File No. S360-30
Order No. GC30-5004-3

DOS

IBM Systems Reference Library

IBM System/360
Disk Operating System
QTAM Message Control Program

Program Number 360N-CQ-470

PREFACE

The first part of this publication contains information of a general nature and is of interest to anyone, including managers, system analysts, and programmers, involved in planning or implementing a QTAM-controlled telecommunications system to operate under the IBM System/360 Disk Operating System (DOS). The topics discussed include: devices supported; concepts and terminology; QTAM facilities; and applications supported.

The remaining two parts of this publication, beginning with the chapters on nonaudio and audio message handling respectively, describe in detail the problem programming necessary for constructing QTAM message control programs to support telecommunications applications. A thorough understanding of this publication requires a basic knowledge of System/360 machine concepts and the Disk Operating System. The prerequisite publications are:

1. IBM System/360 Principles of Operation, GA22-6821.
2. IBM System/360 Disk Operating System, System Control and System Service Programs, GC24-5036.
3. IBM System/360 Disk Operating System, Data Management Concepts, GC24-3427.
4. IBM System/360 Disk Operating System, Supervisor and Input/Output Macros, GC24-5037.
5. IBM System/360 Disk and Tape Operating Systems, Assembler Specifications, GC24-3414.

The reader should also be familiar with the following publications that apply to equipment in his system configuration:

Telecommunications Control Units:

1. IBM 2701 Data Adapter Unit, Principles of Operation, GA22-6864
2. IBM 2702 Transmission Control, GA22-6846
3. IBM 2703 Transmission Control, GA27-2703

Audio Response Units:

1. Component Description, IBM 7770 Audio Response Unit, Models 1, 2, and 3, GA27-2712
2. IBM System/360 Component Description, IBM 7772 Audio Response Unit, GA27-2711
3. IBM 7772 Audio Response Unit Vocabulary, GA27-2710
4. IBM System/360 Disk Operating System, Vocabulary File Utility Program for the IBM 7772 Audio Response Unit, GC27-6924

Terminal Equipment:

1. IBM 1030 Data Collection System, GA24-3018
2. IBM 1050 Data Communication System, Principles of Operation, GA24-3474
3. IBM 1060 Data Communication System, GA24-3034
4. IBM 2260 Display Station, IBM 2848 Display Control, GA27-2700

Fourth Edition (March 1971)

This publication corresponds to Release 24. It is a major revision of, and renders obsolete C30-5004-2. This publication incorporates changes issued in Technical Newsletters N30-5522, dated October 18, 1968, N30-5527, dated April 15, 1969, and N30-5535, dated July 14, 1969. Changes are periodically made to the information herin: before using this publication in connection with the operation of IBM systems or equipment, refer to the latest SRL Newsletter 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.

This manual has been prepared by the IBM System Development Division, Publications Center, Department E01, P. O. Box 12275, Research Triangle Park, North Carolina 27709. A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be sent to the above address.

In the IBM System/360 Disk Operating System, an access method is a procedure for transferring data between main storage and an input/output device. A variety of access methods is available to the user of the Disk Operating System (DOS). One of these, the Queued Telecommunications Access Method (QTAM), controls data transfer between main storage and remote terminals.

QTAM is an input/output control system that extends the techniques of logical IOCS to the telecommunications environment. Files accessed by the problem programmer are queues of messages coming in from, or going out to, remote terminals via communication lines. Although the time and order of the arrival and departure of messages to and from the central processing unit (CPU) are unpredictable, the programmer can handle the messages as if they were organized sequentially.

Unlike other commonly used access methods, QTAM furnishes far more than the mechanics for input/output operations. In addition to the GET/PUT macro instruction support for message processing programs, QTAM provides a high-level, flexible message control language. QTAM-supplied macro instructions can be used to construct a complete message control program that controls the flow of message traffic from one remote terminal to another (message switching application), and between remote terminals and any message processing programs (message processing applications). An installation-oriented message control program can thus be written in hours rather than in the months previously required for such a programming task.

A QTAM message control program is generated from a number of assembler macro instructions coded by the programmer. Although the assembler macro generator is used, the process followed is similar to that used by a high-level compiler. A QTAM message control program is open-ended. That is, the user can include functions not provided through the QTAM language by employing DOS control program macro instructions, and assembler language instructions and macro instructions.

A message control program is completely device dependent, with all communication lines and terminals identified to the system; but, for the Audio Response Units, only the communication lines must be identified to the system. Through file definition and control information macro instructions, the user specifies his equipment configuration and the main storage areas (buffers) required for his applications. These macros generate the tables and lists of control information that define the environment of the system for the QTAM logic. A primary resource in the telecommunications system is the buffers, the number and size of which are specified by the user. For a nonaudio application, the buffers are allocated to a common buffer pool from which QTAM automatically and dynamically uses them in accordance with immediate requirements. For an audio application, the buffers are strictly associated with the audio lines.

QTAM logic modules are also provided for many procedural functions, such as message code translating, routing of messages, and error checking. By selecting the appropriate macro instructions, the user specifies which of these QTAM logic modules are to be incorporated into his message control program. In this way, the system can be tailored to the exact requirements of the applications being supported.

The message processing program services of QTAM enable a programmer to process messages from a telecommunications network with the same easy-to-use macro instructions that he uses for his local input/output devices. Because a QTAM message control program is used to perform the input/output operations, a message processing program can be device independent except for the device control characters associated with a particular terminal type. The applications programmer is, in effect, completely shielded from the time and device dependent aspects of the telecommunications environment. By using some other access method for a sequentially organized file, the user can completely write and test his message processing programs without ever running in the telecommunications environment. (For example, test input from a card reader can be used for this purpose.) Then, by simply changing the definition of a DTF table, he can reassemble the message processing program to operate under QTAM control.

This publication is devoted primarily to the QTAM facilities provided for the construction of a message control program. Message processing programs are discussed in general terms and only when necessary to give a complete picture of a telecommunications system using QTAM. For detailed information on message processing programs

and the services QTAM provides in supporting them, refer to the publication, IBM System/360 Disk Operating System, QTAM Message Processing Program Services, Form C30-5003.

CONTROL UNITS SUPPORTED

Combinations of IBM 7770 and/or 7772 Audio Response Units attached to a multiplexer channel, and/or any combination of IBM 2701, 2702, or 2703 telecommunications control units attached to the same multiplexer channel may be supported by DOS/QTAM. Up to eight control units can be attached to the multiplexer channel. DOS/QTAM additionally supports the IBM 2848 Display Control attached directly either to the multiplexer or a selector channel.

If using the IBM System/360 Model 25 with the Integrated Communications Attachment, operation and program set-up is the same as for the Model 30 with a 2703 Transmission Control Unit attached to a multiplexer channel (switched and nonswitched lines).

TERMINAL TYPES SUPPORTED

DOS/QTAM supports the following types of terminals:

1. Terminals attached to a multiplexer channel through an IBM 2701, 2702, or 2703 telecommunications control unit:
   - IBM 1050 Data Communication System on a switched network or on a nonswitched network.
   - IBM 1060 Data Communication System on a nonswitched network.
   - IBM 1030 Data Collection System on a nonswitched network. See Note 1.
   - IBM 2260-2848 or 2265-2845 Display Complex (remote) on a nonswitched network (2701 only).

   Note: Within this SRL, all comments which refer to the 2260-2848 Remote are also applicable to the 2265-2845.

   - AT&T 83B3 Selective Calling Stations on a nonswitched network.
   - Western Union Plan 115A Outstations on a nonswitched network.
   - Common Carrier (8-level code) TWX Stations on a switched network (for example, WU Model 33 or 35 Teletypewriter Terminal--dial service).
   - World Trade telegraph terminals (WTTA terminals) on a nonswitched network, attached to a 2701, 2702, or 2703 Control Unit. See Note 2.
   - IBM 2740 Communications Terminal on a nonswitched network, four types:

     Basic 2740

     Basic 2740 with station control

     Basic 2740 with station control and checking

     Basic 2740 with checking.

   - IBM 2740 Communications Terminal on a switched network, four types:

     Basic 2740

     Basic 2740 with transmit control and checking

     Basic 2740 with checking

     Basic 2740 with transmit control.

   DOS/QTAM supports the following device attached directly either to the multiplexer or a selector channel:

   - IBM 2260-2848 Display Complex (Local).

2. Terminals attached to a multiplexer channel through an IBM 2702, or 2703 transmissions control unit equipped with the Auto Poll feature:
   - IBM 1050 Data Communication System on a nonswitched network.
   - IBM 1060 Data Communication System on a nonswitched network.
   - IBM 1030 Data Collection System on a nonswitched network. See Note 1.
   - IBM 2740 Communication Terminal on a nonswitched network when equipped with the Station Control feature:

     Basic 2740 with station control

     Basic 2740 with station control and checking

   - IBM 2740 Model 2 Communication Terminal on a nonswitched network when equipped with station control, with or without checking, and with or without the Buffer Receive feature.
3. Audio terminals attached to a multiplexer channel through an IBM 7770 Model 3 or 7772 Audio Response Unit on a switched network:

- IBM 1001 Data Transmission Terminal
- IBM 1092/3 Programed Keyboard
- IBM 3944 Dial Terminal (World Trade terminal attached only to the IBM 7772 with Dial Terminal Feature) with rotary dial telephone
- Standard rotary dial, push button telephones, or similar terminals

Note: Any reference in this document to the IBM 7770 or 7772 implies their associated terminals.

Note 1: The IBM 1032 Digital Time Unit cannot be attached through a 2701 Data Adapter Unit.

Note 2: Throughout this publication, 'World Trade telegraph terminal' (WTTA terminal) refers to a terminal as defined on page 30 , connected to a control unit that incorporates a World Trade Telegraph Adapter. A World Trade line (WTTA line) is a line connected in the same manner to a WTTA terminal.

MACHINE AND DEVICE REQUIREMENTS

QTAM operates on any System/360 having at least 64K of main storage. The only additions to the minimum requirements of the System/360 Disk Operating System are:

- All telecommunications terminals, except the IBM 2260-2848 local, must be attached to an IBM 2701 Data Adapter, or an IBM 2702 or 2703 Transmission Control Unit, or an IBM 7770 or 7772 Audio Response Unit; they cannot be attached directly to a channel.
- All IBM 2701, 2702[1], 2703, 7770 or 7772 Control Units that operate under QTAM must be attached to the System/360 via the multiplexer channel.
- No device may be operated in burst mode on the multiplexer channel concurrently with the operation of QTAM, except when the QTAM operation involves only the 2260 Local.
- The storage protection feature is required.
- The IBM 1052 Printer-Keyboard is mandatory.

The following additional features may be required if certain optional functions provided by QTAM are desired:

- The interval timer feature, if time-of-day information in messages, the polling interval function, the checkpoint interval function, or the operator control INTREL function is desired or if IBM 2740 Model 2 Terminals are included in the system.
- The line correction feature on IBM 1050 terminals if automatic retry is desired when a transmission error occurs.

GENERAL REQUIREMENTS AND CAPABILITIES

To construct a telecommunications system to operate under control of QTAM, in the Disk Operating System environment, the user must write:

1. A message control program, and
2. Any message processing programs required by his application.

A telecommunications control system created through the use of the QTAM message control language can:

- Establish contact and control message traffic between computer and terminals,
- Dynamically assign and use buffers as required,
- Perform editing of incoming and outgoing messages (for example, code translation, insertion of new fields in message headers),
- Forward messages to destination terminals and message processing programs,
- Take corrective action and provide special handling for messages containing errors,
- Maintain statistical information about message traffic.

---

[1]A switch on the CE panel on the 2702 can be used to place a given line in CE mode for equipment checking. Care must be taken to ensure that no lines are in CE mode when QTAM is used since no ending status will be returned to a SIO command.

This section describes the line and terminal configuration of a telecommunications system operated under control of QTAM and the IBM System/360 Disk Operating System, and defines some terms used in this publication. A telecommunications system (or network) consists of a number of input, output, or combined input/output devices, usually in geographically dispersed locations, connected by one or more communication lines. As used in this publication, a telecommunications system operating under DOS/QTAM may be more specifically defined as a network of terminals connected to a central computer by one or more half-duplex communication lines. A half-duplex line is a line over which data can flow in either direction, but only in one direction at a time.

In communication terminology, special terms are used to represent the media that connect the physical components of a system: communication line, data link, data path, circuit, and channel. In this publication, the term communication line (or line) is used to refer to any medium that connects the physical components of a system, whether a telegraph circuit, a telephone circuit, a private circuit, etc. The term audio communication line (or audio line) is used for lines attached to the audio response units, and always refers to a telephone circuit. Conversely, wherever the audio response units are specified, the associated communication line (or line) is implied.

A terminal is the unit or units of equipment that accepts keyed or punched data as input for sending to the computer and/or produces printed, punched, or visually displayed data as output received from the computer. All messages from one terminal to another pass through the computer; in addition, the computer may itself receive and originate messages for the terminals.

A terminal consists of a control unit and one or more input/output devices. Each such device is called a component. Each input device and each output device is considered a separate component, regardless of whether they are physically combined. For example, an IBM 1050 is referred to as a terminal; its constituent devices, or components, include the IBM 1053 Printer, 1054 Paper Tape Reader, keyboard section of the 1052 Printer-Keyboard, printer section of the 1052 Printer-Keyboard, etc.

An audio terminal is the unit of equipment associated with an audio response unit, that accepts keyed or dialed data as input for sending to the computer and/or produces an audio response as output received from the computer. When an audio terminal has established the connection through an audio communication line, the computer receives the input message and sends the audio output message to the same audio terminal. Audio response units may be referred to as terminals.

Terminals in a telecommunications system operated under DOS/QTAM control are classified as either remote or local terminals. The main distinction is how the terminals are attached to the computer. The IBM 7770 or 7772 Audio Response Unit (ARU) and terminals attached to the computer channel through an IBM 2701, 2702, or 2703 Telecommunications Control Unit (TCU) are classified as remote. With the exception of the IBM 2260 Local, all supported terminals fall under this classification. Remote terminals are usually separated from the computer by a distance sufficient to require common-carrier facilities and transmission techniques to communicate with the computer. The system, however, may include terminals at the same location as the computer, attached to it through a TCU and local cables. Units that are connected directly to a System/360 channel, such as the 2260-2848 Display Complex (Local), are classified as local.

Each remote terminal, each TCU, and each ARU is connected to a communication line by a data set, a modem (modulator demodulator), etc., depending on the kind of communication line and kind of terminal involved. (Terminals connected to the TCU by local cables do not require data sets.) The precise functions of these units vary, but the overall purpose is the same: to provide an interface between terminal and line. This publication uses the term data set to represent any of these units. The programmer need not concern himself in any way with these data sets because their presence exerts no influence on programming. They are defined only to provide a complete, accurate picture of the line and terminal configuration.

The term station means the aggregate of equipment and controls attached to any of the several ends of a communication line. A station can also be described as a terminal (including the terminal components)

plus the equipment by which the terminal is attached to the line.

In this publication, computer is used as a general term for the equipment and programs at the central processing location (CPU, TCU, ARU, etc.), when reference to a specific unit of equipment or programming is not necessary.

TELECOMMUNICATIONS NETWORKS

A telecommunications system consists of either a nonswitched network, a switched network, or a combination of the two. Figure 1 shows the configuration of an installation that includes both a nonswitched network and a switched network.

A nonswitched network consists of a number of private or leased lines that connect the computer to one or more remote terminals. The computer and the terminals are physically connected; that is, the circuits making up the communication lines are continuously established for predetermined time periods during which data transmission may proceed between the computer and the terminals. In this type of system, the computer can, under certain conditions, send messages to more than one terminal on the same line at the same time. The lines comprising a nonswitched network are known as either private, leased, or dedicated lines. Such lines usually are furnished by a common carrier on a contract basis, between specified locations for a continuous period or regularly recurring periods at stated hours, for the exclusive use of one customer.

A switched network consists of a number of remote terminals with which the computer can communicate. The computer and the several terminals are each continually connected, by access lines, to the common-carrier exchanges serving their respective locations. A complete, continuous data path is established between computer and terminal only for the period of time in which transmission takes place. The connection is established by dialing the telephone number of the unit (either terminal or CPU) on the other end. In this type of system, communication can be established between the computer and only one terminal at a time on each line. In this case, line refers to a discrete data path between the telecommunications control unit and the common-carrier exchange. The service provided by the common carrier is typically on a time-used basis.

In a nonswitched network, the physical circuit connections determine which terminals are associated with each line into the computer. In a switched network, the user can, by several means, specify which terminals can communicate with the computer over each line.

A switched network is used by the audio response units, but the connection is always established from an audio terminal by dialing the telephone number of an audio communication line enabled (activated) by the computer. Depending on the dialed number, the terminal can communicate with the computer on a specified line of a specified ARU.

Some communication networks have characteristics typical of both switched and nonswitched networks. In this publication, the term switched network refers to any network in which a direct physical connection between computer and terminal must be established by dialing in order for data transmission to occur. The term nonswitched network refers to a network in which the communication lines linking computer and terminals are continuously established, thus requiring no dialing.

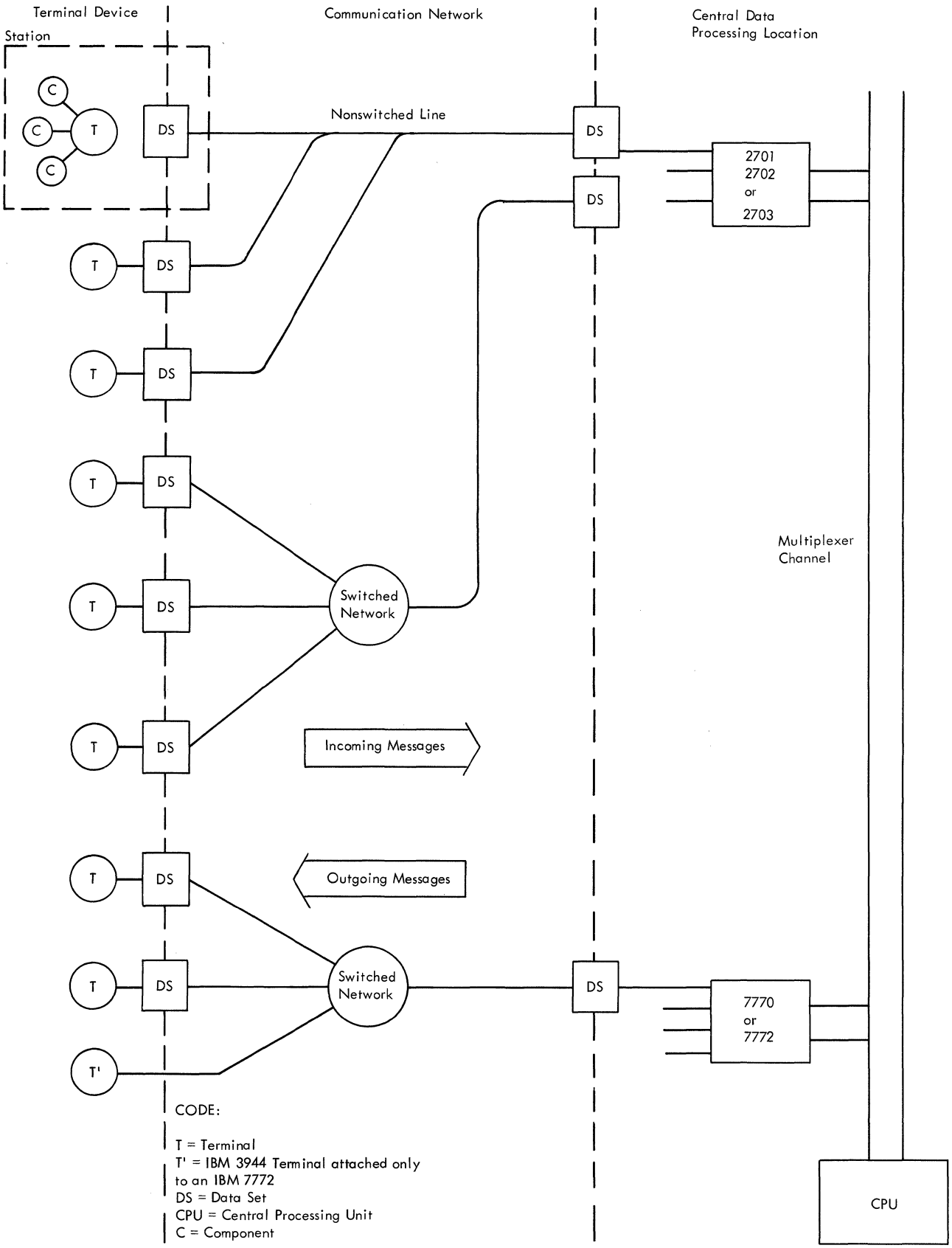


Figure 1. Configuration of a Teleprocessing Installation

MESSAGE CONTROL

Accomplishing efficient systematic supervision of message traffic is the job of the QTAM message control facilities. In some respects, the functions performed by QTAM message control procedures parallel those performed in telecommunications systems that are not computer-oriented.

This section provides a general description of telecommunications systems followed by a discussion of the main functions performed in a computer-based system operated under QTAM control. Subsequent sections of this publication explain how these functions are implemented. (In the following discussion, the term computer includes both the physical equipment at the central location and the Disk Operating System programs. The word terminal is used for the equipment at remote or local locations.)

In any telecommunications system, contact between terminals must be established before a message is sent. In some systems, terminals wishing to send a message contend with one another for use of the line. The first terminal to initiate contact on a line that is not currently in use seizes the line and prevents its use by other terminals until it has concluded its message transmission. A system operated in this manner is called a contention system. Audio communication lines, WTTA lines, and the IBM 2260-2848 local always operate as contention systems.

In other systems, one of the terminals is specified as the control station. This control station initiates all contacts for all other terminals on the line, using a procedure known as polling. Polling is a flexible, systematic, centrally controlled method of permitting terminals on a multiterminal line to transmit without contending for use of the line. The control station periodically contacts the other terminals and invites them to send any messages they have ready. In addition, the control station itself may elect to send a message. A system operated in this manner is called a polling system.

Polling is accomplished by sending on the line one or more polling addresses each of which consists of one or more polling characters. Typically, two characters are used; the first selects the terminal, the second selects the specific component of that terminal. The terminal identified by these characters then sends a response to the control station--a positive response if it has a message to send, a negative response if it does not. The control station may poll a number of terminals and components, in turn, until one is found that has a message ready.

Similarly, when the control station terminal, or any other terminal, has a message to send, it transmits on the line one or more addressing or call-directing characters. As in polling, two characters are often used: the first selects the terminal; the second selects the component. The terminal identified by these characters returns a response. A positive response is returned if the terminal is able to accept the message; a negative response is returned if the terminal cannot accept the message.

The nonaudio communication lines operate within a polling system in which the computer acts as the control station (although certain types of the IBM 2740 Communication Terminal employ techniques similar to those used by a contention system). The entire contention and/or polling telecommunication system operated under QTAM is a centralized system; that is, terminals send their messages, not to other terminals, but to the computer. The computer then relays the messages to the appropriate destination terminals (or to a message processing program) or sends the response to an audio terminal.

QTAM POLLING SYSTEM

The polling and addressing functions are performed in both switched and nonswitched systems, with minor variations.

