			
60 (3C) **LCBINCAM** IOS		**LCBTTBIN** Index to Terminal to be Connected **LCBERRCT** IOS Error Counters	

64 (40)
LCBUCBX
UCB Index

65 (41)
LCBRCBFR
Pointer to the Recalled Buffer

LCBLSPCI
Address of the Last Serviced PCI

68 (44)
LCBRECOF
Offset to the Current Block

70 (46)
LCBSTATE
Status Bits

LCBSTAT1
First Status Byte

71 (47) **LCBSTAT2**
Second Status Byte

72 (48)
LCBTSTSW
Test-and-Set Switch

73 (49)
LCBRECAD
Address of the Current Message Block

76 (4C) **LCBERBKY**
ERB Key
LCBERB
Element Request Block

77 (4D)
LCBERBQB
ERB QCB

80 (50)
LCBERBPY
ERB Priority

81 (51)
LCBERBLK
Address of the Next Item in the Chain

84 (54)
LCBERBST
Status of ERB

85 (55)
LCBERBCH
Address of the Chain to be Assigned Buffers

88 (58)
LCBERBCT
Count Fields

90 (5A)
LCBTTCIN
Index to the Terminal Currently Connected

92 (5C)
LCBMSGFM
Bits to Control BSC Line

93 (5D)
LCBSCBA
Address of the Current SCB

96 (60)
LCBERMSK
Error Recording Mask

97 (61)
LCBINVPT
Address of the Current Entry in the Invitation List

100 (64)
LCBTPCD
TP OP Codes

112 (70)
LCBSNSV
Save Area for Sense Byte

113 (71)
LCBCSWSV
Save Area for Channel Status Word

120 (78)
LCBERCCW
3 ERP Commands

141 (8D)
LCBSTICS
Characteristics Work Area

144 (90)
LCBSTICS
(Cont.)
LCBCPA
Channel Program Area

IEDQLCBX

0 (0)
LCBXFLAG
Device Flags

1 (1)
LCBXDCT
Device Characterstics Table Storage Area

4 (4)
LCBXRADR
ERP Polling Characters Address

Offset		*Name*	*Bytes*	*Description*		
0	(0)	LCBRCB	8	Resource control block for this LCB		
0	(0)	LCBKEY	1	Key field of the RCB		
1	(1)	LCBQCBA	3	QCB address		
4	(4)	LCBPRI	1	Priority of the RCB		
5	(5)	LCBLINK	3	Address of the next element in the chain in which this RCB is currently located		
8	(8)	LCBRSKEY	1	Receive Scheduler key field		
9	(9)	LCBSTCBA	3	Address of the first STCB when the LCB is functioning as a QCB		
12	(C)	LCBRSPRI	1	Receive Scheduler priority field		
13	(D)	LCBRSLNK	3	Address of the next item in the chain in which this STCB currently resides		
16	(10)	LCBEOLTD	2	End of the invitation list time delay interval		
18	(12)	LCBTDL	1	Time delay queue offset to QCB address for LCB = X‘14’		
19	(13)	LCBTSOB	1	TSO status byte:		
			Name	*Bit*	*Value*	*Meaning*
			LCBPREP	0	X‘80’	Prepare on line
			LCBWRBRK	0	X‘80’	Write break in progress
			LCBTSBUF	1	X‘40’	Buffer has TSO prefix
			LCBSATRD	2	X‘20’	Simulated attention request
			LCBSOPL	3	X‘10’	Start of polling list
			LCBREAD	4	X‘08’	Reading partial line
			LCBCIRCD	5	X‘04’	Circle D sent to 2741
			LCBINHBN	6	X‘02’	Use inhibits for this terminal
			LCB2741N	7	X‘01’	2741 on 2741/1050 line
20	(14)	LCBCHAIN	1	Disposition status byte:		
			Name	*Bit*	*Value*	*Meaning*
			LCBSCRNN	0	X‘80’	Screen change requested
			LCBSCRNF	0 Off	X‘7F’	Mask to specify no screen change requested
			LCBEXCP	1	X‘40’	Delay EXCP until association
			LCBERMSG	2	X‘20’	ERP message waiting
			LCBNORTY	3	X‘10’	Text retry not possible
			LCBUREQN	4	X‘08’	Unit request in progress
			LCBUREQF	4 Off	X‘F7’	Mask to specify that a unit request is not in progress
			LCBBFRSZ	5	X‘04’	Queue management flag
			LCBTETEN	6	X‘02’	User requested tete-a-tete
			LCBTETEF	6 Off	X‘FD’	Mask to specify that tete-a-tete is not requested

Offset		*Name*	*Bytes*	*Description*		
			LCBABRTN	7	X‘01’	Abort sequence must be sent
			LCBABRTF	7 Off	X‘FE’	Mask to specify that an abort sequence is not required
21	(15)	LCBINSRC	3	In-source chain		
24	(18)	LCBNTXT	1	Temporary save area for PRFNTXT		
25	(19)	LCBSCBDA	3	Address of the SCB directory		
26	(1A)	LCBLNENT	1	Termname table offset to the line entry reserved		
28	(1C)	LCBISZE	1	Count of idles (reserve characters)		
29	(1D)	LCBFSBFR	3	First buffer assigned to this LCB		
29	(1D)	LCBLSBFR	3	Last buffer assigned to this LCB		
32	(20)	LCBFLAG1	1	IOS flags 1		
33	(21)	LCBFLAG2	1	IOS flags 2		
34	(22)	LCBSENS0	1	Sense byte 0		
35	(23)	LCBSENS1	1	Sense byte 1		
36	(24)	LCBECBCC	1	Completion code		
37	(25)	LCBECBPT	3	ECB address		
40	(28)	LCBFLAG3	1	IOS flags 3		
			Name	*Bit*	*Value*	*Meaning*
			LCBOBRRD	2	X‘02’	TP error record processing
			LCBSOHC	4	X‘08’	SOH% C message
			LCBSOHR	6	X‘20’	SOH% R message
41	(29)	LCBCSW	7	Last CSW		
48	(30)	LCBSIOCC	1	SIO condition code		
49	(31)	LCBSTART	3	Address of the channel program		
52	(34)	LCBDCBPT	4	Address of the corresponding DCB		
56	(38)	LCBRESTR		Start of error message data		
56	(38)	LCBRCQCB	4	Address of the QCB to which recalled buffers are to be tposted		
60	(3C)	LCBINCAM	2	IOS		
62	(3E)	LCBTTBIN	2	Index of the terminal to be connected		
62	(3E)	LCBERRCT	2	IOS error counters		
64	(40)	LCBUCBX	1	UCB index		
65	(41)	LCBRCBFR	3	Pointer to a recalled buffer		
65	(41)	LCBLSPCI	3	Address of the last serviced PCI		
68	(44)	LCBTRST	2	Offset to the start of the Buffer Translation routine		
70	(46)	LCBSTATE	2	Status bits		
70	(46)	LCBSTAT1	1	First status byte; bit definitions are:		

Offset	Name	Bytes	Description		
		Name	*Bit*	*Value*	*Meaning*
		LCBRCLLN	0	X'80'	Recall being performed
		LCBRCLLF	0 Off	X'7F'	Mask to specify that no recall is being performed
		LCBCTLMD	1	X'40'	Line is in control mode
		LCBCVRSP	1	X'40'	First BSC output conversational block
		LCBOCNI	2	X'20'	Non-immediate operator control operation is in progress
		LCBINITN	3	X'10'	Receiving initiate mode message
		LCBINITF	3 Off	X'EF'	Mask to specify no initiate mode message
		LCBCONT	4	X'08'	Continue or reset operation in progress
		LCBFREEN	5	X'04'	Line is free
		LCBFREEF	5 Off	X'FB'	Mask to specify that the line is not free
		LCBRECBN	6	X'02'	Line is receiving
		LCBSENDN	7	X'01'	Line is sending (Line is stopped if bits 5,6, & 7 are off.)
71 (47)	LCBSTAT2	1	Second status byte; bit settings are:		
		Name	*Bit*	*Value*	*Meaning*
		LCBTRACE	0	X'80'	I/O trace active for this line
		LCBLOCK	0	X'80'	Line is in lock mode
		LCBTRCOF	8 Off	X'7F'	Mask to specify that I/O trace is not active for this line
		LCBMSGNN	1	X'40'	MSGGEN or start-up message
		LCBMSGNF	1 Off	X'BF'	Mask to specify that this is not a MSGGEN or start-up message
		LCBBEOTN	2	X'20'	EOT from a buffered terminal (no EOM)
		LCBBEOTF	2 Off	X'DF'	Mask to specify a regular EOM if EOT from a buffered terminal
		LCBSNDPR	3	X'10'	Send priority switch set by the Send Scheduler
		LCBNEGRP	4	X'08'	Negative response to polling
		LCBSYNC	5	X'04'	Line is binary synchronous
		LCBDIAL	6	X'02'	This is a dial LCB
		LCBRESP	7	X'01'	A response needs to be sent to this line
72 (48)	LCBTSTSW	1	Test-and-set switch:		
		Name	*Bit*	*Value*	*Meaning*
		LCBCONCT	0	X'80'	Connection established
73 (49)	LCBRECAD	3	Address of the current message block		
76 (4C)	LCBERB	4	Start of the ERB for this LCB		

Offset	Name	Bytes	Description
76 (4C)	LCBERBKY	1	Element request block key field
77 (4D)	LCBERBQB	3	Address of the QCB to which this ERB is currently tposted
80 (50)	LCBERBPY	1	ERB priority
81 (51)	LCBERBLK	3	Address of the next item in the chain in which this ERB currently resides
84 (54)	LCBERBST	1	ERB status; bit settings are:

Name	Bit	Value	Meaning
LCBMSG	0	X'80'	End of initiate mode
LCBEOMSG	1	X'40'	End of message read from disk
LCBRDERR	2	X'20'	Logical read error
LCBRDERF	2 Off	X'DF'	Mask to specify no read error
LCBINQ	3	X'10'	ERB is waiting for buffers from IEDQHM
LCBERROR	5	X'04'	Error on the send side
LCBPRCPG	6	X'02'	After the initial request is satisfied, tpost the ERB to the QCB specified in LCBRCQCB
LCBCOMPL	6	X'02'	Disk request is complete
LCBDLNKN	7	X'01'	Delink switch—ERB is not tposted, but is eligible to be tposted
LCBDLNKF	7 Off	X'FE'	Mask to specify that the ERB is tposted, so PCI cannot tpost it again

Offset	Name	Bytes	Description
85 (55)	LCBERBCH	3	Address of the chain to be assigned buffers
88 (58)	LCBERBCT	2	Count fields
90 (5A)	LCBTTCIN	2	Index to the terminal that is currently connected
92 (5C)	LCBMSGFM	1	Bits to control the BSC line

Name	Bit	Value	Meaning
LCBNAK	0	X'80'	Request to send a NAK response
LCBACK1	1	X'40'	ACK counter

The following bits indicate whether a scan of line control has been accomplished and the type of line control received.

LCBVSTRT	2	X'20'	Valid start sequence
LCBRSTRT	3	X'10'	Error start sequence
LCBTTD	4	X'08'	Temporary time delay received
LCBENQ	5	X'04'	ENQ received
LCBEOT	6	X'02'	EOT first character
LCBOLT	7	X'01'	Address of the current SCB

93 (5D)	LCBSCBA	3	Address of the current SCB
96 (60)	LCBERMSK	1	Error recording mask
97 (61)	LCBINVPT	3	Address of the current entry in the invitation list

Offset		Name	Bytes	Description
100	(64)	LCBTPCD	12	TP operation codes
112	(70)	LCBSNSV	1	Save area for the sense byte
113	(71)	LCBCSWSV	7	Save area for the CSW
120	(78)	LCBERCCW	24	Three ERP commands
141	(8D)	LCBSTICS	3	Characteristics work area
144	(90)	LCBCPA	8	Channel program area
The following is the LCB extension:				
0	(0)	LCBXFLAG	1	Device flags
1	(1)	LCBXDCT	3	Device characteristics table storage area
4	(4)	LCBXRADR	4	ERP polling characters address

On-Line Test Control Block

The major control block used by the Teleprocessing On-Line Executive is the On-Line Test Control Block (OLTCB), which contains the buffers, pointers, flags, parameter lists, and data fields that must be preserved after the modules that set them up have been deleted. The OLTCB also contains control fields and queue pointers to allow the TOTE parent task to communicate with and control the On-Line Test subtasks.

Modules IEDQWA and IEDQWB have an eight-byte extension at the beginning of the OLTCB. This extension contains the forward and backward pointers for the OLTCB queue.

The format of the on-line test control block is illustrated in the following layout. Descriptions of the fields follow the layout. The offsets represented are for all modules except IEDQWA and IEDQWB, whose offsets are eight greater.

Eight-byte extension for Modules IEDQWA and IEDQWB:

TOTOBPTR			
TOTELKEY	DS	C	OLTCB queue element key
TOTELQCB	DS	AL3	Address of OLTCB QCB
	DS	C	Unused
TOTELLNK	DS	AL3	OLTCB queue element link field

TOTOLTCB

0 (0) **$ERRLPCT** Loop on Error Count		2 (2) **$TESTOPT** Test Option Field	3 (3) **$ERROPT** Error and Option Field
4 (4) **$RT0108** Routine Masks 1-8	5 (5) **$RT0916** Routine Masks 9-16	6 (6) **$DRIVER** Driver Identification	7 (7) **$SPARE 1** Unused
8 (8) **$TSSSYM** Reserved for TSS		10 (A) **$PDEVFLG** Primary Device Flags	11 (B) **$CDSFLGS** Device CDS Flags
12 (C) **$PDEVADR** Primary Device Address			
16 (10) **$PDEVDSC** Primary Device Descriptors			
20 (14) **$CDS8T19** Primary Device CDS Bytes 8-19			
32 (20) **$RMSKCNT** Routine Mask Count Length	33 (21) **$EXECFLG** Executive Program Flags	34 (22) **$OLTSIZE** OLT Region Size	
36 (24) **$OLTFLGS** OLT Functional Flags	37 (25) Unused	38 (26) **$TOTFLG1** TOTE 1st Flag Byte	39 (27) **$TOTFLG2** TOTE 2nd Flag Byte
40 (28) **$R017024** Routine Mask 17-24	41 (29) **$R025032** Mask 25-32	42 (2A) **$R033040**	43 (2B) **$R041048**
44 (2C) **$R049056**	45 (2D) **$R057064**	46 (2E) **$R065072**	47 (2F) **$R073080**
48 (30) **$R081088**	49 (31) **$R089096**	50 (32) **$R097104**	51 (33) **$R105112**
52 (34) **$R113120**	53 (35) **$R121128**	54 (36) **$R129136**	55 (37) **$R137144**
56 (38) **$R145152**	57 (39) **$R153160**	58 (3A) **$R161168**	59 (3B) **$R169176**
60 (3C) **$R177184**	61 (3D) **$R185192**	62 (3E) **$R193200**	63 (3F) **$R201208**
64 (40) **$R209216**	65 (41) **$R217224**	66 (42) **$R225232**	67 (43) **$R233240**
68 (44) **$R241248**	69 (45) **$R249255**	70 (46) **$RETMASK**	71 (47) Spare

TOTOLTCB

72 (48)	Reserved
80 (50)	**$TABLE** Address of Branch Table
84 (54)	**$PASS** Address of Passon Area
88 (58)	**$ETX** Address of External Data
92 (5C)	SCT Expansion Area
116 (74)	**TOTSMGRT** Service Manager Return Save
120 (78)	**TOTSAVE1** OLT Subtask 1st Save Area
192 (C0)	**TOTSAVE2** OLT Subtask 2nd Save Area
264 (108)	**TOTSAVE3** OLT Subtask 3rd Save Area
336 (150)	**TOTSAVE4** OLT Subtask 4th Save Area
408 (198)	**TOTSAVE5** OLT Subtask 5th Save Area
480 (1E0)	**TOTSVEND** End of TOTE Save Areas
488 (1E8)	**TOTLNKPL** Service Mgr Link Parameter List
488 (1E8)	**TOTLNKNM** Link Name
496 (1F0)	**TOTBKASN** OLT Main Storage Blocks
498 (1F2)	**TOTBKRQD** OLT Main-Storage Blocks Required
500 (1F4)	**TOTMMSPC** OLT Unused Main Storage
504 (1F8)	**TOTCTENT** Control Terminal Entry

TOTOLTCB

504(1F8) **TOTCTUCB** C. T. UCB Address		506(1FA) **TOTCTOFF** Offset to C. T. TNT Entry	
508(1FC) **TOTCTRLN** C. T. Relative Line No.	509(1FD) **TOTCTLCB** C. T. LCB Address		
512(200) **TOTCTTNT** C. T. Terminal Name Table Entry Address (Dummy Entry)			
516(204) **TOTCRTNT** C. T. Terminal Name Table Entry Address (Real Entry)			
520(208) **TOTCTTLN** Length of C. T. TTE Area		522(20A) **TOTCTFLG** Control Terminal Flags	523(20B)
524(20C) **TOTCTNAM** Control Terminal Name in EBCDIC			
532(214) **TOTCTDFL** Control Terminal Initial Conditions			
536(218) **TOTCUTST** Control Information For $CUTEST Macro			
536(218) **TOTCUFLG** $CUTEST Flags	537(219) **TOTCU#AD** $CUTEST No. of Contig. Addrs.	538(21A) **TOTCUCUU** $CUTEST Starting Address	
540(21C) **TOTCUSAV** $CUTEST Save Area			
544(220) **TOTCURS1** $CUTEST Resv. Byte 1 Code Parm.	545(221) **TOTCURS2** Byte 2	546(222) **TOTCURS3** Byte 3	547(223) **TOTCURS4** Byte 4
548(224)			
644(284) **TOTAPENT** Alternate Printer Entry **TOTAPUCB** A. P. UCB Address		646(286) **TOTAPOFF** Offset to A. P. TNT Entry	
648(288) **TOTAPRLN** A. P. Relative Line No.	649(289) **TOTAPLCB** Application Program LCB Address		
652(28C) **TOTAPTNT** App. Prog. Terminal Name Table Entry Address (Dummy Entry)			
656(290) **TOTARTNT** App. Prog. Terminal Name Table Entry Address (Real Entry)			
660(294) **TOTAPTLN** Length of App. Prog. TTE Area		662(296) **TOTAPFLG** Printer Flags	663(297) **TOTAPDFL** Alt. Ptr. Init. Cond..
664(298) **TOTAPDCB** Local Printer DECB			

TOTOLTCB

Offset	Field	Description
752(2F0)	**TOTPDECB**	Local Printer DECB
772 (304)		Unused
776 (308)	**TOTOLTMQ**	Subtask Message Queue
780 (30C)	**TOTOTECB**	TOTE Subtask ECB
784 (310)	**TOTRESSV**	Pointer to Mother Task Save Area
788 (314)	**TOTTCBAD**	Subtask TCB Address
792 (318)	**TOTCMPCD**	Subtask Completion Code
796 (31C)	**TOTOLTPL**	OLT Input Parameter List
804(324)	**TOT#TBLE**	Branch Table Address
808 (328)	**TOTWTORP**	WTOR Parm List for Operator Communication
808 (328)	**TOTINCNT**	Reply Byte Count
809(329)	**TOTINADR**	Reply Buffer Address
812 (32C)	**TOTINECB**	Reply ECB Address
816 (330)	**TOTWTOPL**	WTO & WTOR PL for Operator Communication
816 (330)	**TOTWTOPL**	WTO & WTOR Parm. List
817 (331)	**TOTOTCNT**	Out Message Count
818 (332)		Reserved
820 (334)	**TOTOTBUF**	Output Area
904(388)	**TOTINBUF**	Reply Buffer
984 (3D8)	**TOTOLTID**	TOTE OLT Identification
986 (3DA)	**TOTRTCOD**	Service Module Return Code
987 (3DB)		Unused
988 (3DC)		Unused

TOTOLTCB

992 (3E0) **TOTRQRLN** Request Relative Line No.	993 (3E1) **TOTRQUCB** Address of Request Line UCB		
996 (3E4) **TOTPLNKQ** Plink Module Queue			
996 (3E4) **TOTPLFWD** Queue Forward Pointer			
1000 (3E8) **TOTPLBKW** Queue Backward Pointer			
1008 (3F0) **TOTWKSPC** TOTE Work Area			
1136 (470) **TOTWKEND** End of TOTE Work Area			
1136 (470) Unused			
1156 (484) **TOTAVTPT** Address of AVT			
1160 (488) **TOTRESPL** Address of TOTE Resident Parameter List			
1164 (48C) **TOTFLG01** TOTE 1st Flag Byte	1165 (48D) **TOTFLG02**	1166 (48E) **TOTFLG03**	1167 (48F) **TOTFLG04**
1168 (490) **TOTFLG05**	1169 (491) **TOTFLG06**	1170 (492) **TOTFLG07**	1171 (493) **TOTFLG08**
1172 (494) **TOTFLG09**	1173 (495) **TOTFLG10**	1174 (496) **TOTTTBEL** Terminal Name Table Entry	1175 (497) Unused
1176 (498) **TOTEXT** External Data Buffer			
1232 (4D0) **TOTPASS** Pass-on Date Buffer			
1296 (510) **TOTTRMBF** TRM Buffer for TRM Analysis			
1296 (510) **TOTPRIBK** Primary Test Device I/O Control Blocks			
1296 (510) **TOTPRECB** Primary ECB			

TOTOLTCB

1300 (514)	**TOTPRENT**	TOTE Primary Test Device Entry
1300 (514)	**TOTPRUCB**	Primary Device UCB Address
1302 (516)	**TOTPROFF**	Offset to Primary TNT Entry
1304 (518)	**TOTPRRLN**	Primary Device Relative Line No.
1305 (519)	**TOTPRLCB**	Primary LCB Address
1308 (51C)	**TOTPRTNT**	Primary Terminal Name Table Entry Address (Dummy Entry)
1312 (520)	**TOTTNTPR**	Primary Terminal Name Table Entry Address (Real Entry)
1316 (524)	**TOTPRTLN**	Length of Primary TTE Area
1318 (526)	**TOTPRFLG**	Primary Flags
1319 (527)		Unused
1320 (528)	**TOTPDTBF**	Response Buffer Address
1324 (52C)	**TOTPDTCT**	Response Buffer Size
1326 (52E)	**TOTPFLGS**	Flags
1327 (52F)		Unused
1328 (530)		Unused
1332 (534)	**TOTPTECB**	Primary TECB Address
1336 (538)	**TOTPRIOB**	Primary IOB
1380 (564)	**TOTPRDCB**	Primary DCB
1424 (590)	**TOTPRDEB**	DEB Address
1452 (5AC)	**TOTSCIBK**	Secondary Test Device I/O Control Blocks
1452 (5AC)	**TOTSCECB**	Secondary ECB
1456 (5B0)	**TOTSCENT**	TOTE Secondary Test Device Entry
1456 (5B0)	**TOTSCUCB**	Seconday Device UCB Address
1458 (5B2)	**TOTSCOFF**	Offset to Secondary TNT Entry

TOTOLTCB

1460 (5B4)	TOTSCRLN	Secondary Device Relat. Line No.
1461 (5B5)	TOTSCLCB	Secondary LCB Address
1464 (5B8)	TOTSCITNT	Secondary Terminal Name Table Entry Address (Dummy Entry)
1468 (5BC)	TOTSRITNT	Secondary Terminal Name Table Entry Address (Real Entry)
1472 (5C0)	TOTSCTLN	Length of Secondary TTE Area
1474 (5C2)	TOTSCFLG	Secondary Flags
1475 (5C3)		Unused
1476 (5C4)	TOTSDTBF	Response Buffer Address
1480 (5C8)	TOTSDTCT	Response Buffer Size
1482 (5CA)	TOTSFLGS	Flags
1483(5CB)		Unused
1484(5CC)		Unused
1488 (5D0)	TOTSTECB	Secondary TECB Address
1492 (5D4)	TOTSCIOB	Secondary IOB
1536 (600)	TOTSCDCB	Secondary DCB Address
1580(62C)	TOTSCDEB	DEB Address
1608 (648)	TOTTRMND	End of TRM Buffer
1608(648)	TOTCROLT	Current OLT I. D.
1616 (650)	TOTOLTTB	OLT I. D. Table
1646(66E)	TOTOLTTE	End of Table
1696 (6A0)	TOTTDTBL	TOTE Test Device Table

TOTOLTCB

1732(6C4)	**'TOTLETGO** Last Entry Pointer	
1736 (6C8)	**TOTTDEND** End of Table	
1736(6C8)	**TOTWECBA** Pointer to ECB for Test Device	
1740 (6CC)	**TOTECBPT** Pointer to Subtask ECB	
1744 (6D0)	**TOTTLCNT** Test Loop Count	
1752 (6D8)	**TOTWAITI** WAITIO Time Interval	
1760 (6E0)	**TOTGRABP** Current Secondary Device Pointer	
1764 (6E4)	**TOTAPNAM** Alternate Printer Name	
1772 (6EC)	**TOTOLTEN** Entry Point Address of OLT	
1776(6F0)	**TOTOBEND** End of OLTCB	

Offset		*Name*	*Bytes*	*Description*		
0	(0)	$ERRLPCT	3	Loop on error count		
2	(2)	$TESTOPT	1	Test option field		
			Name	*Bits*	*Value*	*Meaning*
			$NOPRT	5	X'04'	No print option flag
			$INDEFLP	6	X'02'	Indefinite error loop flag
			$FSTCOMM	7	X'01'	First error communication flag
3	(3)	$ERROPT	1	Error and option field		
4	(4)	$RT0108	1	Routine mask 1—8		
5	(5)	$RT0916	1	Routine mask 9—16		
6	(6)	$DRIVER	1	Driver identification		
7	(7)	$SPARE1	1	Unused		
8	(8)	$TSSSYM	2	Reserved for TSS		
10	(A)	$PDEVFLG	1	Primary device flags		
			Name	*Bits*	*Value*	*Meaning*
			$FPMOLD	0	X'80'	File protect flag
			$EXFILPT	1	X'40'	Additional file protect flag
			$TSSSYS1	2	X'20'	Reserved for TSS
			$LASTDEV	3	X'10'	Last device of subsystem flag
				4	X'08'	Spare
			$TERMNDX	5	X'04'	Reserved for TSS
			$PATHDEF	6	X'02'	Reserved for TSS
			$LASTSUB	7	X'01'	Last device of last subsystem flag
11	(B)	$CDSFLGS	1	Device flags from CDS flag byte		
			Name	*Bits*	*Value*	*Meaning*
			$FPM	0	X'80'	File protect flag
			$SHARED	1	X'40'	Shared device flag
			$CEVOL	2	X'20'	CE volume flag
			$EXTINTC	3	X'10'	Device address associated with external flag
			$SYMNAME	4	X'08'	Symbolic name flag
			$TWOCHSW	5	X'04'	Two channel switch flag
			$CUSTSYM	6	X'02'	Customer assigned symbolic name flag
			$COMMCN		X'01'	Line connection required flag
12	(C)	$PDEVADR	4	Primary device address		
16	(10)	$PDEVDSC	4	Primary device descriptors		
20	(14)	$CDS8T19	12	Primary device CDS bytes 8—19		
32	(20)	$RMSKCNT	4	Routine mask count length		

Offset		*Name*	*Bytes*	*Description*		
33	(21)	$EXECFLG	4	Executive program flags		
			Name	*Bits*	*Value*	*Meaning*
			$CECCMIN	0	X'80'	Reply in to outstanding CECOM flag
			$MULTDEV	1	X'40'	More than one device entry flag
			$RTNSLCT	2	X'20'	Operator mode routine selection flag
			$ERRCNT	3	X'10'	Operator specified error loop count
			$LSTPDEV	4	X'08'	Do not assign more primary devices flag
			$CLEANUP	5	X'04'	Section entered for cleanup flag
			$CTRLMOD	6	X'02'	Control mode available flag
			$QSCTMOD	7	X'01'	Quiescent mode available flag
34	(22)	$OLTSIZE	2	OLT region size		
36	(24)	$OLTFLGS	1	OLT functional flags		
			Name	*Bits*	*Value*	*Meaning*
			$MANINTV	0	X'80'	Manual intervention routine flag
			$CLEANRT	1	X'40'	OLT has cleanup routine flag
			$RETAIN	2	X'20'	RETAIN is active flag
			$CONTCB	3	X'10'	Contingent connection broken flag
			$RETCODE	4	X'08'	
			$TRACE	5	X'04'	
			$LASTSEC	6	X'02'	Last section scheduled flag
37	(25)		1	Unused		
38	(26)	$TOTFLG1	1	TOTE 1st flag byte		
			Name	*Bits*	*Value*	*Meaning*
			$LINESHR	0	X'80'	Line can be shared flag
			$TDATBLK	1	X'40'	TOTE message clocking flag
39	(27)	$TOTFLG2	1	TOTE 2nd flag byte		
40	(28)	$R017024	1	Routine mask 17-24		
41	(29)	$R025032	1	Routine mask 25-32		
42	(2A)	$R033040	1	Routine mask 33-40		
43	(2B)	$R041048	1	Routine mask 41-48		
44	(2C)	$R049056	1	Routine mask 49-56		
45	(2D)	$R057064	1	Routine mask 57-64		
46	(2E)	$R065072	1	Routine mask 65-72		
47	(2F)	$R073080	1	Routine mask 73-80		
48	(30)	$R081088	1	Routine mask 81-88		
49	(31)	$R089096	1	Routine mask 89-96		
50	(32)	$R097104	1	Routine mask 97-104		
51	(33)	$R105112	1	Routine mask 105-112		

Offset		*Name*	*Bytes*	*Description*
52	(34)	$R113120	1	Routine mask 113-120
53	(35)	$R121128	1	Routine mask 121-128
54	(36)	$R129136	1	Routine mask 129-136
55	(37)	$R137144	1	Routine mask 137-144
56	(38)	$R145152	1	Routine mask 145-152
57	(39)	$R153160	1	Routine mask 153-160
58	(3A)	$R161168	1	Routine mask 161-168
59	(3B)	$R169176	1	Routine mask 169-176
60	(3C)	$R177184	1	Routine mask 177-184
61	(3d)	$R185192	1	Routine mask 185-192
62	(3E)	$R193200	1	Routine mask 193-200
63	(3F)	$R201208	1	Routine mask 201-208
64	(40)	$R209216	1	Routine mask 209-216
65	(41)	$R217224	1	Routine mask 217-224
66	(42)	$R225232	1	Routine mask 225-232
67	(43)	$R233240	1	Routine mask 233-240
68	(44)	$R241248	1	Routine mask 241-248
69	(45)	$R249255	1	Routine mask 249-255
70	(46)	$RETMASK	1	Return code mask
71	(47)		1	Spare
72	(48)		8	Spare
80	(50)	$TABLE	4	Address of branch table
84	(54)	$PASS	4	Address of pass-on area
88	(58)	$EXT	4	Address of external data
92	(5C)		24	SCT expansion area
116	(74)	TOTSMGRT	4	Service manager return save area
120	(78)	TOTSAV1	72	OLT subtask 1st save area
192	(CD)	TOTSAV2	72	OLT subtask 2nd save area
264	(108)	TOTSAV3	72	OLT subtask 3rd save area
336	(150)	TOTSAV4	72	OLT subtask 4th save area
408	(198)	TOTSAV5	72	OLT subtask 5th save area
480	(1EO)	TOTSVEND		End of TOTE save area
480	(1EO)	TOTLNKPL	6	Service manager link parameter list
480	(1EO)	TOTLNKNM	8	Link name
496	(1F0)	TOTBKASN	2	Main-storage blocks assigned to this OLT
498	(1F2)	TOTBKRQD	2	Main-storage blocks required by this OLT
500	(1F4)	TOTMMSPC	4	OLT unused main-storage
504	(1F8)	TOTCTENT		Control terminal entry

Offset	Name	Bytes	Description
504 (1F8)	TOTCTUCB	2	C.T. UCB address
506 (1FA)	TOTCTOFF	2	Offset to C.T. TNT entry
508 (1FC)	TOTCTRLN	1	C.T. relative line no.
509 (1FD)	TOTCTLCB	3	C.T. LCB address
512 (200)	TOTCTTNT	4	C.T. terminal name table entry address (dummy entry)
516 (204)	TOTCRTNT	4	C.T. terminal name table entry address (real entry)
520 (208)	TOTCTTLN	2	Length of CT TTE area
522 (20A)	TOTCTFLG	1	Control terminal flags
523 (20B)	TOTCTNAM	9	Control terminal name in EBCDIC
532 (214)	TOTCTDFL	4	Control terminal initial condition

Name	Bits	Value	Meaning
TOTCTCRM	0	X'80'	Terminal device flag
TOTCTLIN	1	X'40'	Line address flag
TOTCTGP	3	X'10'	General Poll initially active on test line
TOTCTLST	4	X'08'	CT initially stopped
TOTCTHD	5	X'04'	CT initially held
TOTCTIAC	6	X'02'	CT invitation list entry initially active

Offset	Name	Bytes	Description
536 (218)	TOTCUTST		Control information for CUTEST macro
536 (218)	TOTCUFLG	1	CUTEST flags

Name	Bits	Value	Meaning
TOTCUDON	0	X'80'	CU test issued
TOTCUCUP	1	X'40'	CU test clean up required

Offset	Name	Bytes	Description
537 (219)	TOTCU#AD	1	CUTEST number of contiguous address
538 (21A)	TOTCUCUU	2	CUTEST starting address
540 (21C)	TOTCUSAV	4	CUTEST save area
544 (220)	TOTCURS1	1	CUTEST reserved byte 1 code parameter
545 (221)	TOTCURS2	1	CUTEST reserved byte 2 code parameter
546 (222)	TOTCURS3	1	CUTEST reserved byte 3 code parameter
547 (223)	TOTCURS4	1	CUTEST reserved byte 4 code parameter
548 (224)		95	
644 (284)	TOTAPENT		Alternate printer entry
644 (284)	TOTAPUCB	2	A.P. UCB address
646 (286)	TOTAPOFF	2	Offset to A.P. TNT entry
648 (288)	TOTAPRLN	1	A.P. relative line no.
649 (289)	TOTAPLCB	3	A.P. LCB address
652 (28C)	TOTAPINT	4	A.P. terminal name table entry address (dummy entry)
656 (290)	TOTARINT	4	A.P. terminal name table entry address (real entry)
660 (294)	TOTAPTLN	2	Length of AP TTE area

Offset	*Name*	*Bytes*	*Description*		
662 (296)	TOTAPFLG	1	Printer flags		
663 (297)	TOTAPDFL	1	Alternate printer initial condition		
		Name	*Bits*	*Value*	*Meaning*
		TOTAPTER	0	X'80'	Terminal device flag
		TOTAPLIN	1	X'40'	Line address flag
		TOTAPGP	3	X'10'	General Poll initially active on test line
		TOTAPLST	4	X'08'	AP initially stopped
		TOTAPHD	5	X'04'	AP initially held
		TOTAPIAC	6	X'02'	AP invitation list entry initially inactive
664 (298)	TOTAPDCB	88	Local printer DCB		
752 (2F0)	TOTPDECB	20	Local printer DECB		
772 (304)		4	Unused		
776 (308)	TOTOLTMQ	4	Subtask message queue		
780 (30C)	TOTOTECB	4	TOTE subtask ECB		
784 (310)	TOTRESSV	4	Pointer to mother task save area		
788 (314)	TOTTCBAD	4	Subtask TCB address		
792 (318)	TOTCMPCD	4	Subtask completion code		
796 (31C)	TOTOLTPL	12	OLT input parameter list		
796 (31C)	TOT#TBLE	8	Branch table address		
804 (328)	TOTWTORP		WTOR parameter list for operator communication		
808 (328)	TOTINCNT	1	Reply byte count		
809 (329)	TOTINADR	3	Reply buffer address		
812 (32C)	TOTINECB	4	Reply ECB address		
816 (330)	TOTWTOPL		WTO and WTOR parameter list for operator communication		
816 (330)		1			
817 (331)	TOTOTCNT	1	Out message count		
818 (332)		2			
820 (334)	TOTOTBUF	84	Output area		
904 (388)	TOTINBUF	80	Reply buffer		
984 (3D8)	TOTOLTID	2	TOTE OLT identification		
986 (3DA)	TOTRTCOD	1	Service module return code		
987 (3DB)		1	Unused		
988 (3DC)		4	Unused		
992 (3E0)	TOTRQRLN	1	Request line relative line no.		
993 (3E1)	TOTRQUCB	3	Address of request line UCB		
996 (3E4)	TOTPLNKQ		PLINK module queue		
996 (3E4)	TOTPLFWD	4	Queue forward pointer		
1000 (3E8)	TOTPLBKW	4	Queue backward pointer		

Offset		Name	Bytes	Description		
1008	(3F0)	TOTWKSPC	128	TOTE work area		
1136	(470)	TOTWKEND		End of TOTE work area		
1136	(470)		20	Unused		
1156	(484)	TOTAVTPT	4	Address of AVT		
1160	(488)	TOTRESPL	4	Address of TOTE resident parameter list		
1164	(48C)	TOTFLG01	1	TOTE 1st flag byte		
			Name	*Bits*	*Value*	*Meaning*
			TOTTRREC	0	X‘80’	
			TOTPRSTP	2	X‘20’	Primary test device stopped flag
			TOTSCSTP	3	X‘10’	Secondary test device stopped flag
			TOTRQSTP	4	X‘08’	Requested terminal stopped flag
			TOTPRTAS	6	X‘02’	Primary device TNT entry assigned
			TOTSCTAS	7	X‘01’	Secondary device TNT entry assigned
1165	(48D)	TOTFLG02	1	TOTE 2nd flag byte		
			Name	*Bits*	*Value*	*Meaning*
			TOTPTSRT	0	X‘80’	Primary line start request flag
			TOTSCSRT	1	X‘40’	Secondary line start request flag
			TOTCHKSZ	4	X‘08’	Update subtask storage allocation flag
1166	(48E)	TOTFLG03	1	TOTE 3rd flag byte		
			Name	*Bits*	*Value*	*Meaning*
			TOTAPOER	0	X‘80’	Alternate printer option error flag
			TOTTDFER	1	X‘40’	Test device field error flag
			TOTTSTER	2	X‘20"	Test ID field error flag
			TOTOPTER	3	X‘10’	Optional field error flag
			TOTIMNCP	4	X‘08’	Temporary no count flag
			TOTTSINC	5	X‘04’	Inclusive test ID entry flag
			TOTTCREP	7	X‘01’	Get EXIO response by TCAM flag
1167	(48F)	TOTFLG04	1	TOTE 4th flag byte		
			Name	*Bits*	*Value*	*Meaning*
			TOTNUMDV	0	X‘80’	Numeric TRM entry flag
			TOTDTCHD	1	X‘40’	Subtask already detached flag
			TOTCLOSE	2	X‘20’	TCAM closedown progress flag
			TOTCNFUP	3	X‘10’	Configuration update flag
			TOTCTSWT	4	X‘08’	Control terminal line switched flag
			TOTRTSWT	5	X‘04’	Request terminal line switched flag
			TOTCNCLR	6	X‘02’	Cancel request message flag
			TOTTTSWT	7	X‘01’	Test terminal line switched flag