In a switched network, the line connection must be completed between computer and terminal before message transmission can proceed. The connection may be established by either the computer or a terminal. When the computer is to establish the connection, it dials the telephone number of the terminal (the user provides QTAM with the telephone number of each terminal in the switched network). The connection is established when the terminal responds. The function performed by the computer, in this case, is known as calling. Polling or addressing may then take place. Ordinarily, the computer calls a terminal only for the purpose of addressing the terminal (to send it a message), rather than polling it (to solicit messages). When a terminal is to establish the connection, the user dials the telephone number of the computer (or one of its several numbers). The connection is established when the computer responds; the function performed by the computer in this case is known as answering. Polling or addressing may then take place. Ordinarily, a terminal calls the

computer only in order to be polled for a message it has ready for the computer or another terminal. Note that regardless of whether the computer or the terminal establishes the line connection, message flow from the terminal to the computer is achieved by polling the terminal, and message flow from the computer to a terminal is achieved by addressing the terminal.

Although terminals are permitted to call the computer at any time, the computer, in order to fulfill its function as control station, must be able to accept or reject incoming calls. Therefore, the computer performs a function known as enabling the line. Enabling the line is the process of conditioning the telecommunications control unit to accept incoming calls on a line. The user determines which lines are and which are not to be enabled at a given moment. If a terminal calls in on a line that is currently enabled and that is not in contact with another terminal, the line connection is completed and message transmission (preceded by polling or addressing) can occur. If a terminal calls in on a line that is not currently enabled, contact is not established. However, if the line has been enabled but is occupied with another terminal, the calling terminal receives a busy signal. In either case, the terminal must wait and call again later.

In a nonswitched network, the line connections between computer and terminals are continuously established; hence, the calling, answering, and enabling functions are not required. Only the computer can initiate contact with terminals (except for certain types of IBM 2740 Communications Terminals and the IBM 2260 Display Station (local), which can bid for the computer's attention). After the line connection is established, the computer addresses terminals to send messages to them, and continually and systematically polls terminals to solicit messages from them.

For nonswitched lines, the polling process may be achieved under the control of an entirely programmed capability or under the control of the Auto Poll feature. Throughout this publication, auto polled lines refer to lines polled under the control of the Auto Poll feature, while polled lines refer to lines polled under the control of the program capability.

The contention technique of initiating contact on a line not currently in use is used by WTTA terminals and by the four types of IBM 2740 Communications Terminals which have neither station control nor transmit control.

In a contention system either the terminal operator or the computer can initiate contact on an available line connecting them. In any contention system, it is possible for contact to be initiated at both ends of a communications line at the same time. (For WTTA terminals, refer to the note below.) When this occurs, the teleprocessing system will generally malfunction. This situation is extremely rare (it requires that the terminal operator simultaneously hit the EOT and BID keys when terminating entry of one message and initiating contact to enter another message at the same time that the computer has a message to send to the terminal), and is avoided completely if the terminal operator allows a normal I/O reaction time between EOT and BID. For example, if the terminal operator uses the same finger to hit the EOT and BID keys, "normal" I/O reaction time is provided.

Note: If the first character of an input message is received at the same time as the computer sends a character to the terminal, contention occurs and is resolved as follows:

1. If contention occurs during the user-specified interval required by the terminal motor to reach nominal speed, the terminal is given priority.

2. If contention occurs on a significant character of an output message, the computer is given priority.

AUDIO SYSTEM ON A SWITCHED NETWORK

The line connection must be completed between computer and audio terminal before message transmission can proceed. The connection can be established only by the terminal. The terminal operator dials the telephone number of the computer (or one of its numbers) and the connection is established if the computer allows it.

Although ARU terminals are permitted to call the computer at any time, the computer, in order to fulfill its function as control station, must be able to accept or reject incoming calls. Therefore, the computer performs a function known as enabling the line. Enabling is the process of conditioning the audio control unit to accept incoming calls on a line. The user determines which lines are, at a given moment, to be enabled, and which are not. If a terminal calls in on a line that is currently enabled and that is not in contact with another terminal, the line connection is completed and message transmission can immediately occur. If a terminal calls in

on a line that is not currently enabled, contact is not established. However, if the line has been enabled but is occupied with another terminal, the calling terminal receives a busy signal. In either case, the terminal must wait and call again later.

MESSAGE PROCESSING

Message processing is the most variable of all telecommunications functions. The nature of each user's processing routines depends on the individual application.

QTAM provides macro instructions enabling the user's problem program to obtain messages queued for processing and to place a response message on destination queues. For nonaudio terminals, QTAM also provides a set of macro instructions for examining and modifying control information used by the access method. For detailed information on message processing, refer to the Message Processing Program Services publication listed in the Preface of this manual.

GENERAL CONCEPTS AND FACILITIES

The function of programs constituting support for a telecommunications system is to control, systematically and efficiently, the flow of data in a computer-based telecommunications system, and to perform, concurrently, any required processing of the data. Data enters the system randomly in the form of messages from terminals and/or from programs that generate messages; data is ultimately delivered to one or more terminals or programs that process messages. The messages entered at the terminals consist of two principal parts, the message header, consisting of control information, and the message text or data.

For a number of reasons, the support is logically divided into two categories:

1. The programming required to identify the telecommunications system to the IBM System/360 Disk Operating System, to establish the line control disciplines required for the various types of terminals and modes of connection, and to control the routing of messages in accordance with the user's requirements; and

2. The programming required to process the contents of the messages.

The first category is implemented by routines collectively known as the message control program which is primarily concerned with the message header. The second category is implemented by one or more message processing programs which are primarily concerned with the message text or data.

The paramount reason for dividing telecommunications support into these types of programs is that message flow in the system is random and proceeds at relatively slow speeds (due to the operating speeds of the terminals supported), while the messages, once delivered to the computer, can be processed at computer speeds. To fully utilize the computing system capabilities, message traffic must proceed simultaneously with message processing. Another reason for having separate message control and message processing programs is that while many device-dependent considerations govern the design of a message control program, they do not affect the design of a message processing program. The programmer writing a message processing program need know only the format of the messages and the characteristics of the data they contain to be able to proceed with the program design.

A message control program serves as an intermediary between the remote terminals and any message processing programs. The device-dependent input/output operations are performed by QTAM routines that support the message control program, based on the terminal and line configuration of the user's system as specified in the operands of QTAM macro instructions. To provide maximum efficiency, QTAM uses the operating technique of placing messages on queues on a direct access storage device (DASD), when necessary, and subsequently retrieving these messages for processing. This enables the terminals to be referenced indirectly, in much the same way as local input/output devices are referenced. This is accomplished from a message processing program using language statements such as GET, PUT, OPEN, and CLOSE.

When messages are associated with audio terminals, QTAM places them in main storage queues. A message processing program using language statements such as GET, PUT, OPEN, and CLOSE, retrieves audio input messages from a main storage queue, processes them, and sends the audio response to another main storage queue.

The message control program itself can perform limited processing of the message, in addition to that performed by a message processing program. Some of these processing operations may be required in order for the message control program to perform its function:

- For nonaudio messages, scanning the header to determine routing information and message code translating. Other optional processing operations are provided by QTAM as a convenience to the user. For example, the message control program can insert the time of day in message headers, obviating the need for a message processing routine to do this.

- For audio messages, message code translating. Other optional processing operations are provided by QTAM as a convenience to the user. For example, the message control program can check the input messages to determine if an audio error message must be sent to the calling terminal.

Every telecommunications system operated under QTAM requires one and only one message control program. Depending on the application, one or more processing programs may be required, or none at all (limited to two by the number of available partitions). An example of a nonaudio application requiring no message processing program is a message switching application, in which the sole function of the telecommunications system is to receive messages from terminals and forward them unaltered (except for such processing as the message control program may perform) to one or more other terminals. An example of an audio application requiring no message processing program is a standard audio answering application, in which the sole function is to answer with an invariable audio answer without receiving any input message.

A telecommunications system may include several different terminal types, and both switched and nonswitched line types. For each combination of line type and terminal type, the user must specify a sequence of QTAM message control macro instructions. A separate sequence is normally written for each communication line group. A communication line group consists of one or more communication lines of the same type, over which the same type of terminal can communicate with the computer. Each sequence of message control macro instructions is called a line procedure specification (LPS); the several LPSs collectively constitute the heart of the message control program. For an audio line group, one refers to an Audio LPS.

By way of example, assume that a telecommunications system is to consist of four nonswitched lines to which IBM 1050 terminals are connected, one switched line over which contact with IBM 1050s can be made, three nonswitched lines to which IBM 2260s are connected, and two switched lines over which contact with TWX terminals can be made. The system would then have four line groups: a nonswitched 1050 group, a switched 1050 group, a nonswitched 2260 group, and a switched TWX group. A separate LPS would be required for each group.

Each LPS consists of user-selected macro instructions in two groups: a "receive group", which defines the routines required to operate on incoming messages from any line in the line group; and a "send group", which defines routines required to operate on outgoing messages to any line in the line group.

For the audio communication line groups, each LPS (or Audio LPS) consists of user-selected audio macro instructions, which define the routines required to operate only on incoming messages from any audio line in the line group.

OPERATING ENVIRONMENT

A telecommunications system operating under DOS/QTAM exists in a multiprogramming environment. A message control program is always executed as a foreground-one program, regardless of the presence or absence of other programming components in foreground-two or background. Concurrently with the execution of the message control program, one or two message processing programs can operate in the foreground-two or background partitions. The task selection mechanism of the DOS Supervisor controls the asynchronous operation of all programming components in the system. This method of execution is based on:

1. The completion of awaited events such as I/O termination and the availability of resources (for example, buffers), and

2. The established priorities of foreground-one, foreground-two, and background.

After being assembled, linkage edited, and cataloged into the core image library, a message control program can be loaded and executed in foreground-one. This is accomplished by the foreground initiation routine as the result of an operator message keyed into the system via the 1052 Printer Keyboard. The procedure is the same for a message processing program to be executed in foreground-two. A message processing program to be executed as a background program is initiated by Job Control from the batched-job input stream. In any case, the message control program must be initiated before any message processing program. For detailed information on initiating foreground and background programs, refer to the System Control and System Service Programs publication listed in the Preface of this manual.

With multitasking, it is possible to perform multiprogramming within one or all of the background, foreground-one, and foreground-two partitions. In a multitasking environment, processing in a single partition is possible; the message control program can operate concurrently with one or more processing programs in the foreground-one partition. Within the partition, the subtasks have higher priority than the main task, with the first attached subtask having the highest priority and the last attached subtask having the lowest priority. (For a complete discussion of

Multitasking, see the DOS Supervisor and Input/Output Macros publication.) The main task must first attach the message control program and then the message processing programs. The main task may also deactivate the partition after the QTAM subtasks have closed.

Note: In multitasking the maintask cannot use the STIXIT IT if QTAM is attached with operator control, checkpoint by timer interval, polling interval facilities, or 2740 Model 2.

Depending on the requirements of the user, a System/360 that includes Teleprocessing may be either:

1. Dedicated to Teleprocessing, or
2. Set up to execute non-Teleprocessing jobs concurrently with the execution of Teleprocessing jobs.

The system is dedicated to Teleprocessing when all three partitions are allocated to programs performing Teleprocessing functions; that is, a message control program is executing in foreground-one, and message processing programs are executing in foreground-two and in background.

At the other extreme, if no message processing program is continuously required by the Teleprocessing application, non-Teleprocessing programs can be executing in foreground-two and background. An example of such a configuration is: normal batch processing in the background partition, concurrent peripheral operations in foreground-two, and a message control program performing Teleprocessing functions in foreground-one. Such a configuration can exist only when the message control program can perform the required Teleprocessing functions without the support of a message processing program (for example, a message switching, data collection, or standard audio answering application). However, where it is necessary to terminate operation of the message control program, a message processing program must be initiated in the foreground-two or background partition to perform this function. This requires temporary termination of one non-Teleprocessing program if two such programs are executing at the time the message control program is to be terminated.

Typically, a Teleprocessing application that requires message processing is supported by only one message processing program (in addition to the message control program). One message processing program can be designed to process all message types and to terminate the message control program when necessary. This leaves one partition available for the execution of a non-Teleprocessing program, and thus enables the user to take fullest advantage of the multiprogramming capabilities of DOS. An example of such a configuration is: normal batch processing in background; and a message processing program in foreground-two and a message control program in foreground-one performing Teleprocessing functions.

Multitasking extends the multiprogramming capabilities of the Disk Operating System to execute twelve programs rather than three, since one partition can contain up to ten programs operating concurrently. Message processing programs can be executed in the same partition (foreground-one) as the message control program or in the remaining system partitions.

FILE DEFINITION AND CONTROL INFORMATION

For each file referred to by the message control and message processing programs, a DTF table must be defined by means of file definition macro instructions. A DTFQT macro instruction must be provided for each of the following types of QTAM files:

- Each communication line group (message control program).
- Direct access message queues (message control program--not applicable to audio response units).
- DASD checkpoint records file (message control program).
- 7772 Digitally Coded Voice (DCV) vocabulary (message control program), required when at least one 7772 ARU is present in the system.
- Each main storage process queue (message processing program).
- Each main storage destination queue (message processing program-- not applicable to audio messages).
- Each audio output queue (message processing program).

Similarly, an appropriate DTFxx macro instruction must be provided for each message log used by the message control program. The actual DTFxx used is a function of the storage medium used for the message log (for example, DTFMT would be used for magnetic tape).

The DTF tables in a message control program serve as a logical connector between

the message control program and the associated line group, DASD message queues, 7772 DCV vocabulary, and message log files. The DTF tables defined in a message processing program are not associated with files themselves. They are used to provide control information to QTAM for the transfer of data to and from a message processing program. The main storage (MS) process and destination queues and the audio output queue are the main connectors between a message control program and a message processing program.

In addition to the file definitions, the user must supply control information (in the form of macro instructions) that is used by the message control program to control the sending and receiving of messages. The control information consists of:

- The name and address of each terminal with related information, such as any special distribution lists for sending a message to more than one terminal (not applicable to audio lines).
- The name of each DASD process queue associated with a message processing program to which incoming messages are to be sent.
- A polling list for each line that indicates the order in which the terminals on the line are to be polled (not applicable to audio lines).
- The size and number of main storage buffers that are to be used for sending and receiving messages to and from the terminals. In order to compensate for the differences in the rates of information flow, QTAM automatically and dynamically uses available buffers in accordance with immediate needs (not applicable to audio lines).
- The name of each audio communication line together with related information, such as line group name.
- The size and number of main storage DCV buffers used to transmit messages to the terminals (for each IBM 7772 ARU transmitting DCV words dynamically retrieved from DASD). QTAM automatically and dynamically uses available buffers from the corresponding DCV buffer pool in accordance with immediate needs.
- The name and location of 7772 DCV vocabulary words to be permanently kept in main storage.

QTAM FACILITIES

The QTAM facilities include a comprehensive set of input/output, message control, translating, and editing routines that relieve the programmer of the detailed and specialized programming normally required in writing a message control program for a telecommunications system. Macro instructions are provided that allow the programmer to assemble and linkage edit these routines into an integral message control program designed to meet the exact requirements of an installation.

For nonaudio terminals, the primary capabilities of the telecommunications programs that can be created through the use of QTAM macro instructions are:

- Polling terminals
- Receiving messages from terminals
- Addressing terminals
- Sending messages to terminals
- Dynamically assigning and using available buffers as required
- For incoming messages, performing message editing functions such as: translating from the transmission code in which messages are sent to extended binary coded decimal interchange code (EBCDIC); inserting time-received and date-received information in the header; recording (logging) the message on a secondary storage medium such as magnetic tape; and maintaining a count of the number of messages received from each terminal
- Routing messages to appropriate queues, determined by either the destination code specified in the header of the message, or the source from which the message entered the system
- Queueing messages on a direct access storage device
- Initiating corrective action when an error or unusual condition is detected
- Intercepting transmission of messages in error
- Cancelling messages containing errors
- Rerouting messages
- Transmitting error messages
- Routing messages with erroneous header information to a special queue

- Providing message data, in the work unit specified (message, message segment, or record), to a message processing program
- Placing response messages generated by message processing programs on queues for subsequent transmission
- Retrieving messages already queued for transmission to terminals
- For outgoing messages, performing message editing functions such as: placing time-sent and date-sent information in the header; placing an output sequence number in the header; logging the outgoing message on a secondary storage device; maintaining a count of the number of messages sent to each terminal; and translating the message from EBCDIC code to the appropriate transmission code
- Taking periodic checkpoints of the system, in which the status of the queues and the telecommunications network are saved on a direct access storage device. This information can be utilized by a recovery facility (Restart) in case of subsequent system failure
- Providing operator-to-system communication through a telecommunications system control terminal
- Providing on-line terminal testing for remote IBM terminals
- Keeping counts of line errors
- Providing error recovery procedures.

For audio terminals, the primary capabilities of the telecommunications programs that can be created through the use of QTAM macro instructions are:

- Enabling audio communication lines
- Receiving messages from terminals
- Sending audio messages to terminals
- Providing main storage for DCV word buffering, when an IBM 7772 uses DCV words dynamically retrieved from DASD
- Performing, for incoming messages, message editing functions such as: translating from the transmission code in which messages are received into extended binary coded decimal interchange code (EBCDIC), except for the messages dialed on IBM 3944; inserting time-received information when messages are to be logged on a secondary storage medium such as magnetic tape
- Queuing messages on the main storage process queue
- Initiating corrective action when an error or an unusual condition has been detected
- Providing messages to a message processing program
- Placing response messages in queues for subsequent transmission
- Retrieving messages previously queued for transmission
- Keeping counts of audio line errors
- Providing error recovery procedures.

The following sections describe message control services for networks designed to handle nonaudio messages only. For a discussion of audio message handling and of mixed audio and nonaudio message handling, refer to page 117 .

MESSAGE FORMATS (NONAUDIO MESSAGES)

This section describes the format of nonaudio messages. For a detailed discussion of the formats of messages arriving from various terminal types, those in storage, and those being transmitted to various terminal types, refer to Appendix J.

A nonaudio message usually consists of two parts: header and text. The message header contains control information for the message, such as:

1. One or more destination codes.
2. The code name for the originating terminal.
3. The number of the message relative to the numbers of previous messages received from that terminal (input sequence number).
4. A message type indicator.
5. Various other fields containing control type data.

Operations on the fields in the header are a primary function of the LPS-defined routines in the message control program. The length and format of the header and the information it contains depend solely on the requirements of the application and the user's preferences. The length may be just a few characters or many characters. In some instances, it is possible to omit headers entirely; however, some type of header is usually provided. The text portion of a message consists of the information of concern to the party ultimately receiving the message. This party can be either a terminal or a program that processes the text (message processing program).

The format of the message header, to a great extent, dictates the arrangement of the message control program. For this reason, the control characters used and the sequence of the fields within the header must be predetermined so that the message control program for the telecommunications system can be properly coded.

The destination codes in the message header identifies the terminal(s) or processing program to which the message is to be routed. The message type indicator can be used to identify a header that is to be processed in a special manner. By inserting certain macro instructions in the message control program, the user can insert in the header such data as the date and time the message is received, the date and time it is sent, and the number of the message in relation to other messages sent to a particular terminal (output sequence number).

Depending on the type of work unit (message, segment, or record) with which he is dealing in his system, the user must specify appropriate characters for control purposes.

- A message is that unit of text that is terminated by a special end of transmission (EOT) character or by an EOM or EOT character for WTTA terminals.
- A segment is that portion of a message contained in a single buffer, the size of which is specified by the user.
- A record is that portion of a message terminated by any of the following characters: end of block (EOB), end-of-text (ETX), carriage return (CR), line feed (LF), or new line (NL).
- A message block is that portion of message terminated by an EOB character. There is no EOB character for WTTA terminals.

  Note: The end of an input message sent by a WTTA terminal is indicated by one of the following:

  - EOM character: indicates that another message is to be sent by the terminal operator.
  - EOT character: indicates that the input message is the last to be sent by the terminal operator.
  - A time-out: indicates that no character has been sent by the terminal operator during a 28-second interval. This is recognized as an end-of-transmission signal.

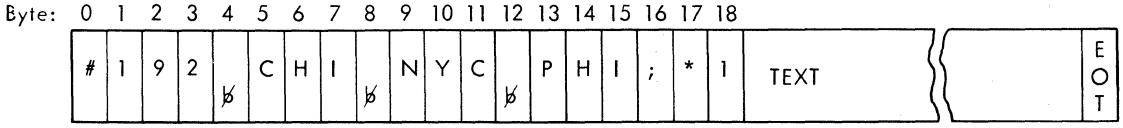


Figure 2. Sample Format for an Incoming Message

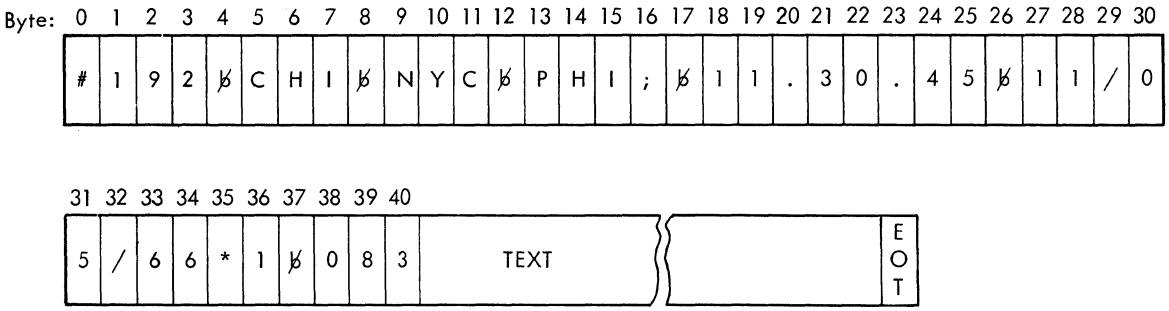


Figure 3. Sample Format for an Outgoing Message

There are many possible variations for the format of a message header. The sample formats shown in Figures 2 and 3 are included for illustrative purposes.

The format shown in Figure 2 could be used in a message switching application. Byte 0 contains a machine end of address (EOA) character. There is no machine EOA character for WTTA terminals. When the message is transmitted, this character signals the end of nonrecorded machine control characters (such as addressing characters and the machine ECA itself) and the beginning of data characters. The 192 in bytes 1 through 3 is the input sequence number. Bytes 5 through 7 contain the code for the terminal that originated the message. Bytes 9 through 11 and 13 through 15 contain destination codes specifying the terminals to which the message is to be sent. In this example, the semicolon in byte 16 has been designated by the user as the program ECA character. Since some of the messages in this application contain multiple destination codes, this control character must follow the last destination code. Bytes 17 and 18 contain characters specifying the priority of the message. The remaining portion of the message is text and is followed by the EOT character.

After the message control program has operated on the message header and before the message is transmitted to the destination terminals, the format of the message could be as shown in Figure 3. When the message comes into main storage, the message control program inserts time received and date received information in the header. The time received information in bytes 18 through 25 indicates that the message was received at 11 hours, 30 minutes, and 45 seconds on the date (November 5, 1966) specified in bytes 27 through 34. Insertion of this information moves the priority data to bytes 35 and 36. The message is then queued by priority on the direct access storage device. When the message reenters main storage prior to transmission to the destination terminals, the message control program places the output sequence number in bytes 38 through 40 of the header. The original text and the EOT character follow the output sequence number.

QTAM, with its complete set of header processing routines and associated macro instructions, allows the user to indicate the header processing functions he wishes performed. He does this by including the appropriate macro instructions in the message control program. Many functions are available, in addition to those described briefly in this section, such as the detection of incorrect or invalid information in the header fields. These functions and the relationship of the message header format to the design of the message control program are discussed in detail in later sections of this publication.

NONAUDIO MESSAGE FLOW

The message flow within the system depends on the type of message. This section describes the flow of nonaudio messages through a system operating under DOS QTAM from the receipt of the message at the computer to its transmission to a destination terminal. Figure 4 illustrates this message flow.

The input message is prepared at the remote terminal location. Messages may be of variable length and consist of two parts: header and text. When polled, the source terminal sends the message to the computer via a communication line. In Figure 4, step 1 shows the message passing through an IBM 2701, 2702, or 2703 control unit and the multiplexer channel, and filling available buffers from the QTAM buffer pool.

The user defines the size of his buffers in the message control program, which must be in the foreground-one partition. QTAM inserts control information (known as a prefix) in the first portion of each buffer. The first 32 bytes of a buffer used to contain a message header are set aside for a header prefix generated by QTAM. This buffer must contain the entire header and may also contain text data. The characters transmitted by the remote terminal begin filling the buffer in byte 32. The first 22 bytes of a buffer used to contain only text data are set aside for a text prefix generated by QTAM. Message data begins filling the buffer in byte 22.

The user can transmit single segment or multisegment messages. (A message segment is that message data that occupies one buffer.) In single segment messages, the entire message is contained within one buffer. In multisegment messages, more than one buffer is needed for a message.

In all buffers except the last for a multisegment message, the segment containing the header is shorter than a segment containing only text; this is because the header prefix generated by QTAM is ten bytes longer than the text prefix. In each buffer used to contain intermediate text, the segments are the same size. In the last buffer for a multisegment message, the message segment can be any length equal to or less than the buffer length minus 22.

The buffers shown in Figure 4 are each 80 bytes in length. The first input buffer thus accommodates a message segment of 48 characters; of these, 26 constitute the header portion of the message and 22 the text portion. In the second input buffer, the message segment is 58 characters; all of these characters are text data. The third and last input buffer contains the remaining characters in the message. Since the input message is 150 characters, the message segment size for this buffer is 44.

As soon as a buffer is filled with the first segment of a message, a portion of the line procedure specification (LPS) called the receive group performs such user-selected functions as: converting codes, logging, updating message counts, incorporating time-received and date-received information, and checking input sequence numbers. The first three functions can also be performed for text segments. In Figure 4, the user has specified that messages to be handled by a message processing program must have six characters of time-received information incorporated into the message header. The header information preceding the position where the time is to be inserted is shifted into the reserved area in the header, and the time is inserted into the space thus created. The insertion of additional fields in the header must not cause the header and prefix size to become greater than the specified buffer size.

In performing its function, the LPS scans and processes header fields in accordance with the order indicated by the relative positions of the individual LPS macro instructions; the operations are performed in the buffer containing the message segment. After performing these functions, the receive group of the LPS routes the prefix (minus the first eight bytes[1]) and the message segment to either a DASD destination queue or a DASD process queue.

Each DASD destination queue contains message segments that are to be transmitted via a certain line, or message segments that are to be transmitted to a certain terminal. A DASD process queue contains message segments that are to be routed to a message processing program.

The receive group of the LPS can check the validity of the name of the originating terminal and the destination code before routing the message to a DASD process queue or DASD destination queue. Each type of queue is maintained on a direct access storage device, and all such queues are regarded as one file (the DASD message queues file).

---

[1]The first eight bytes of a header or text prefix contain control information used only in main-storage buffer handling; therefore, these bytes are not placed on the direct access device.

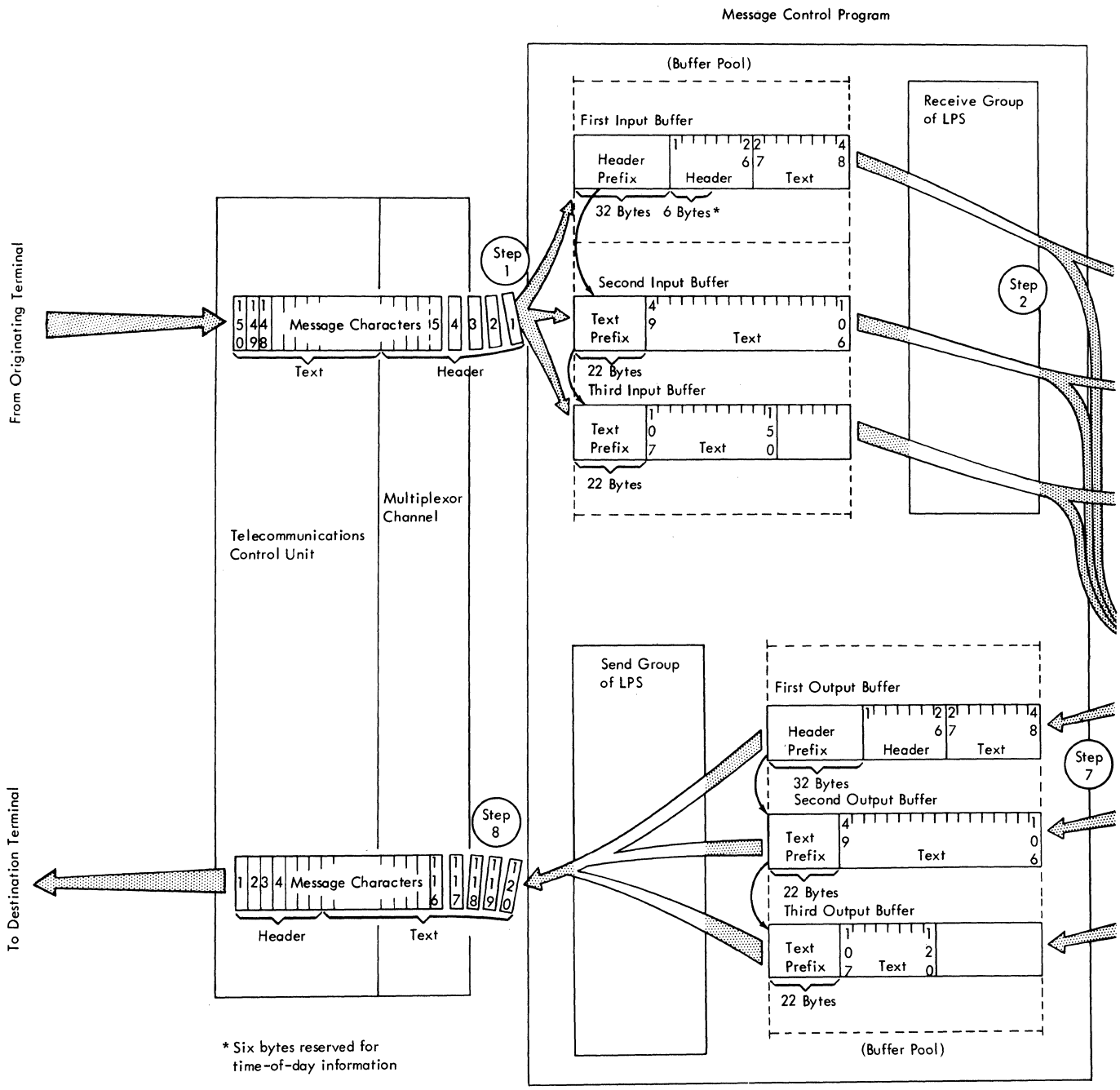


Figure 4. QTAM Message Flow (Part 1 of 2)

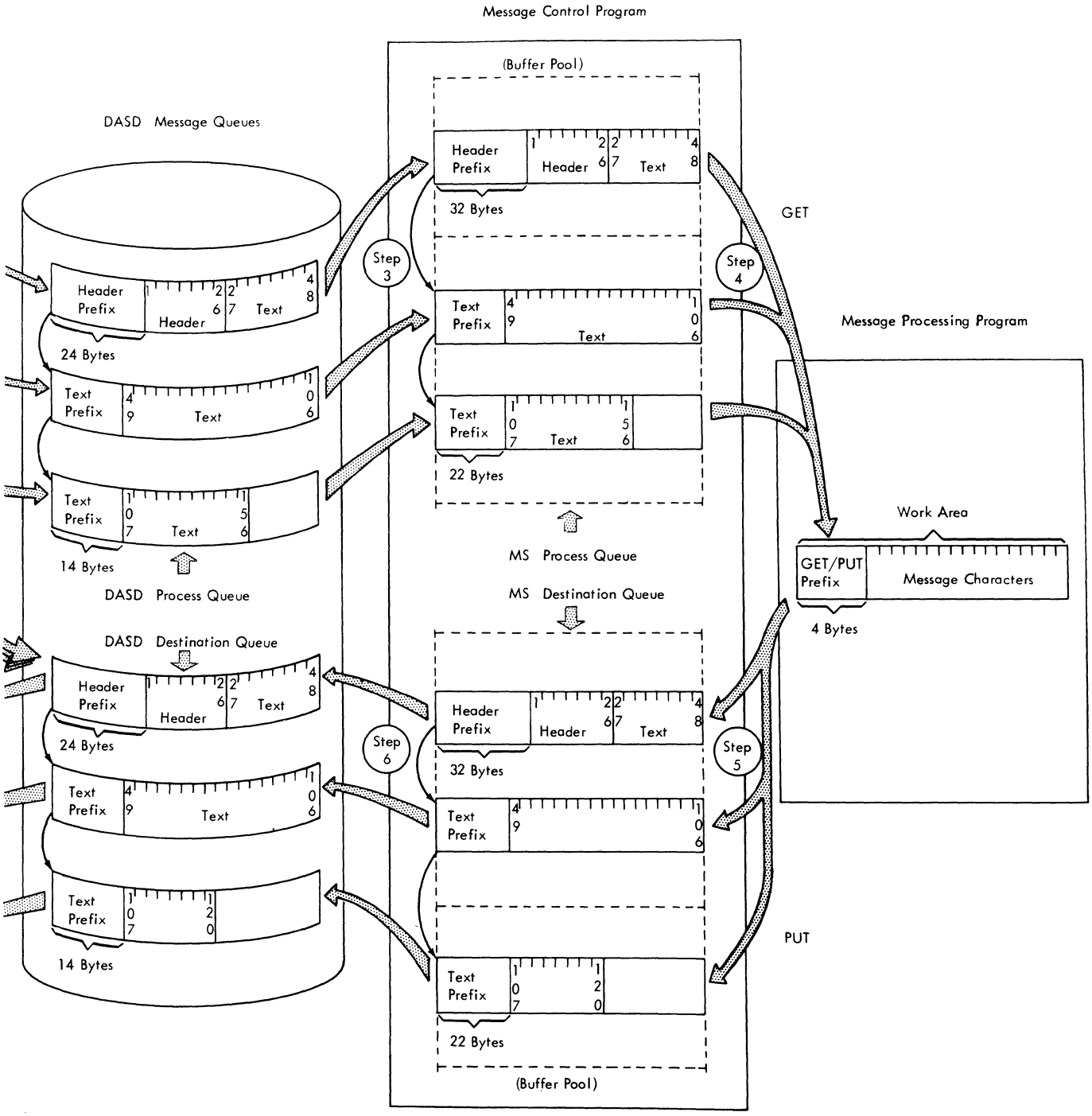


Figure 4. CTAM Message Flow (Part 2 of 2)

Each DASD process queue is associated with a message processing program. Messages requiring text processing should be routed to the DASD process queue associated with the message processing program that processes that type of message. The user controls this routing either via the message header (the destination code is the name of the DASD process queue) or by LPS macro instructions that direct messages of a particular type to a particular queue. Step 2 of Figure 4 shows the LPS routing a message to a DASD process queue. The receive group of the LPS can place messages that do not require text processing (e.g., switched messages) directly on the appropriate DASD destination queues.

For each DASD process queue maintained, QTAM maintains a corresponding queue in main storage. Each main storage (MS) process queue is maintained in buffers from the QTAM buffer pool in the foreground-one partition. The number of buffers allocated to an MS process queue is specified in a DTF table by a message processing program. After the DTF table for the MS process queue has been opened by the message processing program, a QTAM routine automatically passes the message segment from the DASD process queue to the MS process queue (see Step 3 of Figure 4). In moving the prefix and the segment to the buffer, the eight bytes that were deleted when the prefix and the segment were placed on the DASD process queue are restored so that the prefix length is once again 32 (header prefix) or 22 (text prefix).

Each time the message processing program gains control and issues a GET (Step 4 in Figure 4), QTAM passes message data from the MS process queue to a user-specified work area in the message processing program. Message data is provided in the work unit specified by the user in the DTF table. The work unit may be either a complete message, a message segment, or a record. Before moving the message data to the work area, QTAM strips the header and text prefixes from the message segments. In the first four bytes of the work area, QTAM places a 4-byte prefix, which indicates the size and type of work unit with which the user is dealing. After receiving the message data, the message processing program processes it as required by the application.

A message processing program generating a response message must define and open a DTF table governing message transfer before attempting to place the message on a DASD destination queue. This DTF table contains information needed by QTAM to establish an MS destination queue. When a PUT macro instruction is issued by a message processing program (Step 5 in Figure 4), QTAM moves the message data from the user-specified work area into the MS destination queue. The header and text prefixes are attached to the message segments in the buffer areas that make up the MS destination queue. As the message data fills the buffers, QTAM inserts chaining addresses and other necessary control information into the prefix fields. The response message generated by a message processing program can be any size (the one used in Figure 4 is 120 characters).

After the header and text prefixes have been added in the MS destination queue, QTAM places the message into the appropriate DASD destination queue on the DASD message queues file (Step 6 of Figure 4).

QTAM retrieves message segments from the DASD destination queues on a first-in-first-out basis within priority groups. The message segments are brought in from the direct access device and placed in available buffers (Step 7 of Figure 4). The send group of the LPS section in the message control program can then perform such user-selected functions as: converting the code of the message to the device code of the terminal, incorporating time-sent and date-sent information in the header, message logging, and updating of message counts. These operations are performed in the buffers that receive the message segments from the direct access device. QTAM then strips the header and text prefixes from the message segments and transmits the message to the appropriate terminal (Step 8 of Figure 4).

The header and text prefixes described in this section are generated automatically and are used by QTAM routines. No programming considerations are required by the user for the manipulation of the buffers and their prefixes as messages flow through the system. The header and text prefixes are discussed only to give a complete description of the flow of messages through the system. A macro instruction is provided that allows the user to retrieve messages from a queue on the DASD message queues file. When this macro instruction is used to retrieve the segment containing the message header of a multisegment message, the user can access the chain address field in the header or text prefix to retrieve succeeding segments of the message. The formats of the header and text prefixes are shown in Appendix A.

RELATIVE PRIORITY OF RECEIVING VERSUS SENDING IN NONAUDIO OPERATIONS

Message traffic can proceed in only one direction at a time over each of the half-duplex lines that comprise a QTAM-controlled telecommunications network.

The user has the option of specifying, for each line group made up of nonswitched lines, one of three relative priorities of receiving versus sending operations for the lines in the group. He may specify that receiving has priority over sending, the two have equal priority, or sending has priority over receiving. The significance of these options varies with the type of polling process. For lines polled under the control of the program capability (polled lines), the implications are as follows.

If receiving has priority over sending, polling of terminals and receipt of incoming message traffic proceed continuously on a given line except during the time period that a user-specified polling interval is observed. The specified polling interval is observed only when no message traffic is received during a complete pass through the polling list for the line. Outgoing messages (if any are present on the destination queue for the terminal or line) are sent only during this interval, and only until the interval expires. Upon expiration of the interval, outgoing message transmission ends after the current message is sent, regardless of whether any messages still remain queued. Polling and incoming message transmission then resume. It is important to note that if no polling interval is specified, or if there is no lapse in incoming message traffic, outgoing message transmission cannot occur. Assuming that the user specifies a polling interval, he must also make it long enough to accommodate any expected density of outgoing message traffic. In other words, too short an interval will cause outgoing messages to "back up" on the destination queue for that terminal or line.

If receiving and sending have equal priority, polling and incoming message traffic proceed without interruption until all terminals on the line have been polled (i.e., until the end of one polling pass). Then outgoing messages (if any are present on the destination queue for the terminal or line) are sent. Once outgoing message transmission begins, it continues until all messages on the queue have been sent, regardless of whether the user has specified a polling interval. When the destination queue is depleted, polling and incoming message traffic resume. Note that, in contrast to the case where receiving has priority, outgoing message transmission occurs whether or not a polling interval is specified and regardless of the length of the interval.

If sending has priority over receiving, outgoing messages (if any are present on the destination queue) are sent:

1. Each time a negative response to polling is received from a terminal.
2. Each time an EOT is received from a terminal, indicating that a complete message has been sent.

Once outgoing message transmission begins, it continues until all messages on the queue have been sent. Note that when sending has priority, outgoing transmission can occur after each terminal is polled, rather than only after a complete polling pass.

For lines polled under the control of the Auto Poll feature (autopolled lines), the priority options of receiving over sending have the following meanings.

If receiving has priority over sending, no outgoing messages can be sent to the terminals attached to these autopolled lines. The lines are open to incoming messages traffic only.

If receiving and sending have equal priority, the significance of this option for autopolled lines is the same as for polled lines with the exception that no polling interval may be specified.

If sending has priority over receiving, outgoing messages are sent:

1. Each time an EOT is received from a terminal, indicating that a complete message has been sent by this terminal.

2. Each time the end of the polling list is reached.

MANAGEMENT OF WTTA LINES

The name World Trade telegraph terminal (WTTA terminal) refers to any of various European teletypewriters using a start-stop 5-level code with two shifts (letters shift and figures shift) to transfer data over leased point-to-point telegraph lines (referred to as WTTA lines) at 50, 75, or 100 bauds (bits per second). The codes used are either the International Telegraph Alphabet No. 2 (referred to as ITA2) or the Figure Protected Code ZSC3 (referred to as ZSC3). These two codes are illustrated in Figures 44 and 45.

WTTA lines operate in a contention system. The message control program is always ready to handle messages from WTTA terminals since, as soon as traffic ceases, a Read operation is initiated so that the line is prepared to receive the next message. Therefore, only one WTTA terminal can be connected to a given WTTA line.

A message sent to a WTTA terminal (output message) or sent by a WTTA terminal (input message) must always start with twelve LTRS characters. For an input message, these twelve characters are sent by the terminal operator, but they do not enter main storage. For an output message, they are automatically sent by QTAM. When input messages are prepunched into paper tape, the 12 LTRS characters are required. On the other hand, the WRU, EOM, and EOT characters must never be prepunched; they must be sent by the operator.

The user can specify that receiving has priority over sending, that sending has priority over receiving, or that the two have equal priority. If receiving has priority over sending, (or if the two have equal priority), and if there is no traffic over the line, the corresponding Read command is interrupted and an output message is sent to the WTTA terminal (refer to the note below). If an input message is being received, the output message will be sent only after an EOT character has been received or after a time-out has occurred. If sending has priority over receiving, the same procedure is followed, except when an input message is being received; in this case, the output message will be sent as soon as an EOM or EOT character is received or a time-out occurs.

Note: If the first character of an input message is received at the same time the Read command is interrupted, the output message is not sent and the input message is accepted.

Both the CPU and the WTTA terminal can ask for the identification sequence of the other. When an identification exchange is performed, the CPU sends its identification sequence to the terminal (IAM=YES must be specified in the DTFQT macro instruction), and the terminal sends its identification sequence to the CPU (WRU=YES must be specified in the DTFQT macro instruction). An identification exchange can be performed during either

- Receiving operations, each time the terminal operator sends the WRU signal to the CPU; or

- Sending operations, at the beginning of an output message (if the WRU macro instruction is in the Send Header subgroup of the LPS) or at the end of an output message (if the WRU macro instruction is in the End Send subgroup of the LPS).

When the CPU receives the terminal identification sequence, QTAM compares this sequence with that specified in the TERM macro instruction (refer to the description of the TERM Macro Instruction).

MANAGEMENT OF NONAUDIO SWITCHED LINES

Insofar as possible, QTAM management of switched lines parallels the management of nonswitched lines. A number of differences should be understood, however.

In a switched network, terminals are not attached to the computer by specific lines. Rather a line connection between the computer and a terminal is established over any available access line included in the line group. An available line is a line over which message traffic is not currently in progress.

The management of switched lines by QTAM allows all current message traffic (both from CPU to terminal and from terminal to CPU) to be transmitted during one call. That is, once a line connection is established by either party, all messages queued for sending to the terminal are sent, and the terminal is allowed to send to the computer all messages it may have ready. The priority scheme implemented for switched lines is similar to the sending priority previously described for nonswitched lines. That is, each time a message is received

from a terminal, any messages on the destination queue for the terminal are sent before accepting another message from the terminal. This is true regardless of whether the line connection is caused by a computer initiated call or by a terminal initiated call. The primary reason for handling switched lines in this manner (rather than requiring separate calls for incoming and outgoing messages) is in the interest of economy since the rates for a switched network are frequently on a per call basis.

CALLS FROM THE COMPUTER TO A SWITCHED TERMINAL

The Auto Call feature is required for computer initiated calls to a terminal on a switched network. This discussion assumes that this feature is included and that computer initiated calls are desired.

When messages appear on a destination queue for any type of terminal on a switched network, the computer attempts to find an available access line starting with the relative line defined in the terminal table entry for that terminal. If that access line is busy and additional lines were defined in the line group, the computer attempts to initiate the call using one of these lines. If all the defined lines are busy, the attempt to call the terminal is deferred until a line becomes available. When an available access line is found, the computer disables the line and dials the terminal using the dial digits specified in the terminal table entry. If the dialing procedure is completed successfully and if the terminal is ready to receive, the queued messages are sent. (If the dialing procedure is not completed or if the computer receives a negative response to addressing, the messages are not sent unless the user includes INTERCPT and RELEASEM macro instructions to cause later sending of the messages. See the descriptions of these macro instructions.) After all messages are sent to the terminal, the computer turns the line around and accepts any incoming messages the terminal may have ready. If further messages arrive on the destination queue for the terminal while the computer is accepting an incoming message, these messages are also sent before another message is accepted from the terminal. When the last incoming message is received and no further messages appear on the destination queue, the computer breaks the line connection (for the 2740, the connection must be broken at the terminal). It then reenables the access line, making that line available again.

Restriction: A terminal on a switched line not having the Auto Call feature can receive a message only in response to an inquiry when the line is operating in conversational mode. A message cannot be switched to this terminal from another terminal.

For a discussion of calls from computer to switched IBM 1050, IBM 2740, and TWX terminals, see Appendix J.

CALLS FROM A SWITCHED TERMINAL TO THE COMPUTER

The Auto Answer feature is required for the terminal initiated calls to the computer on a switched network. This discussion assumes that this feature is included and that terminal initiated calls are desired.