Offset	*Name*	*Bytes*	*Description*		
1168 (490)	TOTFLG05	1	TOTE 5th flag byte		
		Name	*Bits*	*Value*	*Meaning*
		TOTPRINT	0	X'80'	Access method print flag
		TOTCECOM	1	X'40'	Access method CECOM flag
		TOTREPLY	2	X'20'	Access method CECOM with reply flag
		TOTNTAVL	3	X'10'	Function not available flag
		TOTDEVST	4	X'08'	Start test devices flag
		TOTTMOUT	5	X'04'	Wait time-out flag
		TOTEXIOF	6	X'02'	Access method EXIO flag
		TOTNDMSG	7	X'01'	Send cancelor terminate message flag
1169 (491)	TOTFLG06	1	TOTE 6th flag byte		
		Name	*Bits*	*Value*	*Meaning*
		TOTOTERM	0	X'80'	Section terminate flag
		TOTCBOPN	1	X'40'	Cancel before open
		TOTNCMFG	3	X'10'	Non-current mode flag
		TOTABEND	4	X'08'	OLT has abended flag
		TOTCANCL	5	X'04'	Cancel testing flag
		TOTOTACT	6	X'02'	OLT active flag
		TOTNPERR	7	X'01'	No permanent error flag
1170 (492)	TOTFLG07	1	TOTE 7th flag byte		
		Name	*Bits*	*Value*	*Meaning*
		TOTPRIEX	0	X'80'	Primary device for EXIO/WAITIO flag
		TOTSECEX	1	X'40'	Secondary device for EXIO/WAITIO flag
		TOTMSGMV	2	X'20'	TOTE message already moved
		TOTMSCEC	4	X'08'	TOTE message source flag CECOM
		TOTMSREP	5	X'04'	TOTE message source flag CECOM
		TOTMSPRT	6	X'02'	TOTE message source flag CECOM
1171 (493)	TOTFLG08	1	TOTE 8th flag byte		
			Bits		*Meaning*
			0-3		DPRINT forms control flag
			4-7		DPRINT level control flag
1172 (494)	TOTFLG09	1	TOTE 9th flag byte		
		Name	*Bits*	*Value*	*Meaning*
		TOTAPCON	0	X'80'	Printer = system console flag
		TOTAPOUT	1	X'40'	Printer = system printer flag
		TOTCTCON	2	X'20'	Control terminal = system console flag
		TOTAPTRM	3	X'10'	Printer = terminal flag
		TOTPRENB	4	X'08'	Primary test device enabled flag

Offset		Name	Bytes	Description

TOTSCENB	5	X'04'	Secondary test device enabled flag
TOTPRECT	6	X'02'	No alternate printer flag
TOTMACFT	7	X'01'	Unsupport macro function flags

1173	(495)	TOTFLG10	1	TOTE 10th flag byte

Name	*Bits*	*Value*	*Meaning*
TOTTERMS	0	X'80'	Test devices = terminal flag
TOTFSUCB	1	X'40'	First test device in TRM flag
TOTINCLT	2	X'20'	Inclusive entry flag
TOTTTDEB	3	X'10'	Free test DEB on terminate flag
TOTTNTRQ	4	X'08'	Dummy TNT entry request flag
TOTOLTED	5	X'04'	On-line test ended flag
TOTOLTWT	6	X'02'	On-line test waiting flag
TOTOLTRS	7	X'01'	On-line test restart flag

1174	(496)	TOTTTBEL	1	Terminal name table entry
1175	(497)		1	Unused
1176	(498)	TOTEXT	56	External data buffer
1232	(4DO)	TOTPASS	64	Passon data buffer
1296	(510)	TOTTRMBF		TRM buffer for TRM analysis
1296	(510)	TOTPRIBK		Primary test device I/O control blocks
1296	(510)	TOTPRECB	4	Primary ECB
1300	(514)	TOTPRENT		TOTE primary test device entry
1300	(514)	TOTPRUCB	2	Primary device UCB address
1302	(516)	TOTPROFF	2	Offset to primary TNT entry
1304	(518)	TOTPRRLN	1	Primary device relative line no.
1305	(519)	TOTPRLCB	3	Primary LCB address
1308	(51C)	TOTPRTNT	4	Primary terminal name table entry address (dummy entry)
1312	(520)	TOTTNTPR	4	Primary terminal name table entry address (real entry)
1316	(524)	TOTPRTLN	2	Length of primary TTE address
1318	(526)	TOTPRFLG	1	Primary flags
1319	(527)		1	Unused
1320	(528)	TOTPDTBF	4	Response buffer address
1324	(52C)	TOTPDTCT	2	Response buffer size
1326	(52E)	TOTPFLGS	1	Flags
1327	(52F)		5	Unused
1332	(534)	TOTPTECB	4	Primary TECB address
1336	(538)	TOTPRIOB	44	Primary IOB
1380	(564)	TOTPRDCB	72	Primary DCB
1380	(564)	TOTPRDEB	44	DEB address

Offset	*Name*	*Bytes*	*Description*
1452 (5AC)	TOTSCIBK		Secondary test device I/O control blocks
1452 (5AC)	TOTSCECB	4	Secondary ECB
1456 (5B0)	TOTSCENT		TOTE secondary test device entry
1456 (5B0)	TOTSCUCB	2	Secondary device UCB address
1458 (5B2)	TOTSCOFF	2	Offset to secondary TNT entry
1460 (5B4)	TOTSCRLN	1	Secondary device relative line no.
1461 (5B5)	TOTSCLCB	3	Secondary LCB address
1464 (5B8)	TOTSCTNT	4	Secondary terminal name table entry address (dummy entry)
1468 (5BC)	TOTSRTNT	4	Secondary terminal name table entry address (real entry)
1472 (5C0)	TOTSCTLN	2	Length of secondary TTE address
1474 (5C2)	TOTSCFLG	1	Secondary flags
1475 (5C3		1	Unused
1476 (5C4)	TOTSDTBF	4	Response buffer address
1480 (5C8)	TOTSDTCT	2	Response buffer size
1482 (5CA)	TOTSFLGS	1	Flags
1483 (5CB)		5	Unused
1488 (5D0)	TOTSTECB	4	Secondary TECB address
1492 (5D4)	TOTSCIOB	44	Secondary IOB
1536 (600)	TOTSCDCB	72	Secondary DCB address
1536 (600)	TOTSCDEB	44	DEB address
1608 (648)	TOTTRMND		End of TRM buffer
1608 (648)	TOTCROLT	8	Current OLT I.D.

Name	*Bits*	*Value*	*Meaning*
TOTFTTIM	0	X'80'	
TOTCLRFG	0-7	X'00'	

Offset	*Name*	*Bytes*	*Description*
1616 (650)	TOTOLTTB	30	OLT I.D. table
1646 (66E)	TOTOLTTE		End of table
1646 (66E)		50	Unused
1696 (5A0)	TOTTDTBL	40	TOTE test device table
1696 (6A0)	TOTLETGO	36	Last entry pointer
1736 (6C8)	TOTTDEND		End of table

Name	*Bits*	*Value*	*Meaning*
TOTTDTRM	0	X'80'	Terminal device flag
TOTTDLIN	1	X'40'	Line addfess flag
TOTTDGRB	2	X'20'	Test device grabbed flag
TOTTDSGP	3	X'10'	General poll initially active on test line
TOTTDLST	4	X'08'	Test device initially stopped

Offset	*Name*	*Bytes*	*Description*		
		TOTTDTHD	5	X'04'	Test device initially held
		TOTTDIAC	6	X'02'	Test device invitation list entry initially inactive
		TOTTDLGO		X'01'	Test device let go flag
1736 (6C8)	TOTWECBA	4	Pointer to ECB for test device		
1740 (6CC)	TOTECBPT	4	Pointer ro subtask ECB		
1744 (6D0)	TOTTLCNT	2	Test loop count		
1752 (6D6)	TOTWAITI	8	WAITIO time interval		
1760 (6E0)	TOTGRABP	4	Current secondary device pointer		
1764 (6E4)	TOTAPNAM	8	Alternate printer name		
1772 (6EC)	TOTOLTEN	4	Entry point address of OLT		

(This page left blank intentionally)

Operator Control Address Vector Table

The operator control address vector table (IEDQOPCD) is a fixed-length table that serves as a general work area for the use of operator control. The table is assembled at the end of the Resident Operator Control module. This table is used by the Resident Operator Control module, by the operator control processing modules, and by the checkpoint/restart modules. The table is never referred to unless an operator control command is entered. Once such a command is entered, the operator control address vector table (AVT) contains entry points for modules, two save areas, bit switches, pointers, and a checkpoint element.

The address of the operator control AVT is the AVTOCGET field of the address vector table.

Because the operator control AVT is an attached module, storage is allocated for the table at the time of execution of the INTRO macro. The table is initialized at assembly time.

The Operator Control control module work area is a table of approximately 540 bytes that is attached to the end of the operator control AVT at a displacement of X'D8'. This area is *not* discussed below.

The format of the operator control AVT is illustrated; descriptions of the fields follow.

IEDQOPCD

Offset	Field	Description
0 (0)	**OPCDOUBL**	Doubleword Work Area
8 (8)	**OPCAVTPT**	Address of TCAM AVT
12 (C)	**OPCCOPCE**	Address of Current Element (OPCE)
16 (10)	**OPCDCBLK**	Address of DCB, RLN; Address of Lookup Routine
20 (14)	**OPCGTBLD**	GETMAIN and BLDL List Areas
24 (18)	**OPCTOFLK**	Address to get TNT, Offset, TERM Entry
28 (1C)		Reserved
32 (20)	**OPCLCB**	Address of LCB Setup Routine
36 (24)	**OPCTRMWA**	Work Area for Term Entry Address
40 (28)	**OPCSAVE**	Operator Control Save Area
68 (44)	**OPCQCBAD**	Address of Operator Control QCB
72 (48)	**OPCRSAVE**	Base and Return Save Area
80 (50)	**OPCWORK**	Address of Resident Work Area
84 (54)	**OPCWRKSZ**	Size of Resident Work Area
86 (56)	**OPCFLAG1**	Flag Byte
87 (57)	**OPCFLAG2**	Flag Byte
88 (58)	**OPCBFERB**	Buffer Request ERB
104 (68)	**OPCCKERB**	Checkpoint Request ERB
120 (78)	**OPCAQCTL**	SVC 102 Parameter List
132 (84)	**OPCWAIT**	Operator Control Input Wait List

144 (90) **OPCXCTL** List Form of XCTL			
152 (98) **OPCLDNME** XCTL Module Name			
160 (A0) **OPCWAITC** QCW for Checkpoint Wait Queue			
164 (A4) **OPCWAITL** QCW for LCB Wait Queue			
168 (A8) **OPCWAITN** QCW for BCH Response Wait Queue			
172 (AC) **OPCWAITO** Output Queue QCW			
176 (B0) **OPCWAITR** QCW for Resource Wait Queue			
180 (B4) **OPCNEXT** Address of Current Element			
184 (B8) **OPCGETBF** Address of Buffer Request Routine			
188 (BC) **OPCFREBF** Address of Buffer Unit Free Routine			
192 (C0) **OPCBFREQ** Units Need by Wait List		196 (C2) **OPCCHA8** Halfword Set to Eight	
196 (C4) **OPCASBUF** Units Assigned to OP CTL		198 (C6) **OPCAVBUF** Units in OP CTL Free Pool	
200 (C8) **OPCBFIRS** Address of First Unit in Free Pool			
204 (CC) **OPCBFEND** Current Unit End			
208 (D0) **OPCSPEC** Flag	209 (D1) **OPCOQSW** Switch	210 (D2) **OPCHNEND** Test for End of Buffer Unit	
212 (D4) **OPCHNEND** (cont.)		214 (D6) **OPCSTCBS** Switch	215 (D7) **OPCEND** Test Byte

Offset		*Name*	*Bytes*	*Description*
0	(0)	OPCDOUBL	8	Doubleword work space for checking across units
8	(8)	OPCAVTPT	4	Address of TCAM AVT
12	(C)	OPCCOPCE	4	Address of Current Element
16	(10)	OPCDCBLK	4	Address of DCB, RLN, and address of Lookup routine
20	(14)	OPCGTBLD	4	GETMAIN and BLDL list areas
24	(18)	OPCTOFLK	4	Address to get TNT offset and terminal entry
28	(1C)		4	Reserved
32	(20)	OPCLCB	4	Address of LCB Setup routine
36	(24)	OPCTRMWA	4	Work area for picking up TNT address
40	(28)	OPCSAVE	28	Operator control register save area
68	(44)	OPCQCBAD	4	Address of operator control QCB
72	(48)	OPCRSAVE	8	Base and return save area (IGC0010D)
80	(50)	OPCWORK	4	Address of resident work area
84	(54)	OPCWRKSZ	2	Size of resident work area
86	(56)	OPCFLAG1	1	Flag byte for transient use
87	(57)	OPCFLAG2	1	Flag byte for transient use
88	(58)	OPCBFERB	16	Buffer request ERB
104	(68)	OPCCKERB	16	Checkpoint request ERB
120	(78)	OPCAQCTL	12	SVC 102 parameter list
132	(84)	OPCWAIT	12	Operator control input wait list
144	(90)	OPCXCTL	8	List form of XCTL macro used by transient routines
152	(98)	OPCLDNME	8	Module name for XCTL macro
160	(A0)	OPCWAITC	4	Queue control word (QCW) for checkpoint wait queue
164	(A4)	OPCWAITL	4	QCW for LCB wait queue
168	(A8)	OPCWAITN	4	QCW for branch response wait queue
172	(AC)	OPCWAITO	4	Ouptut queue QCW
176	(B0)	OPCWAITR	4	QCW for resource wait queue
180	(B4)	OPCNEXT	4	Address of current element
184	(B8)	OPCGETBF	4	Address of buffer request routine
188	(BC)	OPCFREBF	4	Address of buffer unit free routine
192	(C0)	OPCBFREQ	2	Number of units needed by wait list
194	(C2)	OPCHA8	2	Halfword set to eight
196	(C4)	OPCASBUF	2	Number of units assigned to operator control
198	(C6)	OPCAVBUF	2	Number of units in operator control freepool
200	(C8)	OPCBFIRS	4	Address of first unit in freepool
204	(CC)	OPCBFEND	4	Current end of unit (IGC0110D)
208	(D0)	OPCSPEC	1	Flags used by IEDQCA

Offset	*Name*	*Bytes*	*Description*		
		Name	*Bit*	*Value*	*Meaning*
		OPCALTD	0	X‘80’	Alternate destination specified
		OPCPART	1	X‘40’	Partial unit requested
		OPCRSTRT	2	X‘20’	Restart in progress
209 (D1)	OPCOQSW	1	Output busy switch (FF)		
210 (D2)	OPCHNEND	4	Test under mask instruction executed for to find end of buffer unit		
214 (D6)	OPCSTCBS	1	STCB busy switch (FF)		
215 (D7)	OPEND	1	Test byte to detect end of unit		

(This page left blank intentionally)

Option Characteristics Table

The option characteristics table (IEDQOPTN) is a variable-length table that contains one entry for each OPTION macro issued in the Message Control Program (MCP). The relative position of an entry in the table directly corresponds to the relative position of an option offset in a terminal table entry. The option offset is an index to the actual option table data for the option entry in the option characteristics table. The option characteristics table allows TCAM routines to use the assembled name for an OPTION macro to locate the option table data for a specific station (terminal).

Each entry in the option characteristics table contains the length of the corresponding option table entry, the type of option field specified, and the user-specified name of the OPTION macro. The length of the table is variable and consists of ten bytes for each OPTION macro issued plus one byte (X'FF') to indicate the end of the table. Storage is allocated and the table is initialized at assembly time. The AVT field AVTOPTPT contains the address of the option table, and the second word of the option table contains the address of the option characteristics table.

The format of an entry in the option characteristics table is illustrated below; descriptions of the fields follow.

Offset	0	+1	+2	9
	Length	Type	Name	

Offset		*Name*	*Bytes*	*Description*	
0	(0)	length	1	The length of the corresponding option table entry minus one, which is equal to the number of bytes of data specified by the TPROCESS and TERMINAL macros plus any necessary alignment bytes	
1	(1)	type	1	The type of option field, indicated by one of the following bit configurations:	
			Hex Code	*Type of Constant*	*Machine Format*
			00 C	Character	8-bit code for each character
			01 Z	Decimal	Zoned decimal format
			40 P	Decimal	Packed decimal format
			81 D	Floating-Point	Long floating-point format; usually a doubleword
			80 E	Floating-Point	Short floating-point format; usually a fullword
			D0 Q	Address	Space reserved for a dummy section offset
			C8 V	Address	Space reserved for external symbol addresses; each address usually a fullword
			C4 S	Address	Base register and displacement value; a halfword
			C2 Y	Address	Value of address; usually a halfword
			C1 A	Address	Value of address; usually a fullword
			F0 F	Fixed-Point	Signed, fixed-point binary format; usually a halfword
			E6 H	Fixed-Point	Signed, fixed-point binary format; usually a halfword
			E4 X	Hexadecimal	4-bit code for each hexadecimal digit
			E2 B	Binary	Binary format
2	(2)	name	8	The name of the option field—this is the actual name the user codes in the name field of the OPTION macro	

Option Table

The option table (IEDQOPT) is a variable-length table that contains the actual data coded by the user in the TERMINAL and TPROCESS macros in the message control program. At assembly time, this data is placed in the table with the the necessary byte alignment in the order in which it is coded. An option data field, which is not directly identifiable by the macro in which it is coded, can be referred to only through the option offset fields of a terminal entry. If only the user-coded name for a macro is known, TCAM uses the option characteristics table and the terminal entry to refer to a specific data field in the option table. (See the discussion of the option characteristics table in this section.)

The user may specify an area to correspond to any entry in the terminal table for use by the COUNTER, ERRORMSG, FORWARD, MSGLIMIT, INSERT, PATH, REDIRECT, STARTMH, and other MH delimiter macro instructions issued in a message handler. The fields are defined by OPTION macros, which must be issued before the TERMINAL and TPROCESS macros that define the terminal table. One-byte offsets to these fields are placed in the terminal entry beginning at the TRMOPT label. The routine for the LOCOPT macro uses these offsets to locate the option field.

An OPTION macro defines each field in the option table. The macro names the option field and defines the type and length of the field. The OPTION macro generates a CSECT to contain the actual option data and another CSECT to contain the field name and characteristics.

Initial values for the option fields are specified by parameters of the TERMINAL or TPROCESS macros.

Each option field requires one OPTION macro. The order of the fields within the option table is determined by the order in which the OPTION macro instructions are specified. The first option field is generated on a doubleword boundary. The maximum size of the option fields for a given terminal is 254 bytes plus the length of the last entry, including required boundary alignment.

For each OPTION specified, space for a one-byte offset is reserved in the offsets field of the terminal table entry. When the TERMINAL or TPROCESS macro that initializes the fields of the option table is issued, a two-byte offset to the option table for this entry is generated. If initial data is supplied, the option field is generated for the terminal or process entry; if a comma is coded, the option field is not generated. If the field is generated, its offset is placed in the offset field of the terminal entry; if the field is not generated, the offset field contains X'FF' to indicate that there is no field. The X'FF' is generated only if defined option fields follow this field.

Each single, group, or process entry in the terminal table contains a one-byte offset in the offset field for each OPTION macro issued. The space needed for the option table depends on the number of fields initialized by the TERMINAL or TPROCESS macros, and on the size of the fields as specified by the OPTION macros.

All OPTION names are kept in a table with their numeric values. This table enables an option field named in an operator control message to be located.

At assembly time the address of the option table is placed in the AVTOPTPT field of the AVT. The first two words of the option table contain the address of the end of the option table and the address of the option characteristics table, respectively. The option data immediately follows these two words. The general format of the option table is illustrated below; descriptions of the fields follow.

IEDQOPT

0 (0)	Address of the End of the Option Table
4 (4)	Address of the Option Characteristics Table
8 (8)	Option Data
7 + n	Length

Offset	*Name*	*Bytes*	*Description*
0 (0)		4	The address of the first byte (7+n) following the option table
4 (4)		4	The address of the first byte of the option characteristics table (IEDQOPTN)
8	Option data	n	The actual data coded by the user with the necessary byte alignment, in the order in which the data is coded
7+n	length	1	The length of the option data for the terminal or process entry that has the longest option data

OS I/O Device Characteristics Table

The OS I/O device characteristics table is a variable-length table that contains one 12-byte entry for each direct access device in the system. The table contains such information as the number of cylinders, the number of tracks per cylinder, the overhead for each intermediate record on the track, and the tolerance factor for each intermediate record. The OS I/O device characteristics table is used by the Checkpoint Disk Allocation routine (IGG01949) to obtain data about the specific direct access device used for the checkpoint data set. The table is also used by the Disk Message Queue Open—Load 1 routine (IGG01930) to determine the number of tracks per cylinder for the current data set being opened (to determine whether the device is a 2311 or a 2314).

The address of the OS I/O device characteristics table is in the CVTZDTAB field of the CVT. The unit control block contains the index to the specific entry in the table.

Storage is allocated for the OS I/O device characteristics table and it is initialized at OS IPL time.

The format of one entry in the OS I/O device characteristics table is illustrated below; descriptions of the fields follow.

IEDQDCTD

<table>
<tr><td>0 (0)
Reserved</td><td>1 (1)
DCTCYL
Cylinder Count</td><td colspan="2">2 (2)
DCTRACKS
Number of Tracks Per Cylinder</td></tr>
<tr><td colspan="2">4 (4)
DCTBYTE
Number of Bytes Per Track</td><td>6 (6)
DCTINTRO
Overhead</td><td>7 (7)
DCTLASTO
Overhead</td></tr>
<tr><td>8 (8)
DCTKEY
Overhead</td><td>9 (9)
Reserved</td><td colspan="2">10 (A)
DCTOLERN
Tolerance Factor</td></tr>
</table>

Offset		*Name*	*Bytes*	*Description*
0	(0)		1	Reserved
1	(1)	DCTCYL	1	Number of cylinders
2	(2)	DCTRACKS	2	Number of tracks per cylinder
4	(4)	DCTBYTE	2	Number of bytes per track
6	(6)	DCTINTRO	1	Overhead for each intermediate record
7	(7)	DCTLASTO	1	Overhead for the last record on a track
8	(8)	DCTKEY	1	Overhead if keys are not used
9	(9)	DCTOLERN	1	Reserved
10	(A)	DCTOLERN	2	Tolerance factor for each intermediate record

(This page left blank intentionally)

Process Control Block

The process control block (IEDQPCB) is a fixed-length table that serves as a named control block to permit inter-region communications between application programs and the message control program. A PCB macro instruction in the MCP defines a PCB. There must be one PCB, hence one PCB macro instruction, for each active application program to be used with the MCP.

The process control block can be addressed by several means. The address of the PCB is in the PEPCBAD field of the process entry work area, the LCBDCBPT field of the application program LCB, the DEBPCBAD field of the data extent block, and the QCBDCBAD field of the destination QCB.

Storage is allocated for the process control block at assembly time for the message control program. The control block is initialized partially at assembly time for the MCP and partially at the application program open time.

The fields PCBBUFIN and PCBBUFO take up one byte in main storage. PCBBUFIN represents the first four bits of the byte and indicates the initial buffer request for PUT or WRITE. PCBBUFO represents the last four bits and indicates the initial buffer request for a GET/READ operation.

The format of the process control block is illustrated below; descriptions of the fields follow.

IEDQPCB

0 (0) Reserved			
8 (8) **PCBRTQCB** Message Retrieval QCB			
20 (14) **PCBBUFIN** PUT/WRITE Buffer Request **PCBBUFO** Max No. of Full QCB Buffers	21 (15) **PCBMH** Address of the Message Handler		
24 (18) **PCBUCNT** Use Count	25 (19) **PCBLINK** Link Field		
28 (1C) **PCBBUFMX** Read-Ahead Buffer Limit	29 (1D) Reserved		
32 (20) **PCBLCBAD** Address of the Line Control Block			
36 (24) **PCBTJID** TSO Job Identifier		38 (26) **PCBCKPT** Checkpoint Offset	
40 (28) **PCBTCBAD** Address of the Task Control Block			
44 (2C) **PCBOFLG** Flag Bit	45 (2D) Reserved		
48 (30) Reserved	49 (31) Reserved		
52 (34) Reserved	53 (35) **PCBUNTCT** Unit Count	54 (36) **PCBBFSZE** Buffer Size	
56 (38) **PCBRSERH** Header Buffer Reserve	57 (39) **PCBRSERT** Text Buffer Reserve	58 (3A) **PCBORC** Open Return Code	59 (3B) Reserved
60 (3C) **PCBWRKA** Operator Control/Application Program Interface Work Area			
88 (58) **PCBEND** End of the PCB **PCBSIZE** PCB Size in Bytes			

Offset	*Name*	*Bytes*	*Description*
0 (0)		8	Reserved
8 (8)	PCBRTQCB	12	Message retrieval QCB
20(14)	PCBBUFIN	1	Initial buffer request for PUT or WRITE
20(14)	PCBBUFO	1	Maximum number of full buffers on the read-ahead QCB
21(15)	PCBMH	3	Address of the Message Handler
24(18)	PCBUCNT	1	Use count
25(19)	PCBLINK	3	Link field
28(1C)	PCBBUFMX	1	Read-ahead buffer limit
29(1D)		3	Reserved
32(20)	PCBLCBAD	4	Address of the line control block
36(24)	PCBTJID	2	TSO job identifier
38(26)	PCBCKPT	2	Checkpoint offset
40(28)	PCBTCBAD	4	Address of the task control block for the related application program
44(2C)	PCBOFLG	1	Flag byte; bit settings for this field are as follows:

Name	*Bit*	*Value*	*Meaning*
PCBRORIN	0	X‘80’	Application program can be rolled out
PCBRORIF	0 Off	X‘7F’	Mask to specify that an application program cannot be rolled out
PCBTSON	1	X‘40’	Application program is TSO
PCBTSOF	1 Off	X‘BF’	Mask to specify that an application program is not TSO
PCBCKPTN	2	X‘20’	Environment checkpoint has been taken in the MCP
PCBCKPTF	2 Off	X‘DF’	Mask to specify that an environment checkpoint has not been taken in the MCP
PCBRETVN	3	X‘10’	Subsequent retrieval
PCBRETVF	3 Off		Mask to specify no subsequent retrieval

45(2D)		3	Reserved
48(30)		1	Reserved
49(31)		3	Reserved
52(34)		1	Reserved
53(35)	PCBUNTCT	1	Unit Count
54(36)	PCBBFSZE	2	Buffer size
56(38)	PCBRSERH	1	Header buffer reserve
57(39)	PCBRSERT	1	Text buffer reserve
58(3A)	PCBORC	1	Open return code
59(3B)		1	Reserved
60(3C)	PCBWRKA	28	Operator Control/application program interface work area
88(58)	PCBEND	1	End of the PCB
88(58)	PCBSIZE	1	Size in bytes of the PCB

(This page left blank intentionally)

Process Entry Work Area

The process entry work area (IEDQPEWA) is a fixed-length table in the message control program. This work area provides a logical extension of the process entry for the associated application program. The work area also provides storage for the control blocks for the GET and PUT Schedulers. The function of the work area varies depending upon the functions of the GET or PUT Scheduler.

The address of the process entry work area is in the TRMSTAT field of a terminal entry when that entry has been generated by a TPROCESS macro instruction. The address is also in the PWAPEWA field of the access method work area in the associated application program.

When a DCB in an application program is being opened, the OPEN/CLOSE subtask (IEDQEU) allocates main storage for and initializes the process entry work area.

The format of the process entry work area is illustrated below; descriptions of the fields follow.

IEDQPEWA

0 (0)	**PEWARES**	Reserved
8 (8)	**PEWAISZE**	Count of Idles Reserved
12 (C)	**PEAQCTL**	AQCTL Parameter List
24 (18)	**PEWAECBA**	Address of the Application Program ECB
28 (1C)	**PEWASOWA**	Work Area Data Length
29 (1D)	**PEWAFLG**	General Flag Byte
30 (1E)	**PEBFCT**	Buffer Limit
32 (20)	**PEUNCT**	Units Per Buffer
	PEPCBAD	Address of Process Control Block
36 (24)	**PERCQCB**	Address of the QCB Associated with the ERB Below
40 (28)		Reserved
44 (2C)	**PEWALCBA**	Address of the LCB
48 (30)		Reserved
52 (34)	**PECBUF**	Address of First Empty Byte in Current Unit – for PUT Address of the Chain of Read-Ahead Buffers Not Processed by MH – for GET
56 (38)	**PEERB**	Element Request Block
76 (4C)	**PEGQWKAR**	Address of Work Area used by Queue Reset Executor
80 (50)	**PEWAELEM**	Special Element
96 (60)	**PERAQCB**	Read-Ahead QCB
108 (6C)	**EOMSAVE**	Address of the Last EOM for GET
	PEWATIC	Current Unit Address for PUT
112 (70)	**PEPSSTCB**	Put Scheduler STCB
	PEGSSTCB	Get Scheduler STCB

120 (78)	**PEWADEB**	Data Extent Block Address
124 (7C)	**PEGFSTCB**	Get FIFO STCB
132 (84)	**PEWAPROC**	Address of the Process Entry
136 (88)	**PESAVE**	Register Save Area

Offset		*Name*	*Bytes*	*Description*
0	(0)	PEWARES	8	Reserved
8	(8)	PEWAISZE	4	Count of idle (reserve) characters reserved
12	(C)	PEAQCTL	12	AQCTL parameter list
24	(18)	PEWAECBA	4	Address of the application program ECB
28	(1C)	PEWASOWA	2	Work area data length
30	(1E)	PEWAFLG	1	General flag byte; bit settings are:

For the GET Scheduler:

Name	*Bit*	*Value*	*Meaning*
ERBBUSY	0	X'80'	ERB tposted to the disk I/O QCB
CFLG	1	X'40'	Closedown in progress
POSTAP	2	X'20'	Need to tpost the application program ERB
FIRSTR	5	X'04'	First-time retrieve flag
MHOK	6	X'02'	Buffer may be tposted to the message handler
RFLG	7	X'01'	Retrieve mode

Offset		*Name*	*Bytes*	*Description*
31	(1F)	PEBFCT	1	Buffer limit—number of buffers that may be on the read-ahead QCB at any one time
32	(20)	PEUNCT	1	Number of units per buffer—fixed per process entry
33	(21)	PEPCBAD	3	Address of the process control block
36	(24)	PERCQCB		Address of the QCB associated with the ERB below
40	(28)		4	Reserved
44	(2C)	PEWALCBA	4	Address of the LCB
48	(30)		4	Reserved
52	(34)	PECBUF	4	For PUT Scheduler—address of the first empty byte in the current unit; GET Scheduler—address of the chain of read-ahead buffers not processed by the message handler
56	(38)	PEERB	24	Element request block
76	(4C)	PEGQWKAR	4	Address of work area used by Queue Reset Execution if QBACK=YES coded on TPROCESS; otherwise zeros.

Offset		*Name*	*Bytes*	*Description*
80	(50)	PEWAELEM	16	Special element
96	(60)	PERAQCB	12	Read-ahead QCB
108	(6C)	EOMSAVE	4	Address of the last EOM for GET
108	(6C)	PEWATIC	4	Current unit address for PUT
112	(70)	PEPSSTCB	8	PUT Scheduler STCB
112	(70)	PEGSSTCB	8	GET Scheduler STCB
120	(78)	PEWADEB	4	Address of the data extent block
124	(7C)	PEGFSTCB	8	GET FIFO STCB
132	(84)	PEWAPROC	4	Address of the process entry
136	(88)	PESAVE	56	Register save area

Queue Control Block

A queue control block (QCB) is used to regulate the sequential use of elements among requesting tasks. Every queue, or item, that is waiting for service in the system is associated with a QCB. There is a master destination QCB for every destination message queue. There is another type of queue control block, called a priority QCB, for each priority level applicable for each destination QCB. The first priority QCB begins at a displacement of 40 (X'28') from the beginning of the destination QCB.

Note: *There is no priority QCB for a TSO dedicated line. The QCB is truncated at the displacement 40 (X'28').*

A QCB has three primary fields: a pointer to the element chain, a link address, and a pointer to the STCB chain. The element chain consists of any elements, other than the requesting resource on the ready queue, that the subtask represented by the STCB chain might need to process. The link field is used to point to another item when a QCB is on a higher queue. The STCB chain consists of pointers to the routines that are associated with the QCB.

The address of the destination QCB is in the TRMDESTQ field of the terminal table entry which is, in turn, pointed to by the termname table entry. The address of the termname table is in the AVTRNMPT field of the address vector table. The LCBSCBDA field of the line control block points to the station control block. Within an SCB is a pointer (SCBDESTQ) to the queue control block.

Storage is allocated for the QCB at assembly time. The QCB is initialized partially at assembly time and partially at open time.

The formats of the master destination queue control block and the priority QCB are shown below; descriptions of the fields follow the illustrations.

Master Queue Control Block DSECT: IEDQQCB

0 (0) QCBDSFLG Flag Byte	1 (1) QCBELCHN Element Chain		
4 (4) QCBPRI Priority	5 (5) QCBLINK Pointer to the Next STCB in a Chain		
8 (8) QCBSTVTO Index to the Entry in the Subtask Vector Table	9 (9) QCBSTCHN STCB Chain		
12 (C) QCBSTPRI Priority of the STCB	13 (D) QCBSLINK Pointer to the Next STCB in a Chain		
16 (10) QCBEOLDT Interrupt Time		18 (12) QCBRETCT TSO Retry Counters QCBLKRLN Lock Relative Line Number	19 (13) QCBSTAT Status of this QCB
20 (14) QCBSCBOF Offset to the Proper SCB	21 (15) QCBINSRC Chain of Source LCBs Currently Sending Initiate Mode Msgs		
	QCBSATCT Sim ATTN Output Line Count	22 (16) QCBTSOF2 Second TSO Flag Byte	23 (17) QCBTSOF1 First TSO Flag Byte
24 (18) QCBINTVL; QCBEXTO Interval for Poll Delay; Offset to EXT		26 (1A) QCBMSGCT Count of Messages in this Queue	
28 (1C) QCBPREN Address of Terminal Table Entry if QCB for a Process Entry			
QCBPRLVL Highest Priority Level Message	29 (1D) QCBLKRRN Lock Relative Line Number		
	QCBCARCT Carriage Position Count	30 (1E) QCBTJID TSO Job Identification	
32 (20) QCBRELLN Relative Line Number	33 (21) QCBDCBAD Address of the DCB		
36 (24) QCBFLAG QCB Status Bits	37 (25) QCBQBACK QBACK Message Chain		

Priority Queue Control Block DSECT: IEDPQCB

40 (28) **QCBDNHDR** Disk Record Number to Put the Next Header Received	43 (2B) **QCBDATFL** Data Flags Field
44 (2C) **QCBPFEFO** Record Number of Message Previous to Last Message Serviced	47 (2F) **QCBDATSQ** Sequence Number
Continued	49 (31) **QCBINTFF** Disk Record Number of the First Intercepted Msg – FEFO Order
52 (34) **QCBPREVF** Record Number of Message Prior to the Message in QCBFFEFO	55 (37) **QCBFFEFO** Disk Rcd. No. of First FEFO Message or Core Rcd. No.
Continued	58 (3A) **QCBLFEFO** Disk Rcd. No. of Last FEFO Msg Core Rcd. No. if Core – Only Queue
Continued	61 (3D) **QCBCFHDR** Core Record No. of First Header Appearing in this Queue
64 (40) **QCBPRIPQ** Priority of this Priority Level QCB	65 (41) **QCBCPVHD** Core Address of Last Address Placed on this Queue

Offset		*Name*	*Bytes*	*Description*

The following is for the master QCB:

0	(0)	QCBDSFLG	1	Flags that indicate a specific destination QCB to the Dispatcher and which message queues data set is to receive the messages for the destination. Bit definitions are as follows:

Name	*Bit*	*Value*	*Meaning*
QCBFQCB	6	X'02'	Indicates a QCB
QCBDRQQ	5	X'04'	Indicates a concentrator data ready queue
QCBALTMH	4	X'08'	Indicates messages to alternate MH
QCBREUS	3	X'10'	Indicates reusable disk queuing
QCBNREUS	2	X'20'	Indicates nonreusable disk queuing
QCBDISK	2,3	X'30'	Disk queues are used
QCBCORE	1	X'40'	Flag for main-storage queues:
	1,3	X'50'	Indicates main-storage queues with backup on reusable disk
	1,2	X'60'	Indicates main-storage queues with backup on nonreusable disk
QCBTSQ	0	X'08'	Indicates time-sharing queues

1	(1)	QCBELCHN	3	Element chain pointer—contains the address of the QCB to be tposted when this QCB is removed from the time delay queue.
4	(4)	QCBPRI	1	Priority
5	(5)	QCBLINK	3	Pointer to the next STCB in a chain
8	(8)	QCBSTVTO	1	Index to an entry in the subtask vector table

Offset		*Name*	*Bytes*	*Description*		
9	(9)	QCBSTCHN	3	STCB chain pointer		
12	(C)	QCBSTPRI	1	Priority of the STCB		
13	(D)	QCBSLINK	3	Pointer to the next STCB in a chain		
16	(10)	QCBEOLDT	2	Interrupt time		
18	(12)	QCBRETCT	1	TSO retry counters		
			Name	*Bit*	*Value*	*Meaning*
			QCBCR	2	X'20'	TSO Carriage Return request
			QCBLF	3	X'10'	TSO Line Feed request
			QCBNL	2,3	X'30'	TSO New Line request
			QCBEND	4	X'08'	TIOC Edit Special output request
			QCBIEND	5	X'04'	TIOC Edit Special output request or 3270 Format bit
18	(12)	QCBLKRLN	1	Lock relative line number		
19	(13)	QCBSTAT	1	Status of this QCB; bit settings are:		
			Name	*Bit*	*Value*	*Meaning*
			QCBEOM	0	X'80'	End of message sent
			QCBTRMHO	1	X'40'	Terminal was held
			QCBBUFRD	2	X'20'	Buffered terminal
			QCBSEND	3	X'10'	Sending to a buffered terminal
			QCBRECEV	4	X'08'	Receiving from a buffered terminal
			QCBSCHDL	5	X'04'	Put in the time delay queue when inactive
			QCBCLOCK	6	X'02'	On=clock, Off=interval
			QCBTIME	7	X'01'	Delay greater than 12 hours
20	(14)	QCBSCBOF	1	Offset to the proper SCB for this transmission; X'00' unless this line has buffered terminals		
21	(15)	QCBINSRC	3	Chain of source LCBs currently sending initiate mode messages to this destination queue		
21	(15)	QCBSATCT	1	Simulated attention output line count (TSO)		
22	(16)	QCBTSOF2	1	Second TSO flag byte; bit settings are:		
			Name	*Bit*	*Value*	*Meaning*
			QCBINHBN	0	X'80'	Use inhibits with this terminal
			QCBBUFQ	1	X'40'	TCAM buffer being held
			QCBPOSTO	2	X'20'	QCB tposted to itself
			QCBDSSMI	3	X'10'	Start MI character sent (TSO)
			QCBSATCH	5	X'04'	Simulated attention by character
			QCBSATTI	6	X'02'	Simulated attention by time
			QCBSATLC	7	X'01'	Simulated attention by line

Offset		*Name*	*Bytes*	*Description*		
23	(17)	QCBTSOF1	1	First TSO flag byte; bit settings are:		
			Name	*Bit*	*Value*	*Meaning*
			QCBWRBRK	0	X'80'	Issue a write break
			QCBTGET	1	X'40'	TGET request
			QCBTPUT	2	X'20'	TPUT request
			QCBNOBUF	3	X'10'	Insufficient buffers
			QCBSATRD	4	X'08'	Simulated attention read request
			QCBPARTO	5	X'04'	Partial output line
			QCBDELAY	6	X'02'	QCB in time delay queue
			QCBDISC	7	X'01'	User to be logged off
24	(18)	QCBEXTO	2	Offset to the QCB extension		
24	(18)	QCBINTVL	2	Interval for poll delay		
26	(1A)	QCBMSGCT	2	Count of messages in this queue		
28	(1C)	QCBPREN	4	Address of the terminal table entry if this is a QCB for a process entry		
28	(1C)	QCBPRLVL	1	Highest-priority message		
29	(1D)	QCBLKRRN	3	Lock relative line number; link field for the QCB when it's on the time delay queue		
29	(1D)	QCBCARCT	1	Carriage position count		
30	(1E)	QCBTJID	2	TSO job identification		
32	(20)	QCBRELLN	1	Relative line number for the line this QCB represents		
33	(21)	QCBDCBAD	3	Address of the DCB		
34	(22)	QCBFLAG	1	QCB status bits; bit settings are:		
			Name	*Bit*	*Value*	*Meaning*
			QCBTSSES	0	X'80'	TSO session in progress
			QCBNOBRK	1	X'40'	No reverse break feature
			QCBREAD	2	X'20'	Read has priority
			QCBRSRV	3	X'10'	Reusability serviced
			QCBTERMQ	4	X'08'	Queue by terminal
			QCBSDFFO	5	X'04'	Currently sending a message
			QCBPROC	6	X'02'	This QCB is for a process entry
			QCBCKPT	7	X'01'	Flag for checkpoint
37	(25)	QCBQBACK	3	Queue-back message chain		
The following is for a priority QCB:						
40	(28)	QCBDNHDR	3	Disk record number of the first unit of the first buffer of the next message		
43	(2B)	QCBDATFL	1	Data flags field of the last message removed from the FEFO queue		

Offset		Name	Bytes	Description
44	(2C)	QCBPFEFO	3	If a terminal on this queue is held, the record number of the message previous to the first message held, otherwise the record number of the message previous to the last one marked serviced
47	(2F)	QCBDATSQ	2	Sequence number of the last message removed from the FEFO queue
49	(31)	QCBINTFF	3	Disk record number of the first held message in FEFO order
52	(34)	QCBPREVF	3	Disk record number of the FEFO message before the message in QCBFFEFO
55	(37)	QCBFFEFO	3	Disk record number of the first message to be completely received. Main-storage record address if this is a main-storage-only queue
58	(3A)	QCBLFEFO	3	Disk record number of the last FEFO message received. Main-storage record address if this is a main-storage-only queue
61	(3D)	QCBCFHDR	3	Main-storage record address of the first buffer of the first message appearing in this queue
64	(40)	QCBPRIPQ	1	The priority of this priority level QCB. This is X'00' if this is the lowest priority level.
65	(41)	QCBCPVHD	3	Main-storage record address of the last address placed on this queue

Queue Control Block Extension

A QCB extension contains the information necessary to execute the OUTMSG subgroup for a terminal that is attached to a concentrator. There is one QCB extension for each master destination QCB, plus on for each priority QCB if priority level queuing is used (that is, QCONTROL=MSG,level).