When QTAM is not sending messages to or receiving messages from a switched terminal over a particular access line, that line is enabled to permit switched terminals to call the computer. A terminal that wishes to send a message causes a line connection to be established by dialing the computer over an enabled line. After answering the call, QTAM receives the first incoming message (if any) from the terminal and then sends any messages on the destination queue for the terminal. From this point, the transmission of further messages between the terminal and the CPU proceeds in the same sequence described for computer initiated calls. When the terminal indicates that it has no more messages and no further messages appear on the destination queue for the terminal, QTAM breaks the line connection and reenables the line. (With the 2740, the connection must be broken at the terminal).

Except that the first message is sent by the terminal, the receiving and sending procedures for a terminal initiated call from either an IBM 1050, IBM 2740, or a TWX terminal are identical to those described in 'Calls from the Computer to a Switched Terminal'.

For a discussion of IBM 2260-2848 Local Operations, refer to Appendix J.

TELECCMMUNICATIONS APPLICATIONS (NONAUDIO)

A telecommunicatiens system cperating under DCS/QTAM can be designed for a wide variety of applicatiens including message switching, collecting data, standard audio answering, processing collected data, and inquiry processing. Each of these applications is described here briefly.

MESSAGE CONTRCL APPLICATIONS

Three applicatiens particularly suited to handling by a message control program are message switching, data collection, and standard audio answering (the last of these is described in the audio sectien of this publicatien).

MESSAGE SWITCHING

Message switching can be acccmplished entirely within the message control program except that a message processing program must be loaded and initiated to terminate the execution of the message control program.

In a message switching application, terminals transmit messages to the central processing unit, which relays the messages to cne or more other terminals. The application does not prevent a terminal from sending a message to be processed. QTAM places these messages on a DASD process queue for handling by a message processing program (either concurrently or at a later time).

When an incoming message is to be switched, the LPS section of the message contrcl program rcutes the message to a DASD destination queue. If desired, information such as the time and date of receipt can be placed in the message header. Validating the codes cf the originating and destinatien terminals and checking the input sequence number in the header can also be performed. Before a message is transmitted to its destination, the IPS can record in the header the date and time the message is sent. It also can record the number of the message in relatien to other messages sent to that terminal. The LPS can also log the messages sequentially on a storage device for subsequent reference by the user with a different access method.

DATA COLLECTION

Data collection, like message switching, can be accomplished entirely within the message control program except that a message processing program must be loaded and initiated to terminate the execution of the message control program.

In a data collection application, terminals send data in the form of messages to the central processing unit (CPU). The messages are accumulated and stored by the CPU, and subsequently processed as a batch.

The message control program can accumulate data in two ways:

1. It stores the data on DASD process queues.
2. It can also store the data on any secondary storage medium.

If the first method is used, the messages can be obtained at any time (concurrently or later) by a message processing program (see the subsequent section Processing Collected Data). The messages are routed to DASD process queues on the DASD message queues file in the same manner as any other messages that have a message processing program as their destination. The messages remain on these DASD process queues until the activation of a message processing program which issues a series of GET macro instructions to obtain and process the messages.

In the second method, the LOGSEG macro instruction in the LPS section of the message control program causes the message to be recorded sequentially on a secondary storage device selected by the user. An access method other than QTAM must be used to retrieve these messages for processing.

MESSAGE PROCESSING APPLICATIONS

A wide variety of telecommunications applications can be processed by a message processing program. Two of these applications are:

- Processing collected data (not applicable to the audio messages).
- Inquiry processing.

PROCESSING COLLECTED DATA

The processing of collected data is the second part of a 2-step application. The first step is the actual collection of the data by a message control program (see the preceding Data Collection discussion).

The message control program places data to be collected on DASD process queues and optionally on any secondary storage device (through the LOGSEG macro instruction). These messages may be retained on the storage device used until it is convenient to process them.

If messages are collected on a DASD process queue, they remain on the queue until a message processing program issues GET macro instructions to obtain and process the messages. The message processing program that processes the collected data can either:

1. Be operated concurrently with the collection of the data by the message control program, or
2. Be loaded and initiated at a later time (for example, to process data at the end of the day after all message traffic has ceased).

In the latter case, if the user wishes to have QTAM retrieve the messages from the DASD process queue, the message control program must remain operational. If termination of the message control program is desired so that the foreground 1 partition is available for another program, the message processing program must implement another access method to perform the input operations.

If the data is collected on a user selected secondary storage device by a LOGSEG macro instruction, the data must be obtained for processing by an access method other than QTAM.

INQUIRY PROCESSING

An inquiry application involves receiving messages from terminals (performed by the message control program), processing the data contained in the messages (performed by the message processing program), and sending replies to the originating terminals (message control program).

The routines called by the message processing program to process the messages need not reside in main storage. For example, in an inquiry processing application that requires processing of many different types of inquiries, it may not be economical to have all of the required processing routines in main storage. The message processing program can contain an analysis routine that determines the type of the message and loads the routine required to process it from the core image library (via a FETCH or LOAD macro instruction). The routines fetched dynamically in this manner must have previously been linkage-edited onto the core image library at a specified address in an available area in the partition in which the message processing program is executing.

An optional feature of the inquiry application is operation in a conversational mode.

For nonaudio lines operating in conversational mode, a terminal transmitting a message into the system is held on the line by the message control program until the message processing program generates a response message for transmission back to the inquiring terminal. The response message is transmitted immediately. Conversational mode is specified through the CONVERSE operand of the MODE macro instruction in the LPS section of the message control program. Another optional function particularly suited for a high volume inquiry application is specified by the EXPEDITE operand of the PROCESS macro instruction in the message control program. The EXPEDITE operand causes messages to be routed directly to the main storage process queue associated with the message processing program. The normal intermediate step of placing the messages on a DASD process queue is therefore bypassed. Both of these optional functions decrease the time required to return an answer to the inquiring terminal.

MESSAGE CONTROL PROGRAM (NONAUDIO MESSAGES)

In every telecommunications system using QTAM, there must be one message control program. The message control program must be executed in the foreground 1 partition of the Disk Operating System. With multitasking, the message control program must be executed as the highest priority task in the foreground-one partition.

For nonaudio lines, message control includes those functions that:

1. Control the flow of messages between the computer and terminals.
2. Prepare the messages for processing and route them to their destination (another terminal or a message processing program).
3. Provide the user with statistical information relating to message traffic.
4. Provide the user with a copy of messages received from or sent to terminals.

The message control program includes both device handling and message handling routines of QTAM. Messages arriving at the computer from terminals are coded in the transmission code of the particular terminal. Facilities are provided to convert these transmission codes to the extended binary coded decimal interchange code (EBCDIC), which simplifies message analysis. Similarly, messages being sent from the computer to a terminal are converted from EBCDIC to the transmission code of the terminal. Refer to Appendix J for a discussion of considerations for specific terminals.

Messages received from terminals can be routed to one or more destinations. QTAM routines check the validity of the destination codes and place messages in DASD queues according to their destinations. From these DASD queues, the messages are normally sent to their destinations on a first-in-first-out basis. However, a message priority scheme may be included to expedite the handling of certain messages. Priority processing of messages is particularly useful in an application, such as inquiry processing, where rapid response to inquiries is required.

The input messages are normally sent to the MS process queue of the corresponding message processing program on a first-in-first-out basis.

To construct his message control program, the user must select, place into order, and assemble macro instructions provided by QTAM. There are four major sections in the message control program and each can be wholly defined by QTAM macro instructions. The four major sections and the order in which they will be discussed are:

1. File definition.
2. Control information.
3. File initialization and activation.
4. Line procedure specification.

File definition and control information macro instructions generate the tables, lists, and buffer areas needed in the system. Initialization and activation macro instructions ready the system for operation.

The line procedure specification (LPS) section is the most important section of the message control program. In the LPS section, the user specifies, through LPS macro instructions, the manner in which he wishes message traffic in his system to be handled. The LPS macro instructions establish the linkage to QTAM routines that perform the message code translating, editing, error checking, logging, and routing functions. The parameters used by these routines originate either in the message header or in the LPS macro instructions supplied by the user.

The information specified in the file definition and control information sections is also used by the LPS routines in performing their functions. There must be one LPS for each communication line group that requires different message handling functions.

QTAM provides DSECTs (see Figure 5) that enable the user to refer symbolically to the various fields in the tables and control blocks used by QTAM. Three such DSECTs are automatically generated in the message control program:

1. Terminal table entry DSECT (includes names assigned to an optional subfield by the user in OPTION macro instructions). This DSECT is always generated.

2. Buffer prefix DSECT (header and text prefixes).

3. Line control block DSECT.

If any other DSECT listed in Figure 5 is desired in the message control program, it can be included at assembly time by including a COPY statement with the name in the specify column of Figure 5 in the operand field. If any DSECT is required in a message processing program, it can be included in the same manner (none of the DSECTs are automatically generated in a message processing program). It should be noted that all QTAM DSECT names have the four-character prefix, IJIQ; therefore, the user should refrain from using names beginning with these four characters.

Table or Control Block	DSECT Name	Specify (for COPY)
Terminal Table Entry	IJLQTBL0	IJLQTBLD
Buffer Prefix	IJLQBFR0	IJLQBFRD
Line Control Block	IJLQLCB0	IJLQLCBD
Queue Control Block	IJLQQCB0	IJLQQCBD
DTF Table	IJLQDTF0	IJLQDTFD

Figure 5. QTAM DSECTs

FILE DEFINITION

A file definition macro instruction must be specified for each file referred to by the message control program. Four types of files are normally used:

- Direct access message queues file.
- Communication line group files.
- Direct access checkpoint records file.
- Message log files.

DIRECT ACCESS MESSAGE QUEUES FILE

One direct access message queues file is required. Message segments awaiting transmission to destination terminals and message segments awaiting processing by a message processing program are placed on queues on a direct access storage device (DASD). (Multiple extents and multiple volumes may be used for the DASD message queues file.) To establish these queues (called DASD destination queues and DASD process queues, respectively), one DTFQT macro instruction must be issued to define the DTF table for the file of message queues. The macro must specify 'TYPE=DA'; however, the DLAB card creating the DASD label must specify type SD.

Before defining and using the DASD message queues file, the following steps must be performed:

1. An area is allocated for the file on the DASD volume to be used, and

2. The entire area allocated for the file is preformatted with dummy records.

The dummy records must have a fixed length equal to the buffer size (as defined by the BUFFER macro instruction) minus eight. This formatting process is necessary only when the system is initially generated, not each time the message control program is initiated. An access method other than QTAM must be used to format the area for this file. Each extent formatted must have a format 1 label. The Clear Disk utility is recommended for formatting.

The DASD message queues file is checked for incorrect formatting at the time the file is opened by an OPEN macro instruction. If the file has been formatted incorrectly, an error message is provided and the job is cancelled.

COMMUNICATION LINE GROUP FILES

A communication line group file consists of messages transmitted via communication lines, or between the computer and a locally attached IBM 2260-2848 Display Complex. One or more files of this type are required. The user must specify one DTFQT macro instruction to define a DTF table for each line group in the system.

A line group can consist of up to 31 lines having the following common characteristics:

- All lines in the group are of the same type, either switched or nonswitched. If switched, lines having the auto-call feature cannot be included in the same line group with lines not having the auto-call feature. If nonswitched, lines having the Auto Poll feature cannot be included in the same line group

with lines not having the Auto Poll feature.

- Association with the same type of terminal devices (for example, all the lines connect IBM 1050's to the system, or all the lines connect IBM 1030's to the system).

- Requirement that the same number of buffers be requested in advance for each transmission of data from a terminal to the computer.

- Operation under the same relative priority specification.

- Use of the same LPS.

- Use of the same polling interval.

Two additional requirements are:

1. The relative position of the device access area in a terminal table entry must be the same for all terminal table entries associated with the lines in the line group (see the section The Terminal Table).

2. No line within the line group can be defined as part of another line group.

The requirements for defining an IBM 2260-2848 Local line group are:

- A DTFQT macro must be specified for each IBM 2848 Display Control. More than one 2848 cannot be defined in the same line group. However, the same 2848 can be defined in more than one line group.

- No IBM 2260 Display Station or 1053 Printer within the line group can be defined as part of another line group.

- The same number of buffers must be assigned in advance for each transfer of data from a 2260 to the computer.

- The same type of Read operation must be used (Read Display Station [DS] Manual Input [MI] or Short Read (DS MI)for each 2260 in the line group.

- The same LPS must be used for each device in the line group.

- The relative position of the device access area in a terminal table entry must be the same for all terminal table entries associated with the line group (see the section The Terminal Table).

DIRECT ACCESS CHECKPOINT RECORDS FILE

The direct access checkpoint records file contains a record of the status of the queues and the telecommunications network. This information is written onto the DASD at user-specified intervals. Two such checkpoint records are maintained in the file along with a pointer to the most recent record. One DTFQT macro instruction is required to define the DTF table for the checkpoint records file.

Before the checkpoint records file is defined, the following steps must be performed:

1. An area is allocated for the file on the DASD volume to be used, and

2. The area allocated for the file is preformatted by the user with one dummy record at the beginning of the file.

The number of tracks required may be calculated from the formula:

$$N = \frac{[288 + 2.1(M)]}{3625}$$

where

N = the number of tracks. Must be rounded to the next higher integer.

M = the number of bytes in the checkpoint record. Must be the same as that specified in the SOWA operand provided checkpoint records. It may be computed by the formula provided in the description of the SOWA operand in Figure 7.

The dummy record in the file must be four bytes in length, and the first byte must be binary 0. Formatting of other records needed is performed by the checkpoint routine. This formatting process is necessary only the first time this area is used, and not each time the message controlprogram is initiated. However, if the file has not been closed, because of a system failure, and the user does not wish to perform a restart operation, he must reinitialize (reformat) this file before initiating the message control program. An access method or utility other than QTAM must be used to format the area for this file.

The checkpoint records file is checked for incorrect formatting at the time the file is opened by an OPEN macro instruction. If the file has been formatted inco-

rrectly, an error message is prcvided, and the jcb is cancelled.

MESSAGE LCG FILE

A message log file consists cf messages that are stcred and maintained sequentially cn secondary storage for accounting purposes. A message log can be prcduced as a byproduct of normal message handling. The appropriate DTFxx macro instruction must be specified fcr each message log required by the user. The DTFxx used depends on the seccndary storage device emplcyed. Magnetic tape (DTFMT) is the storage medium generally used. (Fcr a detailed discussion of the appropriate DTFxx macro instructicn, refer tc the Supervisor and Input/Output Macros publication listed in the Preface.) The QTAM message control program employs logical ICCS tc reccrd the messages cn the lcg.

FILE DEFINITICN MACRC INSTRUCTICNS

DTFQT Macro Instruction

Cne DTFQT macrc instruction must be specified for the DASD message queues file and for each communication line group file. If the Checkpoint/Restart feature is desired, a DTFQT macro instruction must be provided for the DASD checkpoint records file. At assembly time, DTFQT causes the allocation of main storage for a DTF table. Parameters based on the keyword operands specified in the macro instruction are included in the DTF table.

Name	Operation	Operand
dtf	DTFQT	keyword operands

dtf
: Is the name of the macro instruction. It is also the name of the DTF table generated by the expansion of the macro instruction. The name is specified by from one to seven nonblank characters. The eighth byte of the name field is used by QTAM to indicate a 2311 or 2314 direct access device.

keyword operands
: Are the operands that can be included. The operands are separated by commas and are described for each type of file in Figures 6, 7, and 8.

Keyword Operand	Value Description
TYPE=DA	DA Identifies the file organization as that of the DASD message queues for telecommunications.
DEVADDR=SYSnnn	SYSnnn Specifies the symbolic unit to be associated with this logical file. If multiple volumes are used, only the symbolic unit for the first volume is specified. Actual units and channels are assigned to the file at job execution with appropriate ASSGN statements. The file is expanded to accommodate up to 16 extents.
[SEPASM=YES] [SEPASM=NO]	YES Specifies that this DTFQT is to be assembled separately from the rest of the user's code. NO Specifies that this DTFQT is to be assembled with the rest of the user's code. NO is assumed if this operand is omitted.
EOJAD=relexp	relexp Is the address of the instruction that begins a user-written section of code that closes all files opened in the message control program and performs other termination functions. Refer to the section, Deactivating the Telecommunications System, for a discussion of the functions required in this user-written section of code.
[DEVICE=2311] [DEVICE=2314]	2311 Specifies a 2311 direct access device. 2311 is assumed if this operand is omitted. 2314 Specifies a 2314 direct access device.

Figure 6. Keyword Operands for the DASD Message Queues DTFQT Macro Instruction

Keyword Operand	Value Description
TYPE=CK	CK Identifies the file organization as that of the checkpoint records for telecommunications.
DEVADDR=SYSnnn	SYSnnn Specifies the symbolic unit to be associated with this logical file
[SEPASM=YES] [SEPASM=NO]	YES Specifies that this DTFQT is to be assembled separately from the rest of the user's code. NO Specifies that this DTFQT is to be assembled with the rest of the user's code. NO is assumed if this operand is omitted.
SCWA=m	m The number of bytes in the checkpoint record. It may be computed by the formula $m = 17 + T + 11D + 14Q + 3R + 9L + P_1 + \ldots + P_n$ where: T = the number of bytes in the terminal table, less four bytes for each terminal table entry, and not including the table control field. D = the number of DASD destination queues. Q = the number of DASD process queues. R = the number of MS destination queues. L = the number of lines. $P_1+\ldots+P_n$ = the sum of the sizes of the polling lists in bytes, less one byte for each polling position.
[DEVICE=2311] [DEVICE=2314]	2311 Specifies a 2311 direct access device. 2311 is assumed if this operand is omitted. 2314 Specifies a 2314 direct access device.
DQMAX=integer	integer Specifies the number of destination queues for the processing program (i.e., number of DTFDQ macro instructions issued in the processing program). If this operand is omitted, 2 is assumed.

Figure 7. Keyword Operands for the Checkpoint Records DTFQT Macro Instruction

Keywcrd Operand	Value Description
TYPE=LG	LG Identifies the file organization as that of a communication line group.
CLPS=lpsname	lpsname Is the name of the line procedure specification (LPS) section for this line group. This name must be identical to the name specified in the name field of the LPSTART macro instruction that begins the LPS section for this line group.
LINELST= (nnn,...)	nnn Specifies via a sublist the correspondence between symbolic unit (SYSnnn) and relative line number. In the sublist, the user codes one 3-digit number for each line in the line group. The 3-digit number is interpreted as the "nnn" of SYSnnn. The order of coding the 3-digit numbers determines which symbolic units are associated with the individual lines in the line group. As many as thirty-one 3-digit numbers from 000-244 may be coded in the sublist. Example: LINELST=(005,010,007) This results in associating: SYS005 with relative line number 1, SYS010 with relative line number 2, SYS007 with relative line number 3, in a line group comprising three lines. The DTFQT macro expansion generates a Line Control Block (LCB) for each line defined in the sublist. Each LCB contains a command control block (CCB) and other information needed to control I/O operations on the line. An actual unit and channel are assigned to the line at execution time by an ASSGN statement. This operand must be omitted when defining a line group for an IBM 2260-2848 local. The symbolic unit assignment (SYSnnn) for each 2260 Display Station or 1053 Printer included in the line group is specified in the TERM macro instruction for the terminal. (See the description of the TERM macro instruction.) Because data transfer can occur between the computer and only one 2260 (or the 1053) at a time, only one LCB is generated for the entire line group.
SWITCH={YES / NO}	YES Specifies that the lines in this line group are switched lines; that is, the line connection between the system and the terminals is not permanent. NO Specifies that this line group consists of nonswitched lines; that is, the line connection is permanent.
AUTOPOL = YES AUTOPOL = NO	YES Specifies that the polling process, for the lines in this line group, is under the control of the Auto Poll feature. NO Specifies that the Auto Poll feature is not available for this line group. If this operand is omitted, AUTOPOL=NO is assumed.

Figure 8. Keywcrd Cperands for the nonaudio Communication Line Group DTFQT Macro Instruction (Part 1 of 8)

Keyword Operand	Value Description
CU=code	code Defines the type of telecommunications control unit as a 2701, 2702, 2703, or 2848. CU=2701 must be specified if DEVICE=2848 is specified. CU=2848 must be specified if DEVICE=2260 is specified. CU=2702 or 2703 must be specified if AUTOPOL=YES is specified.
DEVICE=code	code Specifies the type of terminal associated with the line group as a 1030, 1050, 1060, 2260, 2848, 83B3, 115A, TW33, TW35, WTTA, 274A, 274B, 274C, 274D, 274E, 274F, 274G, or 274H. (See Note 2). Note 1: DEVICE=2260 applies to the 2260 Local. DEVICE=2848 applies to the 2260 Remote Note 2: IBM 2740 Communication Terminals are identified to the system using the following: On a nonswitched network: 274A: Basic 2740 274C: Basic 2740 with station control 274D: Basic 2740 with station control and checking. 274F: Basic 2740 with checking On a switched network: 274B: Basic 2740 274E: Basic 2740 with transmit control and checking 274G: Basic 2740 with checking 274H: Basic 2740 with transmit control Note 3: If the terminal is an IBM Model 2 with record checking, 274D is specified; if the 2740 Model 2 is without the checking feature, 274C is specified.
CPOLL= (pollname$_1$,...)	pollname$_1$ Is the name of the polling list for the first line in the line group. There must be one value (a polling list name) in the sublist for each line in the line group. The order of these polling list names must correspond to the order of the lines as specified in the LINELST keyword operand. Each polling list name must be identical to the name specified in the POLL macro instruction used to define the list for that line. If a line is used for output only, the name of a polling list with no terminal entries must be specified; any number of output-only lines may refer to this name. Example: CPOLL=(PCLL1,OUTPUT2,POLL3,OUTPUT2). The fourlines to which the values in the sublist refer must be in the same order in the LINELST keyword operand as in the sublist above. OUTPUT2 is the address of a polling list with no terminal entries. The second and fourth lines, used for output only, refer to this address. If the line group is for a 2260-2848 Local, the sublist must contain only one entry: the name of the POLL macro defining all 2260 Display Stations in the line group from which messages can be received.

Figure 8. Keyword Operands for the nonaudio Communication Line group DTFQT Macro Instruction (Part 2 of 8)

Keyword Operand	Value Description
[BUFNO=absexp BUFNO=2]	absexp Is the number of buffers to be requested for each transmission of data from a terminal to the computer. The requests are made in advance of the message transmission, and all buffers after the first are assigned as they are needed. absexp should be equal to or greater than 2, and must not be greater than either 255 or the number of buffers specified in the BUFFER macro instruction, whichever is less. The primary factors to be considered in determining the value of absexp are the line speed, the size of the buffer pool as opposed to the average number of buffers that are active at any one time, the size of each buffer as opposed to the average size of a transmitted block, and total system loading. If this operand is omitted or if an illegal value is specified, 2 is assumed. The following method of calculating BUFNO for each line group may be used. Assume the slowest-speed lines in the system to have a value of one. Then, assign to each of the remaining lines in the system a value whose ratio to one is the same as the ratio of the line's speed to the slowest line's speed. The value of the BUFNO operand for each line group equals the value for the lines comprising the group, plus one. Example: Line Group A has the slowest lines, 60 characters per second (cps). Its value is therefore 1. BUFNO = 1 + 1 = 2. Line Group B has 120-cps lines; therefore its value is 120/60, or 2. BUFNO for this line group is therefore 2 + 1 = 3. Line Group C has 150-cps lines. 150/60, rounded up, is 3. For this line group, BUFNO = 3 + 1 = 4. Line Group D has 180-cps lines. 180/60 = 3. This line group, too, should have BUFNO equal to 3 + 1 = 4. For a 2260-2848 local line group. absexp must define the number of buffers needed to contain the maximum length message to be received from any 2260 Display Station in the line group. Due to the high data rate of this local device, all required buffers must be assigned before data transfer begins. If too few buffers are specified, any excess data is ignored, and the 'insufficient buffers' bit (bit 4) is set in the error halfword for the line (see figure 13).
TYPEFLE={INPUT CMBND OUTPUT}	INPUT Indicates that the line group is to be used for input operations. CMBND Indicates the line group is to be used for both input and output operations. OUTPUT Indicates that the line group is to be used for output operations. If OUTPUT is specified, the CPOLL keyword operand of the DTFQT macro instruction refers to a polling list with no terminal entries. OUTPUT must not be specified if SWITCH=YES or AUTOPOL=YES is specified.

Figure 8. Keyword Operands for the Communication Line Group DTFQT Macro Instruction (Part 3 of 8)

Keyword Operand	Value Description
INTVL=absexp INTVL=0	absexp Is the polling interval (that is, the number of seconds of intentional delay between passes through a polling list) for the lines in this line group. After all the terminals in a polling list for a given line have been polled (beginning to end), and if no message traffic was received from any terminal in the list, a delay equal to the number of seconds specified in this operand occurs before polling is restarted at the beginning of the list. However, if a message is received from any terminal represented in the list, the specified interval is ignored and polling resumes immediately at the beginning of the list. The purpose of the polling interval is to limit nonproductive polling during slow traffic periods. absexp must not be greater than 255. If this operand is omitted, INTVL=0 is assumed; it must be omitted if this line group consists of switched lines, of lines using Auto Poll, of WTTA lines, or is for the IBM 2260-2848 Local. Restriction: If this operand is specified, the interval timer feature is required, and must be assigned to the foreground-one partition. With multitasking the main task cannot use STIXIT IT if QTAM is attached with operator control, checkpoint by timer interval or polling interval facilities.
THRESH=(absexp$_1$, absexp$_2$, absexp$_3$, absexp) THRESH=(255,10,5,5)	Provides the threshold values to be used in determining excessive number of errors (both temporary and permanent) for a specified number of transmissions for each line of this line group. This operand is not applicable and must be omitted for a 2260-2848 Local line group. absexp$_1$ The threshold value for the number of transmissions (must be from 1 to 255 inclusive). If the number of transmissions on any line in this line group reaches this threshold before any of the error counters reach their thresholds, the threshold counters are reset. absexp$_2$ The threshold value for the number of data checks (must be from 1 to 255 inclusive). If the number of data checks on any line in this line group reaches this threshold before the number of transmissions reaches its threshold value, a message is provided (to the 1052 system console or the telecommunication system control terminal if the OPCTL macro instruction is included), and the threshold counters are reset. absexp$_3$ The threshold value for the number of intervention required errors (must be from 1 to 255 inclusive). Same action is taken as in absexp$_2$.

Figure 8. Keyword Operands for the Nonaudio Communication Line Group DTFQT Macro Instruction (Part 4 of 8)

Keyword Operand	Value Description
[CPRI=R CPRI=E CPRI=S]	absexp The threshold value for the number of timeouts, (except text timeouts) must be from 1 to 255 inclusive. Same action is taken as in absexp$_2$. If this operand is omitted for a line group other than the 2260-2848 Local the threshold values of 255, 10, 5, 5 are assumed. R,E, or S Indicates the relative priority to be given to sending and receiving operations on the lines in the line group: R -- receiving has priority over sending. For polling lines, output messages are sent when a polling interval is being observed. It should be noted that the specified polling interval is observed only when no message traffic was received during the polling pass just completed for the line. E -- receiving and sending have equal priority. After each full polling sequence on a given line, all output messages queued for that line are transmitted. S -- sending has priority over receiving. For polled lines, output messages are sent on such a line after polling of a terminal has resulted in a negative response or after an incoming message ending with an EOT has been received. For auto-polled lines, output messages are sent on such a line after an incoming message ending with an EOT has been received or after the end of the polling list has been reached. In any case, polling on a line resumes when the queue of output messages for this line is exhausted. If the line group consists of 2740 Model 2 (buffered) terminals, the relative priority of the lines remains the same, however, if queueing by terminal has been specified, messages are sent to alternating terminals on the line, within the priority with the following exceptions: E -- a polling sequence will be executed when no message can be transmitted on the line because ''buffer busy'' condition has been signalled from all terminals with messages on the queue. Sending is resumed after each full polling sequence. S -- output messages are transmitted until all queues are exhausted unless a ''BID'' condition is detected. In this case message transmission is deferred until the remainder of the messages for the line have been transmitted and one polling sequence executed. For WTTA lines, the relative priority is as follows: R or E -- output messages are sent where there is no traffic over the line after an EOT character has been received or after a time-out has occurred. S -- output messages are sent when there is no traffic over the line, after an EOT or EOM character has been received, or after a time-out has occurred If the line group consists of i) IBM 2740 terminals, types 274A or 274F, ii) switched lines, or iii) IBM 2260-2848 Local terminals, this operand must be omitted. If this operand is omitted, CPRI=S is assumed.

Figure 8. Keyword Operands for the Nonaudio Communication Line Group DTFQT Macro Instruction (Part 5 of 8)

Keywcrd Cperand	Value Description
[RTYPE=SHORT RTYPE=NCRM]	This cperand applies only to an IBM 2260-2848 Local line group. It specifies the type of Read operation to be performed when receiving from the 2260 Display Stations included in the line group. SHCRT Specifies Short Read DS MI. NORM Specifies Read DS MI. If this operand is cmitted and CU=2848 is coded, TYPE=NORM is assumed.
[ACLCC=integer ACLCC=subfield]	integer Is the position, relative to zero, of the device-access field for each terminal table entry defined by a TERM macro and associated with this line group. All terminal table entries begin on a fullword boundary. The value of integer is the sum of: 9 + c + c + r where 9 is the number of bytes placed by QTAM at the beginning of each terminal table entry; c is the maximum number of characters used in the name field of any TERM, LIST, or PROCESS macro within the terminal table IJLQTTID field; o is the total number of bytes (including any bytes necessary for boundary alignment) within the optional-area subfields of each terminal table entry associated with the line group (see the OPTION macro instruction description); and r=2 if the OBR/SDR (Outboard Recorder/Statistical Data Recorder) option is included in the message control program, otherwise r=0. QTAM uses the value specified in this operand to obtain from the device-access area: 1. the polling and addressing characters for terminals on ncnswitched lines, 2. dialing infmation for terminals on switched lines, or 3. the CCB for terminals in a 2260-2848 Local line group, or 4. the terminal identification field for WTTA terminals. Example: If the maximum number of characters in the name field of TERM macros for this line group is eight, the number of bytes used for optional-area subfields is 13, and the OBR/SDR option is not included, "integer" = 9 + 8 + 13 + 0 = 30. Therefore, ACLCC = 30. subfield Is the name of the last optional subfield used in this line group. If SEPASM=YES is specified, this method of specifying ACLCC cannot be used. If no optional subfield is used in this line group and if SEPASM=NO is specified or assumed, the ACLOC operand may be omitted.

Figure 8. Keywcrd Cperands for the Nonaudio Communication Line Group DTFQT Macro Instruction (Part 6 of 8)

Keyword Operand	Value Description
[SEPASM=YES] [SEPASM=NO]	YES Specifies that this DTFQT is to be assembled separately from the rest of the user's code. NO Specifies that this DTFQT is to be assembled with the rest of the user's code.
[MCN=YES] [MCN=NC] (WTTA lines cnly)	YES Specifies that each terminal of the line group is equipped with the optional Mctor-On feature. NO Specifies that the terminals are not equipped with the Motor-On feature. NC is assumed if this operand is omitted.
[MCNDLY=integer] [MCNDLY=15] (WTTA lines only)	integer Specifies the number of Mark characters corresponding to a 1.5-second time-out when the terminal is not equipped with the optional Mctor-Cn feature. MONDLY=10 corresponds to 50-service MCNDLY=15 corresponds to 75-baud service,, and MONDLY=20 corresponds to 100-baud service. When this operand is omitted or integer exceeds 20,, MONDLY=15 is assumed.
[IAM=YES] [IAM=NC] (WTTA lines cnly)	YES Specifies that the terminal can ask for the computer identification sequence by sending FIGS D. NC Specifies that the computer cannot ask for the identification sequence cf the terminal. NO is assumed if this operand is omitted.
[WRU=YES] [WRU=NC] (WTTA lines only)	YES Specifies that by sending FIGS D,, either the computer or the terminal can ask for the identification sequence of the other. When WRU=YES is specified,, IAM=YES is assumed. NO Specifies that the computer cannot ask for the identification sequence cf the terminal. NO is assumed if this operand is omitted (see Ncte 2).
[ECM=WRU] [ECM=X'hh'] [ECM=X'hh1F'] (WTTA lines only)	WRU Specifies that the end-of-message signal is either the WRU signal (when WRU=YES or IAM=YES is specified) or FIGS D (when-both WRU=NC and IAM=NO are specified). FIGS x is set as FIGS D in the adapter (see Note 1). X'hh' Specifies that FIGS x is used as the EOM signal. hh is the hexadecimal representation of FIGS x set in the adapter. X'hh1F' Specifies that FIGS y LTRS is used as the EOM signal. hh is the hexadecimal representation of FIGS y set in the adapter. WRU is assumed if this operand is omitted.

Figure 8. Keyword Cperands for the Nonaudic Communication Line Group DTFQT Macro Instructicn (Part 7 of 8) -- Applicable only to WTTA Lines

Keyword Operand	Value Description
[ECT=2EOM] [ECT=X'hh1F'] (WTTA lines only)	2EOM Specifies that two consecutive EOM signals will be recognized by QTAM as end-of-transmission, except when IAM=YES and ECM=WRU are specified. X'hh1F' Specifies that FIGS y LTRS is used as the EOT signal (see Note 1). Therefore, ECM=X'hh1F' can not be specified for the EOM signal. Note: A time-out is also recognized as EOT. Moreover, two consecutive EOM signals are always recognized as an EOT signal, except when IAM=YES and ECM=WRU are specified.
Note 1: In the preceding description of the ECM and EOT operands, x and y are the values assigned by the user and set in the adapter at the time of installation of the equipment. Note 2: If neither IAM nor WRU is specified, no exchange of identification sequences can be requested.	

Figure 8. Keyword Operands for the Communication Line Group DTFQT Macro instruction (Part 8 of 8--applicable only to WTTA lines)

Examples:

1. A DTFQT macro instruction that defines the DTF table representing a DASD message queues file:

Name	Operation	Operand
DISK	DTFQT	TYPE=DA,DEVADDR=SYS005,EOJAD=FINISH

2. A DTFQT macro instruction that defines the DTF table representing a communication line group file:

Name	Operation	Operand	Col. 72
GROUP1	DTFQT	TYPE=LG,CLPS=IPS1,LINELST=(006,007,008),SWITCH=NO, CU=2702,DEVICE=1050,CPOLL=(POLL1,POLL2,POLL3),BUFNO=3, TYPEFLE=CMBND,ACLOC=16	

CONTROL INFORMATION

In constructing the message control program for his telecommunications system, the user must provide certain control information. This data includes:

- A terminal table that contains all of the terminal codes as well as complete information about the terminals connected to the system. These terminals include the message processing programs, which are the only terminals to be specified for an audio application.
- A polling list, for each communication line, that specifies the sequence in which terminals on the line are to be polled.
- Buffer specifications that define the maximum number of buffers for the QTAM buffer pool and the size of the message segments used in the system

The IBM-provided logic that supports the message control program uses this control information in performing the message handling functions specified by the user. Macro instructions are provided that allow the user to define the terminal table, polling lists, and buffer areas in accordance with the requirements of his application.

TERMINAL TABLE

A telecommunications system using QTAM requires one terminal table.

For the polling system of the telecommunications system, control information macro instructions are used to produce, at assembly time, a terminal table tailored to the user's device configurations and options desired. The terminal table consists of an 8-byte table control field defining the length of the table, and blocks of information about each terminal. Each such block is called a terminal table entry. The four types of entries are: terminal, group code, distribution list, and process program. Each entry in the terminal table begins on a fullword boundary.

Each type of entry is described in Appendix A and in the following paragraphs.

The size, structure, and contents of the terminal table are based on information provided by the user through the TERMTBL, OPTION, TERM, LIST, and PROCESS macro instructions.

- TERMTBL is specified once and defines the limits of the table.
- TERM creates a single terminal or group code entry in the terminal table.
- OPTION names and allocates storage for an optional subfield to be included in the user area of a terminal table entry. The optional subfields can contain information needed to perform various optional functions provided by QTAM (subsequently discussed) or the user or both. The initial contents of each subfield are specified by the TERM macro instruction that defines the entry.
- LIST defines a distribution list entry.
- PROCESS creates a process program entry.

Single Terminal Entry

The terminal table must contain a single terminal entry for each terminal that can send and receive, send only, or receive only a message (except for a terminal in a group code entry discussed below). If a terminal component is individually polled or addressed, it must also have a separate single terminal entry.

Each single terminal entry contains a minimum of seven fields. The names of the first six fields are provided in a DSECT generated by the expansion of the TERMTBL macro instruction. The first five fields in each entry are of standard length and are described below:

Field	Description
IJLQTSZE	Size of the entry.
IJLQTQAD	Address of the queue control block for the DASD destination queue associated with the terminal.
IJLQTSIN	Sequence number for messages incoming from this terminal.
IJLQTSOT	Sequence number for messages outgoing to this terminal.
IJLQTSTA	Status information indicating whether messages to the terminal are to be suppressed, whether messages can be sent to the terminal, and whether messages can be received from the terminal.

The sixth field (IJLQTTID), containing the name assigned to the terminal by the user, appears in each single terminal entry. This name is the same name that can appear in the source or destination code field of the message header. The length of this field is the same in each entry and is based on information provided by the user. If the number of characters in each terminal name varies, the number of bytes in this field is equivalent to the number of characters in the longest terminal name. The user must specify this number in the TERMTBL macro instruction. If the number of characters in each terminal name does not vary, the number of bytes in this field is equivalent to the fixed number of characters. The IJLQTTID field can be a maximum of eight bytes.

Inclusion of the seventh field is optional. This field, called the user area, can contain one or more optional subfields. The name, length, and boundary alignment of each such subfield, if any, are specified by an OPTION macro instruction; the data content of the subfield is specified by a TERM macro. Optional macro instructions such as COUNTER, DIRECT, ERRMSG, INTERCPT, POLLIMIT, and REROUTE introduce routines that either obtain information from or place information into these subfields in order to perform their functions. The user can also store information into this area. Each single terminal and group code entry associated with

the same line group must contain the same optional subfields.

The eighth field, called the device access area, is required. If the terminal is on a nonswitched line, this area contains the polling and/or addressing characters for the terminal. If the device is a WTTA terminal, this area contains the terminal identification sequence. For a terminal on a switched line, this area is composed of the number of digits in the telephone number, the telephone number, and a third field as follows:

1. For an IBM 1050, the addressing characters for the terminal.
2. For a TWX terminal, the characters in the TWX terminal identification sequence.
3. For an IBM 2260 or 1053 in a 2260-2848 local line group, a CCB and other control information required for the device.

The size of the device access area depends on the requirements for the particular device. This field immediately follows the IJLQTTID field if no optional subfields are included in the user area. If optional subfields are included, the device access field follows the last subfield. The total size of the terminal name area, optional user area, and device access area must not exceed 243 bytes.

The TERM macro instruction provides the initial contents for all fields in the single terminal entry. Detailed information on this entry is contained in Appendix A.

Group Code Entry

The group code entry is applicable only to AT&T 83B3, WU Plan 115A, IBM 2740's with station control (with or without checking), and IBM nonswitched 1050 terminals.

The terminal name in a group code entry represents a prespecified group of terminals on a line with special equipment that provides the group code feature. The feature permits simultaneous transmission of a message to a group of terminals through the specification of a single set of unique addressing characters. Several combinations of prespecified terminals can be grouped for this purpose. Each group has a group terminal name and a corresponding group code entry in the terminal table.

A group code entry is identical in structure to the single terminal entry. However, three fields either are not used or are used in a different manner. QTAM increments by one the sequence number for outgoing messages (IJLQTSOT field) when the group is simultaneously sent a message. If any terminal in the group is also represented by a single terminal entry, the output sequence number for that entry is not changed.

The sequence number for incoming messages (IJLQTSIN field in the single terminal entry) is not applicable to the group code entry because the terminal group cannot collectively send a message to the system. For the same reason, there are no polling characters in the device access area of the group code entry. The total size of the terminal name area, optional user area, and device access area cannot exceed 243 bytes.

The TERM macro instruction provides the initial contents for all fields in the group code entry. Detailed information on this entry can be found in Appendix A.

Distribution List Entry

A distribution list entry contains a list of addresses of single terminal entries. These addresses are grouped under the list name. When the list name is used as a destination code for a message, QTAM sends the message via separate transmissions to all terminals indicated by the list. Each terminal in the list must have a corresponding single terminal entry in the terminal table.

Each distribution list entry contains five active fields and the list area. The first four active fields in each entry are of standard length. A description of the five active fields follows:

Field	Description
IJLQTSZE	Size of the entry.
IJLQTQAD	Address of the queue control block for the distribution list queue.
IJLQTLRA	An access key to the start of the list of addresses.
IJLQTSTA	Status information. This field functions in the same way as the corresponding field for a single terminal entry with the following exception: the receive bit in this

field is never used because terminals in the list cannot collectively send a message to the system.

IJLQTIID — Contains the distribution list name. This name serves the same purpose as the terminal name in a single terminal entry and is subject to the same restrictions.

The list of addresses of single terminal entries follows the fifth active field. This list contains, in each of its subfields (reladdr$_1$ through reladdr$_n$), a relative address that locates the corresponding single terminal entry in the terminal table. These addresses are relative to the base address of the table. The last subfield in the list is zero, indicating the end of the distribution list entry. The total size of the distribution list name area and the list area cannot exceed 243 bytes.

The LIST macro instruction provides the initial contents for all fields in the distribution list entry. Detailed information on this entry can be found in Appendix A.

Process Program Entry

The terminal table must include one process program entry for each DASD process queue (that is, one for each message processing program). The structure of this entry is the same as that of the single terminal entry with the following exceptions:

1. The IJLQTSIN field used in the single terminal entry is not used for the process program entry.
2. The receive bit in the IJLQTSTA field is not used for the process program entry.
3. The IJLQTTID field in the process program entry contains the name of a DASD process queue rather than a terminal name.
4. There is no optional area or device access area in the process program entry.

The PROCESS macro instruction provides the initial contents for all fields in the process program entry. Detailed information on this entry can be found in Appendix A.

Terminal Table (TERMTBL) Macro Instruction

The TERMTBL macro instruction causes a table control field to be created for the terminal table and defines the length of the table. Depending upon the type of application, the expansion of this macro instruction also generates up to four DSECTs that enable the user's symbolic references to communicate with those of QTAM. One DSECT provides names for the fields in each terminal table entry. Another DSECT generated by TERMTBL supplies names for the fields in a line control block (LCB); QTAM maintains an LCB for each communication line. Each LCB contains control information about its associated line. The fourth DSECT provides names for the buffer prefix.

One TERMTBL macro instruction is required, and it must precede all other macro instructions used in creating the terminal table.

Name	Operation	Operand
	TERMTBL	entry[,n]
		[,OPCTL=chars]
		[,CPINTV=integer]
		[,OBRSDR=integer]

entry
: Is the name of the last entry in the terminal table.

n
: Is the number of characters in the longest terminal entry, distribution list entry, or process entry name. This operand is not necessary if the lengths of all terminal names are the same.

OPCTL
: The name of the OPCTL macro instruction in the LPS. This operand specifies that error messages are to be sent to the operator control terminal specified in the OPCTL macro instruction, rather than to the 1052 system console. When this operand is specified, error messages that originate from errors on the operator control terminal receiving error messages are printed on the 1052 system console. This operand must not be specified unless the OPCTL macro instruction is specified in the LPS. When this operand is not specified, the error messages continue to be sent to the 1052 system console.

CPINTV
The number of 15-second intervals between checkpoints. It must be an integer from 1 (15 seconds) to 60 (15 minutes) inclusive. Note that if this operand is specified, the interval timer feature is required, and must be assigned to the foreground-one partition. In multitasking, the main task cannot use STIMIT IT if QTAM is attached with operator control, checkpoint by timer interval, polling interval facilities, or 2740 Model 2 in the application.

OBRSDR
The number of 8-byte SDR (Statistical Data Recorder) counter sets to be provided. It is calculated by counting one for each terminal and one for each dial line; if not enough counter sets are specified, the error counts for some terminals will be lost. Note that if this operand is specified, both CBR (Cutboard Recorder) and SDR facilities for recording error information are included in the message control program.

Note: The name field is ignored. IJLQTTBL is the name generated for the terminal table by the macro instruction expansion.

Terminal Table Optional Field (OPTION) Macro Instruction

The OPTION macro instruction names and allocates, at assembly time, a specified amount of main storage in selected single terminal and group code entries in the terminal table. The storage allocated constitutes the optional area field of the entries. One OPTION macro instruction is required for each optional area subfield desired. The order of the subfields within the optional area is determined by the order of the OPTION macro instructions.

Data values inserted into each optional area subfield are specified by the opdata operand of the TERM macro instruction that creates the entry. If a TERM macro instruction does not specify data values for the optional area, the optional area field is not included (that is, main storage is not allocated) in that entry. However, if the optional area is specified for any terminal in the line group, the space is reserved for every terminal in the line group (see the TERM Macro Instruction description).

The OPTION macro instruction(s), if used, must immediately follow the TERMTBL macro instruction. The relative order of the specified subfields must correspond to the order in which the contents for the subfields are specified in the TERM macro instructions.

Six LPS macro instructions link to IBM-provided routines that use fields allocated by OPTION macros in performing their functions. These optional LPS macros are COUNTER, DIRECT, ERRMSG, INTERCPT, POLLIMIT, and REROUTE; the functions provided by these macro instructions are discussed under the individual macro instruction descriptions. User-written routines can also store information into a subfield defined by an OPTION macro.

Name	Operation	Operand
subfield	OPTION	typelength

subfield
Is the name of the optional-area subfield.

typelength
Is the type and length of the subfield in the standard assembler language format (for example, H, CL8, AL3). When the subfield is used in conjunction with the DIRECT, ERRMSG, or REROUTE macro instruction, CLn must be specified where n equals or exceeds the longest name of any terminal table entry. If used in conjunction with the COUNTER macro instruction, "typelength" should be specified as H since COUNTER requires a half-word field aligned on a half-word boundary to perform its function. INTERCPT and POLLIMIT require 3-byte and 1-byte fields, respectively; no boundary alignment If used in conjunction with the POLLIMIT macro instruction, "typelength" must not be specified as C (or CL1), because the field must contain a binary value rather than a character representation.