The offset from the master QCB to the QCB extension is in the QCBEXTO field of the master QCB.

The format of the QCB extension is illustrated below; descriptions of the fields follow.

IEDQQCBE

0 (0)	**QCBEFLG**	Flag Byte
1 (1)	**QCBEHDR** **QCBECONC**	SCBSCHDR Saved Address of the Concentrator Terminal Entry
4 (4)	**QCBEOSEQ** **QCBEDAMT**	SCBOSEQ Saved Amount of Data to Take from the Queue
6 (6)	**QCBETCIN**	TTCIN of the Last Message from the Queue
8 (8)	**QCBELGTH**	Entry Length
9 (9)	**QCBENPLV**	Number of Priority Levels
10 (A)	**QCBEPRI** **QCBEMACR**	SCBPRI Saved SCBMACR Saved
11 (B)	**QCBEFEFO**	SCBFEFO Saved
12 (C)	**QCBEFEFO** (Cont.) **QCBEMACR** (Cont.)	
13 (D)	**QCBEEOB**	SCBEOB Saved
14 (E)	**QCBEQTYP**	Type
15 (F)	**QCBEDROB**	
16 (10)	**QCBEDROB** (Cont.)	SCBDROB Saved
19 (13)	**QCBELRS**	Length of the CTB Characters
20 (14)	**QCBERS**	Start of CTB Characters

Offset		*Name*	*Bytes*	*Description*		
0	(0)	QCBEFLG	1	Flag byte—bit definitions are:		
			Name	*Bit*	*Value*	*Meaning*
			QCBESTAT	0	X‘80’	STATUS specified on QCONTROL
			QCBECNT	1	X‘40’	INTEGER specified
			QCBEOPL	2	X‘20’	Priority level QCB defined
			QCBEHELD	3	X‘10’	Temporary hold
			QCBESRVC	4	X‘08’	QCB is serviced
			QCBEOMSG	5	X‘04’	OUTMSG is pending
			QCBEDATA	6	X‘02’	Data is in the message
			QCBEPEND	7	X‘01’	QACTION operation is pending
1	(1)	QCBEHDR	3	SCBSCHDR field saved		
1	(1)	QCBECONC	3	Address of the concentrator terminal entry		
4	(4)	QCBEOSEQ	2	SCBOSEQ field saved		
4	(4)	QCBEDAMT	2	Amount of data to take from the queue		
6	(6)	QCBETCIN	2	TTCIN of the last message from the queue		
8	(8)	QCBELGTH	1	Length of the entry		
9	(9)	QCBENPLV	1	Number of priority levels		
10	(A)	QCBEPRI	1	SCBPRI field saved		
10	(A)	QCBEMACR	3	SCBMACR field saved		
11	(B)	QCBEFEFO	3	SCBFEFO feild saved		
13	(D)	QCBEEOB	2	SCBEOB field saved		
14	(E)	QCBETYP	1	Type—bit definitions are:		
			Name	*Bit*	*Value*	*Meaning*
			QCBEMM	7	X‘01’	Middle of the message
15	(F)	QCBEDROB	4	SCBDROB field saved		
19	(13)	QCBELRS	1	Length of the CTB characters, a maximum of 8 characters		
20	(14)	QCBERS	1	Start of the CTB		

Resource Control Block

The resource control block (IEDQRECB) is à two-word prefix to an element that allows the TCAM Dispatcher to determine the disposition of an element and to determine the QCB to which an element will be tposted. Each element in the TCAM system is represented by a resource control block (RCB). The first word of the RCB is a pointer to the QCB with which the element is associated; the second word is a link field which, when the element is on a chain, points to the next item on the chain. The first word in the associated QCB may point to the RCB.

Storage is allocated for the RCB at open time for the line group or for the application program. The RCB is initialized at open time and is modified when elements are passed in the system.

There are two types of permanent RCBs:

1. Buffer RCBs
2. Communication line RCBs

Buffers are areas of main storage used to contain message data and/or control information. The first eight bytes of each buffer comprise an RCB. As with all TCAM elements, the identity of a buffer depends solely upon the queue that its representative RCB is chained to at a particular time. The buffer itself is always physically identifiable as a fixed number of bytes of main storage. If the RCB representing the buffer is chained into a destination QCB, the buffer is full; that is, it contains a message segment to be transmitted to a destination. When the same RCB is subsequently chained into the element chain of the buffer request QCB, the element involved is a available buffer, even though there has been no change in the physical storage location of the buffer.

A line control block (LCB) represents a communication line to the TCAM MCP. There is an LCB for each line in the system. When a subtask has control of an LCB, it has control of the line; therefore, the LCB itself is treated as the resource element. The RCB is contained within the first two words of the LCB.

There are two special types of RCBs:

1. Queue control block RCBs

When a queue control block (QCB) appears on the ready queue, it may represent a special case in which the QCB is tposted to itself. The QCB acts as a special element rather than as a system resource, in that the first subtask on the STCB chain of the QCB gains control without an element to process. The subtask must be self-contained and able to locate any data it needs for execution. If there are no elements to process, the QCB has gained the system resourcemtime.

2. Element request block RCBs

An element request block (ERB) on the ready queue can act as a request for a resource or as an actual element itself.

Below is the format of a resource control block; descriptions of the fields follow the illustration.

IEDQRECB

0 (0) **RECBKEY** Key Field	1 (1) **RECBQCBA** QCB Address
4 (4) **RECBPRI** Priority	5 (5) **RECBLINK** Link Field

Offset		*Name*	*Bytes*	*Description*
0	(0)	RECBKEY	1	Key field
1	(1)	RECBQCBA	3	Address of the QCB to which this RCB is tposted
4	(4)	RECBPRI	1	Priority of this RCB
5	(5)	RECBLINK	3	Address of the next RCB in the chain in which this RCB is currently located

Special Characters Table

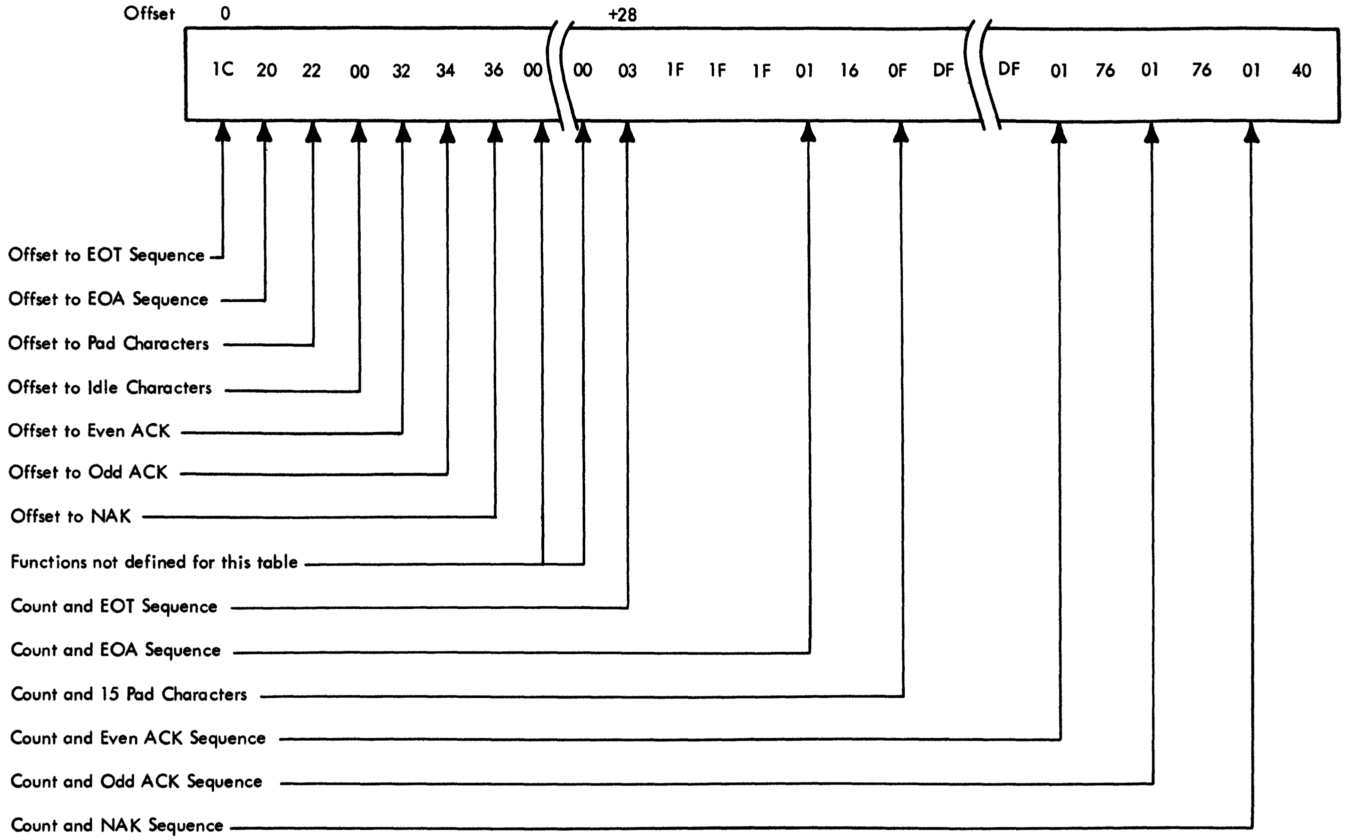
A special characters table (SCT) is a variable-length table that consists of entries giving the special characters required for device I/O for a specific line group. There is one SCT for each type of line group in the TCAM system. Each SCT contains a list of the characters that the associated terminal or line group recognizes. SYS1.SVCLIB contains a special characters table for each line group in the system. The various SCTs are initialized at SYSGEN time, ànd at open time the TCAM Line Group Open routine uses information from the UCB and the terminal entry to load the appropriate special characters table.

An SCT is located by a three-byte address in the DCBSCTAD field of the DCB for the line group. The address of the DCB for the line group is in the LCBDCBPT field of the associated LCB.

An SCT is used to build channel programs. This table is also used by the error recovery procedures to retry certain text errors, and by the message handling routines to initiate on-line test procedures and to determine the message format for line control insertion.

The first 28 bytes of an SCT comprise a fixed-length directory of one-byte offsets, each of which, when added to the SCT pointer in the DCB, points to a one-byte length field. This length field is followed by a special characters entry of the length specified in the length field. There are as many entries in the directory as there are different sets of special characters required by the line group. If a function is not defined for the associated terminal or line group, the offset field in the directory contains a X'00' value.

The following is an example of an SCT entry.



The following is a list of the specific types of characters that each of the offsets in the first 28 bytes of a SCT represent.

Offset	*Special Characters*	*Used By*
0 (0)	EOT sequence	I/O Generator
1 (1)	EOA sequence	I/O Generator
2 (2)	PAD characters	I/O Generator
3 (3)	Idle or reserve characters	I/O Generator
4 (4)	Even ACK	I/O Generator
5 (5)	Odd ACK	I/O Generator
6 (6)	NAK	I/O Generator
7 (7)	ENQ (inquiry)	I/O Generator
8 (8)	EOB/ETB (for BSC DLE ETB)	I/O Generator
9 (9)	DLE ETX (BSC)	I/O Generator
10 (A)	DLE STX (BSC transparent sequence)	I/O Generator
11 (B)	DLE/STX/ENQ (BSC transparent temporary text delay-TTD)	Line End Appendage
12 (C)	SOH (BSC start of header character)	Line End Appendage
13 (D)	On-line Test sequence	Line End Appendage START MH Subtask
14 (E)	WACK (BSC)	Line End Appendage
15 (F)	RVI (BSC reverse interrupt)	Line End Appendage
16 (10)	DLE EOT (BSC dial sequence)	Line End Appendage
17 (11)	DLE ENQ (BSC—use in abort sequence)	ERP Modules
18 (12)	Blocking sequence	MSGFORM function
19 (13)	Subblock sequence	MSGFORM function
20 (14)	Ending sequence	MSGFORM function
21 (15)	EOT sequence	PCI Appendage
22 (16)	EOB sequence	PCI Appendage
23 (17)	ETX sequence	PCI Appendage
24 (18)	ENQ sequence	PCI Appendage
25 (19)	SOH % S sequence	Line End Appendage
26 (1A)	SOH % E sequence	Line End Appendage
27 (1B)	SOH %/CANCEL/ sequence (BSC On-Line Test cancel sequence)	Line End Appendage
28 (1C)	SOH % C sequence	Line End Appendage

Station Control Block

There is at least one station control block (SCB) associated with each LCB in the TCAM system. With buffered terminals there is one SCB per terminal on a line. A buffered terminal receives a block or a part of an entire transmission at a time; while that terminal is transmitting data to the output device, TCAM examines and sends to other terminals on the same line. TCAM uses the SCB for a terminal to keep track of one transmission from that buffered terminal on the line.

If the terminals on a line are not buffered, or if the line with which the SCB is associated is a dial line, one terminal at a time completes its transmission. There is no need to keep track of many transmissions in parallel, so one SCB is sufficient for the entire line. In this case the address of the SCB is the LCBSCBA field of the LCB.

The address of the SCB directory is in the LCBSCBDA field of the line control block. The offset to the current SCB is in the LCBSCBO field of the LCB.

To obtain the address of any SCB associated with a QCB, TCAM first locates the LCB. This is done by multiplying the relative line number (in QCBRELLN) by the size of an LCB (DCBEIOBX) and adding the address of the pseudo-IOB (DCBIOBAD). This gives TCAM the address of the IOB. At a displacement of –X‘20’ from the beginning of the IOB is the beginning of the LCB. TCAM then multiplies the SCB size (located in the AVTSCBSZ field of the address vector table) by the offset in QCBSCBOF and adds that total to the address of the SCB directory (LCBSCBDA). This sum then points to the desired station control block.

Storage is allocated for a station control block at assembly time for leased lines and at open time for dial lines. The SCB is initialized by STARTMH.

The format of the station control block is illustrated below; descriptions of the fields follow.

IEDQSCB

0 (0) **SCBSTATE** Status Bits	1 (1) **SCBDESTQ** Pointer to the Destination QCB		
4 (4) **SCBSNDCT** Message Limit On Send Side **SCBRCVCT** Message Limit On Receive Side	5 (5) **SCBMACR** First/Next IN/OUTMSG Macro to be Executed **SCBMBHEN** Address of the Multiple Header Buffer Entry		
8 (8) **SCBPRI** Priority Index to the QCB	9 (9) **SCBBKFCT** Count of Message Length for Break	10 (A) **SCBEOBSZ** Size of Logical Blocks	**SCBCTBSZ** Size of CTB
12 (C) **SCBSALEV** Simulated Attention Level Req **SCBQTYPE**	13 (D) **SCBMRFPL** Address of Forward Parameter List		
16 (10) **SCBERRST** Error Word Bits			
SCBERR1 First Byte	17 (11) **SCBERR2** Second Byte	18 (12) **SCBERR3** Third Byte	19 (13) **SCBERR4** Fourth Byte
20 (14) **SCBMRFSD** Multiple Router First Secondary Destination		22 (16) **SCBEOBAC; SCBCTBAC** Accumulated Count Between Blocks; Accumulated Count of Data Inserted	
24 (18) **SCBBSCFM** MSGFORM Dynamic Block Changes	25 (19) **SCBMBSSA** Multiple Buffer Scan Save Area		
32 (20) **SCBCPBNO** Number of Next Sequential CPB	33 (21) **SCBDCHDR** Disk Address Current Header		
36 (24) **SCBDESTL** Length of Destination Names	37 (25) **SCBCCHDR** Main Storage Address of the Current Header		
40 (28) **SCBITBSZ** Size of Logical Subblocks **SCBCTBSV** CTBFORM Parameters Saved	41 (29) **SCBSCSEG** Current Segment Being Read **SCBDNSEG** Disk Address of the Next Segment to Write to the Disk		
44 (2C) **SCBHBFNO** Number of Buffers in Multiple Header	45 (2D) **SCBSCHDR** Current Header Being Sent **SCBCLSEG** Main Storage Address of the Last Message Segment Placed in the Main-Storage Queue		
48 (30) **SCBITBAC** Accumulated Count Between ITBs **SCBCTBFL** Concentrator Flag Byte	49 (31) **SCBFEFO** Saved FEFO Pointer **SCBDCSEG** Disk Address of the Current Segment		
52 (34) **SCBDEOB** Disk Information On the Last EOB			
56 (38) **SCBSRCE** Message Buffer Source Saved		58 (3A) **SCBSIZE** Message Buffer Size Saved	
60 (3C) **SCBSTAT1** Status Byte	61 (3D) **SCBXTRA** Address of Additional Records Saved **SCBCORE** Address of the Record in the Core Queue Saved		
64 (40) **SCBSEQ** Sequence Out Number **SCBSCAN** Scan Pointer		66 (42) **SCBTQBCK** Text Segment Chain Saved **SCBNTXT** Address of the Next Text Segment Saved	
Continued Continued	69 (45) **SCBCRCD** Address of the Current Segment Saved		

72 (48) **SCBNHDR** Address of the Next Header Segment Saved **SCBCHDR** Address of the Current Header Segment Saved	75 (4B) **SCBNXCPB** Next CPB Number from Disk **SCBLCSEG** Main Storage Address of Current Segment
Continued	78 (4E) **SCBEOB** Pointer to First EOB Saved
80 (50) **SCBUNTCT** Count in Disk Record of First Byte of Data	81 (51) **SCBTRANS** Current Translate Table Address
84 (54) **SCBRGSAV** Save Area for User MH Registers — if Specified on INTRO	

Offset		*Name*	*Bytes*	*Description*		
0	(0)	SCBSTATE	1	Status bits:		
			Name	*Bit*	*Value*	*Meaning*
			SCBTRANP	0	X‘80’	Message in transparent mode
			SCBMGFMN	1	X‘40’	MSGFORM requested
			SCBMGFMF	1 Off	X‘BF’	Mask to specify MSGFORM not requested
			SCBSEQIN	1	X‘40’	Sequence-in has been executed for the current message
			SCBLCK1F	2 Off	X‘DF’	Mask to specify that a message is not being received in lock mode
			SCBMSGLN	4	X‘08’	Message lock mode
			SCBMSGLF	4 Off	X‘F7’	Mask to specify extended lock mode
			SCBCKPT	5	X‘04’	Checkpoint requested
			SCBPRER	6	X‘02’	Previous EOB/ETB error
			SCBCODE	7	X‘01’	Translation requested
1	(1)	SCBDESTQ	3	Address of the destination QCB		
4	(4)	SCBSNDCT	1	MSGLIMIT on Send side		
4	(4)	SCBRCVCT	1	MSGLIMIT on Receive side		
5	(5)	SCBMACR	3	First or next INMSG or OUTMSG macro to be executed		
5	(5)	SCBMBHEN	3	Address of the multiple-buffer-header entry		
8	(8)	SCBPRI	1	Priority index to the QCB		
9	(9)	SCBBKFCT	3	Count of message length for break		
10	(A)	SCBEOBSZ	1	Size of logical blocks		
10	(A)	SCBCTBSZ	2	Size of the concentrator terminal block (CTB)		
12	(C)	SCBSALEV	1	Simulated attention level request (TSO)		
12	(C)	SCBQTYPE	1	Queuing medium for this message:		
			Name	*Bit*	*Value*	*Meaning*
			SCBCOREQ	1	X‘40’	Main-storage queues
			SCBREUS	2	X‘20’	Reusable disk queues
			SCBNREUS	3	X‘10’	Nonreusable disk queues
			SCBCONC	4	X’08’	Concentrator SCB
				5	X‘04’	Reserved
			SCBBFTM	6	X‘02’	Buffered terminal SCB
			SCBBFMM	7	X‘01’	Buffered terminal in middle of message
13	(D)	SCBMRFPL	3	Address of the FORWARD parameter list		
16	(10)	SCBERRST		Error word bits		

Offset		*Name*	*Bytes*	*Description*		
16	(10)	SCBERR1	1	First byte:		
			Name	*Bit*	*Value*	*Meaning*
			SCBHDRRN	0	X'80'	Incomplete header
			SCBHDRRF	0 Off	X'7F'	Mask to specify not an incomplete header
			SCBNOLOG	0	X'80'	Invalid Logon message (TSO)
			SCBORIGN	1	X'40'	Invalid origin
			SCBORIGF	1 Off	X'BF'	Mask to specify a valid origin
			SCBHANG	1	X'40'	Logon requests hang-up message (TSO)
			SCBNOTRM	2	X'20'	Not a TSO terminal (TSO)
			SCBSEQHN	3	X'10'	Sequence high
			SCBSEQHF	3 Off	X'EF'	Mask to specify that sequence is not high
			SCBNOTSO	3	X'10'	TSO is not in the system (TSO)
			SCBSEQLN	4	X'08'	Sequence low
			SCBSEQLF	4 Off	X'F7'	Mask to specify that the sequence is not low
			SCBNOVAC	4	X'08'	Too many TSO users (TSO)
			SCBNOBFN	6	X'02'	Insufficient buffers
			SCBCUTFN	7	X'01'	CUTOFF error
			SCBCUTFF	7 Off	X'FE'	mask to specify no CUTOFF error
			SCBRVISL	7	X'01'	RVI to selection on a buffered terminal terminal
17	(11)	SCBERR2	1	Second byte:		
			Name	*Bit*	*Value*	*Meaning*
			SCBCRMIN	0	X'80'	Main-storage minimum passed
			SCBCRMAX	1	X'40'	Main-storage maximum passed
			SCBCODER	2	X'20'	Error in dynamic translate (TSO)
			SCBALN	3	X'10'	Automatic line numbering (TSO)
			SCBOLTR	4	X'08'	TOTE not in the system
			SCBABRTN	5	X'04'	Abort—BSC terminal
			SCBFRWDN	6	X'02'	Terminal FORWARD error
			SCBSOHE	7	X'01'	SOH%E, C, or R message
18	(12)	SCBERR3	1	Third byte:		
			Name	*Bit*	*Value*	*Meaning*
			SCBLOSTN	0	X'80'	Message lost (overlaid)
			SCBLOSTF	0 Off	X'7F'	Mask to specify message processed

Offset		*Name*	*Bytes*	*Description*		
			SCBXPI	0	X‘80’	Attention: Send I (TSO)
			SCBTMIDN	1	X‘40’	ID from terminal invalid
			SCBTMIDF	1 Off	X‘BF’	Mask to specify that the terminal identification is valid
			SCBXPD	1	X‘40’	Attention: Send D (TSO)
			SCBSATTN	3	X‘10’	Simulated Attention received (TSO)
			SCBUSERN	4	X‘08’	User error
			SCBUSERF	4 Off	X‘F7’	Mask to specify no user error
			SCBFORMN	5	X‘04’	Format error—BSC message
			SCBATTN	6	X‘02’	Hardware Attention
			SCBXCEPN	7	X‘01’	Unit exception
			SCBXCEPF	7 Off	X‘FE’	Mask to specify no unit exception
19	(13)	SCBERR4	1	Fourth byte:		
			Name	*Bit*	*Value*	*Meaning*
			SCBSLCTN	0	X‘80’	Error during selection
			SCBSLCTF	0 Off	X‘7F’	Mask to specify no selection error
			SCBTXTTN	1	X‘40’	Error during text transfer
			SCBTXTTF	1 Off	X‘BF’	Mask to specify no text transfer error
			SCBCONNN	2	X‘20’	Error in connect/disconnect
			SCBCONNF	2 Off	X‘DF’	Mask to specify no connect/disconnect error
			SCBTRMLN	3	X‘10’	Terminal error
			SCBTRMLF	3 Off	X‘EF’	Mask to specify no terminal error
			SCBCTLUN	5	X‘04’	Error in the control unit
			SCBCTLUF	5 Off	X‘FB’	Mask to specify no control unit error
			SCBCHANN	6	X‘02’	Error in channel
			SCBCHANF	6 Off	X‘FD’	Mask to specify no error in channel
			SCBUNDFN	7	X‘01’	Undefined error—should not occur
			SCBUNDFF	7 Off	X‘FE’	Mask to specify no undefined error
20	(14)	SCBMRFSD	2	Multiple routing first secondary destination		
22	(16)	SCBEOBAC	2	Accumulated count between blocks		
22	(16)	SCBCTBAC	2	Accumulated count of the data inserted		

Offset		*Name*	*Bytes*	*Description*
22	(16)	SCBDLPTR	2	Distribution list pointer
24	(18)	SCBBSCFM	1	MSGFORM dynamic block changes:

Name	*Bit*	*Value*	*Meaning*
SCBTRNSP	0	X'80'	Receiving transparent
SCBNONTR	1	X'40'	Receiving non-transparent
SCBRCVTX	2	X'20'	ETX received from BSC
SCBCNTEN	3	X'10'	Switch for Scheduler to determine the next operation on the line
SCBNOEOT	6	X'02'	BSC dial—no EOT before read
SCBMLMTN	7	X'01'	MSGLIMIT has been exceeded
SCBMLMTF	7 Off	X'FE'	Mask to specify that MSGLIMIT is not exceeded

25	(19)	SCBMBSSA	7	Multiple buffer scan save area
32	(20)	SCBCPBNO	1	Number of the next sequential CPB to be read from disk
33	(21)	SCBDCHDR	3	Disk address of the current header
36	(24)	SCBDESTL	1	Length of destination names
37	(25)	SCBCCHDR	3	Main-storage address of the current header
40	(28)	SCBITBSZ	1	Size of logical subblocks
40	(28)	SCBCTBSV	1	CTBFORM parameters saved
41	(29)	SCBSCSEG	3	Current segment being read
41	(29)	SCBDNSEG	3	Disk address of the next segment to write to the disk
44	(2C)	SCBHBFNO	1	Number of buffers in the multiple-buffer header
45	(2D)	SCBSCHDR	3	Current header being sent
45	(2D)	SCBCLSEG	3	Main-storage address of the last segment placed in the main-storage queue
48	(30)	SCBITBAC	1	Accumulated count between ITBs
48	(30)	SCBCTBFL	1	Concentrator flag byte
49	(31)	SCBFEFO	3	Saved FEFO pointer
49	(31)	SCBDCSEG	3	Disk address of the current segment
52	(34)	SCBDEOB	4	Disk information on the last EOB

Note: *The section in bytes 54-79 is a copy of the last buffer prefix processed.*

54	(38)	SCBSRCE	2	Message buffer source
58	(3A)	SCBSIZE	2	Message buffer size
60	(3C)	SCBSTAT1	1	Status byte—concentrator support only:

Name	*Bit*	*Value*	*Meaning*
SCBCBGN	0	X'80'	Concentrator message beginning
SCBCEND	1	X'40'	Concentrator message end
SCBNIDLE	2	X'20'	Buffers should not be put in the idles loop yet

Offset		*Name*	*Bytes*	*Description*
			SCBNOPST	3 X'10' Buffers should not be tposted
61	(3D)	SCBXTRA	3	Address of additional records
61	(3D)	SCBCORE	3	Address of the record in main storage
64	(40)	SCBOSEQ	2	Sequence-out number
64	(40)	SCBSCAN	2	Scan pointer address
66	(42)	SCBTQBCK	3	Text segment queue-back chain
66	(42)	SCBNTXT	3	Address of the next text segment
69	(45)	SCBCRCD	3	Address of the current segment
72	(48)	SCBNHDR	3	Address of the next header segment
72	(48)	SCBCHDR	3	Address of the current header segment
75	(4B)	SCBNXCPB	1	Next CPB number from disk; if zero, no multiple routing
75	(4B)	SCBCCSEG	3	Main-storage address of the current segment
78	(4E)	SCBEOB	2	Pointer to the first EOB
80	(50)	SCBUNTCT	1	Count of the first byte of data in the disk record
81	(51)	SCBTRANS	3	Current translation table address
84	(54)	SCBRGSAV	4	Save area for user MH registers if specified on INTRO

Subtask Control Block

A subtask control block (IEDQSTCB) is a variable-length table that represents a routine that performs the work of the TCAM system. The purpose of an STCB is to cause a routine to be executed. The TCAM Dispatcher uses the STCB to determine the entry point of a subtask that is waiting for work and uses the activation key of the STCB to determine the type of STCB present. The address of the STCB is in the third word of the QCB. Determination of the actual address of the subtask varies according to the type of STCB. When the address is available, the TCAM Dispatcher exits to the routine itself.

For each attached task (Operator Control, On-Line Test, Checkpoint, and FE Common Write) there is a special QCB that has an event control block (ECB) in the second word. The TCAM Dispatcher posts the ECB when the attached task is to vie for control of the system. An element that is to be passed to the attached task is chained into the QCB element chain.

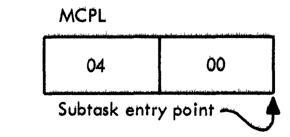
Storage is allocated for the STCB at various times depending upon the type of QCB containing the STCB address. If the QCB is a destination QCB, storage is allocated for the STCB at assembly time. If the QCB is in a line control block or is a read-ahead QCB, storage is allocated for the STCB at open time for the line group or for the application program DCB. If the QCB is in the AVT, storage is allocated at assembly time. In cases where the QCB is a prefix to a module, storage is allocated for the STCB at assembly time.

In the same manner, initialization of the STCB depends upon the related QCB. If the QCB is a destination QCB, the STCB is initialized at assembly time but is modified at open time for the DCB to which it is related. If the QCB is in the LCB or is a read-ahead QCB, the STCB is initialized at open time. If the QCB is in the AVT, the STCB is initialized at assembly time and at link-edit time. If the QCB is a prefix to a module, the STCB is initialized at assembly time.

The following figure shows the formats and attributes of the different types of STCBs.

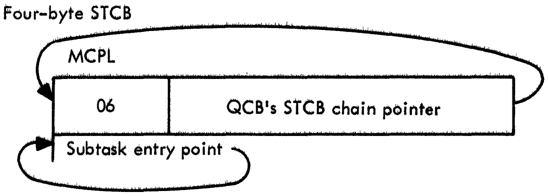
Format: Attributes:

Two-byte STCB



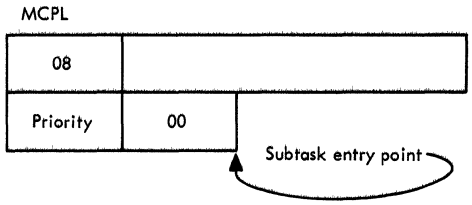
- QCB located in the AVT or assembled in main storage
- QCB has only one STCB
- STCB is never chained to any other QCB

Four-byte STCB



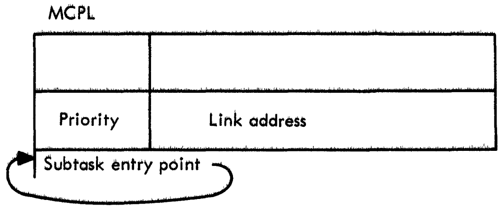
- QCB is part of the subtask code
- QCB and STCB are combined - the STCB is the third word of the QCB
- QCB has only one STCB
- STCB is never chained to any other QCB

Six-byte STCB



- STCB is always the last STCB in the STCB chain of a QCB

Eight-byte (Full) STCB



- STCB can appear in any position of the STCB chain of a QCB

Below is the format of a full (eight-byte) STCB; descriptions of the fields follow the illustration.

IEDQSTCB

0 (0) **STCBVTO** Activation Key	1 (1) Reserved
4 (4) **STCBPRI** Priority	5 (5) **STCBLINK** Link Field

Offset		*Name*	*Bytes*	*Description*
0	(0)	STCBVTO	1	Activation key
1	(1)	STCBINDX	1	Index to Common Buffer data area
2	(2)	STCBTCIN	2	TTCIN of destination terminal
4	(4)	STCBPRI	1	Priority
5	(5)	STCBLINK	3	Link address—address of the next STCB in this STCB chain

Terminal Table

The terminal table (IEDQTRM) is a variable-length table that contains blocks of device-dependent information about each terminal in the TCAM system; each such block is called a terminal entry. There are six types of terminal entries (shown below), each of which is used for a different type or group of terminals depending upon the configuration of the teleprocessing system.

The terminal table entries are assembled and initialized according to the specifications of the TERMINAL, TLIST, TPROCESS, TTABLE, LOGTYPE, and OPTION macro instructions. The size, structure, and contents of the terminal table are based on the information provided by the user in the above-listed macros. Each entry in the terminal table begins on a fullword boundary. The terminal entries are located through the address portion of the entries in the termname table. TTABLE is specified once and defines the limits of the table. One TERMINAL macro instruction is issued to create each single or group entry. OPTION macro instructions and data supplied by the TERMINAL and TPROCESS operands caused storage to be allocated for any option fields to be included in the option table for a terminal entry. The option fields can contain information needed to perform various optional functions provided by TCAM or the user. The initial contents of each option field are specified by the TERMINAL or TPROCESS macro instruction that defines the entry. TLIST defines a distribution or cascade entry. TPROCESS creates an entry for an application program. LOGTYPE creates an entry for logging messages.

If the user codes an OPTION macro, three fields in the terminal table entry are initialized, and bit 6 in the TRMSTATE field is set to 1. The TRMOPNO field contains the number of option fields specified for the entry. The option offsets are positional in nature, and the number of offsets is equal to all the offsets up to and including the last option specified by the user. The next field, TRMOPTBL, contains the offset to the beginning of the option table data for this terminal entry. The third field, TRMOPT, is the first of the actual option offsets to the option table data, the beginning of which is pointed to by the TRMOPTBL field. Each option offset is a one-byte index to the corresponding option table data. There is an option offset for each possible option up to and including the last option specified for this terminal entry. If a particular option within that span is omitted, that option offset field is initialized to X'FF'.

The device-dependent fields of an entry in the terminal table are used to indicate such information as the dial digits or addressing characters of the terminal. The specific type of information in these fields is noted in the two bytes of the device-dependent field flags field (TRMDEVFL) of the terminal table. The actual entries in the device-dependent fields consist of one byte, which contains the length of the entry, followed by the actual information. The location of the device-dependent field is indicated by the bit settings in the first byte of the terminal table. If bit 6 (TRMOPTFN) in the status byte (TRMSTATE) is off, the device-dependent field is located at +17 (X'11') in the table. If bit 6 is on, indicating that there are option offset fields in the table, the device-dependent field starts at location 20 (X'14') plus the value in the number of option offsets field (TRMOPNO). Each option offset is one byte long, and the first option offset is located at offset 20 in a terminal entry; the device-dependent field starts immediately after the last option offset.

There is one terminal entry for each terminal in the system, and each terminal table entry is referred to by a pointer from the termname table, and each terminal entry beings on a fullword boundary.

Single Entry

A *single entry* in the terminal table defines a single terminal or component. A single entry must be defined for each terminal or component that can enter only, accept only, or both enter and accept messages (except for a terminal in a group entry). If a terminal component is to be selected individually, the component must have a separate single entry.

Bits 0 through 2 of byte 0 of the control information field are set to binary 000 to indicate a single (or group) entry. If there is no option area for an entry, the offset and count fields are omitted. The required selection sequence field contains the selection characters for the terminal and, if it is a switched terminal, its telephone number and the number of dial digits.

A single entry in the terminal table is defined by a TERMINAL macro.

Group Entry

A *group entry* represents a prespecified group of terminals on a line that has special equipment to permit simultaneous transmission of a message to the group. A single set of addressing characters is used to contact the group. Several combinations of prespecified terminals can be grouped for this purpose. Each group has a group terminal name and a corresponding group entry in the terminal table. A group entry in the terminal table has the same format as a single entry, except that, since the entry is for output transmissions only, the input sequence counter field is not used.

A group entry is defined by a TERMINAL macro.

Distribution Entry

A *distribution entry* contains a list of pointers to single, process, or group entries. The pointers are grouped under the entry name. When a message contains a distribution entry name as its destination code, TCAM sends the message by separate transmissions to all destinations indicated by the list. Each terminal on the list must have a corresponding single or group entry in the terminal table. The TCAM MCP cannot receive messages through the distribution list method.

The format of a distribution entry in the terminal table is the same as that for a single entry, except that the setting of the status bits is binary 010, and the input sequence number field (bytes 4 and 5) contains a count of the entries in the list. Two-byte pointers to the single or group entries that make up the list follow this count field.

For distribution and cascade entries, bytes 1 to 3 contain the address of a distribution or cascade destination QCB.

A distribution entry in the terminal table is defined by a TLIST macro.

Cascade Entry

A *cascade entry* is identical in appearance to a distribution entry, but is handled differently. The message is queued for the available terminal that has the fewest messages queued for it in the list. An available terminal is one that is currently capable of accepting a message. The terminal must not be held. To be available, a dial terminal must not be involved in a time delay. If more than one of the available terminals have the same number of messages queued and that number is the fewest number of messages queued, the message is sent to the first available terminals on the list. If the message cannot be sent to any terminal at this time, it is queued for the first terminal in the list. The TCAM MCP cannot receive messages through a cascade list.

The format of a cascade entry is the same as that for a single entry, except that the setting of the status bits is binary 010 and the input sequence number field contains a count of the entries in the list. Two-byte pointers to the single or group entries that make up the list follow this count field.

A cascade entry in the terminal table is defined by a TLIST macro.

Process Entry

A *process entry* in the terminal table represents a queue of messages for an application program. There must be a process entry for each queue to which an application program can issue a GET or READ macro and at least one for all the PUT or WRITE macros from the same application program. The format for a process entry in the terminal table is the same as that for a single entry, except that the setting of the status bits is binary 001. Also, for a GET/READ operation, bytes 1 to 3 contain the address of the destination QCB.

A process entry is defined by a TPROCESS macro.

Logtype Entry

A *logtype entry* in the terminal table represents a queue of messages for a logging medium. The setting of the status bits for a log entry is binary 011.

A logtype entry is defined by a LOGTYPE macro.

Line Entry

A *line entry* in the terminal table defines a switched line that is used for input operations. A line entry contains the device characteristics for stations that call in on a switched line before supplying identification and for stations that call in and never supply identification data.

The format of a line entry is the same as for a single or group entry except that the setting of the status bits is binary 100.

A line entry is defined by the UTERM operand on a TERMINAL macro.

The formats of the various types of terminal entries are illustrated below; descriptions of the fields follow.

(This page left blank intentionally)

TERMINAL TABLE ENTRY TYPE

Single and Line

Offset

0	1	4	6	8	10 (A)	12 (C)	14 (E)	15 (F)	16 (10)	17 (11)	18 (12)	20 (14)	20 + n
Status byte	Destination QCB address	Input sequence number	Output sequence number	Alternate destination offset	Device dependent field flags	Number start I/Os	Number temporary errors	Intensive mode recording indicator	DCT index	Number option offsets	Option Table offset	Start of option offsets	Start of device dependent fields
TRMSTATE	TRMDESTQ	TRMINSEQ	TRMOUTSQ	TRMALTD	TRMDEVFL	TRMSIO	TRMTEMPR	TRMSENSE	TRMCHCIN	TRMOPNO	TRMOPTBL	TRMOPT	

Group

Offset

0	1	4	6	8	10 (A)	12 (C)	14 (E)	15 (F)	16 (10)	17 (11)	18 (12)	20 (14)	20 + n
Status byte	Destination QCB address	Unused X'0000'	Output sequence number	Alternate destination offset	Device dependent field flags	Number start I/Os	Number temporary errors	Intensive mode recording indicator	DCT index	Number option offsets	Option Table offset	Start of option offsets	Start of device dependent fields
TRMSTATE	TRMDESTQ		TRMOUTSQ	TRMALTD	TRMDEVFL	TRMSIO	TRMTEMPR	TRMSENSE	TRMCHCIN	TRMOPNO	TRMOPTBL	TRMOPT	

Distribution

Offset

0	1	4	6
Status byte	Distribution List QCB address	Number entries in the list	Offset to the first entry in the list
TRMSTATE	TRMDESTQ	TLISTCNT	TLISTEN

Cascade

Offset

0	1	4	6
Status byte	Cascade list QCB address	Number entries in the list	Offset to the first entry in the list
TRMSTATE	TRMDESTQ	TLISTCNT	TLISTEN

Process

Offset

0	1	4	6	8	10 (A)	12 (C)	16 (10)	17 (11)	18 (12)	20 (14)
Status byte	Process QCB address	Input sequence number	Output sequence number	Alternate destination offset	Device dependent field flags	Process Entry Work Area address	Work unit record delimiter character	Number option offsets	Option Table offset	Start of option offsets
TRMSTATE	TRMDESTQ	TRMINSEQ	TRMOUTSQ	TRMALTD	TRMDEVFL	TRMSTAT	TRMCHCIN	TRMOPNO	TRMOPTBL	TRMOPT

Logtype

Offset

0	1	4	6	8	10 (A)	12 (C)	14 (E)	15 (F)	16 (10)	17 (11)	18 (12)
Status byte	Destination QCB address	Unused X'0000	Unused X'0000'	Unused X'0000'	Device dependent field flags	Unused X'0000'	Unused X'00'	Unused X'00'	Unused X'00'	Unused X'00'	Buffer size (2 bytes)
TRMSTATE	TRMDESTQ				TRMDEVFL						

Insert foldout page 577 at end of book.

IEDQTRM

0 (0) **TRMSTATE** Status Byte	1 (1) **TRMDESTQ** Destination QCB Address		
4 (4) **TRMALNCT** Automatic Line Number Count			
TRMINSEQ Input Sequence Number **TLSTCNT** TLIST Count of Entries		6 (6) **TRMOUTSQ** Output Sequence Number **TLISTEN** First TLIST Entry Address	
8 (8) **TRMALTD** Alternate Destination Termname Table Offset		10 (A) **TRMDEVFL** Device Dependent Field Flags	
12 (C) **TRMSTAT** Error Statistics **TRMSIO** Start I/O Count		14 (E) **TRMTEMPR** Temporary Error Count	15 (F) **TRMSENSE** Intensive Mode Recording Indicator
16 (10) **TRMCHCIN** DCT Index	17 (11) **TRMOPNO** Option Field Count	18 (12) **TRMOPTBL** Option Table Offset	
20 (14) **TRMOPT** Start of Option Offsets			

Offset	*Name*	*Bytes*	*Description*		
0 (0)	TRMSTATE	1	Status byte; bit definitions are:		
		Name	*Bit*	*Value*	*Meaning*
			0-2		Type of entry:
				B'000'	Terminal, single, or group
				B'001'	Process
				B'010'	Cascade or Distribution list
				B'100'	Line
				B'101'	Log
			3		Reserved
		TRMACPTN	4	X'08'	Terminal can accept an entry for processing
		TRMHELDN	5	X'04'	Terminal is held or a process entry is specified SYNC=YES
		TRMOPTEN	6	X'02'	Option fields are used
		TRMSCNYN	7	X'01'	Secondary operator control terminal
1 (1)	TRMDESTQ	3	Address of the destination QCB for the entry or of the distribution or cascade entry QCB.		
4 (4)	TRMINSEQ	2	Input sequence number		
4 (4)	TLISTCNT	2	Count of entries in a TLIST		
6 (6)	TRMOUTSQ	2	Output sequence number		

Offset		Name	Bytes	Description		
6	(6)	TLISTEN	2	First entry in a TLIST		
8	(8)	TRMALTD	2	Termname table offset of the alternate destination		
10	(A)	TRMDEVFL	2	Device-dependent field flags to indicate which fields are present		
			Name	*Bit*	*Value*	*Meaning*
				0	X'8000'	BUFSIZE specified
				1	X'4000'	Dial digits present
				2	X'2000'	Addressing characterspresent
				3	X'1000'	BLOCK specified
				4	X'0800'	SUBBLOCK specified
				5	X'0400'	Transparent block length specified
				6	X'0200'	BFDELAY specified
				7	X'0100'	Display device
			TRMCONC	8	X'0080'	Concentrator or a terminal attached to a concentrator
			TRMLMD	9	X'0040'	LMD=YES specified
				10	X'0020'	RETRY specified
				11-15	Reserved	
12	(C)	TRMSTAT		Error statistics		
12	(C)	TRMSIO	2	Number of START I/O instructions		
14	(E)	TRMTEMPR	1	Number of temporary errors		
15	(F)	TRMSENSE	1	Intensive mode recording indicator		
16	(10)	TRMCHCIN	1	Index to the device characteristics table for this entry		
17	(11)	TRMOPNO	1	Number of option fields for this entry		
18	(12)	TRMOPTBL	2	Offset to the option table for this entry		
20	(14)	TRMOPT	1	Start of option offsets		

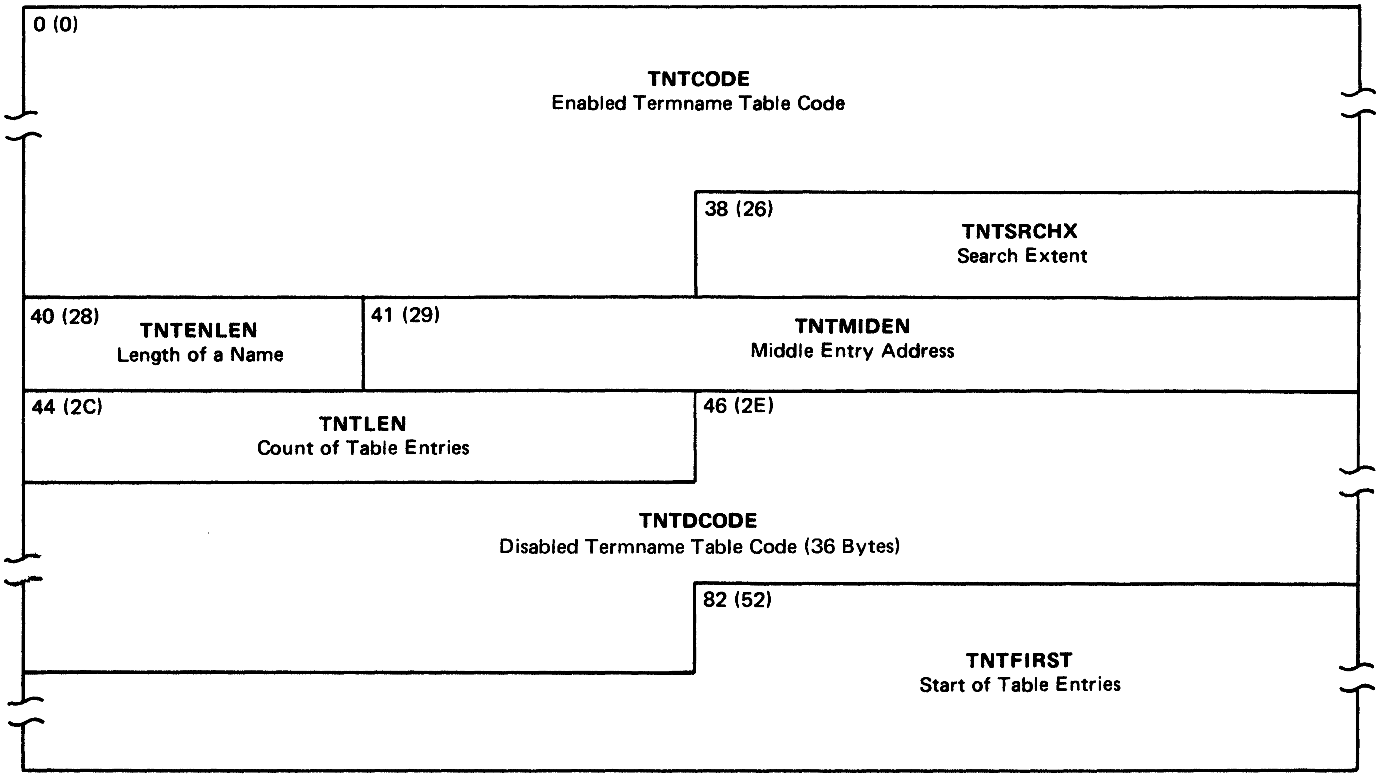
Termname Table

The termname table has an entry that contains the name and terminal entry address for each terminal, terminal component, application program, list of terminals, and logging media in the TCAM system. These entries are generated at assembly time from the TERMINAL macros in the order in which the macros are coded. At MCP initialization time the entries are sorted into collating sequence to permit binary searches for locating terminal names and for finding terminal-dependent information.

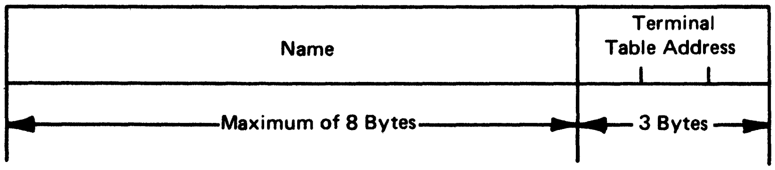
The beginning of the termname table contains code (the Termname Table Code-IEDQTNT) that is used to convert the relative position field in the invitation list to the address of the corresponding entry in the terminal table. The code can be executed as a subroutine by other TCAM modules. Following the code there are two bytes of control information for the Binary Search routine. The next fields in the termname table contain the number of bytes in the name of an entry, the address of the middle entry in the table, and the total number of entries in the table. Each entry consists of the terminal name and the three-byte address of the terminal table entry for that terminal. The length of the field for the terminal name is determined by the longest terminal name; each terminal name field is as long as the longest name (the names are padded with blanks on the right, if needed).

The address of the termname table is in the AVTRNMPT field of the AVT. However, the individual termname table entries are referred to by the relative position offsets that precede the control data in each invitation list. When a TCAM module needs to find a specific entry in the terminal table, the module activates the termname table code, which translates the relative position offset in the invitation list to the address of the corresponding terminal table entry.

IEDQTNTD



Format of a Termname Table Entry:



Offset		*Name*	*Bytes*	*Description*
0	(0)	TNTCODE	38	Enabled termname table code (IEDQTNT) to convert the relative offset to a terminal table address
38	(26)	TNTSRCHX	2	Binary search extent—used by the Binary Search routine (IEDQA1)
40	(28)	TNTENLEN	1	Length in bytes of the name field of an entry
41	(29)	TNTMIDEN	3	Address of the middle entry in the termname table—used by the Binary Search routine (IEDQA1)
44	(2C)	TNTLEN	2	Total number of entries in the termname table
46	(2E)	TNTDCODE	36	Disabled termname table code
82	(52)	TNTFIRST		The beginning of the termname-table entries

Test Event Control Block (TTECB)

The test event control block (TTECB) is used by modules IEDQWS and IEDQW44 in reporting the results of an I/O operation back to the OLT. This block contains the CSWs, condition codes, I/O addresses, and sense information passed by TOTE; it is of variable length.

TTECB

0 (0) **TECBFDCT** Number of Event Fields	1 (1) **TECBFDLN** Length of Event Fields	2 (2) **TECBSNLN** Length of Sense Field	
4 (4) **TECBFLGS** Flags	5 (5) **TECBSNCT** No. of Sense Fields	6 (6) **TECBSNOC** No. of Senses Occurred	7 (7) **TECBEVOC** No. of Events Occurred
8 (8) **TECBFDO1** 1st Event Field			
As many additional identical fields as specified in **TECBFDCT**			

Offset		*Name*	*Bytes*	*Description*
0	(0)	TECBFDCT	1	Number of event fields
1	(1)	TECBFDLN	1	Length of event fields
2	(2)	TECBSNLN	2	Length of sense field
4	(4)	TECBFLGS	1	Flags
5	(5)	TECBSNCT	1	Number of sense fields
6	(6)	TECBSNOC	1	Number of senses occurred
7	(7)	TECBEVOC	1	Number of events occurred
8	(8)	TECBFDO1	4	1st event field

(This page left blank intentionally)

TOTE Resource Control Block

The TOTE resource control block (RESPL) is used in communication between TOTE's resident module (IEDQWA) and the resource management module (IEDQWB) for the initialization of an on-line test. RESPL contains the addresses of the first and the last OLTCB in the active queue, the last TOTE termname table entry, the current storage blocks, TOTE's extended area used for dummy TTEs or QCBs, the end-of-task exit routine, the TOTE service manager routine, and the queue handlers routine. It is assembled as part of the resident module, IEDQWA.

RESPL

Offset	Field	Description
0 (0)	**RESTECBS**	Parm List for TOTE WAIT
4 (4)		
5 (5)		
8 (8)	**RESTECB1**	Subtask Request ECB
12 (C)	**RESOBQCB**	OLTCB Queue Control Block
	RESOBFWD	OLTCB Queue Forward Pointer
16 (10)	**RESOBBKW**	OLTCB Queue Backward Pointer
20 (14)	**RESTTLST**	Last TOTE TNT Entry Address
24 (18)	**RESBKTOT**	Total TOTE Storage Blocks
26 (1A)	**RESBKAVL**	Current Free Storage Blocks
28 (1C)	**RESEFQCB**	Extended Area Free Queue Control Block
32 (20)	**RESTNTCT**	No. of TNT Entries
33 (21)	**RESDMTTE**	Pointer to TOTE Extended Area for TTEs and QCBs
36 (24)	**RESETXRA**	End of Task Exit Routine Address
40 (28)	**RESSMGRA**	Service Manager Entry Address
44 (2C)	**RES#OLTS**	Max. Simult. OLTs
45 (2D)	**RESWBFNC**	IEDQWB Function Request Code
46 (2E)	**RESFLGS**	TOTE Resident Flags
47 (2F)		Unused
48 (30)	**RESTRMQ**	TOTE TRM Queue Control Block
52 (34)		Unused
53 (35)	**RESTNTPT**	Address of Start of TOTE TNT Entries
56 (38)	**RESQHTBL**	Queue Handler
	RESREMFR	Address to Remove Element From Front Queue Handler
60 (3C)	**RESREMEL**	Address to Remove Specified Element Queue Handler

RESPL

Offset	Field
64 (40)	**RESADDND** Address to Add Element to End Queue Handler
68 (44)	**RESADDAF** Address to Add Element After Another Specified Element Queue Handler

Offset		*Name*	*Bytes*	*Description*
0	(0)	RESTECBS	4	Parm list for TOTE WAIT
4	(4)		1	
5	(5)		3	
8	(8)	RESTECB1	4	Subtask request ECB
12	(C)	RESOBQCB		OLTCB queue control block
12	(C)	RESOBFWD	4	OLTCB queue forward pointer
16	(10)	RESOBBKW	4	OLTCB queue backward pointer
20	(14)	RESTTLST	4	Last TOTE TNT entry address
24	(18)	RESBKTOT	2	Total storage blocks for TOTE
26	(1A)	RESBKAVL	2	Current free storage blocks
28	(1C)	RESEFQCB	4	Extended area free queue control block
32	(20)	RESTNTCT	1	Total number of TNT entries
33	(22)	RFSDMTTE	3	Pointer to TOTE extended area for TTEs and QCBs
36	(24)	RESETXRA	4	End of task exit routine address
40	(28)	RESSMGRA	4	Service manager entry address
44	(2C)	RES#OLTS	1	Maximum simultaneous OLTs
45	(2D)	RESWBFNC	1	IEDQWB function request code
46	(2E)	RESFLAGS	1	TOTE resident flags
			Name	*Bits* *Value* *Meaning*
			RESINIT	0 X'80' TOTE initialized flag
47	(2F)		1	Unused
48	(30)	RESTRMQ	4	TOTE TRM queue control block
52	(34)		1	Unused
53	(35)	RESTNTPT	3	Address of start of TOTE TNT entries
56	(38)	RESQHTBL		Queue handlers
56	(38)	RESREMFR	4	Address to remove element from front queue handler
60	(3C)	RESREMEL	4	Address to remove specified element queue handler
64	(40)	RESADDND	4	Address to add element to end queue handler
68	(44)	RESADDAF	4	Address to add element after another specified element queue handler

TSO TSINPUT Control Block

The TSO TSINPUT control block is generated as a queue control block (QCB) for the time-sharing subtask and as an extension of the address vector table (AVT) for time-sharing support.

The DSECT names of the TSINPUT fields are shown in the following layout. A more detailed description of the fields and the data they contain follows the data area layout.

TSINPUT

<table>
<tr><td>0 (0)
TSIFLAG</td><td>1 (1)
TSIELCHN
QCB Element Chain</td></tr>
<tr><td>4 (4)
TSIPRI
Priority</td><td>5 (5)
TSLINK
Pointer to Next QCB in Chain</td></tr>
<tr><td colspan="2">8 (8)
TSINPUT
Address of IEDAYI (TSINPUT)</td></tr>
<tr><td colspan="2">12 (C)
TSISTAE
Address of IEDAYT (STAE)</td></tr>
<tr><td colspan="2">16 (10)
TSIEDIT
Address of IEDAYE (Edit Routine)</td></tr>
<tr><td colspan="2">20 (14)
TSIHANG
Address of IEDAYH (Hangup Routine)</td></tr>
<tr><td colspan="2">24 (18)
TSISIMAT
Address of IEDAYS (Simultaneous Attention)</td></tr>
<tr><td colspan="2">28 (1C)
TSISCHED
Address of IEDAYZ</td></tr>
<tr><td colspan="2">32 (20)
TSIBUFQ
Chain of Held TCAM Buffers</td></tr>
<tr><td colspan="2">36 (24)
TSITSBQ
Chain of TSBS Holding TCAM Buffers</td></tr>
<tr><td colspan="2">40 (28)
TSIABEND
ECB Posted When TSC Abends</td></tr>
<tr><td colspan="2">44 (2C)
TSIMSGEN
Address of IEDAYM (MSGEN Routine)</td></tr>
<tr><td colspan="2">48 (30)
TSIHALT
Address of IEDAYF (Halt I/O)</td></tr>
</table>

QCB For Asynchronous Time
Delay QCB Removal Routine

52 (34)	**TSIDYQCB**	QCB is Always Posted to Itself
	TSIDYQFG	Flag Byte
56 (38)	**TSIDYPRI**	Priority
57 (39)	**TSIDYLINK**	Pointer to Next Element
60 (3C)	**TSIDYDLY**	Address of IEDAYY
64 (40)	**TSIABLST**	
64 (40)	**TSIABACT**	AQCTL Action Code
65 (41)	**TSIABWDI**	First Word of Parm List

QCB for TS
Input Abend Interface

68 (44) **TSIABQCB**	
TSIABWD2 Second Word of Parm List	
72 (48) **TSIABPTY** Special Element Priority	73 (49) **TSIABLNK** Special Element Link Field
76 (4C) **TSIABVCN** STCB Address	
80 (50) **TSIDEST** Address of TCAM Destination Scheduler	
84 (54) **TSICPBI** Address of TCAM CPB Initialization Routine	
88 (58) **TSICPBC** Address of TCAM CPB Cleanup Routine	
92 (5C) **TSIATTEN** Address of IEDAYA (Attention Routine)	
96 (60) **TSITSDST** Address ot TSO Destination Scheduler	
100 (64) **TSI3270** Address of IEDAYB (3270)	

Offset		*Name*	*Bytes*	*Description*		
0	(0)	TSIFLAG	1	Flag byte		
			Name	*Bits*	*Value*	*Meaning*
			TSIQCB	6	X'02'	Flag indicating a QCB
			TSIPOST	4	X'08'	QCB posted to itself
1	(1)	TSIELCHN	3	QCB element chain		
4	(4)	TSIPRI	1	Priority		
5	(5)	TSILINK	3	Pointer to next QCB in chain		
8	(8)	TSINPUT	4	Address of IEDAYI (TSINPUT)		
12	(C)	TSISTAE	4	Address of IEDAYT (STAE)		
16	(10)	TSIEDIT	4	Address of IEDAYE (Edit Rtn)		
20	(14)	TSIHANG	4	Address of IEDAYH (Hangup)		
24	(18)	TSISIMAT	4	Address of IEDAYS (Simult. attn)		
28	(1C)	TSISCHED	4	Address of IEDAYZ (Scheduler)		
32	(20)	TSIBUFQ	4	Chain of held TCAM buffers		
36	(24)	TSITSBQ	4	Chain of TSBS holding TCAM buffers		
40	(28)	TSIABEND	4	ECB posted when TSC abends		
44	(2C)	TSIMSGEN	4	Address of IEDAYM (MSGEN Rtn)		
48	(30)	TSIHALT	4	Address of IEDAYF (Halt I/O)		
52	(34)	TSIDYQCB	4	QCB is always posted to itself		
52	(34)	TSIDYQFG	4	Flag byte		
			Name	*Bits*	*Value*	*Meaning*
			TSIDYQB	6	X'02'	Flag indicating a QCB
			TSIDYPOS	4	X'08'	QCB posted to itself
56	(38)	TSIDYPRI	1	Priority		
57	(39)	TSIDYLNK	3	Pointer to next element		
60	(3C)	TSIDYDLY	4	Address of IEDAYY		
64	(40)	TSIABLST				
64	(40)	TSIABACT	1	AQCTL action code		
65	(41)	TSIABWD1	3	First word of parameter list		
68	(44)	TSIABQCB				
68	(44)	TSIABWD2	4	Second word of parameter list		
72	(48)	TSIABPTY	1	Special element priority		
73	(49)	TSIABLNK	3	Special element link field		
76	(4C)	TSIABVCN	4	STCB address		
80	(50)	TSIDEST	4	Address of TCAM destination scheduler		
84	(54)	TSICPBI	4	Address of TCAM CPB initialization routine		
88	(58)	TSICPBC	4	Address of TCAM CPB cleanup routine		

Offset		*Name*	*Bytes*	*Description*
92	(5C)	TSIATTEN	4	Address of IEDAYA attention routine
96	(60)	TSITTSDST	4	Address of TSO destination scheduler
100	(64)	TSI3270	4	Address of IEDAYB (3270)

Section 6: Diagnostic Aids

SCB Error Word Usage by Module

SCBERR1 (Byte 0)

Bit	*Bit Indication (On/Off)*	*Module Action*
0	Is/is not an incomplete header	Checked by IEDQAT and IEDQA4. Checked by IEDQBD for IN/OUT message macro instructions. Set by IEDQA4. Checked by IGCD910D.
1	Is/is not an invalid origin	Checked the same as bit 0. Set by IEDQAM.
2	TSO is not/is in system	Checked the same as bit 0.
3	Is/is not high sequence	Checked the same as bit 0. Set by IEDQAH.
4	Is/is not low sequence	Checked the same as bit 0. Set by IEDQAH.
5	Message is not/is sent/received	Checked the same as bit 0.
6	Are not/are sufficient buffers	Checked the same as bit 0. Set by IEDQAK.
7	Is/is not a cutoff error	Checked the same as bit 0.
	RVI to a selection device for BSC buffered terminals	Set by Line End Appendage. Checked by the user-coded macros.

SCBERR2 (Byte 1)

Bit	*Bit Indication (On/Off)*	*Module Action*
0	Main storage minimum is/is not exceeded	Set by IEDQBD from AVTSYSER. Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D.
1	Main storage maximum is/is not exceeded	Set by IEDQBD from AVTSYSER. Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D.
2	Error is/is not in a dynamic translation operation	Set and checked the same as bit 1.
3	Is/is not automatic line numbering	Set and checked the same as bit 1.
4	TOTE is not/is in the system or a TOTE request is not honored	Set by IEDQAA. Checked by the error macros.
5	BSC abort sequences are/are not received	Set by Line End Appendage. Checked by the error macros.
6	Forward terminal error	Set by IEDQA4 and IEDQA5. Checked by the INMSG macro.
7	Message is/is not SOH% E, C, or R	Set by IGE0904H. Checked by the error macros.

SCBERR3 (Byte 2)

Bit	*Bit Indication (On/Off)*	*Module Action*
0	Message is lost/processed	Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D. Set by IEDQFA and IEDQFQ for a lost message.
1	Terminal ID is invalid/valid	Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D.

Bit	*Bit Indication (On/Off)*	*Module Action*
2	Terminal is inoperative/operative	Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D.
3	Simulated attention is/is not received	Checked the same as bit 2.
4	User error has/has not occurred	Checked the same as bit 2.
5	Is/is not format error in BSC message	Checked the same as bit 2.
6	Is/is not hardware attention	Checked the same as bit 2. Set by IGG019RO.
7	Is/is not unit exception	Checked the same as bit 2. Set by IGE0504G and IGE0204G.

SCBERR4 (Byte 3)

Bit	*Bit Indication (On/Off)*	*Module Action*
0	Is/is not selection error	Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D. Set by IGG019RQ.
1	Is/is not error during text transfer	Checked by IEDQAA. Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D. Set by IGE0004H, IGE0104G, IGE0104H, IGE0204H, IGE0504G, and IGG019R0.
2	Is/is not error in connect/disconnect	Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D. Set by IGE0304G.
3	Is/is not terminal/line error	Checked by IEDQBD for IN/OUT message macro instructions. Checked by IGCD910D. Set by IGE0504G.
4	Reserved	
5	Is/is not error in control. unit	Checked the same as bit 3. Set by IGE0104G, IGE0104H, IGE0204G, and IGE0004G.
6	Is/is not channel error	Checked the same as bit 3. Set by IGE0104G, IGE0804G, and IGE0804H.
7	Is/is not undefined error	Checked the same as bit 3. Set by IGE0504G and IGE0504H.

LCB Status Byte Usage by Module

LCBSTAT1

Bit	*Bit Indication (On/Off)*	*Module Action*
0	Recall being/not being performed	Checked, turned off, and reset by IEDQBD. Checked by IEDQFA and IGCD910D. Cleared by IEDQAA, IEDQAS, and IEDQAT.
1	Line is/is not in control mode	Set by IGG019R1. Checked and reset by IEDQKA, IEDQKB, IEDQKC, IEDQKD, and IEDQKE. Checked by IGCD910D. Cleared by IEDQAA.
2	Operator control is not/is immediate	Set by IEDQHK. Checked and cleared by IEDQAA. Checked by IGCD910D, IGG019R1, IGG019R3, IGG019R4, IGG019Q1, IGG019Q6, and IGG019Q7.
3	Is/is not initiate mode	Reset by IEDQBD. Checked by IEDQFA, IEDQHM, and IGCD910D. Checked and cleared by IEDQAA.
4	Is/is not continue/reset operation	Set by IEDQCU. Checked by IGCD910D. Cleared by IEDQFA and IEDQAA.
5	Line is/is not free	Checked by IGCD910D, IGCZ010D, IEDQHK, and IEDQNK. Cleared by IEDQAA.
6	Line is/is not receiving	Set by IGCV310D. Checked by IEDQGA, IGCZ010D, IEDQNK, IEDQAS, IEDQFA, and IGCD910D. Cleared by IEDQAA.
7	Line is/is not sending	Set by IGG019R4, IGG019Q6 and IGG019Q7. Checked by IEDQAS, IEDQAG, IEDQAN, IEDQAW, IEDQA4, IEDQBD, IGCZ010D, IGCD910D, IEDQFA, IEDQNK, IEDQHM, IEDQGA, and IGG019RN. Checked and cleared by IEDQAA.

Notes: *If both bits 6 and 7 are off, the line is inoperative. When a stop line function is being performed, IEDQHK sets LCBSTAT1 equal to X'00'. Also, IEDQAA and IGC0710D set LCBSTAT1 to X'00' when TOTE asks for control; IGCV310D, IGCV110D, and IGCV210D test for this condition.*

LCBSTAT2

Bit	*Bit Indication (On/Off)*	*Module Action*
0	I/O trace is/is not active for this line	Set and checked by IGCM610D. Checked by IGCD910D, IGG019R0, IGG019Q2, IGG019Q3, IGG019Q4, and IGG019Q5.

Bit	*Bit Indication (On/Off)*	*Module Action*
1	Is/is not MSGGEN/start-up message	Turned off by IEDQBD. Checked by IGCD910D, IEDQKA, IEDQKB, IEDQKC, IEDQKD, IEDQKE, IGG019R0, IGG019Q2, IGG019Q3, IGG019Q4 and IGG019Q5
2	EOT from a buffered terminal without/with EOM	Checked by IGCD910D.
3	Send priority switch is/is not set by the Send Scheduler	Checked by IGCD910D.
4	Negative response to polling is/is not received	Checked by IGCD910D, IGG019R1, IGG019R4, IGG019Q6 and IGG019Q7. Set by IEDQHK, IGG019R0, IGG019Q2, IGG019Q3, IGG019Q3, IGG019Q4, and IGG019Q5. Reset by IEDQKA, IEDQKB, IEDQKC, IEDQKD, and IEDQKE
5	Line is/is not BSC	Checked by IGG019RN,IGG019R0, IGG019Q2, IGG019Q3, IGG019Q4, IGG019Q5, IGG019Q6, IGG019Q7, IGG019R4, IGCD910D, and IEDQKA.
6	Is/is not a dial LCB	Checked by IEDQAG, IGCD910D, IGCV210D, IGCV410D, IGCM410D, IEDQHK, IEDQKA, IGGO19R1, and IGG019R4.
7	Do/do not owe a terminal a response	Set by IGG019R0, IGG019Q2, IGG019Q3, IGG019Q4 and IGG019Q5. Checked by IEDQAK, IGCD910D, IEDQKA, IEDQKB, IEDQKC, IEDQKD, and IEDQKE. Checked and cleared by IEDQA4.

Table of Message Origins

This table lists each of the messages generated by the TCAM executable modules. The originating module names and the message routing codes are included by each message.

Routing Codes:

* This routing code indicates that the message must be routed back to the console that initiated the request.

1 MASTER CONSOLE.
This routing code is for messages that must be sent to the master console because some action is required by the master console operator, or because the message contains information considered critical to the continued operation of the system.

2 MASTER CONSOLE INFORMATIONAL.
This routing code is for informational messages to the master console operator. Informational messages usually require no action from the operator. If they do, that action should be at the operator's discretion.

8 TELEPROCESSING CONTROL.
This routing code is for messages relating to teleprocessing.

10 SYSTEM ERROR/MAINTENANCE.
This routing code is used for any message that indicates a system error or an incorrectable I/O error, or any message associated with system maintenance.

11 PROGRAMMER INFORMATION.
This routing code is for messages of interest to the programmer. The message is sent to an operator console and not to the system output device.