Example: The following is an example of the use of the TERMTBL and OPTION macro instructions:

Name	Operation	Operand
	TERMTBL	KCHI
POLLMT	OPTION	FL1
COUNT	OPTION	H
ALTNTERM	OPTION	CL4
INTECPT	OPTION	XL3

TERMTBL defines KCHI as the terminal name in the last entry in the terminal

table. The OPTICN macro instructions allocate a 10-byte optional area for single-terminal and group-code entries in the terminal table. The opticnal area consists of four subfields: the PCILMT subfield ccntains cne byte of data to be used by the POLLIMIT macro; the COUNT subfield contains a half-wcrd for decimal data to be used by the COUNTER macro; the ALTNTERM subfield contains fcur bytes for a character string to be used by the DIRECT macrc; and the INTECPT subfield contains three bytes fcr hexadecimal data to be used by the INTERCPT macrc.

The halfword specification fcr the COUNT subfield causes the assembler tc perform boundary alignment; however, in this case, no adjustment is necessary because the COUNT subfield already begins on a halfword boundary. If, in some other case, bytes are skipped by the assembler in performing boundary alignment, the number of bytes skipped must be included in the calculatien cf the ACICC keyword operand in DTFQT macro for the line group affected.

Terminal Table Entry (TERM) Macro Instruction

The TERM macro instruction causes a terminal name and associated terminal information tc be included as an entry in the terminal table. If a single terminal or compcnent is invclved, TERM produces a single terminal entry. If a group of terminals having the group code feature is involved, TERM prcduces a grcup code entry. Cne TERM macro instructicn is required fcr:

1. Each terminal (bcth switched and nonswitched) that can send, receive, or send and receive messages.

2. Each group of nonswitched terminals equipped with the group code feature. Terminals can only receive messages under the grcup code feature. Each terminal in the group that can also send messages must be represented by a single terminal entry.

All TERM macrcs in the same line group must be grouped together. If messages are to be queued by line rather than by terminal, the individual TERM macrcs must be grouped by relative line number within this TERM coding sectien. TERM entries within a line group must have exactly the same format. The same option fields and the same number cf dial digits and addressing and polling characters must be specified.

Name	Operation	Operand
symbol	TERM	qtype,filename,rln [,adchars] [,(opdata,...)] [,CALL=integer] [,CALL=NONE] [,ID=hexchars] [,DEVADDR=SYSnnn] [,PRNTR=YES]

symbol
: Is the terminal name containing one to eight nonblank characters; it must be specified. This name is the same name that appears in the source or destination code field of a message header.

qtype
: Specifies the type of message queueing.

  T specifies that outgoing messages are to be queued by terminal; that is, all messages for a given terminal are sent before any messages for other terminals are sent. (This is economically advantageous when the destination terminal is on a switched line.) Highest priority messages for the given terminal are sent first. T must be specified for switched terminals, and may be specified for nonswitched terminals.

  L specifies that outgoing messages are to be queued by communication line; messages for all terminals on the line are sent on a first-in-first-out basis within priority groups. If L is specified, all TERM macros for each line must be grouped together. L may be specified for nonswitched terminals; it cannot be specified for switched terminals. It is recommended that L be specified for terminals in a 2260-2848 Local line group.

  Note 1: Since only one terminal can be connected to a WTTA line, T and L are equivalent.

  Note 2: If the terminal is a 2740 Model 2, T should be specified.

filename
: Is the name of the DTF table for the communication line group in which the terminal is included.

rln
: Is the relative line number, within the line group, of the line over which the computer and the terminal communicate. Within a line group, the relative line numbers specified must be

consecutive integers beginning with one (zero is invalid). For a switched terminal, any value up to the highest numbered access line in the line group may be specified. When the computer calls a terminal, it attempts to make the call using the specified line number. If that line is busy, the call is retried using the next higher numbered line, and so on until a free line is found. If all remaining lines in the group are busy, the message is not sent until a line becomes available. A relative line number of one must be specified for terminals in a 2260-2848 local line group, because only one LCB is generated for the entire line group.

adchars

Are the addressing and/or polling characters for the terminal. The TERM macro expansion places these characters in the device access area of the terminal table entry. Refer to Figure 9 for the number and kind of characters to be specified for each type of terminal and line. The characters are specified by writing the hexadecimal equivalent of the appropriate device code representation. This operand must be omitted for a TWX terminal, for WTTA terminals, for IBM 2740 terminals, Types 274A, 274B, 274E, 274F, 274G, and 274H, and for IBM 2260 or 1053 terminals in a 2260-2848 local line group. Its omission must be indicated by a comma if the positional opdata operand is used. If the opdata operand is not used, no comma is coded to denote omission of the adchars operand. No polling characters are specified for a switched IBM 1050 terminal. If polling is required for a switched IBM 1050 terminal, the polling characters are specified in a PCIL macro instruction.

Examples: (1) If the addressing and polling characters for a nonswitched IBM 1050 are R9 and R0, adchars is written D213D215 (D213 and D215 are the hexadecimal equivalents of the device code representation of R9 and R0). (2) If the polling characters of a nonswitched IBM 1050 that is only to be polled (not addressed) are K0, adchars is written as xxxxC515, where xxxx represents the hexadecimal equivalent of any two fill characters. (3) If the addressing and polling characters of a nonswitched IBM 1030 that is to be polled and addressed are K and L, adchars is written as 45xx46, where xx represents the hexadecimal equivalent of any fill character. (4) If the same IBM 1030 is to be addressed only, adchars would be written as 45 (no fill character needed).

opdata

Is the actual data to be inserted into the optional area subfield(s) of the terminal table entry for this terminal. This operand allows flexibility because data can be specified for different subfields depending on the optional functions required for messages sent to or received from the terminal. All entries, however, representing terminals included in the same line group must have data specified for the same optional subfields. (Figure 10 shows an example of data specified for different subfields in entries not associated with the same line group.) The maximum length and type of data specified for each subfield must correspond to the length and type specified by the OPTION macro instruction that allocates the additional storage required for the subfield. A comma is used to:

1. Delimit the data for each subfield (except the last).

2. Indicate that no data is specified for an intermediate subfield.

Framing characters such as X, C, and quotes are not coded.

CALL

Is the telephone number of the terminal. This operand must be specified for switched terminals only. NONE must be specified if the line is a WTTA line or if the lines over which contact with the terminal is to be established does not have the autocall feature.

ID

Is the terminal identification sequence for the TWX or WTTA terminal represented by the terminal table entry created by this TERM macro instruction.

For TWX terminals, this operand is specified only when the computer is to call a terminal. It is specified by writing the hexadecimal equivalent of the 8-level TWX code. When the computer calls the terminal, the terminal automatically sends an identification sequence. QTAM compares the sequence with the sequence specified by this operand, as a check that the intended terminal has in fact been reached. An equal compare permits message transmission to proceed. An unequal compare is treated as a negative response; the message is not sent.

For WTTA terminals, this operand is specified only when the WRU=YES is present in the DTFQT macro instruction for the line group. It is specified by writing the hexadecimal equivalent of the 5-level code used by the terminal.

When an identification exchange is performed (with WRU=YES specified in the DTFQT macro instruction), QTAM compares the terminal identification sequence with the sequence specified by the ID operand. On an unequal compare, the transmission-error bit is set on in the error halfword for the line. Moreover, if the identification exchange has been performed at the beginning of an output message, the message-not-sent bit is set on in the error halfword for the line.

DEVADDR

Specifies the symbolic unit to be associated with this logical device. An actual unit and channel is assigned to the device at execution time by an ASSGN statement. This operand must be specified for IBM 2260 or 1053 terminals in a 2260-2848 Local line group; for all other terminals, this operand must not be specified. It generates a CCB and other control information required for the device.

PRNTR

PRNTR=YES is specified if the terminal is an IBM 1053 Printer in a 2260-2848 Local line group or if the terminal is an IBM 2740 Model 2.

Warning: Note carefully the discussion pertaining to the omission or inclusion of commas to indicate omitted adchars and opdata operands.

Terminal Type	Line Type	Specify
IBM 1050, 1060, 2260-2848 Remote AT&T 83B3, WU 115A	Nonswitched	AAPP (if polling and addressing are required) ffPP (if only polling is required)[1] AA (if only addressing is required)
IBM 1030 IBM 2740, Types 274C and 274D	Nonswitched	AfP (if polling and addressing are required) ffP (if only polling is required) A (if only addressing is required)
IBM 2740, Types 274C and 274D	Nonswitched Auto Poll	AfPs (if polling and addressing are required) ffPs (if polling only is required)
IBM 2740, Types 274A and 274F	Nonswitched	No polling or addressing characters are to be specified
IBM 1050	Switched	AA (if addressing is required)[2]
WU TWX 33, 35	Switched	CR LF I_1 I_2 . . . I_n CR LF XOn (This is the TWX terminal identification sequence; it is specified only if the computer is to call the terminal.)[3]
IBM 2740, Types 274B, 274F, 274G, and 274H IBM 2260-2848 Local	Switched	No polling or addressing characters are to be specified.

Legend: A = one addressing character; CR = carriage return; P = one polling character; LF = line feed; s = one space character; f = one fill character; XOn = transmitter on; I_1 I_2 . . . I_n = a sequence of characters identifying the TWX terminal

[1]The second "P" must be specified as hexadecimal FF if a general poll of all 2260s attached to a 2848 is desired.

[2]If polling is required, the polling characters are specified in a POLL macro instruction.

[3]If the terminal is to call the computer, the computer identification sequence must be specified in a POLL macro instruction.

Figure 9. Addressing and Polling Characters for the TERM Macro Instruction

Terminal Table List (LIST) Macro Instruction

The LIST macro instruction causes the name of a list of terminals, relative terminal table addresses of the entries for the terminals in the list, and associated information on the list to be included as an entry in the terminal table. The list of terminals is called a distribution list, and the entry produced is a distribution list entry.

One LIST macro instruction must be provided for each such distribution list to be created. Terminals can only receive messages through the distribution list transmission method; they cannot send them.

Name	Operation	Operand
symbol	LIST	entry

symbol
Is the name of the list. This must be

specified, and may be from one to eight nonblank characters.

entry,...
Includes the names of the terminals that are to be in the distribution list.

Restriction: A name representing another distribution list must not be included as an operand of the LIST macro instruction. All terminal names in the list must be defined by either a TERM or a PROCESS (without EXPEDITE option) macro instruction.

Terminal Table Process (PROCESS) Macro Instruction

The PROCESS macro instruction causes the name and associated information of a DASD process queue to be included as an entry in the terminal table. The entry produced is a process program entry. It differs from other terminal table entries in that it does not have an optional area or device access area, and the IJIQTSIN field is not used. A message processing program, like a terminal, can be a destination for a message. However, unlike a terminal, the processing program is not associated with a communication line and does not need addressing and polling characters.

One PROCESS macro instruction must be included for each MS process queue that is defined by a DTFQT macro instruction in a message processing program.

The EXPEDITE operand permits the user to speed the processing of messages by the message processing program associated with the process program entry. This function is valuable for an application requiring immediate transmission of data to a message processing program. A combination of the EXPEDITE function and conversational mode (subsequently discussed) provides the most rapid response to an inquiry.

Name	Operation	Operand
symbol	PROCESS	[EXPEDITE]

symbol
Is the name of the process program entry in the terminal table. The name must be specified, and must be the same as the PROCESS keyword operand specified in the MS-process queue DTFQT macro instruction in a message processing program. The name can contain from one to eight nonblank characters.

EXPEDITE
This optional operand specifies that messages are to be routed directly to the message processing program's MS-process queue, bypassing the normal intermediate step of placing the messages on a DASD-process queue. EXPEDITE should not be specified if multisegment messages are expected, because segments from different messages may be intermixed as they are delivered to the queues. This restriction does not apply for multisegment messages received from an IBM 2260 Local.

Note: The RETRIEVE macro instruction cannot be used to retrieve messages from a process queue when the EXPEDITE operand is used. Also, the EOA, EOB, EOBLC, CANCELM, ERRMSG, and REROUTE macro instructions cannot be used for any message whose destination is a processing program identified by a PROCESS macro instruction with the EXPEDITE operand.

Example -- Terminal-Table Definition

Figure 10 shows a coding sequence used to create a terminal table. The terminals are IBM 1050's attached to the computer by nonswitched lines. There is only one message processing program.

No.	Name	Operation	Operand
1		TERMTBL	CPU,
2	COUNT	OPTION	FL2
3	LIMIT	OPTION	FL1
4	DEST	OPTION	CL3
5	NYC	TERM	L,GROUP1,1,E407E40D,(0,8,BOW)
6	BOS	TERM	L,GROUP1,1,E207E20D,(0,3,NYC)
7	WAS	TERM	L,GROUP1,2,E407E40D,(0,1,NYC)
8	PHI	TERM	L,GROUP2,1,E407E40D,(0,,NYC)
9	PIT	TERM	L,GROUP3,1,E407E40D,(0)
10	RAL	TERM	L,GROUP4,1,E407E40D
11	BCW	LIST	(BOS,WAS)
12	CPU	PROCESS	

Figure 10. Example of Coding Sequence Used to Create a Terminal Table

Instruction 1 (TERMTBL): Identifies the last entry in the terminal table. Omission of the second operand indicates that all terminal names are of equal length.

Instructions 2 through 4 (OPTION): Define the names and sizes of three optional area subfields used for functions specified by the COUNTER, POLLIMIT, and DIRECT macro instructions, respectively (refer to the Line Procedure Specifications section for a detailed discussion of the functions performed by these macro instructions). The single terminal entries in which these subfields are included and used depends on whether the optional functions are specified in the IPS that handles the line group associated with the entry.

Instructions 5 through 10 (TERM): Define the single terminal entries for the terminals in four line groups. The operands of each TERM macro instruction provide information for the fields of the respective entries. In this example, outgoing messages are queued by line. Therefore, the first operand is an L in each case.

The second operand of each TERM specifies the name of the DTF table for the line group in which the terminal is included. In this case, there are four line groups (hence, four DTF tables).

The third operand of each TERM is the relative line number of the line to which the terminal is attached. The user establishes the relative line number of each line at system generation. The relative line number can be modified by an ASSGN statement. In the line group associated with the DTF table named GROUP1, the New York and Boston terminals are attached to one line (relative line number 1), and the Washington terminal is attached to another line (relative line number 2). Since the messages are queued by line, individual TERM macro instructions must be grouped by relative line number. For example, it would be incorrect if the TERM macro instructions in this line group were in the order: NYC, WAS, BOS.

The fourth operand of each TERM contains the addressing and polling characters for the terminal. These characters are specified in the hexadecimal equivalent of the device code. In instruction 5, E407 is the hexadecimal equivalent of B3 and E40D is the hexadecimal equivalent of B6 in IBM 1050 code. B3 constitutes the addressing characters; for IBM 1050s, the first addressing character identifies the terminal, and the second identifies the component (the number 3 indicates the component addressed is a card punch). B6 constitutes the polling characters (6 is the identification of a card reader). Translation of the addressing polling character representations in instruction 6 is A3A6; translations of instructions 7 through 10 are all B3B6. If all these terminals were on the same line, and all were to be addressed or polled individually, a unique

set of addressing and polling characters would have to be assigned to each terminal.

The fifth operand of each TERM contains the data to be inserted in the subfields defined by the OPTION macro instructions. Instructions 5 through 7 specify data for all three optional subfields, because the LPS that operates on messages for this line group (GROUP1) includes the COUNTER, POLLIMIT, and DIRECT macro instructions. COUNTER uses the COUNT subfield to keep a count of all messages received by the New York, Boston, and Washington terminals, respectively. The count is set initially to zero in all entries. POLLIMIT uses the value in the LIMIT subfield to restrict the number of messages that can be sent by a terminal during one polling pass. The New York (NYC), Boston (BOS), and Washington (WAS) terminals can send a maximum of 8, 3, and 1 messages, respectively, during each polling pass. The DIRECT macro instruction uses the name specified in the DEST subfield to determine where to send the messages originated by each terminal. All messages sent by the New York terminal are directed to the Boston and Washington terminals because the distribution list entry, BOW, is specified. Messages sent by the Boston and Washington terminals are directed to the New York terminal.

Instruction 8 specifies data for only the COUNT and DEST subfields; a POLLIMIT macro instruction requiring information from the LIMIT subfield is not included in the LPS that operates on messages for this line group (GROUP2). It should be noted that an additional comma is required to specify that no data is inserted in the LIMIT subfield. Instruction 9 specifies data only for the COUNT subfield because neither the POLLIMIT nor DIRECT macro instruction is in the LPS for the line group (GROUP3). In this case, no additional commas are required because no subfield following the COUNT subfield is used. Storage is not allocated for the LIMIT and DEST subfields in the terminal table entry for the Pittsburgh (PIT) terminal. Instruction 10 does not specify data for any of the optional subfields since none of the optional functions are specified in the LPS for the line group (GROUP4). No storage is allocated for the optional area field in the entry for the Raleigh (RAL) terminal.

Instruction 11 (LIST): Creates a distribution list entry in the terminal table. When BOW (the name of the list) is found as a destination code in a message header or in the DEST subfield of the terminal sending the message, the message is routed to the Boston and Washington terminals.

Instruction 12 (PROCESS): Creates a process program entry in the terminal table. When CPU is specified as the destination code for a message, the message is routed to the message processing program represented by this process program entry. This input message is either directly routed to the message processing program represented by the CPU process program entry, or it is queued on the waiting chain located in the expansion of this macro instruction.

POLLING LISTS

Polling is a centrally-controlled method of permitting each terminal on a multiterminal nonswitched line to send messages without contending for use of the line. QTAM contacts the terminals in the order established by a user-specified polling list. The polling list consists of control information followed by a series of pointers to those terminal table entries representing terminals to be polled. In operation, QTAM steps through the pointers one by one. For each pointer, QTAM finds the polling address in the indicated terminal table entry and sends that address on the line. As each terminal recognizes its unique address, it either sends a message if one is ready for transmission, or it sends a negative response if no message is ready.

After a message is received from a terminal, the same terminal is again polled and it again sends a message if one is ready. This process is repeated until the terminal has no more messages to send or until the user-established polling limit for that terminal is reached, whichever occurs first. (The POLLIMIT macro instruction is used to set the limit.)

Each time a negative response is received by the computer or the polling limit is reached, QTAM repeats the polling process using the next pointer in the list. This operation is repeated until the terminal represented by the last pointer in the polling list has sent a negative response or has sent its last message. When this occurs, the polling process is repeated, starting at the beginning of the list. If the user has specified a polling interval in the DTFQT macro instruction for the line group and if no message traffic was encountered during the polling pass just completed, the next pass through the polling list is deferred for the time specified; otherwise, polling proceeds continuously.

A polling list must be specified for each nonswitched line in the system. In defining a list for a nonswitched line,

theuser may enter terminal names as many times as he wishes, and in any order. A list can include terminals on one line only. If a line is used for output only, the user must specify a polling list with no terminal entries.

If there are only a few terminals on a line, QTAM performance may be improved by specifying the terminal names more than once in the polling list. This improvement is due to the CPU time involved in cleaning up after a negative response to polling and in preparing to poll the line again, relative to the time spent in polling the line. There should be about ten entries in the polling list for optimum performance.

Another advantage gained by specifying the terminal names more than once in the polling list is that some of these entries may subsequently be treated as dummy entries for the purpose of expanding the polling list. Refer to the discussion of the CHNGP macro instruction for a discussion of this procedure.

The polling process has a different meaning for switched lines. For non-switched lines, the computer always initiates contact with the terminals. However, for switched lines, the terminal normally initiates the contact. The polling function, in this case, consists only of sending the polling address to the terminal that initiates the contact. The terminal responds by sending one or more messages. The polling address is sent by the computer after each message is received. The polling list for a switched line does not contain pointers to terminal table entries. Rather, it contains a single polling address (except for TWX terminals), in addition to control information. When a terminal dials the telephone number associated with the line represented by this polling list, QTAM sends the polling address on the line.

In the case of the TWX terminals, the polling function consists of sending on the line a character sequence rather than a polling address. Otherwise, the polling function is identical.

For WTTA lines, the polling list contains only the CPU identification sequence to be sent to the terminal each time an identification exchange is to be performed.

The polling list has a different meaning for a 2260-2848 Local line group. One, and only one, polling list must be specified for the line group. This polling list must contain the names of all IBM 2260 Display Stations in the line group from which messages are to be received. A terminal must not appear in the list more than once. Messages are not accepted from 2260s not defined in the list. If the operator of an undefined terminal attempts to enter a message by pressing the ENTER key, the subsequent Attention interrupt is ignored.

QTAM uses the polling list at open time, when the CCB for each 2260 Local terminal in the list is queued on the DOS channel scheduler queue in anticipation of receiving a read request (Attention interrupt) from the device. Thereafter, the CCB is always maintained on the channel scheduler queue except when QTAM is sending a message to the terminal.

The POLL macro instruction is used to define polling lists for both switched and nonswitched lines. QTAM also provides routines for examining and modifying polling lists. The macro instructions used for implementing these routines are described in the section, Examining and Modifying the Telecommunications System. The structure of polling lists is shown in Appendix A.

Polling List Definition (POLL) Macro Instruction

POLL generates a polling list for a specific line attached to the telecommunications control unit (TCU), or for all 2260 Local terminal from which messages may be received in a line group. For a non-switched line, it defines the order in which terminals on the line are to be polled. For a switched line, it specifies the polling address or identification sequence to be sent to any terminal that calls the computer on the line represented by the list. For a WTTA line, this operand specifies the CPU identification sequence to be sent during an identification exchange. For a 2260-2848 Local line group, it defines all 2260s from which messages may be received. One POLL macro instruction must be included for each switched and nonswitched line in the system.

Name	Operation	Operand
pollname	POLL	[(entry,...)] [,AUTOPOL={1 | 2}] polladdr nid 2740

pollname
Is the name of the polling list for the line. The name must be specified and must be identical to a name

specified in the sublist of the CPOLL keyword operand in the DTFQT macro instruction for the line group. In addition, the polling list defined must be the polling list for the line indicated by the relative position of the name in the CPOLL sublist.

entry,...
Are the names of terminals on a nonswitched line in the order in which the terminals are to be polled. All the terminals specified must be on the same line. For a 2260-2848 Local, the sublist must contain the names of all 2260s in the line group from which messages may be received. Each name specified must be the name of a TERM macro instruction defining a single terminal entry. If the line (or 2260-2848 local line group) is used for output only, or if the line is a WTTA line, the entry operands mustbe omitted. This operand is to be specified only for nonswitched lines or for 2260-2848 Local line groups.