	Message	*Origin*	*Routing Code*
IED001I	TCAM JOB jobname, stepname, procstepname ADDRESS OF AVT address	IEDQOB	2,11
IED002A	SPECIFY TCAM PARAMETERS	IEDQOB	1
IED003A	INVALID KEYWORD keyword	IEDQOB	1
IED004A	REQUIRED PARAMETER MISSING. SPECIFY xxx	IEDQOB	1
IED005A	MSUNITS (M) SPECIFICATION NOT PERMITTED. CONTINUE RESPONSE	IEDQOB	1
IED006A	INVALID OPERAND ON KEYWORD. RESPECIFY keyword	IEDQOB	1
IED007I	termname IS AN ILLEGAL DESTINATION	IEDQOA	11

	Message	*Origin*	*Routing Code*
IED008I	TCAM OPEN ERROR xxx=y IN DCB zzz descriptor	IGG01930 IGG01931	11
IED009I	CHECKPOINT DISK ALLOCATION ERROR—DATA SET NOT OPENED	IGG01942	11
IED010I	CHECKPOINT—INSUFFICIENT CORE {ENVIRON INCIDENT CKREQ name} DATA SET NOT OPEN INCIDENT RECORD IGNORED	IEDQNR IGG01941 IEDQND	11
IED011I	SYSTEM INTERVAL CANNOT BE ALTERED	IGCM410D	*
IED012I	TSO SESSION ON LINE xxx COMMAND REJECTED	IGCV110D	*
IED013I	STOP REQUEST FOR SELF—VARY COMMAND COMMAND REJECTED	IGCV110D	*
IED014I	TCAM ALREADY IN SYSTEM	IEDQOB	2,11
IED015I	TCAM AP OPEN ERROR 043-x yyy zzz	IGG01933	2,8,11
IED016I	STATION name NOT FOUND	IGCD010D IGCM010D IGCV010D IGCH010D IGCR010D	*
IED017I	LINE {ddnname,rln address} NOT OPEN	IGCD010D IGCM010D IGCV010D	*
IED018I	command field COMMAND INVALID	IGCD010D IGCM010D IGCV010D IGCH010D IGCR010D	*
IED019I	{termname grpname,rln address} ALREADY STARTED	IGCV310D IGCV410D	*
IED020I	{termname grpname,rln address} STARTED	IGCV310D IGCV410D	*
IED021I	AUTO POLL STARTED FOR grpname,rln address	IGCM210D	*

	Message	*Origin*	*Routing Code*
IED022I	AUTO POLL ALREADY STARTED FOR $\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$	IGCM210D	*
IED023I	TRACE STARTED FOR $\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$	IGCM610D	*
IED024I	TRACE ALREADY STARTED FOR $\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$	IGCM610D	*
IED025I	$\left\{\begin{array}{l}\text{termname}\\ \text{grpname,rln}\\ \text{address}\end{array}\right\}$ ALREADY STOPPED	IGCV110D IGCV210D	*
IED026I	$\left\{\begin{array}{l}\text{termname}\\ \text{grpname,rln}\\ \text{address}\end{array}\right\}$ STOPPED	IGCV110D IGCV210D	*
IED027I	AUTO POLL STOPPED FOR $\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$	IGCM210D	*
IED028I	AUTO POLL ALREADY STOPPED FOR $\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$	IGCM201D	*
IED029I	TRACE STOPPED FOR $\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$	IGCM610D	*
IED030I	TRACE ALREADY STOPPED FOR $\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$	IGCM610D	*
IED031I	statname QUEUE SIZE=integer, QUEUETYP=type, STATUS=status,...	IGCD210D	*
IED032I	$\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$ LNSTAT=status,... ERR=error,...	IGCD910D	*

IED033I	statname STATUS=status,...
	INTENSE= $\left\{\begin{array}{l}\text{sense count}\\ \text{NO}\end{array}\right\}$
	IN-SEQ=integer,OUT-SEQ=integer
		IGCD510D	*

IED034I	statname HAS NO opfldname OPTION	IGCM810D IGCD810D	*
IED035I	statname OPTION opfldname=data	IGCD810D	*
IED036I	$\left\{\begin{array}{l}\text{grpname,rln}\\ \text{address}\end{array}\right\}$ ACTIVE= $\left\{\begin{array}{l}\text{statname...}\\ \text{NONE}\end{array}\right\}$	IGCD310D	*

	Message	*Origin*	*Routing Code*
IED037I	$\left\{\begin{matrix}\text{grpname,rln}\\\text{address}\end{matrix}\right\}$ INACTIVE= $\left\{\begin{matrix}\text{statname...}\\\text{NONE}\end{matrix}\right\}$	IGCD310D	*
IED038I	statname IS ON LINE $\left\{\begin{matrix}\text{ddname}\\\text{grpname,rln}\\\text{address}\end{matrix}\right\}$	IGCD610D	*
IED039I	NO STATIONS INTERCEPTED	IGCD410D	*
IED040I	INTERCEPTED STATIONS=statname,...	IGCD410D	*
IED041I	PRIMARY= $\left\{\begin{matrix}\text{statname}\\\text{SYSCON}\end{matrix}\right\}$	IGCM710D IGCD110D	*
IED042I	$\left\{\begin{matrix}\text{statname}\\\text{SYSCON}\end{matrix}\right\}$ ALREADY PRIMARY	IGCM710D	*
IED043I	SECONDARY=statname	IGCD110D	*
IED044I	statname NOT ELIGIBLE FOR PRIMARY	IGCM710D	*
IED045I	SYSTEM INTERVAL ALREADY ACTIVE	IGCM410D	*
IED046I	LINE FOR statname IS OUTPUT ONLY STATION	IGCV410D	*
IED047I	SYSTEM INTERVAL IS data	IGCM410D	*
IED048I	POLLING DELAY FOR statname IS data	IGCM410D	*
IED049I	OLT CONTROLS LINE line COMMAND REJECTED	IGCV310D	*
IED050I	statname OPTION opfield MODIFIED	IGCM810D	*
IED051I	statname SET FOR HOLD, SEQ-OUT=integer	IGCH010D	*
IED052I	statname ALREADY SET FOR HOLD	IGCH010D	*
IED053I	statname ALREADY RELEASED	IGCR010D	*
IED054I	statname RELEASED,SEQ-OUT=integer	IGCR010D	*
IED055I	I/O TRACE CANNOT BE ALTERED	IGCM610D	*
IEDQ56I	termname opfldname DATA FORMAT INVALID	IGCM810D	*
IED057I	address NOT CAPABLE OF AUTO POLL	IGCM210D	*
IED058I	$\left\{\begin{matrix}\text{grpname,rln}\\\text{address}\\\text{statname}\end{matrix}\right\}$ SENSECOUNT=count, SETTING=sense	IGCM510D	*

	Message	*Origin*	*Routing Code*
IED059I	$\left\{\begin{matrix}\text{grpname,rln}\\\text{address}\end{matrix}\right\}$ LIST STATUS=status	IGCD710D	*
IED060I	statname CANNOT BE HELD	IGCH010D	*
IED061I	POLLING DELAY FOR statname CANNOT BE ALTERED	IGCM410D	*
IED062I	statname OPTION opfldname CANNOT ACCEPT SPECIFIED DATA	IGCM810D	*
IED063I	CLOSEDOWN IN PROGRESS COMMAND REJECTED	IGC0010D	*
IED064I	LINE addr CONTROL UNIT NOT OPERATIONAL	IGE0204G	8
IED065I	INITIALIZATION ERROR return code	IEDQOA	2,11
IED067I	TCAM INITIALIZATION BEGUN	IEDQXA	2,11
IED068I	UNABLE TO OPEN IEDQDATA	IEDQXA	11
IED069I	INVALID KEYLEN FOR IEDQDATA	IEDQXA	11
IED070I	IEDQDATA DOES NOT SPECIFY CONTIG SPACE IN CYLINDERS	IEDQXA	11
IED071I	UNEQUAL PRIMARY AND SECONDARY EXTENTS ON IEDQDATA	IEDQXA	11
IED072I	I/O ERROR ON IEDQDATA	IEDQXA	2,10,11
IED074I	TCAM INITIALIZATION COMPLETE	IEDQXA	2,11
IED075I	END OF EXTENT. RECORD COUNT IS number, TIME IS time SEC	IEDQXA	11
IED076I	TCAM NON-REUSABLE DISK THRESHOLD CLOSEDOWN	IGG019RC	2,11
IED077I	termname opfldname DATA CHARACTER INVALID	IGCM810D	*
IED078I	DLQ TERM ERROR	IEDQOA	11
IED079I	ENDING STATUS NOT RECEIVED FROM LINE address—LINE UNAVAILABLE	IGG01948	8
IED080I	START OF TCAM SYSTEM DELAY	IEDQHI	2
IED081I	END OF TCAM SYSTEM DELAY	IEDQHI	2
IED082I	CHECKPOINT DISK ERROR—DATA SET NOT OPENED	IGG01942	11

	Message	*Origin*	*Routing Code*
IED083I	CHECKPOINT DISK ERROR—RECOVERY FROM PREVIOUS RECORD	IGG01942	11
IED084I	CHECKPOINT DISK ERROR—RECOVERED	IEDQNQ	11
IED085I	CHECKPOINT DISK ERROR— CKREQ RECORD IGNORED INCIDENT	IEDQND IGG01944	11
IED086I	CHECKPOINT DISK ERROR—ENVIRONMENT CKREQ,name	IEDQNP	11
IED087I	CHECKPOINT DISK ERROR—CONTROL RECORD	IGG012041 IEDQNQ	11
IED088I	termname ON DIAL LINE—CANNOT BE VARIED	IGCV210D IGCV410D	*
IED090I	statname IS NOT SINGLE ENTRY	IGCV210D IGCV410D IGCD610D	*
IED091I	LINE FOR statname NOT OPEN	IGCD210D IGCD610D IGCV210D IGCV410D	*
IED092I	BISYNC ERROR—LINE xxx CANNOT BE STARTED	IGCV310D	*
IED093I	SET SYSTEM INTERVAL COMMAND ACCEPTED	IGCM410D	*
IED094I	CORE REQUESTED FOR ON-LINE TEST NOT AVAILABLE	IEDQND	11
IED095I	MODIFY OLT REJECTED—OLT NOT ACTIVE	IGCMA10D	8,11
IED096I	CHECKPOINT OPERATOR CONTROL NO LONGER ACTIVE COMWRITE	IEDQNA2	2,11
IED097I	TCAM IS CLOSED DOWN	IEDQNA2	2,11,*
IED098I	DCB OPEN FOR MESSAGE PROCESSING PROGRAM—jobname	IGCZ010D	2,11
IED099I	ROUTINE LOADED	IGCM910D	8
IED100I	ROUTINE DEACTIVATED	IGCM910D	8
IED101I	RESTART IN PROGRESS	IGCM910D	8
IED102I	INVALID OPERAND	IGCM910D	8

	Message	*Origin*	*Routing Code*
IED103I	ROUTINE IS ACTIVE	IGCM910D	8
IED104I	ROUTINE NOT ACTIVE	IGCM910D	8
IED105I	RETURN CODE=xxxx	IGCM910D	8
IED106I	MULTIPLE REQUEST	IGCM910D	8
IED107I	COMWRITE NOT ACTIVE	IGCM910D	8
IED109I	ROUTINE NOT DELETED	IGCM910D	8
IED127I	OLT REQUEST REJECTED, CONTROL TERMINAL UNIDENTIFIED	IEDQWC	10
IED128I	ALTERNATE PRINTER REQUESTED BY OLT ALREADY IN USE	IEDQWC	10
IED130I	OLT REQUEST REJECTED, CONTROL TERMINAL NOT OPEN	IEDQWC	10
IED132I	CAN OLT USE FOR NON-CONCURRENT MODE LINES xxx,xxx,xxx,xxx,... (up to 11 lines)	IEDQWC1 IEDQWJ2	10
IED133I	C. T. REQUESTED BY OLT ASSIGNED TO ANOTHER OLT	IEDQWC	10
IED135I	ALREADY CONFIGURED; REQUEST CHANGE FUNCTION TO MODIFY	IEDQWIA	10
IED135I	MODIFICATION/DELETION NOT PERMITTED FOR THIS DEVICE	IEDQWID	10
IED135I	UNSUPPORTED DEVICE TYPE	IEDQWID IEDQWIE	10
IED135I	INVALID LINE ADDRESS	IEDQWIA IEDQWID IEDQWI5U	10
IED135I	NAME NOT FOUND IN TCAM TERMINAL TABLE	IEDQWIA IEDQWID IEDQWIE	10
IED135I	OLD ENTRY DELETED FROM CDS	IEDQWIA IEDQWID	10
IED135I	NEW ENTRY ADDED TO CDS	IEDQWIA	10
IED135I	CONFIGURATOR STARTED	IEDQWI	10
IED135I	CONFIGURATOR COMPLETED	IEDQWI, IEDQWIA,	10

	Message	Origin	Routing Code
		IEDQWID, IEDQWIE,	
IED135I	LINE NOT OPENED	IEDQWI9	10
IED135I	OPTION ENTRY INVALID	IEDQWJ1	10
IED135I	ENTER ONE OPTION OR NONE	IEDQWJ1	10
IED135I	EPN—WHERE N IS LEVEL OF PRINTED OUTPUT WANTED	IEDQWJ1	10
IED135I	CM, NCM, NEP, AP, NAP, EXT=DATA—NNN IS A 4 DIGIT DECIMAL NUMBER	IEDQWJ1	10
IED135I	VALID OPTIONS ARE TLNNNN, NTL, ELNNNN, NEL, CP, NCP, NMI, MI	IEDQWJ1	10
IED135I	DEFAULT OPTIONS ARE CP, NTL, NEL, CM, NAP, NMI, AND EP	IEDQWJ1	10
IED135I	ON LINE TESTING ENDED	IEDQWE	10
IED135I	START OR STOP LINE FAILED—ABORT	IEDQWD	10
IED135I	***CONTROL TERMINAL ID IS nn***	IEDQWC	10
IED135I	xxxxxxxx NOT OPENED	IEDQWC1 IEDQWC2	10
IED135I	S xxxxy UNIT zzz	IEDQWE	10
IED135I	*T xxxxy UNIT zzz T xxxxy UNIT zzz	IEDQWE	10
IED135I	MACRO FUNCTION NOT SUPPORTED	IEDQWM2	10
IED135I	MACRO LEVEL NOT SUPPORTED	IEDQWM2	10
IED135I	ON LINE TESTING CANCELED	IEDQWE	10
IED135I	TRM REJECTED—270X NOT CONNECTED TO PROPER CPU	IEDQWD	10
IED135I	TEST DEVICE DOES NOT BELONG TO TCAM	IEDQWE	10
IED135I	REENTER TRM LATER—RESOURCE IN UNSHAREABLE STATE	IEDQWD	10
IED135I	TRM REJECTED—I/O ERROR LOADING CONFIGURATION DATA	IEDQWD	10
IED135I	TRM REJECTED—NO CONFIGURATION DATA	IEDQWD	10

	Message	*Origin*	*Routing Code*
IED135I	INVALID ENTRY FOR ADDITIONAL TESTS—VALID ENTRIES ARE	IEDQWJ1	10
IED135I	TRM BUFFER TOO SMALL FOR LAST ENTRY	IEDQWJ IEDQWJ1 IEDQWJ2	10
IED135I	ERROR IN TEST LOOP OR ERROR LOOP NUMBER	IEDQWC1, IEDQWJ1	10
IED135I	1060 CANNOT BE CONTROL TERMINAL FOR PROMPT OR CONFIG	IEDQWJ	10
IED135I	DIAGMSG DD CARD MISSING FROM JCL	IEDQWJ	10
IED135I	DIAGMSG OPEN FAILED	IEDQWD	10
IED135I	OPERATOR WILL ONLY ALLOW CONCURRENT MODE—TRM REJECTED	IEDQWJ2	10
IED135I	ERROR IN OPTION FIELD	IEDQWJ	10
IED135I	ERROR IN TEST FIELD	IEDQWJ	10
IED135I	ERROR IN TEST DEVICE FIELD	IEDQWJ	10
IED135I	TRM PROMPTER RUNNING	IEDQWJ	10
IED135I	PROMPTING NOT ALLOWED ON 1060, REENTER TRM	IEDQWJ	10
IED135I	DIAL TEST TERMINAL NOT ALLOWED WITH LEASED ONES	IEDQWJ	10
IED135I	INVALID RESPONSE	IEDQWI, IEDQWIA, IEDQWID, IEDQWIE IEDQWI9 IEDQWJ IEDQWJ1 IEDQWJ2	10
IED135I	INVALID TEST DEVICE ENTRY	IEDQWJ	10
IED135I	ALREADY HAVE 9 TEST DEVICES—TEST DEVICE PROMPTING FINISHED	IEDQWJ	10
IED135I	INVALID ROUTINE ENTRY	IEDQWJ	10
IED135I	INVALID TEST NAME	IEDQWJ	10
IED135I	TRM REJECTED—CONFIGURATION DATA SET CANNOT BE OPENED	IEDQWD	10

	Message	*Origin*	*Routing Code*
IED135I	ON LINE TESTING ACTIVE	IEDQWC	10
IED135I	OLT MODULE xxxxxxxx CANNOT BE LOADED	IEDQWE	10
IED135I	CLASS NOT TP, OR SUPPORTED GRAPHIC—ABORT	IEDQWD	10
IED135I	NOT ENOUGH CORE FOR SECTION xxxxxxxx	IEDQWE	10
IED136D	DO YOU WISH TO CONTINUE? (YES OR NO)	IEDQWIA, IEDQWID, IEDQWIE	10
IED136D	ARE THERE OTHER ENTRIES TO DELETE? (YES OR NO)	IEDQWID	10
IED136D	ARE THERE OTHER ENTRIES TO EXHIBIT? (YES OR NO)	IEDQWIE	10
IED136D	ENTER TYPE OF TERMINAL	IEDQWI9	10
IED136D	ARE THERE OTHER ENTRIES TO ADD? (YES OR NO)	IEDQWIA	10
IED136D	ARE THERE OTHER ENTRIES TO CHANGE? (YES OR NO)	IEDQWIA	10
IED136D	ENTER FUNCTION: ADD, CHANGE, DELETE, EXHIBIT, OR NONE	IEDQWI	10
IED136D	ENTER A LINE ADDR. OR A SYMBOLIC TERMINAL NAME	IEDQWIA, IEDQWID, IEDQWIE	10
IED136D	ENTER LINE ADDRESS (FORMAT CCU)	IEDQWI9	10
IED136D	ENTER ONE OPTION OR NONE	IEDQWJ1	10
IED136D	SYSOUT—SYSCON—SYMBOLIC NAME	IEDQWJ2	10
IED136I	ENTER ALTERNATE PRINTER LOCATION. VALID ENTRIES ARE	IEDQWJ2	10
IED136D	DO YOU WANT TO CONTINUE PROMPTING? (YES OR NO)	IEDQWJ IEDQWJ1 IEDQWJ2	10
IED136D	YOU CAN REENTER (R), CANCEL (C), OR USE TRM AS IS (GO)	IEDQWJ IEDQWJ1 IEDQWJ2	10
IED136D	ENTER NEXT MESSAGE SEGMENT	IEDQWH	10

	Message	Origin	Routing Code
IED135D	ARE THERE ANY MORE TEST DEVICES? ANSWER YES OR NO	IEDQWJ	10
IED136D	ENTER SYMBOLIC NAME OF TERMINAL OR CCU OF TCU TO BE TESTED	IEDQWJ	10
IED136D	DO YOU WANT TO BE PROMPTED? ANSWER YES OR NO	IEDQWJ	10
IED136D	INVALID RESPONSE, PLEASE ENTER YES OR NO	IEDQWJ IEDQWJ1 IEDQWJ2	10
IED136D	ENTER ALPHA CHARACTERS SEPARATED BY COMMAS FOR OTHER SELECTIONS	IEDQWJ	10
IED136D	DO YOU WANT OTHER TEST SECTIONS RUN ON THIS DEVICE? ANSWER YES OR NO	IEDQWJ	10
IED136D	ENTER ROUTINE NUMBER—EXAMPLE 1, 4-6, 9	IEDQWJ	10
IED136D	DO YOU WANT TO SELECT ROUTINES IN THIS TEST? ANSWER YES OR NO	IEDQWJ	10
IED136D	ENTER TEST TO BE RUN—FORMAT NNNNB/ANNNNB—EXAMPLE 2700A/T2700A	IEDQWJ	10
IED136D	INVALID EP LEVEL—ENTER 1, 2 OR 3	IEDQWJ	10
IED138I	ERROR SORTING DEVICE ID TABLE,xxxx	IEDQOA	11
IED139I	PRINTING STOPPED	IGG019RC	2
IED140I	TCAM DISK ERROR aa, bbbbbbbb, cccccccc cccccccc, ddd, ee, ffffff	IGG019RC	2
IED143I	gpstatname GENERAL POLL STARTED	IGC0190D	*
IED144I	gpstatname GENERAL POLL STOPPED	IGC0910D	*
IED145I	gpstatname GENERAL POLL ALREADY STARTED	IGC0910D	*
IED146I	gpstatname GENERAL POLL ALREADY STOPPED	IGC0910D	*
IED147I	gpstatname COMMAND INVALID FOR GENERAL POLL	IGC0910D	*
IED148D	IS C.U. FOR xxx CONNECTED TO THIS SYSTEM?	IEDQWD	10

	Message	*Origin*	*Routing Code*
IED148I	OLT ABEND xxxyyy	IEDQWB	10
IED149I	TOTE BUSY	IEDQWA	10
IED150D	TCAM REUSABLE Q WRAPPED—REPLY 'D' TO DUMP ENTIRE MSG DATA SET OR 'C' TO CANCEL	IEDQXC	1
IED151I	cuu tttt yy ERS z cuu xx tttt THRESHLD cuu xx tttt yy eeee zzzz yy eeee zzzz yy eeee zzzz yy eeee zzzz cuu ww tttt eeeeeee zzzz eeeeeee zzzz eeeeeee zzzz	IGE0904H	2
IED152I	CHECKPOINT BLKSIZE TOO SMALL—300 WAS USED	IGG01949	11
IED153I	CHECKPOINT BLKSIZE TOO BIG—3520 WAS USED	IGG01949	11
IED154I	TOTE CANNOT RETURN DEVICE xxx TO ORIGINAL STATUS	IEDQWE	10
IED156I	statname ON CONCENTRATOR—CANNOT BE VARIED	IGCV210D IGCV410D	11
IED157I	TCAM SYSTEM DELAY ACTIVE—HALT COMMAND REJECTED	IGCV010D	11

Register Usage Conventions in TCAM

Although each module in TCAM uses registers as necessary to perform its functions, some conventions are used in various groups of modules. For specific register usage by module refer to the microfiche of the module. The general register usage by module type follows.

Save Area Management

In TCAM, save area management occurs when a subroutine returns to the routine that called it. A save area "belongs" to a routine when that routine sets register 13 to point to a save area in the AVT. A subroutine of the routine can then store the registers of the routine in the specified save area. If a routine does not call a subroutine, it does not have a save area since it does not modify the contents of register 13.

TCAM maintains four 18-word save areas and one 10-word save area in the AVT After the standard entry linkage to a routine that uses save area management, certain words of the save area contain specific addresses:

- The second word of the save area points to the address of the save area for the calling routine.
- The third word of the save area for the calling routine has the address of the save area for the called routine.
- Register 13 has the address of the save area for the called routine.

During the standard exit linkage of a routine that uses save area management, the save area address for the calling routine is restored from the second word of the save area for the called routine. The registers of the calling routine are also restored from this area, and the calling routine can regain control.

Subroutine Linkage

TCAM uses standard subroutine linkage and requires saving and restoring registers through the SAVE and RETURN macros. These macros are coded in the following manner:

```
SAVE (14,12),,*

RETURN (14,12),T
```

Appendages

Register	*Use*
1	Request element address
2	IOB address
8	IOS register
9	IOS register
13	Save area address
14	Return address
15	Entry point

Application Program Routines

Register	*Use*
0	Termname table entry address
1	Application program work area
13	Save area address
14	Return address
15	Entry point; return code

Checkpoint Routines

Register	*Use*
2	Checkpoint work area address
3	Address of the request element this module is to process
4	Disk record address
9	AVT address
12	IEDQNF base register
14	Return address
15	Entry point; offset to the next module to gain control

Error Recovery Procedure Routines

Register	*Use*
1	Request element address
2	LCB address
4	DCB address
11	AVT address
13	Linkage for the next load module
14	XCTL register
15	Base register

Message Handling Routines

Register	*Use*
1	Parameter list register
3	SCB address
4	LCB address
6	Current buffer address
9	AVT address
12	Base register
13	Save area address
14	Return register
15	Entry point and return code

Open/Close Routines

Register	*Use*
2	DCB address
3	TIOT address; TCB base register
4	DCB work area address
5	DCB parameter list address
6	Where-to-go table address

7	Current entry in the DCB parameter list
8	Current entry in the where-to-go table (the second word in the entry is the address of the open work area)
9	AVT address
11	DEB address
12	Base register

Operator Control Routines

Register	*Use*
1	Operator Control AVT address; appropriate response message
4	AVT address
5	Termname table entry address
6	Terminal table entry address
12	Base register
13	Save area address
14	Return register
15	Entry point and return code

Queuing Routines

Register	*Use*
1	Parameter list address
3	SCB address
4	LCB address
6	Current buffer address
7	QCB address
10	DCB address
11	TCAM Dispatcher address
12	Base register
13	AVT address
14	Return address
15	Entry point

Scheduler Routines

Register	*Use*
1	LCB or buffer address
3	STCB address
4	LCB address
7	QCB address
11	TCAM Dispatcher address
13	Save area address
15	Entry point

(This page left blank intentionally)

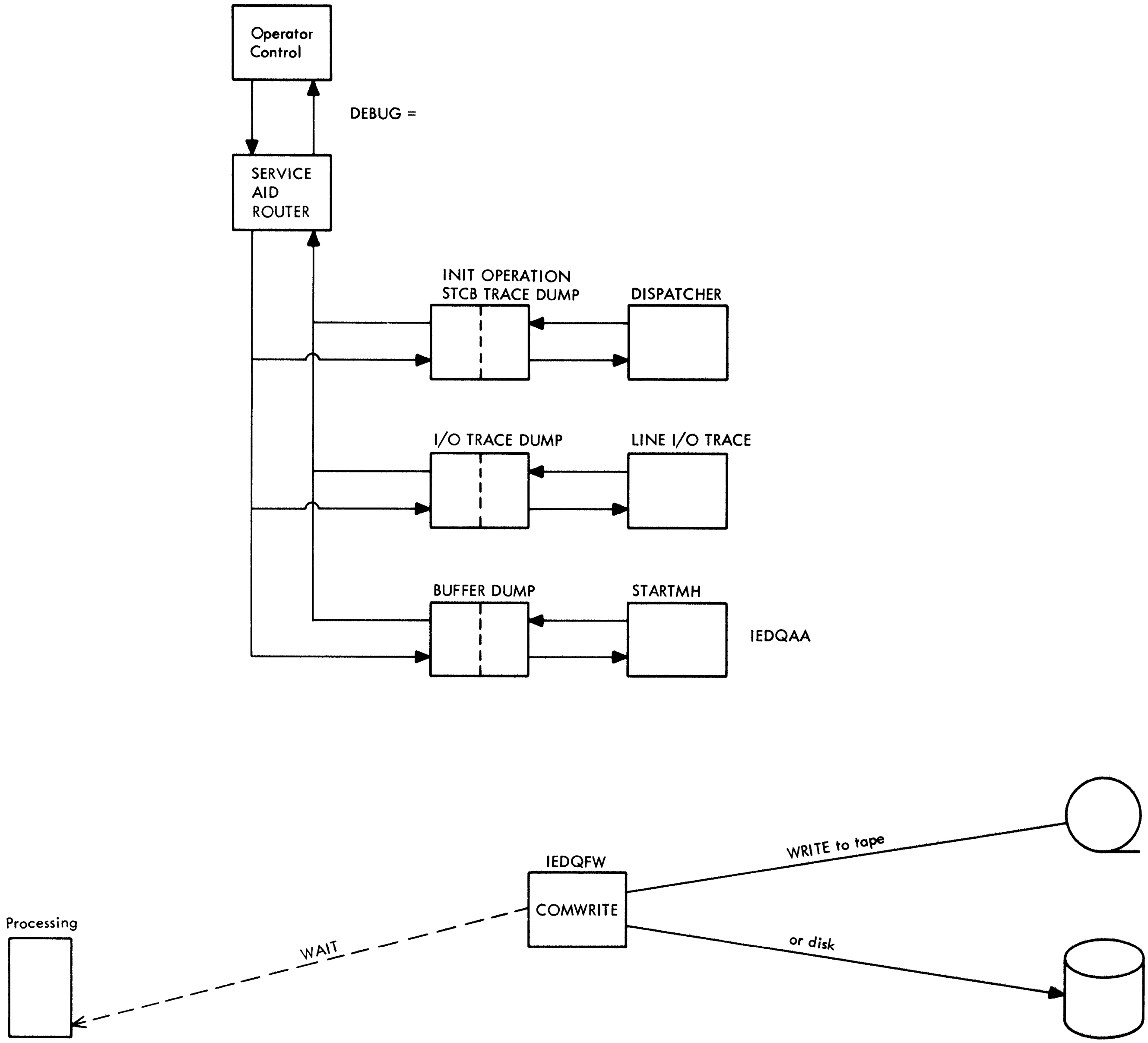
TCAM Service Aids

The Service Aids Programs are an optional TCAM facility. They provide the Customer Engineer and customer programming personnel with the ability to save portions or all of the following TCAM tables and buffers:

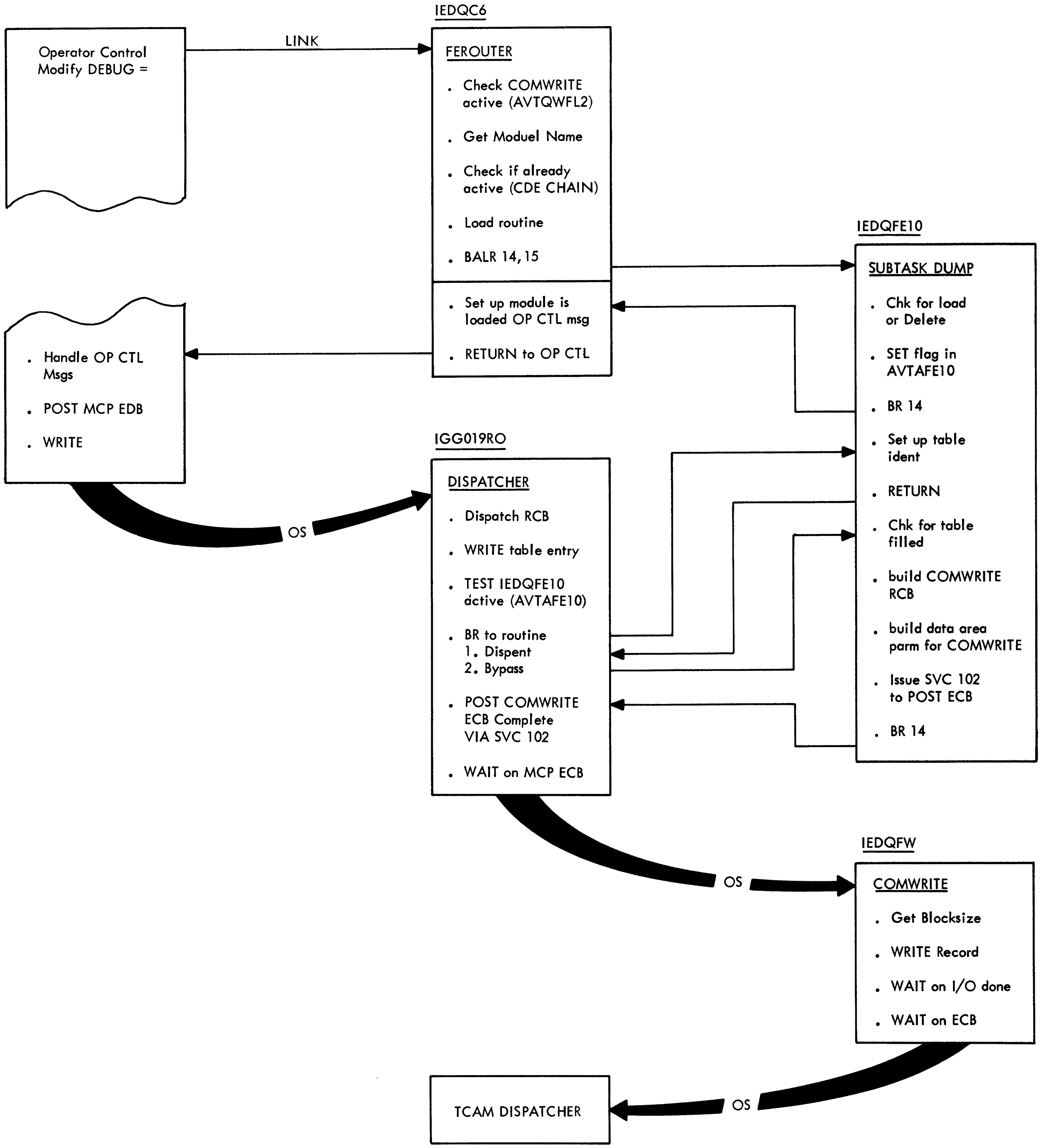
- Subtask Control Block Trace Table
- Line I/O Trace Table
- Message Buffers (main storage and secondary storage)

These areas are stored, using programs from the Service Aids, on either tape or direct access devices. The areas may be edited and printed in formatted form for use as a debugging tool. For detailed information on these areas see chapter 4, *TCAM Diagnostic Aids,* of the *OS TCAM User's Guide* (GC30-2025).

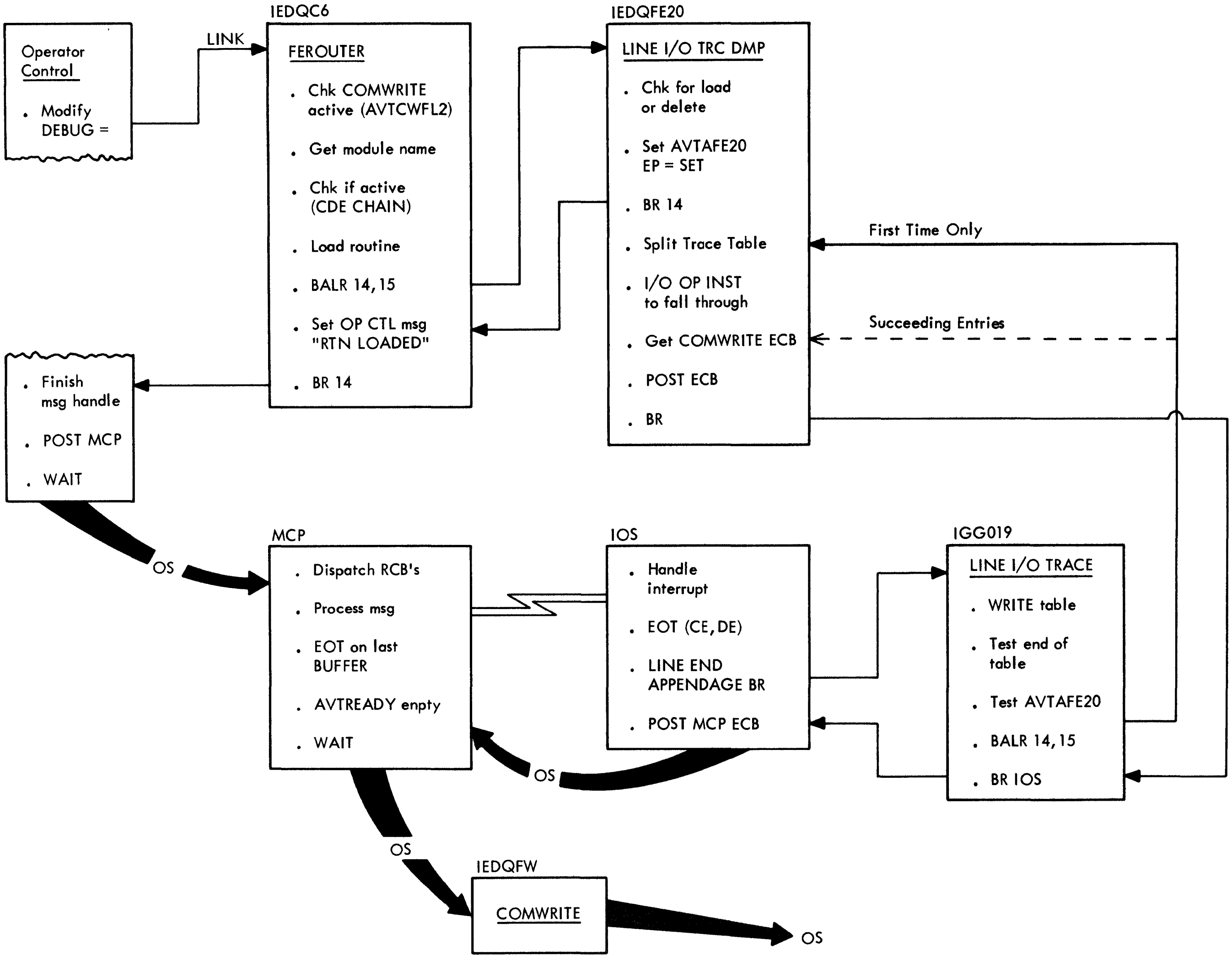
Service Aids Flow



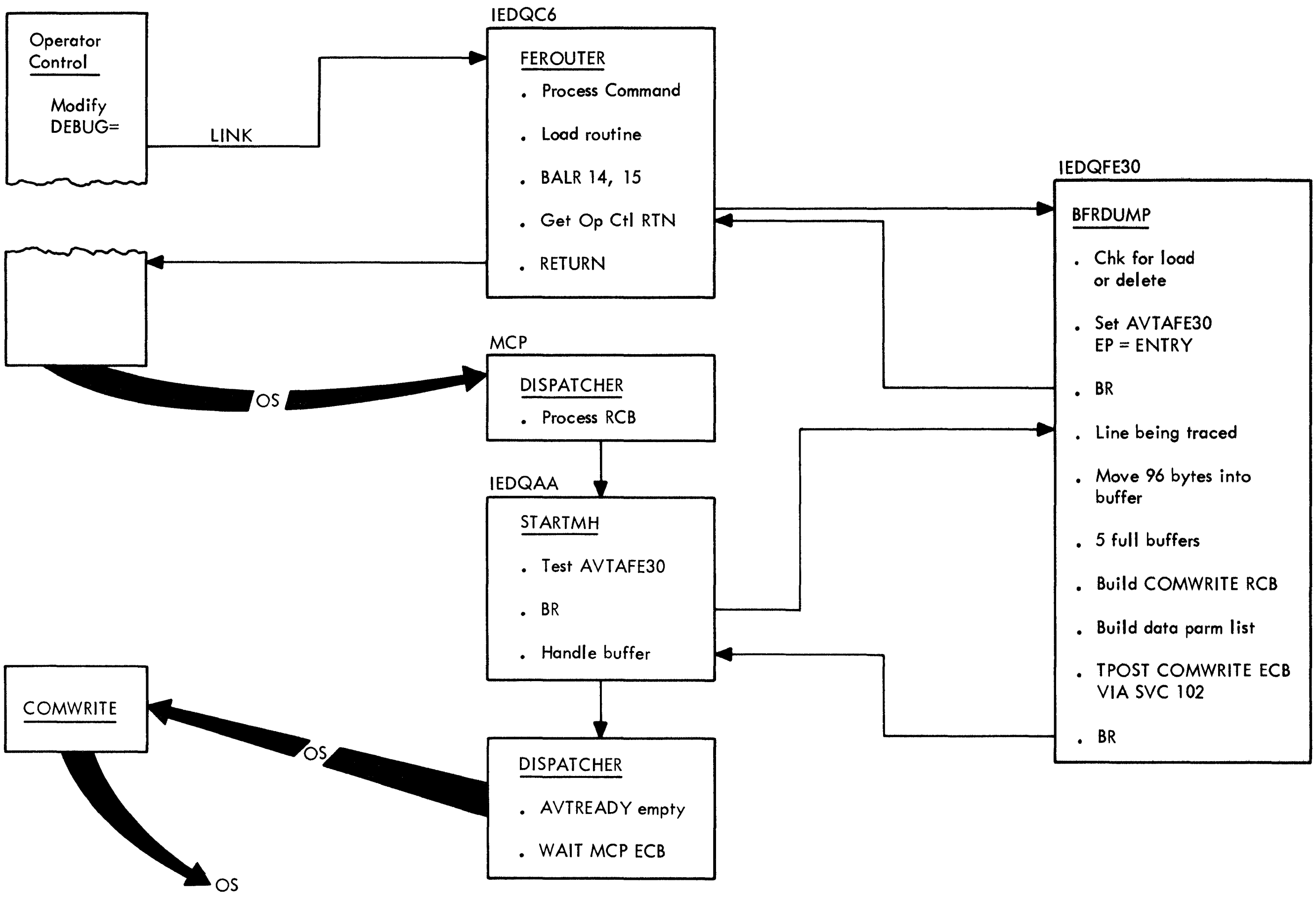
IEDQFE10 Subrask Control Block Trace Table Dump Flow



IEDQFE20 I/O Trace Dump Flow



IEDQFE30 Buffer Trace Dump Flow



Appendix A: List of TCAM Modules by Library

This appendix identifies the modules that constitute TCAM. The modules are organized in alphabetical order by name within the libraries in which they reside. For those modules that represent macro instruction implementing routines, the mnemonic operation code for the macro is included in parentheses.

All resident TCAM modules are in SYS1.TELCMLIB. Transient modules reside in SYS1.LINKLIB, and all Open, Close, Get, and Put modules are in SYS1.SVCLIB. The system nucleus modules are in SYS1.NUCLEUS. The TCAM module IEDQTNT is not stored in a library; rather, it is assembled as part of the termname table. TCAM macros are in SYS1.MACLIB.

SYS1.LINKLIB

IEDQCA	Resident Operator Control Module
IEDQEC	Put Scheduler
IEDQET	Operator Control/Application Program Interface Routine
IEDQEW	Get Scheduler
IEDQEZ	Get Scheduler FIFO Routine
IEDQE1	TCOPY Service Routine
IEDQE2	QCOPY Service Routine
IEDQE3	TCHNG Service Routine
IEDQE4	ICOPY Service Routine
IEDQE6	Password Scramble Routine
IEDQE7	Retrieve Scheduler
IEDQGQ	Queue Reset Executed
IEDQHI	System Delay Routine
IEDQNA2	Nonresident Closedown Completion Routine
IEDQNB	Application Program/Checkpoint Interface Routine
IEDQND	Ready Routine (READY)
IEDQNF	Checkpoint Executor
IEDQNG	Build Incident Record for MH Routine
IEDQNH	Build Incident Record for TCHNG Routine
IEDQNJ	Incident Checkpoint for Operator Control Routine
IEDQNK	Environment Checkpoint Routine
IEDQNM	Build CKREQ Disk Record Routine
IEDQNO	Checkpoint Queue Manager
IEDQNP	Checkpoint Disk I/O Routine
IEDQNQ	Checkpoint Notification and Disposition Routine
IEDQNR	Checkpoint–No Available Core Routine
IEDQNS	Checkpoint–No Incident Records Routine

IEDQNX	Operator Awareness Message Router
IEDQOA	GETMAIN, Termname Table Sort, and Attach Routine
IEDQOB	WTOR Interpreter Routine
IEDQOT01	TCAM ABEND Routine
IEDQWA	TOTE Resident Module
IEDQWAB	DTIME Service Module
IEDQWAJ	CUTEST Service Module
IEDQWB	Resource Management Module
IEDQWB1	TRM Analysis Buffer Analyzer
IEDQWC	Test Request Message Analysis Module 1
IEDQWC1	Test Request Message Analysis Module 2
IEDQWC2	Test Request Message Analysis Module 3
IEDQWD	TOTE Dispatcher Module
IEDQWE	TOTE Test Control Module
IEDQWF	OLT Test Control Module 2
IEDQWH	Numeric Test Request Message Handler
IEDQWI	TOTE Configurator Scheduler
IEDQWIA	Configurator and Scheduler
IEDQWID	Configurator Delete/Change Scheduler
IEDQWIE	Configuration Exhibit Module
IEDQWI5U	Configurator Submodule
IEDQWI7	Configurator Submodule
IEDQWI8	Configurator Submodule
IEDQWI9	Configurator Submodule
IEDQWJ	Test Request Message Prompter Module 1
IEDQWJ1	Test Request Message Prompter Module 2
IEDQWJ2	Test Request Message Prompter Module 3
IEDQWK	TOTE Message Module
IEDQWL	TOTE Message Submodule 1
IEDQWL1	TOTE Message Submodule 2
IEDQWL2	TOTE Message Submodule 3
IEDQWL3	TOTE Message Submodule 4
IEDQWM2	Trace Function Module
IEDQWN	EXIO Service Module (alias IEDQW35)
IEDQWO	Access Manager
IEDQWP	DPRINT Service Module 1 (alias IEDQW39)
IEDQWP1	DPRINT Service Module 2
IEDQWP2	DPRINT Service Module 3
IEDQWQ	CECOM Service Module (alias IEDQW37)

IEDQWR	PLINK Service Module (alias IEDQW28)
IEDQWS	Wait I/O Service Module (alias IEDQW36)
IEDQWV	TOTE GRAB/LETGO Service Module (alias IEDQW21, IEDQWAC)
IEDQWX	TOTE Convert Service Module (alias IEDQW41)
IEDQWY	GETCONFIG Service Module (alias IEDQW16)
IEDQW24	READD Service Module
IEDQW42	MORECORE/FREECORE Service Module (alias IEDQW43)
IEDQW44	DIO Service Module
IEDQW47	Routine Service Module
IEDQ31	Enabling Module
IEDQXA	Disk Message Queue Initializer

SYS1.MACLIB

ATTEN	Activates the TSO/TCAM Attention Processing Routine
CANCELMG	Cancels messages
CARRIAGE	Processes characters that move the carriage
CHECKPT	Takes an incident checkpoint record of the option fields
CHNGP	No-Op (QTAM/TCAM)
CHNGT	No-Op (QTAM/TCAM)
CKREQ	Checkpoints the MCP
CLOSEMC	Closes down the telecommunications system (QTAM/TCAM)
CODE	Translates the data in the buffer currently being handled
COMMBUF	Moves current buffer into a data area
COPYP	No-Op (QTAM/TCAM)
COPYQ	No-Op (QTAM/TCAM)
COPYT	No-Op (QTAM/TCAM)
COUNTER	Maintains a count of complete messages or of message segments received from or sent to a terminal
CTBFORM	Provides for data insertion in the CTB
CUTOFF	Specifies the maximum allowable incoming message length
DATETIME	Inserts the date and time in an incoming or outgoing message header
ERRORMSG	Sends an error message when an error occurs
FORWARD	Queues messages for specified destinations
HANGUP	Checks for I/O errors
HOLD	Suspends transmission to a terminal
ICHNG	Modifies an invitation list
ICOPY	Examines the contents of an invitation list
IEDQCHAR	Internal assembly macro to check character strings

IEDQCHI	Internal assembly macro to determine device characteristics
IEDQCKO	Internal assembly macro to perform validity checking on terminal operands
IEDQFEA	Internal assembly macro for TSO
IEDQGCH	Internal assembly macro to generate device-dependent fields for a terminal entry
IEDQMASK	Internal assembly macro to analyze mask operands
IEDQSCAN	Internal assembly macro to search for a character string
IEDQTO	Internal assembly macro to generate the option fields specified by a TERMINAL macro
IEDQTQ	Internal assembly macro to generate QCBs
IEDQTT	Internal assembly macro to generate a termname table entry
IEDQVCON	Internal assembly macro to provide proper branching addresses for all the macros
INBLOCK	Identifies a subgroup that handles incoming logical messages
INBUF	Identifies a subgroup that handles incoming message buffers
INEND	Identifies the end of the MH incoming group
INHDR	Identifies the beginning of an inheader subgroup
INITIATE	Sends message segments immediately to their destination
INMSG	Identifies the beginning of an MH inmessage subgroup
INTRO	Creates the AVT
INVLIST	Generates the invitation list for a line
INVLIST1	Internal assembly macro to generate an invitation list
INVLIST2	Internal assembly macro to generate an invitation list
INVLIST3	Internal assembly macro to generate an invitation list
INVLIST4	Internal assembly macro to generate an invitation list
INVLIST5	Internal assembly macro to generate an invitation list
INVLIST6	Internal assembly macro to generate an invitation list
LINEGRP	TSO MCP generation macro
LISTTA	TSO MCP generation macro
LOCK	Locks one terminal on a line to an application program
LOCOPT	Locates a field in the option table
LOG	Logs complete messages or message segments
LOGON	Connects a terminal user to TSO or TCAM
LOGTYPE	Initializes for using the TCAM logging facility
MCOUNT	Provides the count of complete messages for an application program
MCPCLOSE	Initiates closedown of the telecommunications system
MHGET	Makes data in current buffer available to user routine
MHPUT	Moves data from user work area into current buffer
MRELEASE	Releases messages queued for a destination

MSGEDIT	Inserts and/or removes specified characters at specific locations in a message
MSGFORM	Inserts line control characters in outgoing messages
MSGGEN	Generates an unqueued message
MSGLIMIT	Limits the number of messages during a single transmission sequence
MSGTYPE	Controls the path of a header through an MH
OPTION	Defines the option table
ORIGIN	Checks the validity of the origin field in a message header
OUTBUF	Identifies a subgroup that handles outgoing message buffers
OUTEND	Identifies the end of any MH outgoing group
OUTHDR	Identifies the beginning of an outheader subgroup
OUTMSG	Identifies the beginning of an MH outmessage subgroup
PATH	Dynamically varies the path of a message through an MH
PCB	Generates a process control block (PCB) in an MCP to interface with an application program
PRIORITY	Specifies priority handling for messages
QACTION	Provides a user exit to analyze status information given by a concentrator or a station attached to a concentrator
QCOPY	Examines the contents of a QCB
QRESET	Resets QCBFFEFO pointer backward to desired output sequence number and re-sends messages from reset point forward
QSTART	Differentiates between a QTAM and a TCAM application program (QTAM/TCAM)
READY	Initializes and activates the MCP
REDIRECT	Queues a message for an additional destination
RELEASEM	Releases messages queued for a destination (QTAM/TCAM)
RETRIEVE	Retrieves a message for reprocessing (QTAM/TCAM)
RETRY	Tries to initiate contact with a switched station
RTAUTOPT	Resumes automatic prompting (after a null line) (TSO)
SCREEN	Modifies the Write operations for display terminals
SEQUENCE	Checks the input sequence number of an incoming message
SETEOF	Indicates an EOF message
SETEOM	Determines the handling of data that follows an EOM control character
SETSCAN	Moves the scan pointer forward or backward or returns the address of the last character of a specific character string
SGIEC3TP	Moves BTAM, QTAM, and TCAM modules into SYS1.SVCLIB at system generation time
SGIEC5TP	Moves BTAM, QTAM, and TCAM modules into SYS1.LINKLIB, SYS1.SVCLIB, and SYS1.TELCMLIB at system generation time

SGIEC2PT	Generates UCBs at system generation time
SGIEC519	Moves the proper macros into SYS1.MACLIB at system generation time
SIMATTN	Handles a simulated attention string or code (TSO)
SLOWPOLL	Suspends further polling on a line when an error occurs
SPAUTOPT	Stops automatic prompting (TSO)
STARTLN	Activates a line or line group (QTAM/TCAM)
STARTMH	Establishes addressability for an MH routine
STATTN	Sets up a simulated attention string or code, time or lines (TSO)
STAUTOCP	Starts automatic character prompting (TSO)
STAUTOLN	Starts automatic line numbering (TSO)
STBREAK	Allows the user to specify the presence of the reverse break feature (TSO)
STCC	Allows the user to specify line and character deletion characters (TSO)
STCLEAR	Specifies the character string used to clear the 2260 screen (TSO)
STCOM	Specifies whether to allow other TSO stations to send the user messages (TSO)
STOPLN	Deactivates a line or line group (QTAM/TCAM)
STSIZE	Specifies the length of a line or the length of and the number of lines for a 2260 (TSO)
STTIMEOU	Specifies whether a 1050 has the time-out suppression feature (TSO)
TCHNG	Places specified data in a terminal table entry
TCLEARQ	Allows the user to clear the TSO input or output queue (TSO)
TCOPY	Examines the contents of a terminal table entry
TERMINAL	Creates a single or group entry in the terminal table
TERRSET	Sets a bit in the error record
TGET	Transfers a line of input from a TSO terminal to the user's data area (TSO)
TGOTO	Provides communication between message handlers for processing logical messages within a concentrated message
TLIST	Defines a cascade-list or distribution-list entry in the terminal table
TPDATE	Specifies whether RECDELs (record delimiters) should be deleted from application program input
TPEDIT	Edits MDI control characters for IBM 50 Magnetic Data Inscriber
TPROCESS	Interfaces between the MCP and an application program
TPUT	Transfers a line of output from the user's data area to a TSO terminal (TSO)

TRANLIST	Generates a control table for use by the Dynamic Translation routine (IEDQA3)
TSINPUT	Generates a QCB for the TSO subtask and creates an extension of the AVT for TSO support
TSOMCP	TSO MCP generation macro
TSOMH	TSO MCP generation macro
TTABLE	Defines the terminal table
TYPETABL	Sets up branch table for MSGTYPE macro
UNLOCK	Removes a terminal from extended lock mode

SYS1.NUCLEUS

IEDQATTN	Attention Routine
IEDQEB	AQCTL SVC 102 Routine

SYS1.SVCLIB

IGC0010D	Operator Control Control Module—Load 0
IGC0110D	Operator Control Control Module—Load 1
IGC0310D	Operator Control Message Module—Load 1
IGC0410D	Operator Control Message Module—Load 2
IGC0510D	Operator Control Message Module—Load 3
IGC0610D	Incident Checkpoint Request Interface
IGC0710D	Output Writer and On-Line Test Interface Routine
IGC0810D	Operator Control Message Module—4
IGC0910D	Operator Control Message Module—5
IGCD010D	Scan/Map/Dispatch Display Function Routine
IGCD110D	Copy Operator Control Terminal Routine
IGCD210D	Copy QCB Information Routine
IGCD310D	Copy Invitation List Entry Routine
IGCD410D	Copy Held Terminals Routine
IGCD510D	Copy Terminal Information Routine
IGCD610D	Copy Line Information Routine
IGCD710D	Copy Invitation List Status Routine
IGCD810D	Display Options Routine
IGCD910D	Copy LCB Information Routine
IGCH010D	Stop Terminal Transmission Routine
IGCI010D	ICHNG Deactivate Routine
IGCI110D	ICHNG Move/Activate Routine
IGCMA10D	Scan/Map/Dispatch Modify Function Routine—1
IGCM010D	Scan/Map/Dispatch Modify Function Routine—2
IGCM110D	Modify Successful Message Routine
IGCM210D	Modify Poll Routine

IGCM410D	Change Interval Type Routine
IGCM510D	Modify Intense Routine
IGCM610D	Alter Trace Status Routine
IGCM710D	Change Control Terminal Routine
IGCM810D	Modify Options Routine
IGCM910D	Debug Service Aid Router
IGCR010D	Resume Terminal Transmission Routine
IGCV010D	Scan/Map/Dispatch Vary Function Routine
IGCV110D	Stop Line Routine
IGCV210D	Stop Terminal Routine
IGCV310D	Start Line Routine
IGCV410D	Start Terminal Routine
IGCV510D	Stop General Poll Routine
IGCV610D	Start General Poll Routine
IGCZ010D	MCP Closedown Processing Routine—1
IGCZ110D	MCP Closedown Processing Routine—2
IGC1303D	TCAM Command Scheduler—SVC 34 (alias of IED1303D)
IGE0004G	Start/Stop ERP Control Module
IGE0104G	Read/Write Unit Check (Except Time-Out) ERP Module
IGE0204G	Non-operational Control Unit, Unit Exception, and Unit Check with Time-Out Module
IGE0304G	Unit Check for Non-read, Non-write, and Non-poll CCWs ERP Module
IGE0404G	Auto Poll and Read Response to Poll Unit Check and Unit Exception ERP Module
IGE0504G	Error Post and Second Level CCW Return Module
IGE0604G	Unit Check and Unit Exception on Read/Write CCWs for Audio and 2260 Local Devices ERP Module
IGE0804G	Start/Stop Channel Check Module
IGE0904G	Closedown Terminal Statistics Recording Module
IGE0004H	BSC ERP Control Module
IGE0104H	BSC Read/Write Equipment Check, Lost Data, Intervention Required, and Unit Exception ERP Module
IGE0204H	BSC Read/Write Data Check, Overrun, and Command Reject ERP Module
IGE0404H	BSC Second Level CCW Return Module
IGE0504H	BSC Error Post Module
IGE0804H	BSC Channel Check ERP Module
IGE0904H	TPER Recorder Module
IGG019AO	TOTE's Start I/O Appendage
IGG019AP	TOTE Channel End and Abnormal End Appendage

IGG019Q0	Line I/O Interrupt Trace Routine
IGG019Q1	Local Receive Scheduler
IGG019Q2	Line End Appendage for BSC Lines
IGG019Q3	Line End Appendage for Start/Stop Lines
IGG019Q4	Line End Appendage for Leased and Start/Stop Lines and No TSO
IGG019Q5	Line End Appendage for a QTAM Compatible System
IGG019Q6	Send Scheduler for Leased Lines and No TSO
IGG019Q7	Send Scheduler with No TSO
IGG019Q8	Checkpoint Continuation Restart Subroutine
IGG019Q9	Concentrator Send Scheduler
IGG019RA	Checkpoint Disk End Appendage
IGG019RB	TCAM Dispatcher
IGG019RC	EXCP Driver
IGG019RD	Buffered Terminal Scheduler
IGG019RE	COMMBUF Send Scheduler
IGG019RF	EXCP Drive for a Single CPB
IGG019RG	GET/READ Routine
IGG019RH	Get Compatible Routine
IGG019RI	PUT/WRITE Routine
IGG019RJ	Put Compatible Routine
IGG019RK	Disk End Appendage for a Single CPB
IGG019RL	Check Routine (CHECK)
IGG019RM	Point Routine (POINT)
IGG019RN	PCI Appendage
IGG019RO	TCAM Dispatcher with Subtask Trace
IGG019RP	Reusability–Copy Subtask
IGG019RQ	Post Pending Routine
IGG019RR	IBM 1030, 1050, 1060, 2740, 2741 Special Characters Table
IGG019RS	IBM 2260 Remote Special Characters Table
IGG019RT	AT & T 115A or Western Union 83B3 Special Characters Table
IGG019RU	AT & T TWX, with Odd Parity Special Characters Table
IGG019RV	IBM 2260 Local Special Characters Table
IGG019RW	World Trade Teletype Adapter (WTTA) Special Characters Table
IGG019RX	AT & T TWX, with Even Parity Special Characters Table
IGG019RY	Audio Special Characters Table
IGG019R0	Line End Appendage
IGG019R1	Dial Receive Scheduler

IGG019R2	Disk End Appendage
IGG019R3	Leased Receive Scheduler
IGG019R4	Send Scheduler
IGG019R5	Attention Handler
IGG019R6	Start-up Message Routine
IGG019R7	BSC EBCDIC Code Special Characters Table
IGG019R8	BSC USASCII Code Special Characters Table
IGG019R9	BSC 6-bit Code Special Characters Table
IGG01930	Disk Message Queues Open—1
IGG01931	Disk Message Queues Open—2
IGG01933	Open Error Handler
IGG01934	Disk Message Queues Open—3
IGG01935	Line Group Open—1
IGG01936	Line Group Open—2
IGG01937	Line Group Open—3
IGG01938	Line Group Open—4
IGG01939	Line Group Open—5
IGG0194B	Application Program Open Error Interface Routine
IGG01940	Line Group Open—6
IGG01941	Checkpoint Open Routine
IGG01942	Checkpoint Disk Initialization Routine
IGG01943	Checkpoint/Restart from Environment Record Routine
IGG01944	Checkpoint/Restart from Incident and CKREQ Records Routine
IGG01945	Checkpoint Continuation Restart Routine
IGG01946	GET/PUT and READ/WRITE Open Executor—1
IGG01947	GET/PUT and READ/WRITE Open Executor—2
IGG01948	Line Group Open—7
IGG01949	Checkpoint Disk Allocation Routine
IGG02030	Disk Message Queues Close Routine
IGG02035	Line Group Close Routine—1
IGG02036	Line Group Close Routine—2
IGG02041	Checkpoint Close Routine
IGG02046	GET/PUT and READ/WRITE Close Executor—1
IGG02047	GET/PUT and READ/WRITE Close Executor—2

SYS1.TELCMLIB

IEDAYA	TSO Attention Routine
IEDAYB	TSO TIOC 3270 Edit Routine
IEDAYC	TSO Carriage Subroutine

IEDAYD	Time Sharing Destination Scheduler
IEDAYE	TSO TIOC Edit Routine
IEDAYF	TSO IOHALT Routine
IEDAYH	TCAM/TSO Hang-up Routine
IEDAYI	TSINPUT Routine
IEDAYL	TSO Logon Routine
IEDAYM	TSO Message Generation Routine
IEDAYO	TSOUTPUT Routine
IEDAYR	STARTMH Subtask for TCAM-TSO Mixed
IEDAYS	TSO Simulated Attention Routine
IEDAYT	TSO Abend Interface Routine
IEDAYX	TSO INMSG/OUTMSG Linker Routine
IEDAYY	TSO Asynchronous Time Delay Removal Routine
IEDAYZ	Time Sharing Scheduler
IEDQAA	STARTMH Subtask (STARTMH)
IEDQAB	STARTMH Continuation
IEDQAC	Date and Time Provision Routine (DATETIME)
IEDQAD	Output Sequence Number Provision Routine (SEQUENCE)
IEDQAE	Locate Option Field Address Routine (LOCOPT)
IEDQAF	Insert Data Routine
IEDQAG	Message Limit Routine (MSGLIMIT)
IEDQAH	Input Sequence Number Insertion Routine (SEQUENCE)
IEDQAI	Skip Forward and Scan Routine (SETSCAN)
IEDQAJ	Skip to Character Set Routine (SETSCAN)
IEDQAK	Line Control Insertion Routine
IEDQAL	Compare at Offset Routine
IEDQAM	Origin Routine (ORIGIN)
IEDQAN	Multiple Insert/Remove Routine (MSGEDIT)
IEDQAO	Unit Request Interface Routine
IEDQAP	Remove at Offset Routine (MSGEDIT)
IEDQAQ	Operator Control Interface Routine
IEDQAR	Cancel Message Routine
IEDQAS	Hold/Release Terminal Routine
IEDQAT	Create an Error Message Routine (ERRORMSG)
IEDQAU	Cutoff Message Transmission Routine and Subtask (CUTOFF)
IEDQAV	Look-up Terminal Entry Routine
IEDQAW	Translate Buffer Routine (CODE)
IEDQAX	Buffer Scan Routine
IEDQAY	Screen Routine (SCREEN)

IEDQAZ	Redirect a Message Routine (REDIRECT)
IEDQA0	Skip Backward Routine (SETSCAN)
IEDQA1	Binary Search Routine
IEDQA2	Insert at Offset Routine (MSGEDIT)
IEDQA3	Dynamic Translation Routine
IEDQA4	Incoming/Outgoing Message Delimiter Routine
IEDQA5	Forward Routine (FORWARD)
IEDQA6	Line Control Initialization Routine (MSGFORM)
IEDQA7	Counter Routine (COUNTER)
IEDQA8	Multiple Insert at Offset Routine (MSGEDIT)
IEDQA9	Re-dial Routine (RETRY)
IEDQBA	Multiple Routing Subtask
IEDQBB	Checkpoint Request Routine
IEDQBC	Distribution List Subtask
IEDQBD	Buffer Disposition Subtask
IEDQBE	Lock Routine
IEDQBF	Unlock Routine
IEDQBG	Cascade List Subtask
IEDQBH	Concentrator Buffer Disposition Subtask
IEDQBL	Message Generation Routine (MSGGEN)
IEDQBM	Origin Routine for a System with Concentrated Message Handling Support
IEDQBN	Data Attach Routine
IEDQBO	SETEOM Routine (SETEOM)
IEDQBP	TGOTO Routine (TGOTO)
IEDQBR	Count Module for SETEOM
IEDQBQ	QACTION Routine (QACTION)
IEDQBT	EOB/ETB Handling Subtask
IEDQBU	CANCELBK Subtask (CANCELMG with LEVEL=BLK)
IEDQBV	COMMBUF Routine
IEDQBX	Log Segment Routine
IEDQBY	Log Message Routine
IEDQBZ	Log Scheduler
IEDQB1	MCOUNT Routine (MCOUNT)
IEDQB2	TPDATE Routine
IEDQB3	DATETIME Insertion Routine for a Processing Program
IEDQB4	Slow Poll Routine
IEDQES	Retrieve Service Routine
IEDQEU	Open/Close Subtask
IEDQE6	Password Scrambler Routine

IEDQE8	Binary Search Routine for a Processing Program
IEDQFA	CPB Initialization Module
IEDQFA1	CPB Initialization–Main-Storage-Only Queuing
IEDQFA2	CPB Initialization–Disk-Only Queuing
IEDQFE	TCAM Service Aids Routine
IEDQFE10	STCB Trace Utility Routine
IEDQFE20	Buffer Trace Utility Routine
IEDQFE30	Line Trace Utility Routine
IEDQGA	Buffer Management Module
IEDQGH	CTBFORM Routine (CTBFORM)
IEDQGP	MHGET/MHPUT Routine
IEDQGR	QRESET Service Routine
IEDQGT	Transparent Transmission CCW Building Routine
IEDQHG	Time Delay Subtask
IEDQHK	Stop Line I/O Subtask
IEDQHM	Destination Scheduler
IEDQHM1	Destination Scheduler–Main-Storage-Only Queuing
IEDQHM2	Destination Scheduler–Disk-Only Queuing
IEDQKA	Activate–I/O Generator Subtask
IEDQKB	Activate–I/O Generator Subtask for BSC Lines
IEDQKC	Activate–I/O Generator Subtask for Start/Stop Lines
IEDQKD	Activate–I/O Generator Subtask for Leased and Start/Stop Lines and No TSO
IEDQKE	Activate–I/O Generator Subtask for a QTAM Compatible System
IEDQNA	Resident Closedown Completion Routine
IEDQUI	User Interface Routine
IEDQ10	IBM 1030 Translate Table
IEDQ11	IBM 1050 Translate Table
IEDQ12	IBM 1050 Folded Translate Table
IEDQ13	IBM 1060 Translate Table
IEDQ14	IBM 2260 Translate Table
IEDQ15	Alias for IEDQ14
IEDQ16	IBM 2740 Translate Table
IEDQ17	IBM 2740 Folded Translate Table
IEDQ18	World Trade Teletype Adapter (WTTA), ITA2 Translate Table
IEDQ19	World Trade Teletype Adapter (WTTA), ZSC3 Translate Table
IEDQ20	AT & T 115A or Western Union 83B3 Translate Table
IEDQ21	AT & T TWX, with Parity Translate Table

IEDQ22	AT & T TWX, without Parity Translate Table
IEDQ23	IBM 2780, 6-bit Code Translate Table
IEDQ24	USASCII Code Translate Table
IEDQ25	Dummy Table (EBCDIC to EBCDIC)
IEDQ26	IBM 2741, BCD Code Translate Table
IEDQ27	IBM 2741, EBCD Code Translate Table
IEDQ28	IBM 2741, Correspondence Code Translate Table

Appendix B. TCAM Queues and QCBs

TCAM Queues

Checkpoint disk I/O queue—Checkpoint disk records wait for this queue to be written to disk. The records are queued in FIFO order. The first word of the record is the link field. Each time an environment checkpoint record is put on the checkpoint disk I/O queue, the IEDQNO routine scans the queue. If there are any incident checkpoint disk records on the queue, the IEDQNO routine removes them and frees them. Since the information in the incident checkpoint record is included in each environment record, it is not necessary to write both records to disk. The Checkpoint Executor routine (IEDQNF) looks at the queue when a record is put on the queue, and gives control to the Checkpoint Disk I/O routine (IEDQNP).

Communication queue—This is a queue of command input blocks in FIFO order, chained by the first word in each CIB. The communication queue is used to queue command input blocks containing operator control commands from the console. An SVC 34 from the Command Scheduler places the CIBs on the queue, and the SVC 34 routine removes them. The second word of the queue is the communication ECB.

Copy buffer queue—When a message is to be copied from one queue medium to another, the first buffer of the message is tposted to COPY, which places the buffer on the copy buffer queue, pointed to by the AVTCOPY field. This field also points to the copy QCB whose first two words are used as a FIFO queue of buffers. Each message stays on the copy buffer queue until a CPB is available to be used to copy the message. One CPB is used per message as CPBs become available, the use of this queue ensures that messages will be copied in the order that the copy operation was requested. Buffers are chained by their second word. A zero is in the second word of the last buffer.

CPB free pool queue—The AVTFCPB field contains the address of the first of a chain of available CPBs. They are chained by CPBNEXT, with a zero in the last CPB. This is *not* a FIFO queue (as are other CPB queues) but a LIFO (last-in-first-out) queue. If the user specifies too many CPBs (INTRO CPB=integer), the CPBs at the end of this free pool chain will never have been used. The user should look at a TCAM dump for unused CPBs and specify a smaller number next time to save main storage.

Disabled ready queue—The disabled ready queue is a FIFO queue that contains elements passed from an application program's disabled appendages and attached tasks for processing by the MCP. The TCAM Dispatcher merges the contents of this queue into the enabled ready queue.

Disk end queue—There are two disk end queues. The address of the first is at AVTDKAPQ. This queue is used to pass CPBs from the Disk End Appendage to the CPB Cleanup routine. The address of the second queue is at AVTDKENQ. This queue is used as an alternate in the disabled/enabled interface to pass CPBs from the Disk End Appendage to the CPB Cleanup routine. If the AVTBPLKN bit is on, the Disk End Appendage cannot put a CPB in the disk end queue pointed to by AVTDKAPQ, but must place it in the queue pointed to by AVTDKENQ.

Enabled ready queue—see ready queue.

EXCP queue—This is a chain of CPBs for the one cylinder, in one extent of a disk message queues data set, that is currently ready for I/O execution. CPBs are ordered on this chain by FIFO order. CPB Initialization waits on this queue for I/O to complete so it can build a new CPB and do another EXCP.

EXCP driver input queue—This is a chain of CPBs that the EXCP Driver processes until it is empty. Only Read or Write CCW op codes and the buffer unit address are in the channel program. The disk address is an absolute disk address in the same form as when taken from the CPBRADDR or CPBNADDR field. An indication of reusability or nonreusability is in the CPBFLAG. The EXCP Driver removes the CPBs in FIFO order, places each one on the new queue by cylinder, and then completes the channel program. No EXCP is issued until the input queue is emptied. A doubleword queue pointer is in the AVTINCPQ field.

FEFO queue—first-ended-first-out—A FEFO message queue is ordered so that the message that ends first will be sent out first, regardless of the order in which the messages were received.

FIFO queue—A FIFO queue is any queue of elements managed on a first-in-first-out basis. When an element is placed on the queue, it is placed in the order in which it was received, and the first element on the queue is the first to be removed.

Hold queue—A hold queue is a FEFO-ordered queue that is a part of the priority level QCB for each destination QCB. If a terminal is intercepted (held), its messages are placed in this queue while messages for other terminals on this destination QCB are sent.

New queue—The new queue is a queue on the IOB chain of CPBs being built by the EXCP Driver. The CPBs are sorted on this queue by absolute cylinder number and are in FIFO order for any cylinder group. The CPBs are placed on the queue one at a time from the input queue by the EXCP Driver. They are removed by cylinder group and are placed on the retry queue.

No-buffer queue—This is a FIFO-ordered queue of CPBs for read operations when no buffers are in the buffer unit pool. This is an internal queue used by IEDQFA and IEDQFQ. The elements are linked by the CPBNEXT field.

No-CPB queue—This is queue of buffers and ERBs waiting for CPBs. The queue is located in the AVT and serves as a place to keep elements until CPBs are built for them.

Operator control queue—This is FIFO queue of buffers, dummy CIBs from application programs and TOTE, stopped LCBs, and dummy ERBs with their associated buffers. The second word of the queue is the operator control ECB. The queue is used as a communication link between the TCAM MCP and Operator Control. All commands other than those from the console are placed on this queue, as well as elements (LCBs, ERBs) requested by Operator Control.

Ready queue—This is a priority-FIFO ordered queue of TCAM elements that are to be processed by the TCAM subtasks.

Retry queue—This is a chain of CPBs for one cylinder in an extent of the disk message queues data set. These CPBs are next in line for I/O execution after the CPBs on the EXCP queue are processed. When the Disk End Appendage receives control after the CPBs on the EXCP queue are finished, it requests IOS to do a retry after moving the CPBs on this queue to the EXCP queue. This last move avoids an extra EXCP and permits the channel to begin work on the new disk channel program faster.

System delay queue—This is a chain of LCBs pointed to by the seventh word of the system delay QCB, which is pointed to by AVTHI. The System Delay subtask (IEDQHI) waits on the queue until all the LCBs are on the queue and then begins the system delay interval. When a system delay is requested, the Leased Receive Scheduler and the Buffered Terminal Scheduler tpost LCBs to the system delay queue, rather than continue I/O on the lines. When the count of LCBs is the same as the number of LCBs received by the System Delay subtask, a time request (the system delay QCB) is posted to the Time Delay subtask (IEDQHG). After the interval is complete, each LCB is removed and tposted to itself to resume line activity.

Time delay queue—This is a relative-time-of-interrupt ordered chain of elements that are requesting a system STIMER interrupt. The elements are chained by the eighth word in the element. The time delay QCB is always the last element in the queue. The purpose of this queue is to inform the routine tposting the element when a specified time has elapsed.

TCAM QCBs

Buffer disposition QCB—The address of the Buffer Disposition subtask (IEDQBD) is the first address in the list pointed to by the AVTMSGS field of the AVT. The buffer disposition QCB comprises the first three words of the routine. The Incoming/Outgoing Message Delimiter routine (IEDQA4) tposts the last segment of the incoming message to the QCB, and the Line End Appendage routine (IGG019R0) tposts the last segment of the outgoing message to the QCB to execute the INMSG and OUTMSG macro instructions. The Line End Appendage routine tposts the LCB to the QCB when the routine reaches the end of the polling list to clean up the line.

Buffer request QCB—The buffer request QCB address is located in the AVTBFREB field in the AVT. The Receive Schedulers (IGG019R1 and IGG019R3) tpost to the QCB ERBs requesting buffers for receiving operations. Buffer units are chained from the first word of the QCB to form the buffer unit pool.

Buffer return QCB—The buffer return QCB address is located in the AVTBFRTB field in the AVT. Routines that are no longer using buffers tpost them to the QCB to be returned to the buffer pool.

Checkpoint QCB—The checkpoint QCB address is located in the AVTCKPTB field in the AVT. This is a special type of QCB for attached tasks, and the QCB is also the STCB. An ECB is in the second word of the QCB. The Checkpoint Executor (IEDQNF) waits on the ECB. The TCAM Dispatcher posts the ECB when it puts a request element on the chain. The checkpoint QCB is never tposted to itself. However, when a checkpoint request element is tposted to the QCB, the Checkpoint Executor is given control.

Closedown completion element QCB—the QCB address is located in the AVTCLOSB field in the AVT. The MCP Closedown Processing routine (IEDQC0), and the Checkpoint Notification and Disposition routine (IEDQNQ) tpost the QCB to itself to give control to the Resident Closedown Completion routine (IEDQNA). The QCB is used as an element with the lowest priority of any element in the system. It is the only element ever tposted to the QCB.

Copy QCB—The address of the copy QCB is in the AVTCOPY field of the address vector table. The TCAM Dispatcher activates the Copy subtask when a buffer has been tposted to this QCB to have a message copied from one queuing medium to another.

CPB Cleanup QCB—The address of the CPB cleanup QCB is located in the AVTCPBCB field in the AVT. The Disk End Appendage (IGG019R2), upon completion of an I/O operation, chains the completed CPBs on the AVTDKAPQ queue and tposts the QCB to itself to activate the CPB Cleanup routine (IEDQFQ) in CPB Initialization (IEDQFA).

Cutoff QCB—The cutoff QCB is located within the Cutoff routine (IEDQAU). The Cutoff routine places the address of the QCB in the first word of the LCB. Line End Appendage (IGG019R0) tposts the LCB being cut off to the QCB when a channel program check occurs or when the Read Skip or Write Break sequence initiated by the Cutoff routine is complete.

Delete from time delay QCB—The address of the delete from time delay QCB is in the AVTCPRMB field of the AVT. Attached tasks tpost a special four-word element to this QCB. The element defines another element and requests the Time Delay subtask (entry point IEDQHG03) to search the time delay queue for a particular element. If the Time Delay subtask finds the element on the time delay queue, it removes that element. After this process, the subtask tposts the four-word element back to the requester to indicate the completion of the request.

Destination QCB—A pointer to a specific destination QCB is in each terminal entry. This pointer does not change, but, as messages are received or sent, the SCB points to the destination QCB involved. For dial or buffered terminals, the Time Delay subtask (IEDQHG) tposts the QCB to itself at the end of a time delay. Routines tpost full buffers to be queued to the destination QCB. The Destination Scheduler (IEDQHM) is always the last subtask represented on the STCB chain of a destination QCB. A destination QCB is made up of a master QCB, which contains the send scheduler STCB for this QCB and other information pertinent to the entire QCB; and one or more priority level QCBs, which contain all the queuing pointers for messages for that particular priority level.

Disk I/O QCB—The disk I/O QCB address is located in the AVTDSIOB field in the AVT. Buffers requesting writing on disk or servicing'of a bit are tposted to the disk I/O QCB for processing by CPB Initialization. The schedulers tpost to this QCB ERBs requesting full buffers to send.

Log destination QCB—There is a pointer to a log destination QCB in every logtype terminal table entry. When a log message is specified, a LOGTYPE macro must be specified in the terminal table to generate a terminal entry, an LCB, and an SCB. The Log Message routine (IEDQBY) tposts a duplicate header to the log destination QCB after the complete message is received or sent.

Master QCB—This is the basic format of a destination QCB. This QCB contains ten words of destination-specific data.

Multiple routing QCB—The address of the multiple routing QCB is in the list of VCONs pointed to by the AVTMSGS field in the AVT. The FORWARD parameter list has the index to it. Elements chained on the QCB are either IEDQFA recalled buffers or the IEDQFA ERB for the line.

On-line test QCB—The address of the on-line test QCB is in the AVTOLTQB field of the AVT. Test request messages (messages requesting TOTE to run an on-line test through TCAM) are tposted to this QCB.

Operator control QCB—The address of the operator control QCB is located in the AVTOPCOB field in the AVT. This is a special QCB for attached tasks, and the second word of the QCB is an ECB. When the Dispatcher receives an element for this QCB at the top of the ready queue, the ECB is posted complete. The Translation Test routine (IEDQA3) tposts buffers containing operator commands to the QCB. The Application Program/Operator Control Interface routine (IEDQNB) tposts dummy CIBs from application programs to the QCB. The Buffer Management module–Buffer Request routine (IEDQGA) tposts dummy ERBs containing requested buffers to the QCB. The Stop Line I/O subtask (IEDQHK) tposts stopped LCBs to the operator control QCB.

PCB QCB—The PCB QCB is located in words 2 through 4 of the PCB. This QCB is used in support of the QTAM-compatible RETRIEVE macro. The Dispatcher dispatches the Retrieve Scheduler (IEDQE7) from this QCB. The element chain contains retrieved buffers.

Priority QCB—Priority QCBs follow the master QCB and are logically a part of the master destination QCB. IEDQHM queues messages on one of the priority QCBs that is associated with the master destination QCB to which the message was tposted. The Send Scheduler (IGG019R4) sends messages queued on the highest-priority QCB first.

Put process QCB—The address of the put process QCB is in a process entry in the terminal table. This QCB provides compatability and symmetry so that all terminal entries will look alike to TCAM modules.

QCB for IEDQBD02—The QCB is located within the IEDQBD02 Buffer Disposition subtask (IEDQBD02 entry point). The subtask (IEDQBD) tposts the LCB to this QCB when an INMSG/OUTMSG subgroup has been executed.

Read-ahead QCB—The address of the read-ahead QCB is in the DEBQCBAD field of the application program data extent block, the location of which is within the process entry work area PERAQCB. The element chain contains buffers processed by the application program output message handler, but not processed by the GET/READ logic. The Dispatcher uses this QCB to dispatch the Get Scheduler (IEDQEW).

Recall QCB—The address of this QCB is in the LCBRCQCB field of the LCB. This is a pointer to the QCB of the subtask wishing control to be passed to it with a recalled buffer. The ERB is tposted to the QCB indicated in LCBRCQCB.

REUS QCB—The address of the REUS QCB is in the AVTIA field of the address vector table. The QCB is located at an offset of 4 from the beginning of the IGG019RP module. The Destination Scheduler (IEDQHM) tposts the REUS QCB to itself when the adjusted value in AVTRADDR is greater than four times that of AVTLODPT to activate Reusability to service a zone.

STARTMH QCB—The address of the STARTMH QCB is in the DCBMH field of the DCB for the line group. Buffers are tposted to this QCB by Line End Appendage and PCI Appendage on input when they are filled. On output, the buffers are tposted to the QCB by Line End Appendage after a positive response to addressing. When buffers are tposted to the QCB, IEDQAA receives control unless EOB checking is requested, in which case IEDQBT receives control.

QCB for the Stop Line I/O subtask—The address of this QCB is in the AVTHK field in the AVT. The Stop Line routine (IEDQCK) tposts stop line requests to this QCB. The various schedulers tpost LCBs to it.

System delay QCB—The system delay QCB is located in the first three words of the System Delay subtask (IEDQHI). The address of the subtask is in the AVTHI field of the AVT. The System Delay subtask tposts the QCB to the Time Delay subtask (IEDQHG) to start a wait. At the end of the wait, the Time Delay subtask tposts the QCB to itself to activate the System Delay subtask.

Time delay QCB—The time delay QCB is the last element on the time delay queue. The AQCTL SVC 102 routine (IEDQEB) tposts the QCB to itself as a result of the STIMER exit routine. This QCB is used by the STIMER exit routine to activate the Time Delay subtask (IEDQHG).