AUTOPOL
Must be specified if the line is an autopolled line. AUTOPOL=1 specifies that the terminals are IBM 1030. AUTOPOL=2 specifies that the terminals are IBM 1050, or IBM 1060, or IBM 2740 (type 274C or 274D). This operand must be specified for nonswitched autopolled lines only.

polladdr
Is the polling address to be sent to any switched IBM 1050 terminal that dials the computer on the line represented by this polling list. All such terminals that can dial the computer on this line must recognize the same polling address. This operand must be specified in the hexadecimal representation of the device code appropriate to the type of terminal on this line. This operand is to be specified only for switched lines on which the computer can poll the terminal.

nid
Describes the identification sequence of the computer to be sent to any TWX or WTTA terminal on the line represented by this polling list. Consists of the number of characters in the identification sequence, followed by the characters themselves. Both must be written as one continuous character string in hexadecimal notation; that is, the number of characters in hexadeximal, followed by the characters themselves in the hexadecimal representation of the 8-level TWX code or the 5-level code used by WTTA terminals. This operand is to be specified only for switched lines on which the TWX terminal can call the computer or for leased WTTA lines.

For WTTA lines, nid is the number of characters of the CPU identification sequence, followed by the characters themselves. Both must be written as one continuous character string in hexadecimal notation; that is, the number of characters in hexadecimal, and the characters themselves in hexadecimal representation of the 5-level code used by the WTTA terminal.

2740
Must be specified for IBM 2740 switched terminals (Types 274B, 274E, 274G, and 274H) to be used for both input and output. For output only on these terminals, the operand must be omitted.

Example: The following POLL macro instructions create the required polling lists for two nonswitched input lines, one nonswitched output line, one switched line on which IBM 1050s can dial the computer, one switched line on which TWX terminals can dial the computer, one line on which the computer can dial TWX terminals, one autopolled line, and one nonswitched WTTA line.

```
|   |Name     |Operation|Operand                |
|---|---------|---------|-----------------------|
|1  |POLLINE1 |POLL     |(CHI,BOS)              |
|2  |POLLINE2 |POLL     |(NYC,BOS,NYC)          |
|3  |OUTLINE3 |POLL     |                       |
|4  |POLLINE4 |POLL     |E215                   |
|5  |POLLINE5 |POLL     |0CB150FF72A3EB824B     |
|   |         |         |  D2E15088             |
|6  |POLLINE6 |POLL     |03884DC9               |
|7  |POLLINE7 |POLL     |(PHI,WAS,ATL),         |
|   |         |         |  AUTOPOL=2            |
|8  |POLLINE8 |POLL     |0A020830352D38212D     |
|   |         |         |  0208                 |
```

These macro instructions create polling lists used to:

1. Poll the Chicago and Boston terminals in that order.
2. Poll the New York, Boston, and New York terminals in that order.
3. Represent the output-only line.
4. Poll a switched IBM 1050 whose polling address is A0 (E215 is the IBM 1050 device code representation of A0, in hexadecimal notation).

5. Send the computer's identification sequence, preceded and followed by control characters, to any TWX terminal that calls the computer on this line. The operand for the last POLL macro is the device code representation of 12 CR LF DELETE N E W A R K CR LF X-On, in hexadecimal notation.

6. Send a "turnaround" sequence to any answering TWX which the CPU has dialed to turn on the paper tape transmitter of the TWX. The operand of this POLL macro instruction is the transmission code representation of x-on 2 x-off, in hexadecimal format.

7. Autopoll the Philadelphia, Washington and Atlanta terminals in that order.

8. Send the CPU identification sequence (preceded and followed by control characters) to any WTTA terminal with which an identification exchange is to be performed. The operand for this POLL macro instruction is the device-code representation of:

   10 CR LF 3 6 0 - 5 0 CR LF

   in hexadecimal notation.

BUFFERS

The user must specify the size and number of the main storage areas required by QTAM for input and output buffering. How to define these buffers depends on the type of application.

For a nonaudio application, a buffer is specified by including one, and only one, BUFFER macro instruction in the message control program. These main storage areas collectively form a buffer pool that is allocated to and used dynamically by QTAM to handle the transfer of message segments from and to all communication lines, direct-access queueing devices, and processing queues.

All buffers in the buffer pool are the same length. Since the entire header portion of a message must fit in the buffer that receives the first message segment, the length specified must be equal to or greater than the size of the message header used plus 32 (the size of the header prefix generated and used by QTAM routines).

Buffer request blocks (BRBs) are QTAM control blocks used to dynamically request buffers prior to their actual allocation from the buffer pool. The user should determine the number of BRBs required by the QTAM system.

The number of BRBs required in the system is a function of a number of variable factors. The most important factor is the number of lines that are to be polled at the same time. The user should initially specify a value that represents the maximum number of BRBs that could be in use. If the user has specified a reasonable value in the BUFNO operand of the DTFQT macro instruction for each communication line group, the maximum number of BRBs he may need to specify in the BUFFER macro instruction may be calculated as follows:

> For each line group, multiply the number of buffers specified in the BUFNO operand by the number of lines in the group; add the products obtained for each line group. To this figure, add one buffer for each MS destination queue and each MS process queue defined by message processing programs. The sum is the maximum value desired.

Example: Assume three line groups, GPONE, GPTWO, and GPTHREE, consisting of five, twelve, and eight lines, respectively. The BUFNO operands of GPONE, GPTWO, and GPTHREE specify 3, 3, and 5, respectively. Also assume that there is one message processing program that defines one MS process queue and one MS-destination queue. The maximum number of BRBs to be specified in the BUFFER macro instruction is calculated as:

$$(3x5) + (3x12) + (5x8) + 1 + 1 = 93.$$

The value calculated using the above formula represents the maximum number of BRBs that could be needed at one time. In actual operation, the greatest number required at one time would be somewhat lower, and the user may wish to specify a lesser value. Experience within the operating environment of a particular application can best demonstrate the practicality of specifying a lesser number of buffers.

An exception arises when all the lines consist of dial lines or lines polled with the Auto Poll feature. In these cases the value derived above is the actual value that should be specified. In other cases experience within the operating environment of a particular application can best demonstrate the practicality of specifying a lesser number of BRBs.

The number of buffers specified must be equal to or greater than the number in use at any one time. If the number of buffers

needed to accommodate message traffic at any time exceeds the number of buffers available, loss of message data can occur. Therefore, the user should specify a sufficient number of buffers in the BUFFER macro instruction to prevent this problem from arising under any expected operating conditions.

Because the actual number of buffers that will be in use at any particular moment depends on several variable factors (whose cumulative effect varies more or less unpredictably), the user should initially specify a value that represents the maximum number that could be in use.

This maximum value is related to the number of lines and the number of BRBs by following formula:

Number of buffers = L + f(BRB - L)

where L = number of lines

BRB = number of BRBs

f = a factor whose value is between 0 and 1.

For small systems with few lines, the factor f approaches one. For larger systems a value of 0.5 might be adequate. Experience within the operating environment of a particular application can best demonstrate the appropriate value.

Example: In the previous example there were 25 lines and 93 BRBs. A reasonable number of buffers for most applications would be

number of buffers = 25 + 0.5(93 - 25) = 59.

Management of data buffers is an important factor in running a QTAM system with optimum efficiency. There are four factors to be considered in weighing the trade-off of time and main storage:

1. The user must specify the correct number of buffers to assure no loss of or undue delay of data.
2. The user must select the size of the buffer to accommodate his messages.
3. The user must decide on the number of BRBs needed for a reliable system.
4. The user must decide on the number of BRBs to be assigned each line (set in BUFNO operand of DTFQT macro instruction).

Figure 11 is provided to aid in deciding the effect of these factors. The figure shows the advantages in specifying more or less of the quantity with other considerations equal.

QUANTITY	ADVANTAGES
larger buffers	1. Requires fewer buffers for a message, resulting in less overhead by QTAM to manipulate the buffers. 2. Decreases the probability of losing data, since there is less chance of missing a program controlled interrupt. 3. Makes better use of disk tracks if buffers are filled. 4. Decreases the disk time, since there are fewer disk accesses.
smaller buffers	1. Requires a shorter amount of time to fill up buffers. This results in more dynamic use of main storage, and hence main storage is not tied up unprofitably. 2. Increases likelihood of filling the entire buffer, therefore making better use of main storage. If a message does not fill up the buffer (as in larger buffers), main storage is wasted.
more buffers	1. Decreases the chance of losing data of incoming messages. 2. Assures that outgoing messages are not delayed because they are waiting for a buffer. 3. Allows more CPU time for other tasks.
fewer buffers	1. Uses main storage more efficiently. No more buffers than the amount needed for incoming and outgoing messages are used, thus speeding throughput and saving main storage.
more ERBs per line	1. Reduces the chance of losing data due to a missed program controlled interrupt. 2. Saves main storage if low activity allows fewer buffers than ERBs to be specified.
fewer ERBs per line	1. Saves buffers. Since there are as many buffers used at one time, when transmitting or receiving on a line, as there are ERBs assigned to the line, buffers are not unnecessarily tied up. (Only one buffer per line is assigned during polling.)

Figure 11. Aids in Specifying ERBs and Buffers

BUFFER Macro Instruction

BUFFER specifies the main storage buffer areas required by QTAM. The buffers are allocated to QTAM as a block of main storage called the buffer pool.

Name	Operation	Operand
	BUFFER	nnn,length[,mmm][,brb]

nnn
: Is the number of buffers to be reserved (see the example calculation above).

length
: Is the length, in bytes, of each buffer. All buffers in the buffer pool are the same length. The length specified must equal or exceed the length of the longest message header used in the system plus 32 (the size of the header prefix generated by QTAM). The length of the message segment size used in the system is based on the buffer length. The maximum allowable buffer length is 276 bytes. The value specified (if any) in the optional operand of the LPSTART macro instruction must also be considered in determining the value of the length operand (see the LPSTART macro instruction description).

mmm

Is the number of channel command words QTAM must generate for sending the idle characters specified by the PAUSE macro instructions in the LPS sections of the message control program. The number of CCWs required depends on a number of variables whose cumulative effect changes during system operation. (The principal factors are the number of appearances in messages of each control character that requires idle character insertion, and the number of lines over which outgoing messages are being sent at the moment.) Because determining the actual number of CCWs that could be needed at any given moment is impractical, the user should initially specify a "worst-case" value (that is, a value representing the maximum number of CCWs that could be required under any operating condition). This value may be calculated as follows:

$$mmm = L_1\ (I_1) + L_2\ (I_2) + \ldots\ L_n\ (I_n)$$

where L equals the number of lines in the line group and I equals the expected number of appearances of control characters per outgoing message-filled buffer, for which idle character insertion is required; L(I) is calculated for each of the line groups 1 through n. Insertion of idle characters is not required for terminals in a 2260-2848 Local or Remote line group; therefore, this operand may be omitted if this is the only type of line group defined in a message control program. If this operand is omitted, no CCWs are generated and the PAUSE macro cannot be used.

Example: Assume that the LPS for the first line group includes a PAUSE macro instruction that causes insertion of idle characters each time a NL (new line) control character is encountered in an outgoing message-filled buffer. Also assume that the expected number of appearances of this control character in an outgoing buffer is two. I therefore equals 2. If the line group consists of five lines, $L_1(I_1)$ equals 5(2). If the system includes two other line groups for which L(I), calculated similarly, equals 3(3) and 7(4), then mmm = 5(2) + 3(3) + 7(4) = 47.

In most applications this "worst-case" value will considerably exceed the actual number of CCWs required. Therefore, the user may reduce the value during system testing. If this operand is omitted, zero is assumed.

brb

Is the number of buffer request blocks (BRBs) to be reserved. (See the sample calculation above.) This number must be greater than or equal to the number of buffers specified. If this operand is omitted or the number specified is less than the number of buffers, the number of BRBs is set equal to the number of buffers. If this operand is specified and the mmm-integer is omitted, this operand must be preceded by two commas.

Example: Assume a system in which:

1. 59 buffers of 100 bytes each are required.
2. The number of CCWs required for insertion of idle characters is 47.
3. The number of BRBs required is 93.

The BUFFER macro instruction would then be written:

```
BUFFER 59,100,47,93
```

FILE INITIALIZATION AND ACTIVATION

The file initialization and activation section of the message control program begins with an OPEN macro instruction and ends with the ENDREADY macro instruction. Within the message control program, this section must precede the LPS section. When the instructions in this section have been executed, the system is ready to handle message traffic.

The OPEN macro instruction completes the initialization for and activation of the DASD message queues, communication line group, message log, and checkpoint records files. Opening a communication line group file causes all lines in the line group to be prepared for operation. The lines are activated automatically for message transmission.

The ENDREADY macro instruction must be the last instruction in the file initialization and activation section. When ENDREADY has been executed, the system is ready to handle message traffic. The expansion of this macro instruction causes a branch to the IBM-provided logic that supports the message control program, where procurement of the first message is awaited (for nonaudio lines, the first message procured can be either a message coming in from a terminal, or a message being sent to a terminal by a message processing program. When the first message is procured, control

is returned to the LPS section of the message control program for handling of the message. For this reason, no executable code may be included between ENDREADY and the LPSTART macro instruction that begins the LPS section.

Once the LPS is initially entered via the expansion of the ENDREADY macro instruction, execution in the message control program is restricted to the LPS section; that is, the LPS is continually reentered to handle messages entering and, except for the audio messages, leaving the computer as long as the message control program is active. Deactivation of the message control program and the telecommunications system is accomplished by closing the message control program files. The section, Deactivating the Telecommunications System, contains a discussion of the procedures necessary to terminate the message control program.

The STARTLN, COPYC, CHNGP, COPYT, CHNGT, and COPYQ macro instructions may be used in the initialization and activation section of the message control program. This is useful if the user wishes to modify the status of his system at the time the message control program is initiated. If any of the above macro instructions are used in this section, they must precede the ENDREADY macro instruction. Generally, however, these macro instructions are employed in a message processing program (in the foreground-two or background partition) so that the status of the system can be dynamically examined and modified as needed.

OPEN Macro Instruction

OPEN is used in the message control program to complete the initialization and activation of the DASD message queues, communication line group, message log, and checkpoint records files. All these files can be opened separately or with one OPEN. However, the user must open the DASD message queues file, if any, before any other file used by QTAM. If the Checkpoint Restart feature is used, the DASD checkpoint records file must be opened after the DASD message queues file. The user specifies, as the operands of the OPEN macro instruction, the names of the DTF tables for the files.

If the DTF table for a communications line group is specified, the OPEN routine completes the initialization for all lines in the line group and automatically activates the lines for message transmission. If the TYPEFLE keyword operand of the DTFQT macro instruction for a nonswitched line group specifies INPUT or CMBND (input and output), OPEN performs initialization procedures necessary for polling on those lines in the line group that have an active polling list with terminal entries. Polling on these lines commences after execution of ENDREADY. If a line does not have an active polling list with terminal entries, or if TYPEFLE=OUTPUT is specified, polling is not initiated. If the OPEN is for a switched line group with TYPEFLE=INPUT or CMBND, the OPEN routine issues commands to enable each access line in the line group.

If the communications line group being opened is for a 2260-2848 Local, the Open routine causes the CCB for each 2260 Local from which messages are to be received to be placed on the DOS channel scheduler queue. When this initialization procedure is completed, QTAM is prepared to service read requests (signalled by Attention interrupts) from each 2260 Local defined in the polling list for the line group. Attention interrupts that arrive before Open is completed or from 2260 Locals not defined in the polling list are ignored.

If the Checkpoint Restart feature is used, the DASD checkpoint records file is examined. If the file has been properly closed, initialization is completed for placement of checkpoint records on the DASD. If the file has not been closed, a restart operation is performed. Restart reestablishes the queues and the telecommunications network to the status it had at the time of the most recent checkpoint, except for the audio lines. If the file has not been closed, because of a system failure, and the user does not wish to perform a restart operation, he must reinitialize the checkpoint records file before loading the program.

The formats for the message queues file and for the checkpoint records file are checked when the particular file is opened. If the file is formatted incorrectly, an error message is provided, and the job is cancelled.

If a message log DTF table is specified, initialization is completed for placement of messages on the device to be used for logging.

Name	Operation	Operand
[symbol]	OPEN	filename,...

symbol
Is the name of the macro instruction. The name is optional.

filename
Specifies the name of the DTF table associated with the file being opened. Several files may be opened with one OPEN by entering their file names as operands.

If register notation is used, any general register in the range 2 through 12 may be specified. The programmer must load the address in the register before issuing the OPEN.

Note: It is permissible to intermix register notation and relocatable expressions in the same OPEN. For example:

Name	Operation	Operand
OPENER	OPEN	DISKFILE,(3),(4),LOG

Examples: Shown below are two examples that illustrate the two methods that can be used to open the message control program files. The first example opens each file with a separate OPEN macro instruction; the second opens the files collectively. The DASD message queues file (DISKFILE) is opened first as required in both examples:

Name	Operation	Operand
	OPEN	DISKFILE
	OPEN	INQRCUP1
	OPEN	INQRCUP2
	OPEN	LOGFILE

Name	Operation	Operand
	OPEN	DISKFILE,INQRCUP1,
		INQRCUP2,LOGFILE

ENDREADY Macro Instruction

The file initialization and activation section must be ended by an ENDREADY macro instruction. ENDREADY is essentially a wait-type instruction; the event awaited is the procurement of the first message. Only one ENDREADY macro instruction can be included, and it must be the last in the group of file initialization/activation instructions. Prior to issuing ENDREADY, the user must ensure that register 13 contains the address of an 18-word save area. ENDREADY saves the user's registers in this area by standard register saving conventions. When control returns to the user at the address specified in the EOJAD keyword operand of the DTFQT for the first file opened in the message control program, the registers (except for register 14) are restored. No operand is required.

Name	Operation	Operand
	ENDREADY	

LINE PROCEDURE SPECIFICATION (LPS)

The procedure followed by a message control program in operating upon messages being received from or, except for audio lines, sent to terminals is defined by a line procedure specification (LPS). An LPS is a sequence of IBM-provided statements called LPS macro instructions. The user must provide an LPS for each communication line group in his system. However, one LPS may be used by more than one line group if each of the groups requires identical message control procedures.

The purpose of the LPS is to define macro-introduced routines that:

1. Examine and process control information in the message header.
2. Perform functions necessary to prepare a message segment for processing or transmission.

Preparing an LPS consists primarily of selecting the appropriate IBM-provided macro instructions and writing them in a particular sequence, according to the requirements of the installation and of the line group. The user may also insert his own coding sequences to better adapt QTAM to his requirements. Considerations such as the message header formats, the processing requirements for various types of messages (if messages having different handling requirements are directed to the same LPS), the type of terminal in the line group, and whether the LPS is operating on messages from switched or nonswitched lines must be carefully analyzed.

Two major types of macro instructions are used in the LPS: functional macro instructions and delimiter macro instructions. In general, the functional macro instructions perform the specific operations required on messages directed to the LPS. Delimiter macro instructions for nonaudio lines classify and identify

sequences of functional macro instructions. Delimiter macro instructions also direct control to the appropriate sequence, according to whether the message segment is incoming or outgoing, and whether it is a header segment or a text segment.

COMPONENTS OF THE LPS

The LPS is divided into two major groups of macro instructions: the Receive group, which handles incoming messages; and the Send group, which handles outgoing messages. In the coding of the LPS, the Receive group must precede the Send group. Each of the major groups is further divided into three coding subgroups. The Receive Segment and Send Segment subgroups contain macro instructions concerned with all portions (both header and text) of incoming and outgoing messages, respectively. The Receive Header and Send Header subgroups contain macro instructions concerned only with the headers of incoming and outgoing messages. Macro instructions in the End Receive and End Send subgroups perform error-handling procedures for incoming and outgoing messages.

The Receive Header and Receive Segment subgroups may each be used more than once within the Receive group. Similarly, the Send Header and Send Segment subgroups may be used more than once within the Send group. For example, an application might require that different operations be performed for several different types of messages directed to the LPS. Each of the message types could require a different header format. In such a case, there could be a separate Receive Header subgroup to process the header of each message type. The user can include in his header formats a special message-type character for each type of message. The MSGTYPE functional macro instruction can be used to examine the message-type character and direct control to the appropriate Receive Header subgroup.

The sequence of the Receive Header and Receive Segment subgroups within the Receive group, and the sequence of the Send Header and Send Segment subgroups within the Send group, may depend on which functional macro instructions are specified within the subgroups. For example, assume that the TIMESTMP macro is included in the Receive Header subgroup, and that the TRANS macro is included within the Receive Segment subgroup. The TIMESTMP macro enters the time of day into the message header in EBCDIC form, and the TRANS macro translates all message segments from transmission code into EBCDIC. It is evident that the EBCDIC time information must be inserted <u>after</u> the header has been translated to EBCDIC, not before. The TRANS-introduced translate routine must therefore be executed before TIMESTMP, hence, the Receive Segment subgroup, which contains the TRANS macro, must be executed before the Receive Header subgroup.

The End Receive and End Send subgroups may each be used only once and must be the last sections within the Receive and Send groups, respectively.

If only the IBM-provided macro instructions and associated macro-introduced routines are used in coding an LPS, the Receive Header subgroup is mandatory. The user may omit any other subgroup if it is not required for a particular application. For example, the Receive Segment and Send Segment subgroups may be omitted in a message switching application if all terminals involved operate in the same device code (that is, translation of the message text is not required) and none of the other functions restricted to use in these subgroups is desired. Any or all coding subgroups may be omitted if the user prefers to write his own routines for the functions he requires. An LPS must contain, as a minimum, the LPSTART, POSTRCV, ENDSEND, AND POSTSEND delimiter macro instructions to provide the linkage between the LPS and IBM-provided logic that supports the message control program. An LPS must also contain either a ROUTE or a DIRECT macro instruction. The use of all other functional macro instructions is optional.

<u>Note</u>: Any user code within the LPS should be re-entrant

Figure 12 shows the various coding subgroups that can be included in an LPS, the delimiter macro instructions associated with each subgroup, and the functional macro instructions (in alphabetical order) that can be used in each subgroup.

RECEIVE MACRO INSTRUCTIONS			SEND MACRO INSTRUCTIONS		
Coding Subgroup	Delimiter	Functional	Coding Subgroup	Delimiter	Functional
	IPSTART[1]			SENDSEG	
Receive Segment	RCVSEG	BREAKOFF COUNTER[3] [4] LOGSEG TRANS	Send Segment		COUNTER[4] LOGSEG PAUSE TRANS
Receive Header	RCVHDR	COUNTER[4] DATESTMP[2] DIRECT[4] ECA[2] LOGSEG MODE[2] MSGTYPE[2] OPCTL[2] [4] PCLLIMIT[4] ROUTE[2] SEQIN[2] SKIP[2] SOURCE[2] TIMESTMP[2] TRANS	Send Header	SENDHDR	COUNTER[4] DATESTMP[2] LOGSEG MODE[2] MSGTYPE[2] PAUSE SEQOUT[2] SKIP[2] TIMESTMP[2] TRANS WRU
			End Send	ENDSEND[1]	EOB EOBLC ERRMSG[4] INTERCPT[4] REROUTE[4] WRU
End Receive	ENDRCV	CANCELM ECB ECBLC ERRMSG[4] PCLLIMIT[4] REROUTE[4]		POSTSEND[1]	
	PCSTRCV[1]				

[1]Required delimiter macro instruction.
[2]Functional macro instruction must be in the same relative order as the corresponding message-header field on which it operates.
[3]COUNTER can be used in RCVSEG for nonswitched lines only.
[4]These macro instructions may require optional terminal table subfields.