TSINPUT QCB—The address of this QCB is in the AVTTSOPT field of the AVT. The QCB is tposted to the TSINPUT routine (IEDAYI) to remove the system WAIT and unlock the keyboard.

Appendix C. List of Relative Priorities in TCAM

TCAM routines apply relative priorities to elements through the use of the TPRIOR macro. The names and values presented in this table are established by this internal macro.

Name	*Value*	*Use*	*Routines Using*
PRIINTRQ	E4	to request full buffers from Disk I/O	Send Scheduler Receive Scheduler Get Scheduler Put Scheduler Create an Error Message routine
PRIFSPCI	E8	to request empty buffers from buffer request QCB; to request full buffers from Disk I/O	PCI Appendage (on first PCI only) Multiple Routing subtask
PRISBPCI	E0	to request empty buffers from buffer request QCB; to request full buffers from Disk I/O	PCI Appendage (all PCIs except the first)
PRIDSKRQ	EC	to request an empty unit by chaining the ERB on the buffer return QCB	CPB Cleanup
PRIACTIV	E4	in tposting ERB to the activate QCB to request building an initial contact program and EXCP for the line	CPB Cleanup Buffer Request Buffer Return
PRIDKEOB	E0	to enable EOB to recall; to tpost to EOB Handling after an EOB error; must be lower priority than PRIMHBFR	CPB Cleanup CPB Initialization
PRIRECAL	E0	to request from Disk I/O a copy of the header	All routines requesting recalled headers Multiple Routing subtask
PRIRCQCB	E0	to return the ERB to any routine specified in LCBRCQCB	CPB Cleanup (after recall) Create an Error Message routine
PRIAPERB	D0	to request full buffers	Application Program
PRIEDISP	E0	to tpost ERB to itself on send operations when an error occurs before EOM; must be lower priority than PRIMHBFR	Buffer Disposition

Name	*Value*	*Use*	*Routines Using*
PRIMHBFR	E4	to have a buffer processed by MH	PCI Appendage CPB Cleanup Line End Appendage (receive—last buffer only)
PRIUREQ	E8	to request an empty unit for insert function in MH; must be higher than PRIMHBFR	Unit Request
PRIAPBFR	DC	to tpost a buffer to an application program	Incoming/Outgoing Message Delimiter routine
PRILNEND	E4	to have Buffer Disposition finish processing macros and clean up the line	Line End Appendage (send—last buffer only)
PRIRCBFR	E0	to return a duplicate header to a specified routine	CPB Cleanup Destination Scheduler
PRIBFRTB	E4	to return a buffer or unit to the buffer unit pool	Incoming/Outgoing Message Delimiter routine PCI Appendage CPB Cleanup Destination Scheduler Multiple Routing subtask
PRIDSKBF	EC	to give a unit to CPB Cleanup	Buffer Return
PRICOPY	E0	to have a message copied to a different data set	Destination Scheduler
PRIDESTQ	E4	to put a buffer on a message queue	Incoming/Outgoing Message Delimiter routine Multiple Routing subtask Create an Error Message routine
PRIDKWRT	E4	to have a full buffer written on disk	Destination Scheduler
PRIDKSRV	EC	to have a message flagged serviced	Buffer Cleanup
PRIDKCNC	E0	to have a message canceled in the message queue	Cancel Message
PRIDKINT	E0	to have a message intercepted	Hold/Release Terminal routine

Name	*Value*	*Use*	*Routines Using*
PRICKPLN	EC	to tpost the LCB to Checkpoint requesting a checkpoint	Buffer Disposition
PRIMULTR	E0	to tpost the LCB to the Multiple Router routine to continue	Buffer Disposition TLIST
PRIOPCTL	DC	to tpost an operator control buffer	Message Handling routine Operator Control Interface routine
PRIDSPLB	E4	to tpost last buffer of message to buffer disposition QCB; must be lower than any PCI tpost of an ERB	Incoming/Outgoing Message Delimiter routine
PRIONLT	DC	to request On-Line Test	STARTMH subtask
PRILAEND	E4	to start error processing	Line End Appendage
PRIMHUNT	E8	to tpost a unit to MH; must be greater than PRIMHBFR	Unit Request
PRIRELSE	E0	to release a subtask from Time Delay or Operator Control	Operator Control Hold/Release Terminal
PRIRLCB	EB	to return the LCB	Buffer Disposition
PRILCB	E7	to tpost the LCB for cleanup	Line End Appendage
PRICPBCL	E8	to Post CPB Cleanup complete	Disk End Appendage
PRICKPT	DC	to request a complete checkpoint	Reusability–Copy subtask MCPCLOSE Time Delay subtask
PRILNFRE	E8	to free a line; must get to Destination Scheduler before line is free	Buffer Disposition Put Scheduler Send Scheduler
PRICLSDN	10	to request closedown; must be lowest priority	
PRIAPCKP	DC	to request an incident checkpoint	Application Program
PRIOPCKP	DC	to request an incident checkpoint	Operator Control
PRILNCL	EC	to clean up buffers and to free a line;	Line End Appendage
		to tpost a line to Buffer Disposition	INEND OUTEND

Name	*Value*	*Use*	*Routines Using*
PRILOGLB	E0	to tpost the Log LCB to itself	LOG Scheduler
PRISSOLT	DC	tposted to Operator Control to request Startline/Stopline to return an element from the time delay queue	On-Line Test Time Delay
PRIATTN	DC	to tpost the attention element for local devices	Attention Handler
PRISYSDL	DC	to initiate system delay	Operator Control
PRISYSDT	D8	to tpost the system delay QCB to Time Delay	System Delay
PRILCBDL	20	to indicate to Environment Checkpoint that an LCB is on the time delay queue	System Delay subtask Environment Checkpoint
PRIREUSX	E8	to tpost the REUS QCB to itself to activate Reusability servicing of a zone of reusable disk	Reusability–Copy
PRIFEFO	EE	to tpost a buffer to disk I/O to cause a FEFO pointer to be written to take a message off the FEFO chain	Reusability–Copy
PRILCBAT	E9	to tpost the LCB to the Stop Line I/O subtask for attention interrupt determination	Line End Appendage

Appendix D: TCAM Channel Program and TP Operation Codes

The format of the TCAM channel command word (CCW) is as follows:

Offset	0 7	8 31
	Command Code	Data Address

32 36	37 39	40 47	48 63
Flags	000	Reserved	Count

The TCAM channel programs are generated by the I/O Generator module (IEDQKA). Channel programs are listed by operation types within communication line types. The description of each channel program begins with a representation of the model channel program according to the followingcategories:

1. **Operation**—The command code with a brief description of the information that is being transferred.
2. **Address**—The data address that is set in the CCW before execution:
   Buffer refers to the buffer CCW address.
   Table refers to the appropriate location in the special characters table.
   List refers to the applicable invitation or addressing list entry.
   LCB refers to the line control block.
   Entry refers to addressing characters, dial digits, etc., in a terminal entry.
   Idles refers to an idles loop that is used to process data.
3. **Flags**—The flags that are set in the generated CCW are: chain command (CC), chain data (CD), and suppress length indication (SLI).
4. **Count**—The data count that is set in the generated CCW before execution.

A TP Op code differentiates among the types of CCWs on which interrupts can occur. In TCAM, the Activate–I/O Generator subtask builds a string of TP Op codes for any given channel program in the LCB. There is one TP Op code for each CCW. These codes are retrieved and used by Line End Appendage. A TP Op code with an even-numbered value represents a text or non-text CCW for which an interrupt is anticipated. A TP Op code with an odd-number value represents a CCW for which no interrupt is anticipated. The following is a list of the TCAM TP Op codes:

Name	*Value*	*Description*
TPWREOT	X'01'	Write EOT for selection
TPOPEN	X'02'	Open TP Op Code
TPWRPOLL	X'03'	Write Polling Characters
TPRDRESP	X'04'	Read Response to Polling
TPWRPAD	X'05'	Write pad characters
TPENABLE	X'06'	Enable on Dial Line
TPWRAD	X'07'	Write Addressing Sequence
TPRDRSPD	X'08'	Read Response to Addressing

Name	*Value*	*Description*
TPWREOA	X'09'	Write EOA Sequence
TPRDRPEB	X'0A'	Read Response to EOB/ETB
TPWRCPU	X'0B'	Write CPU ID
TPRDENQ	X'0C'	Read ENQ
TPWRENQ	X'0D'	Write ENQ
TPRSPENQ	X'0E'	Read Response to ENQ
TPWRDLET	X'0F'	Write DLE EOT
TPRDID	X'10'	Read ID (TSO)
TPNULL	X'11'	Non-Read/Write CCWs for which no Interrupt is anticipated
TPBREAK	X'12'	Write BREAK (TSO)
TPENQAD	X'13'	Write ENQ after Selection Response
TPRDLC	X'14'	Read LCOUT
TPWRACK	X'15'	Write Response Prior to Text
TPWRAKNK	X'16'	Write Response
TPWRTONE	X'17'	Write Tone (WTTA BSC)
TPRDIDNQ	X'18'	BSC Read ID ENQ
TPRDIDAK	X'1A'	BSC Read ID ACK
TPRESET	X'1C'	Abort for Send/Receive
TPTWXID	X'1E'	Read TWX ID
TPBUFEOT	X'20'	Buffered Terminal Reset after Block
TPCLOSE	X'22'	Close SDR Recording
TPRSPAD	X'24'	Write Reset after Selection
TPRDSKIP	X'51'	Read Skip Loop
TPWRIDLE	X'53'	Write Idles Loop
TPDLESTX	X'57'	Write DLE STX
TPDLEETX	X'59'	Write DLE ETB (ETX)
TPENQRSP	X'5B'	Write ENQ in Response to Text
TPTEXT	X'FF'	Text CCW

The first two CCWs in Read Initial channel programs are the following:

	Operation	*Address*	*Flags*	*TP Code*	*Count*
A	Read	Skip	CC,SLI	51	1
	TIC	label A			

These CCWs are executed whenever a buffer is not available. The initial contact CCWs are constructed in the channel program area plus 16 (third CCW).

When an Idle character is defined for a device, the first two CCWs in Write Initial channel programs are the following:

	Operation	*Address*	*Flags*	*TP Code*	*Count*
A	Write	Idles	CC,SLI	53	3
	TIC	label A			

CHANNEL PROGRAMS FOR THE AT&T 83B3 SELECTIVE CALLING STATION LINES

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	
Write polling characters	List	CC,SLI	03	
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response	LCB		08	9
Write EOA sequence	Table	CD	09	
TIC	Idles			

The Write Initial channel program places the line in control mode, addresses a terminal, and reads the response. An interrupt is taken on the Read Response, after which buffers are tposted to the outgoing MH. Restart is made at the Write EOA sequence, which transfers-in-channel to the Idles loop and from there writes data.

CHANNEL PROGRAMS FOR WESTERN UNION PLAN 115A OUTSTATION

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	07	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response	LCB		08	9
Write EOA sequence	Table	CD	09	4
TIC	Idles			

The Write Initial channel program places the line in control mode, addresses a terminal, and reads the response. An interrupt is taken on the Read Response, after which buffers are tposted to the outgoing MH. Restart is made at the Write EOA sequence, which transfers-in-channel to the Idles loop and from there writes data.

CHANNEL PROGRAMS FOR IBM 1030 LINES

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write positive (ACK) or negative (NAK) response	Table		16	1

The Read Continue channel program sends a positive or negative response to the previous message block to indicate a response from TCAM.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response	LCB		08	9
Write EOA sequence	Table	CD	09	1
TIC	Idles			

The Write Initial channel program places the line in control mode, addresses a terminal, and reads the response. An interrupt is taken on the Read Response, after which buffers are tposted to the outgoing MH. Restart is made at the Write EOA sequence, which transfers-in-channel to the Idles loop and from there writes data.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	CC,SLI	0A	9
TIC	Buffer			

The Write Continue channel program reads the response to the last message block. If the response is positive, chaining takes place to the next Write Text command.

CHANNEL PROGRAMS FOR IBM 1050 LEASED LINES

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write positive or negative response	Table	CC,SLI	16	1
TIC	Buffer			

The Read Continue channel program writes the appropriate response to a block of data and then chains to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response	LCB		08	9
Write EOA sequence	Table	CD	09	1
TIC	Idles			

The Write Initial channel program places the line in control mode, addresses a terminal, and reads the response. An interrupt is taken on the Read Response, after which buffers are tposted to the outgoing MH. Restart is made at the Write EOA sequence, which transfers-in-channel to the Idles loop and from there writes data.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	CC,SLI	0A	9
TIC	Buffer			

The Write Continue channel program reads the response to the last message block. If the response is positive, chaining takes place to the next Write Text command.

Write Conversational Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOA	Table	CD,SLI	09	1
TIC	Idles			

The Write Conversational channel program writes end-of-address and then chains to writë data.

CHANNEL PROGRAMS FOR IBM 1050 DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		SLI	06	1
Write EOT sequence		CD	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables and then enables the line adapter so that a remote terminal may dial the CPU. An interrupt is taken on the enable so that TCAM can set internal switches. Fifteen pad characters are sent by the CPU, followed by an EOT sequence; this places the terminal in control mode. Two polling characters are sent and then a Read Response that specifies a data count of two, with wrong length indication not suppressed, while the length of the response character is one byte. The effect of this technique is as follows:

1. *Positive response*: The response character and the first byte of the message are read under control of the Read Response CCW. This reduces the data count to zero and causes data chaining to take place. The second and subsequent bytes of the message are read under control of the address and count fields of the Read Data CCW. Execution continues in the channel with an interrupt occurring only at receipt of an EOB or EOT.
2. *Negative response*: This response causes channel end and device end with unit exception and wrong length record indicated.

The Read Initial channel program then transfers-in-channel to the address in the buffer CCW to read data.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write positive (ACK) or negative (NAK) response	Table	CC,SLI	16	1
TIC	Buffer			

The Read Continue channel program sends a positive or negative response to the previous message block and continues reading data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial	T entry	CC,SLI	11	X
Write pad characters	Table	CD,SLI	05	15
Write EOT sequence	Table	CD,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response to address	LCB	SLI	08	9
Write EOA	Table	CD,SLI	09	1
TIC	Idles			

The Write Initial channel program disables the line and then dials a terminal. When the remote terminal answers, the CPU sends the pad characters and the EOT sequence, which places the terminal in control mode. The address characters select the component, which responds to the addressing. End-of-address terminates addressing, and then the Write Initial channel program transfers-in-channel to the Idles loop and from there to write data. The *X* count value depends on the number of dial digits specified in the terminal entry.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	CC,SLI	0A	9
TIC	Buffer			

The Write Continue channel program reads the response to the last message block. If the response is positive, chaining takes place to the next Write Text command.

Write Conversational Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOA	Table	CD,SLI	09	1
TIC	Idles			

The Write Conversational channel program writes End-of-Address character and then transfers-in-channel to a Write Idles loop to write data.

CHANNEL PROGRAM FOR IBM 1050 W/ATTENTION FEATURE FOR TSO MONITOR

Monitor After a Read or Write

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	10	3
Read Response	LCB	CC,SLI,SKP	10	1
Write EOA	Table	CC,SLI	10	1
Prepare	LCB		10	1

This channel program resets the 1050 with the EOT sequence and reads the generated response. The terminal is then put in receive mode and the keyboard is locked. The TCU is then prepared to receive an attention request from the 1050. This request is generated by pressing the Attention Key at the 1050.

CHANNEL PROGRAMS FOR IBM 1060 TERMINALS

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write positive (ACK) or negative (NAK) response	Table	SLI	16	1

The Read Continue channel program sends a positive or negative response to the previous message block and continues reading data to the previous block.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response	LCB		08	9
Write EOA sequence	Table	CD	09	1
TIC	Idles			

The Write Initial channel program places the line in control mode, addresses a terminal, and reads the response. An interrupt is taken on the Read Response, after which buffers are tposted to the outgoing MH. Restart is made at the Write EOA sequence, which transfers-in-channel to the Idles loop and from there writes data.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	CC,SLI	0A	9
TIC	Buffer			

The Write Continue channel program reads the response to the last message block. If the response is positive, chaining takes place to the next Write Text command.

CHANNEL PROGRAMS FOR IBM 2741 LEASED

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	07	3
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program sends a write EOT sequence and then prepares the control unit to receive a message from a terminal. The Sense operation informs the CPU of the status of the terminal through the Read Response. The Read Initial channel program then transfers-in-channel to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOA sequence	Table	CD,SLI	09	1
Write idle characters	Table	CD,SLI	05	15
TIC	Idles			

The Write Initial channel program sends an EOA sequence to set up the terminal and writes 15 idle characters on the line. The program then transfers-in-channel to a write command.

CHANNEL PROGRAM FOR IBM 2741 DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable	LCB	CC,SLI	11	1
Enable	LCB	SLI	06	1
Prepare	LCB	CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables and then enables the line to receive a call. TCAM takes an interrupt on the Enable to set internal switches. The Prepare command conditions the control unit to receive a message. Read Response reads the response from the terminal and then chains to read data by transferring-in-channel.

Note: *The Write Initial channel program for 2741 Dial is the same as for 2741 Leased. TCAM, however, does not dial a 2741; the user calls to establish the connection.*

CHANNEL PROGRAMS FOR IBM 2741 LEASED AND DIAL FOR TSO MONITOR

Monitor After a Read

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOA	Table	CC,SLI	10	1
Prepare	LCB		10	1

This monitor channel program first restores the keyboard of the terminal by sending an EOA. The TCU is then prepared to receive the EOT generated at the terminal when the Attention Key is pressed.

Monitor After Write

Operation	*Address*	*Flags*	*TP Code*	*Count*
Prepare	LCB		10	1

This monitor channel program prepares the TCU to receive the EOT from the terminal. Since the terminal keyboard was locked because it was in receive mode, only a Break (Attention Key) can be sent from the terminal.

CHANNEL PROGRAMS FOR IBM 2740 COMMUNICATION LINES IBM 2740 BASIC CHANNEL PROGRAM

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	07	3
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program sends a write EOT sequence and then prepares the control unit to receive a message from a terminal. The Sense operation informs the CPU of the status of the terminal through the Read Response. The Read Initial program then transfers-in-channel to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CD,SLI	01	3
Write EOA sequence	Table	CD,SLI	09	1
Write idle characters	Table	CD,SLI	05	15
TIC	Idles			

The Write Initial channel program sends an EOT and EOA sequence for preparing the terminal. It then writes 15 idle characters and transfers-in-channel to a Write command.

IBM 2740 WITH CHECKING

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program sends a Write EOT sequence, then prepares the control unit to receive a message from a terminal. The Sense operation informs the CPU of the status of the terminal through the Read Response. The Read Initial program then transfers-in-channel to read data.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write circle Y or circle N	Table	CC,SLI	16	1
TIC	Buffer			

The Read Continue channel program is initiated after a Read Initial operation. The program writes the response character and then transfers-in-channel to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CD,SLI	01	3
Write EOA sequence	Table	CD,SLI	09	1
Write idle characters	Table	CD,SLI	05	15
TIC	Idles			

The Write Initial channel program sends an EOT and EOA sequence for preparing the terminal. It then writes 15 idle characters and transfers-in-channel to a Write command.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	CC,SLI	0A	9
TIC	Buffer			

The Write Continue channel program reads the response after a Write Initial operation and then transfers-in-channel to a Write Text command in the buffer.

Write Conversational Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOA	Table	CD,SLI	09	1
TIC	Idles			

The Write Conversational channel program writes End-of-Address character and then transfers-in-channel to a Write Idles loop to write data.

IBM 2740 WITH DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		SLI	06	1
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables and then enables the line to receive a call. TCAM takes an interrupt on the Enable to set internal switches. The Prepare command conditions the control unit to receive a message. Read Response reads the response from the terminal and then chains to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial	T entry	CC,SLI	11	X
Write pad characters	Table	CD,SLI	05	15
Write EOT sequence	Table	CD,SLI	01	3
Write EOA plus idles	Table	CD,SLI	09	16
TIC	Idles			

The Write Initial channel program disables the line and then dials the specified terminal. The channel program sends 15 pad characters before the EOT sequence. An EOA character plus 15 idle characters are sent and then the program transfers-in-channel to write text. The *X* count value depends on the number of dial characters specified in the terminal entry.

IBM 2740 WITH DIAL AND CHECKING

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		SLI	06	1
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables and then enables the line to receive a call. The Prepare command conditions the control unit to receive a message. Read Response reads the terminal's response and then chains to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial	T entry	CC,SLI	11	X
Write pad characters	Table	CD,SLI	05	15
Write EOT sequence	Table	CD,SLI	01	3
Write EOA plus idles	Table	CD,SLI	09	16
TIC	Idles			

The Write Initial channel program disables the line and then dials the specified terminal. The channel program sends 15 pad characters before the EOT sequence. An EOA character plus 15 idle characters are sent and then the program transfers-in-channel to write text. *X* represents the number of dial digits for the terminal.

IBM 2740 WITH DIAL AND TRANSMIT CONTROL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		SLI	06	1
Write EOT sequence		CD	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables and then enables the line adapter so that a remote terminal may dial the CPU. After the Enable, TCAM waits for an interrupt from the terminal, after which the channel program resumes. Fifteen pad characters are sent by the CPU, followed by an EOT sequence; this places the terminal in control mode. Two polling characters are sent and then a Read Response that specifies a data count of two. The effect of this technique is as follows:

1. *Positive response*: The response character and the first byte of the message are read under control of the Read Response CCW. This reduces the data count to zero and causes data chaining to take place. The second and subsequent bytes of the message are read under control of the address and count fields of the Read Data CCW. Execution continues in the channel with an interrupt occurring only at receipt of an EOB or EOT.
2. *Negative response*: This response causes channel end and device end with unit exception and wrong length record indicated.

The Read Initial channel program then transfers-in-channel to the address in the buffer CCW to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial	T entry	CC,SLI	11	X
Write pad characters	Table	CD,SLI	05	15
Write EOT sequence	Table	CD,SLI	01	3
Write EOA plus idles	Table	CD,SLI	09	16
TIC	Idles			

The Write Initial channel program disables the line and then dials the specified terminal. The channel program sends 15 pad characters before the EOT sequence. An EOA character plus 15 idle characters are sent and then the program transfers-in-channel to write text. *X* represents the number of dial digits for the terminal.

IBM 2740 WITH DIAL, TRANSMIT CONTROL, AND CHECKING

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		SLI	06	1
Write EOT sequence		CD	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables and then enables the line adapter so that a remote terminal may dial the CPU. After the Enable, TCAM waits for an interrupt from the terminal, after which the channel program resumes. Fifteen pad characters are sent by the CPU, followed by an EOT sequence; this places the terminal in control mode. Two polling characters are sent and then a Read Response that specifies a data count of two. The effect of this technique is as follows:

1. *Positive response*: The response character and the first byte of the message are read under control of the Read Response CCW. This reduces the data count to zero and causes data chaining to take place. The second and subsequent bytes of the message are read under control of the address and count fields of the Read Data CCW. Execution continues in the channel with an interrupt occurring only at receipt of an EOB or EOT.
2. *Negative response*: This response causes channel end and device end with unit exception and wrong length record indicated.

The Read Initial channel program then transfers-in-channel to the address in the buffer CCW to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial	T entry	CC,SLI	11	X
Write pad characters	Table	CD,SLI	05	15
Write EOT sequence	Table	CD,SLI	01	3
Write EOA plus idles	Table	CD,SLI	09	16
TIC	Idles			

The Write Initial channel program disables the line and then dials the specified terminal. The channel program sends 15 pad characters before the EOT sequence. An EOA character plus 15 idle characters are sent and then the program transfers-in-channel to write text. *X* represents the number of dial digits for the terminal.

IBM 2740 (DIAL WITH A CONNECTION)

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program sends a write EOT sequence, then prepares the control unit to receive a message from a terminal. The Sense operation informs the CPU of the status of the terminal through the Read Response. The Read Initial program then transfers-in-channel to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CD,SLI	01	3
Write EOA sequence	Table	CD,SLI	09	1
Write idle characters	Table	CD,SLI	05	15
TIC	Idles			

The Write Initial channel program sends an EOT and EOA sequence for preparing the terminal. It then writes 15 idle characters and transfers-in-channel to a Write command.

IBM 2740 WITH CHECKING (DIAL WITH A CONNECTION)

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	11	3
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program sends a Write EOT sequence, then prepares the control unit to receive a message from a terminal. The Sense operation informs the CPU of the status of the terminal through the Read Response. The Read Initial program then transfers-in-channel to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CD,SLI	01	3
Write EOA sequence	Table	CD,SLI	09	1
Write idle characters	Table	CD,SLI	05	15
TIC	Idles			

The Write Initial channel program sends an EOT and EOA sequence for preparing the terminal. It then writes 15 idle characters and transfers-in-channel to a Write command.

IBM 2740 WITH STATION CONTROL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response	LCB		08	9
Write EOA sequence	Table	CD	09	1
TIC	Idles			

The Write Initial channel program places the line in control mode, addresses a terminal, and reads the response. An interrupt is taken on the Read Response, after which buffers are tposted to the outgoing MH. Restart is made at the write EOA sequence, which transfers-in-channel to the Idles loop and from there writes data.

IBM 2740 WITH STATION CONTROL AND CHECKING

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	2
Read Response	LCB		08	9
Write EOA sequence	Table	CD	09	1
TIC	Idles			

The Write Initial channel program places the line in control mode, addresses a terminal, and reads the response. An interrupt is taken on the Read Response, after which buffers are tposted to the outgoing MH. Restart is made at the Write

EOA sequence, which transfers-in-channel to the Idles loop and from there writes data.

IBM 2740 WITH TRANSMIT CONTROL (DIAL WITH A CONNECTION)

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write EOA sequence	Table	CD,SLI	09	1
Write idle characters	Table	CD,SLI	05	15
TIC	Idles			

The Write Initial channel program sends an EOT and EOA sequence for preparing the terminal. It then writes 15 idle characters and transfers-in-channel to a Write command.

IBM 2740 WITH TRANSMIT CONTROL AND CHECKING (DIAL WITH A CONNECTION)

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write EOA sequence	Table	CD	09	
TIC	Idles			

The Write Initial channel program places the line in control mode. The program then issues the write EOA sequence, transfers-in-channel to the Idles loop, and from there writes data.

CHANNEL PROGRAMS FOR WORLD TRADE TELEGRAPH

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Prepare		CC,SLI	11	1
Sense	LCB	CC,SLI	11	1
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program prepares the control unit to receive a message from a terminal. The Sense operation informs the CPU of the status of the terminal through the Read Response. The Read Initial program then transfers-in-channel to a Read Text command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CD,SLI	01	3
Write letters shift	Table	CD,SLI	17	1
Write mark characters	Table	CD,SLI	05	19
Write	WRU	CC,SLI	07	1
Read Response	LCB		08	24
Write EOA sequence	Table		09	1
TIC	Idles			

The Write Initial channel program writes an EOT sequence, sends letters shift to ensure that the terminal motor is on, sends 19 mark characters to condition the line, and writes a WRU on the line, and reads the response. An interrupt is taken on the Read Response, after which the buffers are tposted to outgoing MH. Restart is at the Write EOA sequence, which transfers-in-channel to the Idles loop and writes data.

IBM 2260 REMOTE CHANNEL PROGRAMS

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	3
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. Thus, when there is a one-byte positive response, the response is followed by data. This reduces the count to zero and causes data chaining to read the rest of the data until an EOB or EOT is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write positive (ACK) or negative (NAK) response	Table	CC,SLI	16	1
TIC	Buffer			

The Read Continue channel program sends a positive or negative response to the previous message block and continues reading data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write address	T entry	CC,SLI	07	2
Read Response	LCB		08	9

The Write Initial channel program writes an EOT sequence followed by an address. After the Read Response, the buffers are tposted to MH and data is transferred to the line by EXCP.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	CC,SLI	0A	9
TIC	Buffer			

The Write Continue channel program reads the response to the last message block. If the response is positive, chaining takes place to the next Write Text command.

IBM 2260 LOCAL CHANNEL PROGRAMS

In local mode the channel programs simply read data or write data.

IBM 3270 LOCAL CHANNEL PROGRAMS

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Select,		CC,SLI	11	1
TIC	Buffer			

The Select operation causes transfer of the data in the device buffer to the control unit buffer. The channel program then transfers-in-channel to the first text CCW, which is next to be executed.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Select		CC,SLI	11	1
TIC	Buffer			

Erase All Unprotected Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
EAU		None	26	1

This operation erases all the unprotected fields on the display device. No data is transmitted on an EAU channel program.

CHANNEL PROGRAMS FOR IBM 3670 BROADCAST TERMINAL

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT	Table	CC,SLI	01	03
Write Addressing Characters	T Entry	CC,SLI	07	
TIC	Idles			

The Write Initial channel program places the line in control mode by sending the EOT. The program then sends unique Addressing Characters, which cause the data to go to all broadcast terminals on the line. The channel program then transfers-in-channel to the idles loop to write data.

CHANNEL PROGRAMS FOR IBM 7770 (DIAL)

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		SLI	06	1
Write CPU ID (if ID is specified)	T entry	CC,SLI	0B	X
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables and then enables the line. The CPU ID is written if this is specified, and then the program chains to a Read Response. The *X* count value is the length of the CPU ID specified in the invitation list.

Write Initial Channel Program

This program simply writes data to the 7770.

CHANNEL PROGRAMS FOR TTY MODELS 33 AND 35 TWX LINES

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		SLI	06	1
Write CPU ID	T entry	CC,SLI	0B	X
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program disables the line and sets the enable latch within the line adapter. This permits the terminal to dial the CPU. The Write CPU ID command writes the CPU identification, which is assigned by the invitation list for the line. A Read Response command is then issued, followed by a TIC to a Read Text in the buffer. *X* is the length of the CPU ID specified in the invitation list.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial	T entry	CC,SLI	11	X
Read ID	LCB	SLI	IE	Y

The Write Initial channel program disables and then dials the specified terminal. If the identification received is valid, the program restarts on the Idles loop and writes data. If the ID is invalid, the channel program is terminated. *X* represents the number of dial digits for the terminal and *Y* represents the length of the CPU ID specified in the invitation list.

CHANNEL PROGRAM FOR TWX FOR TSO MONITOR

Monitor After a Read or Write

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write X-On,X-Off	Constant	CC,SLI	10	4
TIC	*–8			

This monitor channel program writes X-On,X-Off characters until the break key is pressed. The X-On,X-Off characters provide an audible indication that the CPU is active and ready to receive data.

CHANNEL PROGRAMS EMPLOYING THE AUTO POLL FEATURE

The devices that use this feature are

IBM 1030
IBM 1050 (nonswitched)
IBM 1060
IBM 2740 (with station control)
IBM 2740 (with station control and checking)
BSC Multipoint

	Operation	*Address*	*Flags*	*TP Code*	*Count*
	Write EOT sequence	Table	CC,SLI	01	X
	Poll	List	CC,SLI	11	Y
	TIC	label A			
	TIC	label B			
A	Poll	List	CC,SLI	11	Z
	TIC	label A			
B	Read	Buffer	CD,SLI	04	2
	TIC	Buffer			

This feature employs the Read Initial type of channel program. First, a write EOT sequence command is sent, followed by a poll of the addresses in the invitation list. If no positive responses are returned, the program transfers-in-channel to poll another list. If there are positive responses, the Read Initial program transfers-in-channel to a Read Response command, and from there chains to a Read Text in the buffer. *X* represents the number of EOTs that depend on the type of terminal (1 for BSC, 3 for all others), *Y* represents the position in the invitation list, and *Z* is the length in bytes of the invitation list.

CHANNEL PROGRAMS FOR IBM BSC MULTIPOINT LINES

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write polling characters	List	CC,SLI	03	2
Read Response	Buffer	CD,SLI	04	2
TIC	Buffer			

The Read Initial channel program places the line in control mode by sending the EOT sequence, polls the terminal, and then reads the response. The Read Response command has a data count of two. This reduces the count to zero and causes data chaining to read the rest of the data until an ETB or ETX is received or the count is zero. A negative response causes channel end and device end with unit exception and wrong length indicated. Line End Appendage finds the polling restart TP code, reinitializes for the next terminal to be polled, and returns control to IOS for execution of the CCWs.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK response	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel programs writes the appropriate response to a block of data and then chains to read data.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT sequence	Table	CC,SLI	01	3
Write addressing characters	T entry	CC,SLI	07	
Read Response	LCB		08	9

The Write Initial channel program places the line in control mode, addresses a terminal, reads the response (ACK-1), and then begins transmission of data.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	CC,SLI	0A	9
TIC	Buffer			

The Write Continue channel program reads the response to the last message block. If the response is positive, chaining takes place to the next Write Text command.

CHANNEL PROGRAMS FOR BSC DEVICES (BINARY SYNCHRONOUS COMMUNICATION)

The devices supported under BSC channel programs are:

IBM 2770
IBM 2780
IBM 2790 Data Communications System
IBM 3780 Data Communications System
IBM 1130 Computing System
IBM System/360, all models 20 and higher

CHANNEL PROGRAMS FOR S/360 to S/360 POINT-TO-POINT

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Prepare		CC,SLI	11	1
Read Inquiry	LCB		0C	11
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program prepares the control unit to receive an inquiry signal, which is read by the Read command. The program then writes an ACK-0 and transfers-in-channel to a Read command in the buffer.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Countt*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write Inquiry	Table	CC,SLI	0D	1
Read Response	LCB	SLI	08	2

The Write Initial channel program writes an inquiry, reads the response (ACK-0), and then begins transmission of data.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI		9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 TO 1130 POINT-TO-POINT

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Prepare		CC,SLI	11	1
Read Inquiry	LCB		0C	11
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program prepares the control unit to receive an inquiry signal, which is read by the Read command. The program then writes an ACK-0 and transfers-in-channel to a Read command in the buffer.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write Inquiry	Table	CC,SLI		1
Read Response	LCB	SLI	08	2

The Write Initial channel program writes an inquiry, reads the response (ACK-0), and then begins transmission of data.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI	0A	9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 TO 2770 POINT-TO-POINT

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Prepare		CC,SLI	11	1
Read Inquiry	LCB		0C	11
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program prepares the control unit to receive an inquiry signal, which is read by the Read command. The program then writes an ACK-0 and transfers-in-channel to a Read command in the buffer.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write Inquiry	Table	CC,SLI	0D	1
Read Response	LCB	SLI	0E	2
Write Escape sequence (STX, ESC or DC,ETB)	T entry	CC,SLI	07	X
Read Response	LCB	SLI	08	2

The Write Initial channel program writes an inquiry, reads the response to that inquiry (ACK-0), writes an escape sequence, reads the response (ACK-1), and then begins transmission of data. *X* represents the length of the addressing sequence specified in the terminal entry.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI		9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 TO 2780 POINT-TO-POINT

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Prepare		CC,SLI	11	1
Read Inquiry	LCB		0C	11
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program prepares the control unit to receive an inquiry signal, which is read by the Read command. The program then writes an ACK-0 and transfers-in-channel to a Read command in the buffer.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write Inquiry	Table	CC,SLI	0D	1
Read Response	LCB	SLI	0E	2
Write Escape sequence STX,ESC or DC,ETB	T entry	CC,SLI	07	X
Read Response	LCB	SLI	08	2

The Write Initial channel program writes an inquiry, reads the response (ACK-0), writes the escape sequence, reads the response to the escape sequence (ACK-1), and then begins transmission of data. *X* represents the length of the addressing sequence specified in the terminal entry.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI		9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 TO 3735 DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		CC,SLI	06	1
Read ID Inquiry	LCB	SLI	18	16
Write ID (if ID is specified)	List	CD,SLI	0B	X
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program disables the line and enables the control unit. The program then reads the inquiry (and writes the CPU ID, if specified). It then writes an ACK-0 and chains to a Read Text command in the buffer. X is the length of the CPU ID.

Read Initial Channel Program with Connection Established

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program reads the inquiry, writes an ACK-0, and then chains to a Read Data command.

Read Initial Channel Program—CPU Initiates Contact

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial digits	T entry	CC,SLI	11	X
Write CPU ID (if ID is specified)	List	CD,SLI	0B	Y
Write Inquiry	Table	CC,SLI	OD	1
Read ID ACK-0	LCB	SLI	1A	17
Write EOT	Table	CC,SLI	01	1
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

This Read Initial channel program disables the line and dials the station. The program writes the CPU ID, if specified, and writes an ENQ character. The response is checked. The channel program then writes an EOT character and reads the inquiry from the station. The Read Initial Channel program then writes an ACK-0 and continues to read data from the station.

Read Initial Channel Program—CPU Yields the Right to Transmit

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT	Table	CC,SLI	01	1
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program writes an EOT character and then reads the inquiry from the station. The Read Initial channel program then writes an ACK-0 and continues to read data from the station.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI	0A	9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 to S/360 DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		CC,SLI	06	1
Read ID Inquiry	LCB	SLI	18	16
Write ID (if ID is specified)	List	CD,SLI	0B	X
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program disables the line and enables the control unit. The program then reads the inquiry (and writes the CPU ID, if specified). It then writes an ACK-0 and chains to a Read Text command in the buffer. *X* represents the length in bytes of the user-specified ID in the invitation list.

Read Initial Channel Program with Connection Established

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	
TIC	Buffer			

The Read Initial channel program reads the inquiry, writes an ACK-0, and then chains to a Read Data command.

Read Initial Channel Program—CPU Yields the Right to Transmit

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT	Table	CC,SLI	01	1
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program writes an EOT character and then reads the inquiry from the station. The Read Initial channel program then writes an ACK-0 and continues to Read Data from the station.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial	T entry	CC,SLI	11	X
Write CPU ID (if ID is specified)	List	CD,SLI	0B	Y
Write Inquiry	Table	CC,SLI	0D	1
Read ID ACK-0	LCB	SLI	1A	17

The Write Initial channel program disables the line and dials the station. The program writes the CPU ID, if specified, and writes an ENQ character. The response is read and the ID is checked. The buffers are tposted to MH, and the channel program restarts at a Write command. *X* represents the number of dial digits for a terminal, and *Y* is the length of the CPU ID.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI	0A	9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 TO 1130 DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		CC,SLI	06	1
Read ID Inquiry	LCB	SLI	18	16
Write ID (if ID is specified)	List	CD,SLI	0B	X
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program disables the line and enables the control unit. The program then reads the inquiry (and writes the CPU ID, if specified). It then writes an ACK-0 and chains to a Read Text command in the buffer. *X* is the length of the CPU ID.

Read Initial Channel Program with Connection Established

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program reads the inquiry, writes an ACK-0, and then chains to a Read Data command.

Read Initial Channel Program—CPU Yields the Right to Transmit

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT	Table	CC,SLI	01	1
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program writes an EOT character and then reads the inquiry from the station. The Read Initial channel program then writes an ACK-0 and continues to read data from the station.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI	0A	9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 TO IBM 2770 DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		CC,SLI	06	1
Read ID Inquiry	LCB	SLI	18	16
Write ID (if ID is specified)	List	CD,SLI		X
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program disables the line and enables the control unit. The program then reads the inquiry (and writes the CPU ID, if specified). It then writes an ACK-0 and chains to a Read Text command in the buffer. *X* is the length of the CPU ID.

Read Initial Channel Program with Connection Established

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	
TIC	Buffer			

The Read Initial channel program reads the inquiry, writes an ACK-0, and then chains to a Read Data command.

Read Initial Channel Program—CPU Yields the Right to Transmit

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT	Table	CC,SLI	01	1
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program writes an EOT character and then reads the inquiry from the station. The Read Initial channel program then writes an ACK-0 and continues to read data from the station.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial digits	T entry	CC,SLI	11	X
Write CPU ID (if ID is specified)	List	CD,SLI	0B	Y
Write Inquiry	Table	CC,SLI	0D	1
Read ID ACK-0	LCB	SLI	1A	17
Write Escape sequence	T entry	CC,SLI	07	Z
Read ACK-1	LCB		08	9

The Write Initial channel program disables the line and dials the station. The program writes the CPU ID, if specified, and writes an ENQ character. The response is checked. The buffers are tposted to MH, and the channel program restarts at the Write Escape sequence. The ACK-1 is read by the program and then the program chains to a Write command. *X* represents the number of dial digits for a terminal, *Y* is the length of the CPU ID, and *Z* is a device-dependent variable.

Write Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Response	LCB	SLI		9

The Write Continue channel program checks the response to the last block of data (ACK-0, ACK-1, RVI) and restarts on a Write Data command.

CHANNEL PROGRAMS FOR S/360 TO IBM 2780 DIAL

Read Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Enable		CC,SLI	06	1
Read ID Inquiry	LCB	SLI	18	16
Write ID (if ID is specified)	List	CD,SLI		X
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program disables the line and enables the control unit. The program then reads the inquiry (and writes the CPU ID, if specified). It then writes an ACK-0 and chains to a Read Text command in the buffer. *X* is the length of the CPU ID.

Read Initial Channel Program with Connection Established

Operation	*Address*	*Flags*	*TP Code*	*Count*
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program reads the inquiry, writes an ACK-0, and then chains to a Read Data command.

Read Initial Channel Program—CPU Yields the Right to Transmit

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write EOT	Table	CC,SLI	01	1
Read Inquiry	LCB		0C	17
Write ACK-0	Table	CC,SLI	15	2
TIC	Buffer			

The Read Initial channel program writes an EOT character and then reads the inquiry from the station. The Read Initial channel program then writes an ACK-0 and continues to read data from the station.

Read Continue Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ACK or NAK	Table	CC,SLI	16	2
TIC	Buffer			

The Read Continue channel program writes a response (ACK or NAK) and transfers-in-channel to a Read Data command in the buffer.

Write Initial Channel Program

Operation	*Address*	*Flags*	*TP Code*	*Count*
Disable		CC,SLI	11	1
Dial digits	T entry	CC,SLI	11	X
Write CPU ID (if ID is specified)	List	CD,SLI	0B	Y
Write Inquiry	Table	CC,SLI	0D	1
Read ID ACK-0	LCB	SLI	1A	9
Write Escape sequence	T entry	CC,SLI	07	Z
Read ACK-1	LCB		08	

The Write Initial channel program disables the line and dials the station. The program writes the CPU ID, if specified, and writes an ENQ character. The response is checked. The buffers are tposted to MH, and the channel program restarts at the Write Escape sequence. The ACK-1 is read by the program and then the program chains to a Write command. *X* represents the number of dial digits for the terminal; *Y* represents the length of the CPU ID specified in the invitation list; and *Z* represents the length of the addressing sequence in the terminal entry.

SPECIAL CHANNEL PROGRAMS

In BSC on a Read Continue operation, when a temporary time delay (TTD) sequence (STX ENQ) is received the channel program is as follows:

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write NAK	Table	CC,SLI	16	2
TIC	Buffer			

When, in response to a text request, TCAM receives two RVIs in succession, a WACK character (except for buffered terminals), or an invalid response, TCAM generates the following channel program to correct the problem.

Operation	*Address*	*Flags*	*TP Code*	*Count*
Write ENQ	Table	CC,SLI	5B	1
Read Response	LCB		0A	9

For two RVIs or an invalid response, TCAM retries this channel program seven times. For a WACK character, TCAM performs no retry operation.

(This page left blank intentionally)

Glossary

The following is a listing of the communications terms used in this manual. For a complete listing of all communications terms, refer to the manual *IBM Data Processing Techniques—A Data Processing Glossary*, Order No. GC20-1699.

access method: a combination of an access technique (either queued or basic) and a given data set organization (for instance, sequential, partitioned, indexed sequential, or direct) that allows the programmer to transfer data between main storage and I/O devices.

access method (ACSMETH) work area: a storage space in an application program. This work area contains data necessary for the interface between the application program and the Message Control Program.

application program: a user-provided program that processes the text portions of messages. Application programs runasynchronously with the Message Control Program and are usually located in another partition or region of main storage. TCAM application programs are optional; there may be many or none, depending on the needs of the user.

Auto Answer: a machine feature that allows either a transmission control unit or a station to respond automatically to a call that it receives over a switched line.

Auto Call: a machine feature that allows either a transmission control unit or a station to automatically initiate a call over a switched line. A dialing operation that originates at the central computer must use the Auto Call machine feature.

Auto Poll: A machine feature of a transmission control unit that permits it to handle negative responses to polling without interrupting the central processing unit. At the end of the invitation list, polling is resumed automatically at the beginning of the list.

binary synchronous communications (BSC): data transmission in which character synchronization is controlled by timing signals generated by the device that originates a message (and the device that obtains the message recognizes the *sync pattern* at the beginning of the transmission—the devices are locked in step with one another); contrast with *start-stop transmission*.

block: that portion of a message terminated by an EOB or ETB line-control character or, if this is the last block in the message, by an ETX or EOT line-control character. When end-of-block checking is specified in the STARTMH macro, messages are checked for certain types of transmission and user-specified logical errors on a block-by-block basis.

BSC: see *binary synchronous communications*.

buffer: an area in main storage into which a message segment is read, or from which a message segment is written. Buffers are temporary data-holding areas that are used to compensate for the difference between the rate at which data can be entered from or accepted by a station and the rate at which it can be processed by the central processing unit; buffers also may be used as work areas in TCAM. The size of TCAM buffers is designated by the user. (See also *hardware buffer*.)

buffer prefix: a control area contained within each TCAM buffer. The prefix for the buffer containing the first segment of a message is 30 bytes long, while the prefix for each buffer containing a subsequent segment of the message is 23 bytes long. The user must allow room for the buffer prefix when he specifies his buffer size. TCAM fills the prefix area with buffer control information.

buffer unit: the basic building block from which TCAM buffers are constructed. All units in a particular TCAM system are the same size; this size is specified by the KEYLEN= operand of the INTRO macro.

buffer unit pool: all the buffer units in a particular TCAM system together constitute the buffer unit pool for that system. The number of units in the pool is equal to the sum of theintegers specified by the LNUNITS= and MSUNITS= operands of the INTRO macro.

buffered terminal: a terminal having a hardware buffer. As used in this book, a buffered terminal is an IBM 2740 Model 2 Station or IBM 2770 station whose TERMINAL macro specifies BFDELAY=integer. When the BFDELAY= operand of TERMINAL is coded, messages are sent to the station segment-by-segment; after a segment is sent, the Message Control Program pauses before sending the next segment to allow the station's buffer to empty. During this pause, the MCP may send segments to other stations on the line.

calling: a procedure that establishes a connection over a switched line; a series of electrical signals, corresponding to the telephone number of the station or computer with which contact is to be made, are sent down the line; these pulses or notes cause automatic switching equipment belonging to the common carrier to establish the connection, if the party being called is free to accept the call.

cascade entry: an entry in the terminal table associated with a cascade list.

cascade list: a list of pointers to single, group, or process entries. A message is queued for the valid entry in the list with the fewest messages queued for it.

channel program block (CPB): a TCAM control block used in the transfer of the data between buffer units and message queues maintained on disk. The CPB= operand of the INTRO macro specifies the number of CPBs to be provided in a TCAM system.

checkpoint data set: an optional TCAM data set that contains the checkpoint records used to reconstruct the MCP environment after closedown or system failure, when the TCAM checkpoint/restart facility is utilized.

checkpoint records: records, located in the checkpoint data set, that are used to reconstruct the MCP environment upon restart following closedown or system failure. There are four types of checkpoint records: environment records, incident records, checkpoint request records, and a control record.

checkpoint request record: a checkpoint record taken as a result of execution of a CKREQ macro issued in an application program. The record contains the status of a single destination queue for the application program. The latest checkpoint request record for a message queue is used during restart to cause sending from that queue to the application program to begin with the message that follows the last message sent to the program from that queue at the time the checkpoint request record was taken, rather than with the message following the last message marked serviced.

checkpoint/restart: a TCAM facility that records the status of the teleprocessing network at designated intervals or following certain events. Following system failure or closedown, the checkpoint/restart facility uses the records it has taken to restore the Message Control Program environment as nearly as possible to its status before the failure or closedown.

cold start: start-up of a TCAM Message Control Program following either a flush closedown, a quick closedown, or a system failure. A cold start ignores the previous environment (that is, the MCP is started as if this were the initial start-up), and is the only type of restart possible when no checkpoint/restart facility is used.

command input buffer (CIB): a communication parameter list that is used by Operator Control to process a command. It describes the command sent from the console and contains the command code, the console identification, and the data in the command.

communication parameter list: the interface between TCAM Operator Control and the Master Scheduler for commands entered from the system console.

concentrator: a remote device that groups blocks of messages into a single physical message for transmission.

concentrator data ready queue (DRQ): a TCAM control block that controls message concentration for output to a concentrator.

concentrator device ID table (DVCID): a TCAM work area that defines a concentrator and each terminal attached to it.

concentrator terminal buffer (CTB): a main-storage area used to contain the physical message transmitted to or from a concentrator.

continuation restart: a restart of the TCAM Message Control Program following termination of the Message Control Program because of system failure; the TCAM checkpoint/restart facility is used to restore the MCP environment as nearly as possible to its condition before failure.

control characters: characters transmitted over a line that are not message data, but which cause certain control operations to be performed when encountered by the computer, transmission control unit, or station; among such operations are polling and addressing, message delimiting and blocking, transmission-error checking, and carriage return.

control record: a record, included in a checkpoint data set, that keeps track of the correct environment, incident, and checkpoint request records to use for reconstructing the Message Control Program environment during restart.

CPB: see *channel program block.*

CTB: see *concentrator terminal block.*

data control block (DCB): an area of main storage that serves as a logical connector between the problem program and a data set. The data control block also can be used to provide control information for any transfer of data. A data control block must be created for each TCAM data set except a message queues data set residing in main storage; a DCB macro instruction is used to create a data control block.

Data set:

1. a named, organized collection of logically related records (program data set). The information is not restricted to a specific type, purpose, or storage medium. Among the data sets specifically related to TCAM are the line group data sets, the message queues data sets, the checkpoint data set, the message log data set, and the input and output data sets for a TCAM-compatible application program.
2. a device containing the electrical circuitry necessary to connect data processing equipment to a communication channel; also called a subset, Data-Phone*, modulator/demodulator, or modem.

dead-letter queue: the destination queue for the station or application program named by the DLQ= operand of the INTRO macro instruction. If an invalid destination is detected in a message header by a FORWARD macro instruction, and if no user-exit is specified in the FORWARD macro, that message is sent to the dead-letter queue.

delimiter macro instruction: a TCAM macro instruction that classifies and identifies sequences of functional macro instructions and directs control to the appropriate sequence of functional macro instructions.

descriptor code: under Multiple Console Support, indicates the means of message presentation and message deletion on display devices.

destination: the place to which a message being handled by a TCAM Message Handler is to be sent. A destination may be either a station defined by a TERMINAL macro, a group of stations defined by a TLIST macro, or an application program defined by a TPROCESS macro. One or more destinations may be specified in fields of the message header that are checked by a FORWARD macro, or a single destination may be specified for all messages handled by a particular inheader subgroup by means of the DEST= operand of a FORWARD macro issued in that subgroup.

*Trademark of the American Telephone & Telegraph Co.

destination field: a field in a message header containing the name of a station or application program to which a message is directed.

destination offset: a two-byte index to the termname table entry of a destination or station.

destination queue: a queue on which messages bound for a particular destination are placed after being processed by the incoming group of a Message Handler. A separate destination queue is created for each station defined by a TERMINAL macro specifying queuing by terminal, one for each line whose stations are defined by TERMINAL macros specifying queuing by line, and one for each application-program process entry (defined by a TPROCESS macro) to which the application program may direct GET or READ macros. Destination queues are maintained in message queues data sets that may be located either on disk or in main storage. Queuing messages by destination permits overlap of line usage in I/O operations. See also *process queue*.

device characteristics table (DCT): a collection of entries that describes the characteristics of the terminals (or devices) in the system.

device ID table: see *concentrator device ID table*.

dial: see *calling*.

dial line: see *switched line*.

disabled module: a module that cannot be interrupted during its execution. It must execute from beginning to end once it has gained control

disabling the line: a process whereby TCAM causes the computer to condition either the transmission control unit or the audio response unit to ignore incoming calls on a switched line. Once this is accomplished, the line is available for TCAM to send queued messages to a station on that line. See *enabling the line*.

distribution entry: an entry in the terminal table associated with a distribution list. A distribution entry is created by a TLIST macro.

distribution list: a list of single, group, cascade, or process entries; when a message is directed to the distribution entry associated with this list, TCAM sends the message to each destination named in the list.

DRQ: see *concentrator data ready queue*.

DVCID: see *concentrator device ID table*.

dynamic buffer allocation: the assignment of buffers to a line on an as-needed basis, after a message has started coming in over the line. Dynamic allocation occurs following program-controlled interruptions, and is specified by the PCI= operand of the line group DCB macro. See also *static buffer allocation*.

element: an individual part of a system resource; for example, a buffer.

element request block (ERB): a control area that is used to make requests for buffers for a line group.

enabled module: a module that can be interrupted at any time by an appendage or external event. When the interruption occurs, the enabled module waits for the appendage to complete its processing and then continues.

enabling the line: a process whereby TCAM causes the computer to condition either the transmission control unit or the audio response unit to respond to incoming calls on a switched line. See *disabling the line*.

end-of-address (EOA) character:

1. a control character or characters transmitted on a line to indicate the end of non-text characters (for example, addressing characters).
2. a TCAM character that must be placed in a message if the system is to accommodate routing of that message to several destinations; the character must immediately follow the last destination code in the message header; and must also be specified by the EOA= operand of the FORWARD macro for the message.

environment record: a record of the total teleprocessing environment at a single point in time. The environment record resides in the checkpoint data set; at restart time, an environment record is updated by the contents of incident records that were taken after the environment record was taken, and the updated environment record is then used to reconstruct the Message Control Program environment as it existed before MCP closedown or system failure.

EOA: see *end-of-address character*.

error record: five bytes assigned to each message being processed by a Message Handler; these bytes indicate physical or logical errors that have occurred during transmission on the line or during subsequent processing or queuing of the message, and are checked by error-handling macros in the inmessage and outmessage subgroups of a Message Handler.

error recovery procedures (ERP): a set of internal TCAM routines that attempt to recover from transmission errors.

exchange: a communications switching center.

event control block (ECB): the communication medium between the various components of the control program, as well as between processing programs and the control program. An ECB is the subject of WAIT and POST macro instructions.

FEFO (first-ended first-out): a queuing scheme whereby messages on a destination queue are sent to the destination on a *first-ended first-out* basis within priority groups. That is, higher-priority messages are sent before lower-priority messages; when two messages on a queue have equal priority, the one whose final segment arrived at the queue earliest is sent first.

FIFO (first-in first-out): a queuing scheme whereby equal-priority messages on the same destination queue are sent in the order that their first segments arrived at the queue.

flush closedown: a closedown of the TCAM Message Control Program during which incoming message traffic is suspended and queued outgoing messages are sent to their destinations before closedown is completed; this form of termination is known as a *flush* closedown because unsent messages are flushed from the message queues. See also *quick closedown*.

functional macro instructions: TCAM macros that perform the specific operations required for messages directed to the Message Handler. See also *delimiter macro instructions*.

group entry: an entry in the terminal table associated with a group of terminals having the group-addressing machine feature.

header: that portion of a message containing control information for the message; a header might contain one or more destination fields, the name of the originating station, an input sequence number, a character string indicating the type of message, a priority level for the message, etc. The message header is operated on by macros in the inheader and outheader subgroups of the Message Handler.

header buffer: a buffer containing a header segment.

header segment: a message segment containing all or part of the message header.

held terminal: a terminal that cannot accept messages because of the effect of a HOLD macro.

identification characters (ID characters): characters sent by a BSC station on a switched line to identify the station. ID characters can also be assigned to the computer (by the CPUID= operand of the INVLIST macro); in this case, the computer and the station can exchange ID sequences. TWX stations also use ID characters.

idle: describes a line that is not currently available for transmission of data because IDLE was coded in the OPEN macro for the line group data set containing the line. Such a line may be activated by a STARTLINE operator command.

incident record: a checkpoint record residing in the checkpoint data set on a DASD. An incident record logs a change in station status or in the contents of an option field that occurred since the last environment record was taken. Incident records are used to update the information contained in environment records at restart time after a closedown or system failure.

incoming group: that portion of a Message Handler designed to handle messages arriving for handling by the Message Control Program. See also *outgoing group*.

incoming message: a message being transmitted from a station to the computer.

input data set: a logical data set for a TCAM-compatible application program. The input data set contains all messages or records being sent to the application program from a single process queue. Though it is not located in a physical medium, the input data set requires a DD statement and a DCB macro for its definition and must be activated and deactivated by OPEN and CLOSE macros. See also *output data set*.

input sequence number: a means of ensuring that messages are received from a source in the correct order. The user may place a sequence number in the header of each message entered by a station or application program, and may code a SEQUENCE macro in the incoming group of his Message Handler. The SEQUENCE macro checks the sequence number for each message; if the number is not one more than that assigned to the previous message received from that origin, a bit is turned on in the message error record.

inquiry processing: a TCAM application in which the Message Control Program receives a message from a station, then routes it to an application program that processes the data in the message and generates a reply; the reply is routed by the Message Control Program to the inquiring station. Response time often may be shortened by specifying the lock mode (by a LOCK macro in the Message Handler) and by locating the message queues data set containing the queues for the application program in main storage.

intercepted station: a station to which no messages may be sent. A station is intercepted by issuing a HOLD macro instruction in the outmessage subgroup of a Message Handler; the suspension is either for a specified time interval or until either an operator command or an application program macro instruction is issued to release messages held for the intercepted station.

invalid destination: a specified destination that does not correspond to a valid terminal table entry.

invitation: the process in which the computer contacts a station in order to allow the station to transmit a message if it has one ready.

invitation delay: a period of time (specified by the INTVL= operand of the line group DCB macro), during which outgoing messages are sent to nonswitched polled stations for which receiving has priority over sending (because CPRI=R is coded in the line group DCB macro). This delay is observed for all such stations on a line when the end of the invitation list for that line is reached. The delay in polling is observed for such stations whether or not the computer has any messages to send them. If no invitation delay is specified for such stations, no messages can be sent to them.

invitation list: a series of sets of polling characters or identification sequences associated with the stations on a line; the order in which sets of polling characters are specified (in the INVLIST macro for the line) determines the order in which polled stations are invited to enter messages on the line.

line control block (LCB): an area of main storage containing control information for operations on a line; one LCB is maintained by TCAM for each line in the system.

line control characters: characters that control transmission of data over a line; for example, line control characters delimit messages, cause transmission-error checking to be performed, indicate whether a station has data to send or is ready to receive data.

line group: a set of one or more communication lines of the same type, over which stations with similar characteristics can communicate with the computer.

line group data set: a Message Control Program data set consisting of all the lines in a line group; the messages that are transmitted on these lines constitute the data in this data set. A line group data set is defined by a line group DCB macro instruction, and by a DD statement for each line in the line group.

line group DCB: a data control block created by a line group DCB macro instruction; information in the data control block defines the line group to TCAM.

lock mode: a TCAM facility, invoked in a Message Handler by the LOCK macro, whereby a station entering an inquiry message for an application program is held on the line by the Message Control Program until a response has been returned to it by the application program. Use of the lock mode decreases response time because there are no interruptions on the line before a response is returned. If LOCK is executed and CONV=YES is coded in the STARTMH macro, tete-a-tete interaction (defined in this *Glossary*) is in effect for the station. A station may be placed in lock mode either for the duration of a single inquiry and response (*message* lock mode) or for the duration of several inquiry-response cycles (*extended* lock mode). The type of lock mode is specified in the LOCK macro.

log: a collection of messages or message segments placed on a secondary storage device for accounting or data collection purposes. The TCAM logging facility is invoked by a functional macro instruction issued in a Message Handler.

log data set: a data set consisting of the messages or message segments recorded on a secondary storage medium by the TCAM logging facility. A log data set is defined by means of a BSAM DCB macro instruction that is issued with the DCB macro instructions defining the line group data sets, the message queues data sets, and the checkpoint data set.

logtype entry: an entry in the terminal table associated with a queue on which complete messages reside while awaiting transfer to the logging medium (a logtype entry is not needed if message segments only are to be logged). A logtype entry is created by a LOGTYPE macro.

main-storage queuing: a situation in which TCAM message queues are maintained in main storage.

MCP: see *Message Control Program*.

MCPL: a subtask control block (STCB) entry code field that identifies the type of STCB and therefore, the method necessary to activate the corresponding subtask.

message: a unit of data received from or sent to a station that is terminated by an EOT or ETX control character or, if the CONV= operand of the STARTMH macro is coded CONV=YES, by an EOB or ETX control character. A TCAM message is often divided into a header portion, which contains control information, and a text portion, which contains the part of the message of concern to the party ultimately receiving it.

Message Control Program (MCP): a set of user-defined TCAM routines that identify the teleprocessing network to the System/360 Operating System, establish the line control required for the various kinds of stations and modes of connection, and control the handling and routing of messages to fit the user's requirements.

Message Handler (MH): a sequence of user-specified TCAM macro instructions in the Message Control Program that examine and process control information in message headers, and perform functions necessary to prepare message segments for forwarding to their destinations. One Message Handler must be assigned to each line group by the MH= operand of the line group DCB macro, and one must be assigned to each TCAM-compatible application program by the MH= operand of the PCB macro. The incoming group of an MH handles messages received from either an originating station or an application program; the outgoing group of an MH handles messages prior to their being sent to a destination station or application program.

message header: the part of a message containing control information, such as the destination code (as distinct from the *text* of the message).

message log data set: a set of messages or message segments that are maintained on secondary storage for accounting or other purposes.

message priority: refers to the order in which messages in a destination queue are transmitted to the destination, relative to each other. Higher-priority messages are forwarded before lower-priority messages. Up to 255 different priority levels may be assigned to a single destination (by the LEVEL= operand of the TERMINAL or TPROCESS macro). The priority for each message sent to the destination may be specified in the message header or assigned by a PRIORITY macro; in either case, a PRIORITY macro should be coded in the inheader subgroup handling the message.

message queue: see *destination queue*.

message queues data set: a TCAM data set that contains one or more destination queues. A message queues data set contains messages that have been processed by the incoming group of a Message Handler and are waiting for TCAM to dequeue them, route them through an outgoing group of a Message Handler, and send them to their destinations. Up to three message queues data sets (one in main storage, one on reusable disk, one on nonreusable disk) may be specified for a TCAM Message Control Program.

message retrieval function: allows the user to retrieve a previously sent message by specifying a combination of the message destination and the input (or output) sequence number of the message. The sequence number is assigned by the SEQUENCE macro.

message segment: the portion of a message contained in a single buffer.

message switching: a telecommunications application in which a message is received from a remote station, stored until a suitable outgoing line is available, and then transmittted to its destination station. TCAM message switching can be handled entirely by the Message Control Program.

MH: See *Message Handler*.

multiple-buffer header: a message header that occupies more than one buffer.

multiple routing: the method of sending a message where more than one destination is specified in the header of the message.

multipoint line: a nonswitched line that connects several remote stations to the computer.

network control: the management of a series of points interconnected by communications channels.

new queue: a chain of CPBs for all cylinders in an extent of a disk message queues data set other than the cylinder currently ready for I/O and the cylinder just after it.

next-buffer location: the value of address (disk relative record number) to be used for the first unit of the next buffer of the message that is currently being placed on the related message queue.

next-message location: the value of address (disk relative record number) to be used for the first unit of the first buffer of the next message received for the related message queue.

no-buffer queue: the chain of CPBs for Read operations when no buffers are in the buffer pool.

no-CPB queue: the chain of elements that are to be processed by CPB initialization.

nonreusable disk queueing: the situation in which each record of a disk record message queues data set may be used only once.

nonswitched line: a communication line that links stations for a continuous period, or for regularly recurring periods; also known as a private, leased, or dedicated line.

non-transparent mode: a mode of binary synchronous transmission in which all control characters are treated as control characters (that is, not treated as text). See *transparent mode*.

on-line test (OLT): an optional TCAM facility that permits either a system console operator or a remote-station operator to test transmission control units and remote stations to find out if they work properly.

operator command: a command entered either at an operator control station or at the system console to examine or alter the status of the telecommunications network during execution.

Operator Control address vector table: an MCP area that contains parameters for the Operator Control module.

operator control station: a station eligible to enter operator commands. An application program and the system console may also serve as operator control stations. Operator control stations are designated as such by the PRIMARY= operand of the INTRO macro and by the SECTERM= operand of the TERMINAL and TPROCESS macros. See also *primary operator control station*.

option field: a storage area containing data relating to a particular station, component, line, or application program. Certain Message Handler routines that need source- or destination-related data to perform their functions have access to data in an option field. User-written routines also have access to data in an option field. Option fields are defined by OPTION macros and initialized for each station, line, component, or application program by the OPDATA= operand of the TERMINAL or TPROCESS macro.

option table: a collection of information provided by the user in OPTION macro instructions.

outgoing group: that section of a Message Handler that manipulates outgoing messages after they have been removed from their destination queues. The outgoing group has three types of subgroups—the outheader subgroup, which executes on outgoing header segments; the outbuffer subgroup, which executes on each outgoing segment; and the outmessage subgroup, which does not execute until after the message has been sent to its destination, if possible. See also *incoming group*.

output data set: a logical data set for a TCAM-compatible application program. The output data set contains the messages or records returned from the application program to the Message Control Program by a process entry in the terminal table. An output data set is defined by a DD statement and a DCB macro, and must be activated and deactivated by OPEN and CLOSE macros. See also *input data set*.

output sequence number: a number placed in the header of a message by TCAM that determines the order in which messages were sent to a destination by the computer. When specified in an outheader subgroup, the SEQUENCE macro causes an output sequence number to be placed in the header of each outgoing message; this sequence number is one greater than the sequence number for the last message sent to this destination. See also *input sequence number*.

path switch: an option field setting used as a switch to indicate the order of or the conditional execution of MH macros.

point-to-point line: a communication line that connects a single remote station to the computer. It may be either switched or nonswitched.

polling: a non-contention line management method whereby the computer invites remote stations on multipoint nonswitched lines and remote terminals on point-to-point lines to enter messages. The computer contacts stations in the order specified by the invitation list; each station contacted is invited to enter messages.

polling characters: a set of identifying characters peculiar to either a station or a component of that station; a response to these characters indicates to the computer whether the station has a message to enter.

prefix: see *buffer prefix*.

primary operator control station: an operator control station that receives, in addition to the responses to commands entered by it, the operator awareness message is sent whenever an I/O error occurs and TCAM's error-recovery procedures are unsuccessful in correcting it. The primary operator control station is designated by the PRIMARY= operand of the INTRO macro.

priority: see *message priority* and *transmission priority*.

problem program mode: operating under the control of the message control or application program, rather than under the control of the OS supervisor.

process control block (PCB): an MCP storage area for data that is necessary for communication between the MCP and an application program.

process queue: a destination queue for an application program (see destination queue). A process queue is defined by a TPROCESS macro.

purge I/O: an SVC issued at close time to remove all traffic from teleprocessing lines.

QCB: see *queue control block*.

QCB extension: A TCAM control area that contains the information necessary to execute the OUTMSG subgroup for a terminal attached to a concentrator.

queue: a set of items consisting of:

1. a queue control block (an area in main storage containing control information for the queue), and
2. one or more ordered arrangements of items (the items may be messages, main-storage addresses, etc.).

queue-back chain: a time-sequential record of the sending and receiving message traffic for the terminal or terminals of a specific destination QCB.

queue control block (QCB): a storage area used to associate elements with appropriate subtasks.

quick closedown: a closedown of the TCAM Message Control Program that involves stopping message traffic on each line as soon as any messages being sent or received at the time the request for closedown is received are transmitted.

read-ahead queue: an area of main storage from which an application program obtains work units in advance of their being requested by the application.

ready queue: a chain of elements that represent the work to be performed in the TCAM system.

recall: a method of retrieving a particular message or a part of a message in order to reprocess it or to redirect it.

recalled buffer: a buffer retrieved from the message queue to be reprocessed. This buffer may be a header or a text buffer.

record: a logical unit of data, the length of which is defined by the user through the use of operands of the input or output DCB macro and delimiting characters in the message.

reentrant module: a module that can be executed by more than one task concurrently; that is, a task may be executing a reentrant module before the previous task has finished executing it.

refreshable module: a module that cannot be modified by itself or by any other module during exeuction; that is, a refreshable module can be replaced by a new copy during execution by a recovery management routine without changing either the sequence or the results of processing.

region control task (RCT): a TSO task that determines which task is to occupy a particular TSO region. There is one RCT for each region. The RCT is activated by the TSIP SVC.

relative line number: a number assigned by the user to a communications line of a line group at system generation time or MCP execution time. If a line group is defined at system generation time by a UNITNAME macro, the lines in the group are assigned relative line numbers according to the order in which their hardware addresses are specified in the UNIT= operand of UNITNAME. The line whose address is specified first is relative line number one, that address specified second is relative line number two, etc. If a line group is defined at MCP execution time by concatenated DD statements, the order in which the DD statements for the lines in the line group are arranged determines the relative line numbers for the lines. The line whose DD statement appears first is relative line number one, the statement that appears second is relative line number two, etc.

resident module: a module that resides in main storage of the TCAM system at all times.

resource: any system facility that is required by a job or task; for example, main storage, I/O devices, data sets, buffer pool.

resource control block (RCB): an eight-byte prefix to an element.

restart: to restructure the execution of a routine or system, using the data recorded at a checkpoint.

retry: an error recovery procedure in which the current block of data (from the last EOB or ETB) is re-sent a prescribed number of times, or until accepted or entered correctly.

retry queue: a chain of one CPB for the cylinder on which to have I/O in an extent of a disk message queues data set after the CPBs on the EXCP queue are processed.

reusable disk queuing: a situation in which messages are queued to a wrapped message queues data set; that is, serviced messages are overlaid by new messages entering the system.

rollout/rollin (RORI): an optional feature of the MVT control program configuration that enables an additional region (or regions) of main storage to be temporarily reassigned from one job step to another.

routing code: under Multiple Console Support, indicates the consoles to which the messages should be sent.

secondary destination: any of the destinations specified for a message except the first destination.

segment: the portion of a TCAM message contained in a single buffer.

selection: the process whereby the computer contacts a remote station to send it a message.

sending: the process in which the central computer places a message on a line for transmission to a station (the station *accepts* the message). Sending and receiving are functions of the central computer.

sequence number: see *input sequence number* and *output sequence number*.

serially reusable module: a module that can be executed by only one task at a time. The module reinitializes itself and restores any instructions or any data in the module that were altered during the execution.

single entry: an entry in the terminal table associated with a single station or station component; one such entry must be created (by a TERMINAL macro) for each station in the TCAM system not defined by a group entry.

source offset: the index value into the termname table for the source terminal.

special characters table (SCT): a collection of entries that contain the special characters required for device I/O for each terminal (or device) in the system.

start-stop transmission: data transmission in which each character being transmitted is preceded by a special control signal indicating the beginning of the sequence of data bits representing the character, and is followed by another control signal indicating the end of the data-bit sequence (character recognition by the device that obtains the data depends on the presence of these control signals for each character); contrast with *binary synchronous communications*.

static buffer allocation: the assignment to a line, before transmission over that line, of all buffers to be used to contain the transmitted data. When PCI=N or PCI=R is coded in the line group DCB macro, the number of buffers specified by the BUFIN= or BUFOUT= operand of the line group DCB macro instruction is assigned to a line before incoming or outgoing transmission begins on that line. Once transmission has started, no more buffers are available to handle the data involved in the transmission.

station: either a remote terminal, or a remote computer used as a terminal.

station control block (SCB): a logical extension of the QCB for each station. The SCB contains information used by TCAM to control buffering.

subblock: that portion of a BSC message terminated by an ITB line control character.

supervisor mode: operating under the control of the system supervisor.

switched line: a communication line on which the connection between the computer and a remote station is established by dialing. Also known as a dial line.

system interval: a user-specified time interval during which polling and addressing are suspended on multipoint lines to polled stations. The system interval is specified by the INTVAL= operand of the INTRO macro, and may be changed during TCAM initialization, by a SYSINTVL operator command. The INTERVAL operator command tells TCAM to begin the system interval. The system interval is used to minimize unproductive polling, to minimize CPU meter time, and to synchronize polling on the polled lines in the system. See also *invitation delay*.

task control block (TCB): the consolidation of control information related to a task.

task I/O table (TIOT): a control block constructed by job management to provide I/O support routines (OPEN, CLOSE, EOV) with pointers to JFCBs and allocated devices.

TCAM/TSO buffer: a buffer residing in the TCAM region in which the PRFTSBUF bit in the buffer prefix is on, indicating that the buffer contains a TSO message.

telecommunications: any transmission or reception of signals, writing, sounds, or intelligence of any nature, by wire, radio, or other electromagnetic media.

teleprocessing: the processing by a computer of data entered at a remote station.

terminal: a point in a system at which data can enter, leave, or enter and leave. A terminal can also be a control unit to which one or more input/output devices can be attached.

terminal I/O coordinator (TIOC): the interface between the TSO subsystem and the version of TCAM that supports TSO.

Terminal On-Line Test Executive (TOTE): the facility for on-line testing available with TCAM used to test various terminal configurations in the user on-line environment.

terminal status block (TSB): a control block containing the status of a terminal for each user. The control block resides in main storage with the user job and is rolled in or out with the user job. The TSB indicates what features are associated with the terminal.

terminal table: an ordered collection of information consisting of a control field for the table and blocks of information on each line, station, component, or application program from which a message can originate or to which a message can be sent.

termname table: a table that contains the name of all the terminals in the system in collating sequence.

tete-a-tete: a mode of message handling in which a station operating in lock mode is polled by the computer. The station responds with a message that ends with a character permitting selection to continue. The computer sends a response message, from an application program, that the station interprets as a positive response.

text: that part of the message of concern to the party ultimately receiving the message (that is, the message exclusive of the header, or control, information).

text segment: a portion of a message that contains no part of the message header.

time delay: a halt of a specific operation for a pre-specified amount of time.

time sharing: a method of using a computing system that allows a number of users to execute programs concurrently and to interact with them during execution.

time sharing input QCB (TSID): an area of main storage that contains the addresses of the time sharing routines.

time sharing job control block (TJB): an area of main storage that contains information about a time sharing user and the status of his job. There is one TJB for each user.

Time Sharing Option (TSO): an optional configuration of the Operating System providing conversational time sharing from remote terminals.

tpost: the technique in TCAM by which an element is passed from one queue to another. The TCAM routines specify the element and the queues, and the TCAM Dispatcher actually performs the action.

transient module: a module that resides in a system library on some type of storage device until it is called into the TCAM system for a limited length of time during the execution of a problem program.

transmission: the transfer of coded data by an electromagnetic medium between two points in a telecommunications network.

transmission control unit (TCU): a control unit that serves as an interface between communication lines and a computer for logical operations. The transmission control units supported by TCAM are the 2701 Data Adapter Unit Model 1, the 2702 Transmission Control Model 1, and the 2703 Transmission Control Model 1.

transmission priority: refers to the order in which sending and receiving occur, relative to each other, for a particular station. Transmission priority is specified on a line-group basis by the CPI= operand of the line group DCB macro. The three transmission priorities possible in TCAM are send priority, equal priority, and receive priority. The exact meaning of each priority depends upon the line configuration and type of station. See also *message priority*.

transparent mode: a mode of binary synchronous transmission in which all data, including normally restricted data-link control characters, is transmitted only as specific bit patterns. Control characters that are intended to be effective are preceded by a DLE character.

twait: the TCAM technique in which a subtask waits for an element to process by having the STCB for that subtask placed in the STCB chain of the QCB to which the needed element will be tposted.

unit: see *buffer unit*.

warm start: a restart of the TCAM Message Control Program following either a quick or a flush closedown. The TCAM checkpoint/restart facility is used to restore the MCP environment as nearly as possible to its condition before failure.

work unit: the amount of data transferred from the Message Control Program to an application program by a single GET or READ macro, or transferred from an application program to the MCP by a single PUT or WRITE macro. The work unit may be a message or a record (or, for QTAM-compatible application programs, a segment).

write-to-operator (WTO): an optional user-coded service whereby a message may be written to the system console operator informing him of errors and unusual system conditions that may need correcting.

write-to-operator with reply (WTOR): an optional user-coded service whereby a message may be written to the system console operator informing him of errors and unusual conditions that may need correcting. The operator must key in a response to this message.

zero-length buffer: a buffer that has a zero in the PRFSIZE field of the buffer prefix. This type of buffer is sent by the Line End Appendage to the Message Handler to indicate that there is an error on the line.

zone: that portion of disk records that reside in an algebraic quarter of the Reusable Disk Message Queue data set.

zone boundary: any of four disk records, one at each of the following positions in the Reusable Disk Message Queue data set: the first record, the records 1/4, 1/2, and 3/4 through the entire data set.

(This page left blank intentionally)

Index

GY30-2029-3

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

READER'S COMMENT FORM

OS/MFT and OS/MVT
TCAM Logic

Order No. GY30-2029-3

- How did you use this publication?

  As a reference source □
  As a classroom text □
  As . . . . □

- Based on your own experience, rate this publication . . .

As a reference source:	. . . . . Very Good	. . . . Good	. . . . . Fair	. . . . . Poor	. . . . . Very Poor
As a text:	. . . . . Very Good	. . . . Good	. . . . . Fair	. . . . . Poor	. . . . . Very Poor

- What is your occupation? . . . .

- We would appreciate your other comments; please give specific page and line references where appropriate. If you wish a reply, be sure to include your name and address.

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

YOUR COMMENTS, PLEASE . . .

Your answers to the questions on the back of this form, together with your comments, help us produce better publications for your use. Each reply is carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in using your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

Cut Along Line

Fold   Fold

FIRST CLASS
PERMIT NO. 569
RESEARCH TRIANGLE PARK
NORTH CAROLINA

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

POSTAGE WILL BE PAID BY . . .

IBM Corporation
P. O. Box 12275
Research Triangle Park
North Carolina 27709

Attention: Publications Center, Dept. E01

Fold   Fold

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)