Figure 12. Nonaudio Line Procedure Specification Macro Instructions

DELIMITER MACRO INSTRUCTIONS

Delimiter macro instructions group the functional macro instructions into the various subgroups. They also perform initialization and control functions within the LPS.

The IPSTART macro instruction identifies the beginning of the LPS and must be the first instruction in every LPS. The code generated by the expansion of IPSTART determines whether the message segment entering the LPS is incoming or outgoing and directs the segment to the Receive group or the Send group accordingly. In an application that directs multisegment messages to the LPS, it is necessary that the functional macro instructions in the header processing subgroups be executed only where the message segment being handled contains the message header. The expansions of the RCVHDR and SENDHDR delimiter macro instructions cause the header processing subgroups to be bypassed when a message segment contains text only.

POSTRCV and POSTSEND identify the ends of the Receive group and the Send group, respectively. These delimiters, along with IPSTART, must appear in every LPS. Each of the remaining delimiters is required only if the user chooses to include in the LPS the coding subgroup associated with that delimiter.

FUNCTIONAL MACRO INSTRUCTIONS

Functional macro instructions perform the specific operations required on message segments. These functions include:

- Message editing (code translation and insertion of time of day, current date, and message sequence numbers in message headers).
- Checking validity of source and destination codes in message headers.
- Routing messages to specified destinations.
- Maintaining logs of messages on an auxiliary storage device.
- Checking for errors in message transmission and taking corrective action.

Functional macro instructions that perform operations related to an entire message segment may appear at any point within the coding subgroup in which they are used. All functional macro instructions in the Receive Segment, Send Segment, End Receive, and End Send subgroups are included in this category.

The majority of the functional macro instructions in the Receive Header and Send Header subgroups perform functions that concern a specific header field. Macro instructions of this type involve either:

1. Use of a QTAM scanning routine to determine the contents of a specific header field (SEQIN and SOURCE, for example); or
2. Insertion of a new field in the message header (TIMESTMP and SEQOUT, for example); or
3. Making of a decision at some point during header processing (MODE and MSGTYPE, for example).

These macro instructions must appear in a specific sequence dependent on the format of the message headers.

In planning a format for message headers, the user may arrange the various header fields in any desired order. Macro instructions involving scanning, insertion of a field, or making a decision must be in the same relative order as the corresponding message-header fields on which they operate. Figure 13 indicates the functional macro instructions that must be sequenced in this manner.

Note: The entire header of each message must be contained within the buffer that receives the first segment of the message (see the BUFFER macro instruction description). The message header must not exceed 244 bytes.

Some functional macro instructions that use the scanning routine provide the option of specifying the length of the header field to be scanned (ROUTE and SOURCE, for example). If the user does not specify the length, the field is assumed to be of variable length and must end with a blank character. No blank character may appear within the field because it will be mistaken for the end-of-field delimiter. If the field length is specified, the field to be scanned need not end with a blank character, and may contain embedded blanks.

ERROR-HANDLING FUNCTIONAL MACRO INSTRUCTIONS

Four functional macro instructions (CANCELM, ERRMSG, INTERCPT, and REROUTE), called error-handling macro instructions, permit the user to test for any condition for which he wishes appropriate action to be taken. These macro instructions are used in conjunction with the error half-word for the communication line involved and/or the information obtained from the device in error by a SENSE command. The error half-word consists of sixteen bits, each of which (except unused bits) indicates the presence (when 1) or absence (when 0) of a specific error or condition that has affected or may affect successful transmission of a message. The meaning of each of the bits is explained in Figure 15, and the meaning of the bits in the sense information is explained in the pertinent telecommunications control unit publication indicated in the Preface.

The user specifies in each error handling macro instruction used a half-word bit configuration called a mask. At the completion of transmission of each message (or each block of a message), the mask is compared to the error half-word. If a 1 is detected in any bit position of both the mask and the error half-word, the function

specified by the macro instruction is performed. A zero is specified in a mask bit position when the error condition represented by the corresponding position in the error half-word is to be ignored. The user may cause the function specified by the macro instruction to be performed unconditionally (that is, for all messages or message blocks) by specifying a mask consisting entirely of zeros.

The user must carefully analyze the requirements of his application to determine which errors or conditions must be detected and which can reasonably be ignored without degrading the performance of his system. The four error-handling macro instructions provide varying methods by which corrective or control functions can be initiated when an error has been detected. The ERRMSG macro instruction is used to send an appropriate message to a designated destination when any error specified by the mask has occurred. For example, if an invalid destination code is detected during receipt of a message, the ERRMSG macro instruction can be used to send a message to the originating terminal stating the nature of the error and requesting that the message be corrected and sent again. The INTERCPT macro instruction suppresses the sending of messages to a terminal when any error specified by the mask has been detected; it is normally used to withhold transmission to a terminal that has become inoperative. The section, FUNCTIONAL MACRO INSTRUCTION DESCRIPTIONS (NONAUDIO LINES), contains detailed discussions of these and the other error-handling macro instructions.

The user may in the LPS access directly the error half-word and the sense information which are contained in the LCB by using the name IJILCLEHW for the error half-word and the name IJILCLSEN for the sense information. Prior to passing control to the LPS, QTAM places the address of the LCB associated with the line involved into register 4. The expansion of the LPSTART macro instruction establishes this register as the base for the LCB DSECT generated by the expansion of the TERMTBL macro instruction.

When entering a message from an IBM 1050 terminal, a terminal operator has the capability of cancelling the message immediately by pressing the alternate and cancel keys and then entering ECB. The terminal is repolled at once so that the message may be reentered if desired.

THE SCAN POINTER

In QTAM, general register 5 is used as the scan pointer register, maintaining a pointer to the current field in the message header. From the user's standpoint, this pointer is the key to QTAM. Through the use of QTAM macro instructions, the user manipulates this pointer, examines fields in the header, and makes decisions based on the contents of these fields. In designing a QTAM message control program, the user must be constantly aware of the header field about to be processed.

QTAM macro instructions perform many varying functions from verifying sequence information to placing messages on destination queues. The user can design a simple message switching application using QTAM macro instructions only, and no user code. More sophisticated applications may require that the user use the scan pointer in his routines.

Basically, two types of LPS macro instructions cause the scan pointer to be moved (Figure 13).

1. Certain macro instructions move the scan pointer along until a user-specified character sequence is found (e.g. SKIP X'15'). After these macro instructions are completed, the scan pointer is positioned to the last character in the sequence.

2. Other macro instructions move the scan pointer a certain number of characters. The number of characters is determined in three ways:

   a. Certain macro instructions have a fixed count of characters (DATESTMP) or an assumed count to be used if no other count is supplied (TIMESTMP). When this type of macro instruction is completed, the scan pointer points to the last character to satisfy the count. Any blank characters encountered are skipped.

   b. With certain macro instructions, the user may specify a number of nonblank characters to be considered as the next field (ROUTE 3). When these macro instructions are completed, the scan pointer is positioned to the last character which satisfies the count. The user may send in RA L, and the field is still considered RAL. The scan pointer points to the L.

c. With certain macro instructions, the field may be variable in length (SOURCE). In this situation, the field length is not specified by the user. The scan pointer is moved forward past any blanks which might precede the field. The field is then scanned for a blank delimiter. When these macro instructions are completed, the scan pointer points to the blank delimiter which follows the field.

When a message is first received for processing by the receive portion of the LPS, the space reserved by the LPSTART macro instruction for expansion has been filled with idle characters (X'17'). The scan pointer is positioned to the last of these idle characters. If no idle characters were specified in the LPSTART macro instruction, the scan pointer points to the last byte of the header prefix.

After the receive section of the LPS is completed, the position of the scan pointer is saved in the ESPI field of the header prefix, and the message is placed on the queue for its destination. When the message comes off the destination queue to go through the send portion of the LPS, the scan pointer is restored to its former position, pointing to the last character of the last field processed. Additional status information may be inserted into the header before the message is finally transmitted.

A message processing program may generate a response message containing idle characters before the header fields. When this message is retrieved from the destination queue to be transmitted, the scan pointer points to the last of these idle characters. If no idle characters are in the message, the scan pointer points to the last character in the header prefix. Macro instructions in the SENDHDR section of the LPS bypass these idle characters in scanning for the beginning of the header field.

Macro instructions in the LPS are placed in the same order as the fields of the header on which they act. The scan pointer controls access to these fields, normally progressing across the header from left to right as the various macro instructions are executed. The user may use the scan pointer in his own routines to perform header analysis not provided by QTAM. However, he must take the responsibility of positioning the scan pointer to its proper position before executing the next QTAM macro instruction. This may be effected by such instructions as LA 5,4(5), which moves the scan pointer four bytes; or, in the case of fixed format headers, the scan pointer may be repositioned relative to the beginning of the buffer by using the address in general register 6. Another method would be to scan the header for a specific character or character sequence (SKIP macro instruction) and reposition the scan pointer there, although this procedure may be time consuming because of the comparisons needed to find the desired position in the header.

Note: When a message is PUT from a message processing program to a message control program, the scan pointer is positioned to the last idle character in the header. This differs from the position of the scan pointer for other messages being handled in the Send portion of the LPS. Therefore it is recommended that the scan pointer be positioned to the last position of the header of all messages being handled in the Send portion of the LPS.

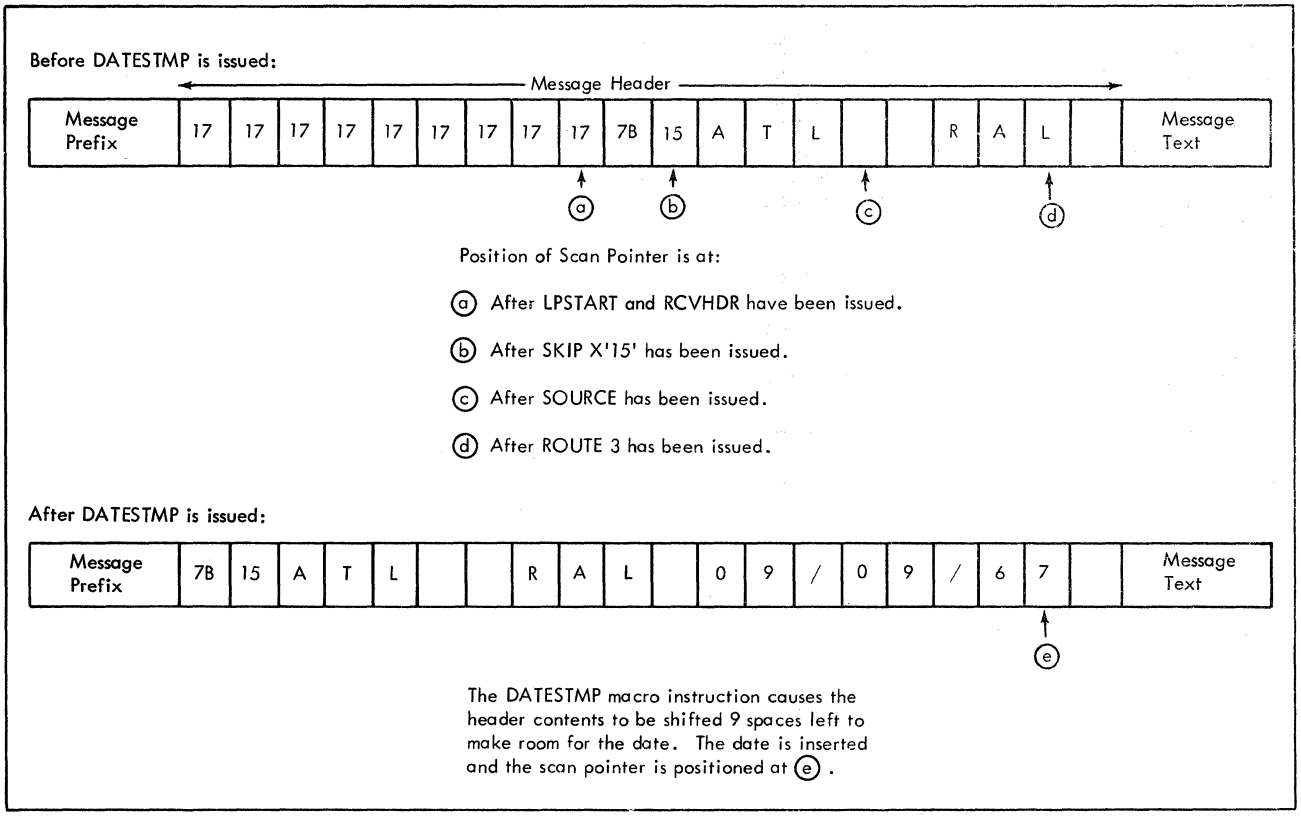


Figure 13. Scan Pointer Positions

ARRANGEMENT OF LPS MACRO INSTRUCTION DESCRIPTIONS

There are two major types of LPS macro instructions:

1. Delimiter macro instructions
2. Functional macro instructions

Since the decision as to what macro instructions should be included in constructing an LPS and how they should be sequenced depends greatly on the particular application, no attempt is made to discuss the macro instructions in any logical order. The macro instruction descriptions are arranged alphabetically by major type for easy access.

Note: The user is cautioned against transferring control between macro instructions within the LPS. A user-written branch to a macro instruction may require that the user also perform functions (such as register saving and restoring) normally provided by the IBM supplied coding. Since user-written branches are the exception rather than the rule, the name fields in the macro instruction formats for the LPS macro instructions (with the exception of LPSTART) have been omitted.

DELIMITER MACRO INSTRUCTION DESCRIPTIONS

End Receive (ENDRCV) Macro Instruction

The ENDRCV macro instruction identifies the beginning of the End Receive coding subgroup of the LPS. The functions specified in this subgroup are performed after an entire message has been received

by the computer. If EOB or ECBIC is specified, the functional macros in this subgroup that precede ECB or ECBIC are performed after each message block has been received. Functional macros following ECB or ECBIC are performed only after the entire message has been received (see the descriptions of the ECB and ECBIC macro instructions).

If the End Receive subgroup is used, it must begin with the ENDRCV macro instruction. It must be the last subgroup in the Receive group and can be used only once in an LPS. No operand is required.

Operation	Operand
ENDRCV	

End Send (ENDSEND) Macro Instruction

The ENDSEND macro instruction identifies the beginning of the End Send subgroup of the LPS. The functional macro instructions included in this subgroup are executed after an entire message has been sent by the computer, or after a message block has been sent if ECB or ECBIC is included as the last macro instruction in the subgroup (see the descriptions of the ECB and ECBIC macro instructions). Functional macros following ECB or ECBIC are performed only after the entire message has been sent.

If the End Send subgroup is included, it must begin with the ENDSEND macro instruction. It must be the last subgroup in the Send group and can be used only once in the LPS. No operand is required.

Operation	Operand
ENDSEND	

Line Procedure Specification Start (LPSTART) Macro Instruction

The LPSTART macro instruction provides an initialization procedure for the LPS LPSTART is required and must be the first macro instruction in every LPS.

The code generated by the expansion of this macro instruction operand includes a test to determine whether the message segment entering the LPS is incoming or outgoing, and directs the segment to the Receive group or the Send group, accordingly.

Name	Operation	Operand
lpsname	LPSTART	nn,
		TERM=(termcode$_1$,...)
		INTRCPT=fieldname

lpsname
: Is the name of the macro instruction and is required. It must be the same as lpsname specified in the CLPS keyword operand of the DTFQT macro instruction for the line group.

nn
: Is the total number of bytes to be reserved in the message header in the first buffer of each input message for insertion of time-of-day, current-date, and output sequence number information, including blanks (see TIMESTMP, DATESTMP, and SEQOUT macro instruction descriptions). If the messages are to be sent to a Western Union Plan 115A terminal, the user should reserve one more byte to contain an initial EOA character. (Refer to End-of-address in Appendix F for further information.) The number of bytes inserted in the LPS Send group that handles messages PUT by a message processing program is not included in this value. A leading zero is not required. If this operand is omitted, no space is reserved. The number of bytes reserved must be included in the calculation of the buffer size (see BUFFER macro instruction).

termcode$_1$,...
: Is the identifying code for the type(s) of terminals that utilize this LPS. The parentheses are coded when the sublist consists of more than a single entry. For example, TERM=(1050). The following values can be included in the sublist:

1. 1030 = IBM 1030 terminals.
2. 1050 = IBM 1050 terminals.
3. 1060 = IBM 1060 terminals.
4. 2848 = IBM 2848 control units (associated with remote IBM 2260 terminals).
5. 2740 = IBM 2740 terminals.

6. 83B3 = AT&T 83B3 Selective Calling Stations.

7. 115A = Western Union Plan 115A outstations.

8. TWX = common carrier TWX stations.

9. 2260 = IBM 2260 Local terminals

10. WTTA = WTTA terminals

INTRCPT
Is the name of the optional subfield of the terminal table specified in the OPTION macro instruction for use by the INTERCPT macro instruction. Must be specified if the intercept function is used.

Example: Suppose that this LPS applies to IBM 1050 and IBM 2740 terminals. Suppose further that TIMESTMP and SEQUT macro instruction are to be included in this LPS. Allowing 6 spaces for the time-of-day (hours and minutes) and 4 spaces for the output sequence number (numbers up to 999), then the LPSTART macro may be coded thus:

```
LPSTWO LPSTART 10,TERM=(1050,2740)
```

The various LPSTART macro instructions identify the start of the various LPS portions of the message control program. A program handling message for TWX and IBM 1050 and 2740 terminals may be organized as shown in Figure 14.

```
QTAMMCP  START
           • (file definition, control,
           • and initialization macro
           • instructions)
         ENDREADY
LPSTWX   LPSTART  9,TERM=(TWX)
           • (delimiter and functional
           • macro instructions for TWX
           • lines)
         POSTSEND
LPS1050  LPSTART  9,TERM=(1050,2740)
           • (delimiter and functional
           • macro instructions for
           • lines with 1050 and 2740
           • terminals
         POSTSEND
         CLOSE
         EOJ

    Alternatively, the above program
 could also be arranged as follows:

MCP2     START
           • (initialization)
         ENDREADY
           • (control macro instruc-
           • tions)
LPS1050  LPSTART 9,TERM=(1050,2740)
           • (functional and delimiter
           • macros for lines with
           • 1050 and 2740 terminals)
         POSTSEND
LPSTWX   LPSTART 9,TERM=(TWX)
           • (functional and delimiter
           • macro instructions for TWX
           • lines)
         POSTSEND
           • (file definition and
           • control macros
           • DTFQT, BUFFER, etc.)
         CLOSE
         EOJ
```

Figure 14. Use of LPSTART in a Message Control Program

Post Receive (POSTRCV) Macro Instruction

The POSTRCV macro instruction identifies the end of the instruction sequence that processes incoming messages; that is, instructions in the Receive Segment, Receive Header, and End Receive coding subgroups.

One POSTRCV macro instruction is required in each LPS, and it must be the last instruction in the Receive group. No operand is required.

Operation	Operand
POSTRCV	

Post Send (POSTSEND) Macro Instruction

The POSTSEND macro instruction identifies the end of the instruction sequence that processes outgoing messages: that is, instructions in the Send Header, Send Segment, and End Send coding subgroups.

One POSTSEND macro instruction is required in each LPS, and it must be the last instruction in the Send group. No operand is required.

Operation	Operand
PCSTSEND	

Receive Header (RCVHDR) Macro Instruction

The RCVHDR macro instruction identifies the beginning of the Receive Header subgroup, which contains instructions concerned only with the header portions of incoming messages. The instructions generated by the expansion of this macro instruction test whether the message segment being operated on contains the message header or text only. If the segment contains text only, the functional macro instructions in the Receive Header subgroup are bypassed; if the segment contains the message header, the instructions in the Receive Header subgroup are executed.

If the Receive Header subgroup is included in the LPS, it must begin with the RCVHDR macro instruction. No operand is required.

Operation	Operand
RCVHDR	

Receive Segment (RCVSEG) Macro Instruction

The RCVSEG macro instruction identifies the beginning of the Receive Segment subgroup, which contains instructions concerned with both header and text portions of incoming messages.

If the Receive Segment subgroup is included in the LPS, it must begin with the RCVSEG macro instruction. No operand is required.

Operation	Operand
RCVSEG	

Send Header (SENDHDR) Macro Instruction

The SENDHDR macro instruction identifies the beginning of the Send Header subgroup, which contains instructions that process only header portions of outgoing messages. The code generated by the expansion of this macro instruction includes instructions that test whether the message segment being operated on contains the message header or text only. The functional macro instructions in the Send Header subgroup are executed if the segment contains the message header; they are bypassed if the segment contains text only.

For messages being sent to a Western Union Plan 115A terminal, the user should insert an EOA character (X'40" before a TRANS macro instruction or X"04" after a TRANS macro instruction) in the first data byte of the header (the thirty-second byte of the buffer) in the Send Header subgroup.

If the Send Header subgroup is included in the LPS, it must begin with the SENDHDR macro instruction. No operand is required.

Operation	Operand
SENDHDR	

Send Segment (SENDSEG) Macro Instruction

The SENDSEG macro instruction identifies the beginning of the Send Segment subgroup, which contains instructions concerned with both header and text portions of outgoing messages.

If the Send Segment subgroup is included in the LPS, it must begin with the SENDSEG macro instruction. No operand is required.

Operation	Operand
SENDSEG	

FUNCTIONAL MACRO INSTRUCTION DESCRIPTIONS

The following macro instructions may be coded in the LPS section for any nonaudio line. The functional macro instructions for audio lines are described in the appropriate section.

Warning: Individual macro instructions in this section do not include a statement of their non-applicability to

audio lines. Audio users must refer to the section FUNCTIONAL MACRO INSTRUCTION DESCRIPTIONS (AUDIO LINES) for applicable macro instructions--no others may be used.

Most of the macro instructions in this section apply to WTTA lines, unless specifically excepted at the beginning of the description. For macro instructions applicable only to WTTA lines, refer to the section FUNCTIONAL MACRO INSTRUCTION DESCRIPTIONS (WTTA LINES).

Halt Receive (BREAKOFF) Macro Instruction

The BREAKOFF macro instruction is used to specify a maximum length for each incoming message. If the message exceeds the maximum length, reception of the message is terminated and an error flag is set in bit ten of the error half-word for the line. This macro instruction also checks if the input buffer is filled with identical characters. If it is, the same action is taken as described above. (A long sequence of identical characters is usually an indication of terminal malfunction.)

This macro instruction can be used only in an LPS section that handles incoming messages from TWX, 115A, 83B3, or 2260 Remote terminals, since the required BREAK command is not valid for the other terminals supported.

Use of BREAKOFF is optional. If used, it must appear within the Receive Segment coding subgroup.

Operation	Operand
BREAKOFF	nnnnn

nnnnn
Is the maximum number of characters for each message. The maximum value of nnnnn is 32767; leading zeros are not required.

Note: Execution of the BREAK command on the 2260 Remote terminal does not delete the START MI from the screen. Thus the next polling of the terminal will cause re-reading of the same message. If this is not desired, a message should be sent to the terminal to clear the START MI.

Cancel Message (CANCELM) Macro Instruction

CANCELM is an error-handling macro instruction that causes immediate cancellation of a message if any of the errors specified by the mask have been detected. Cancellation means that the message is not sent to the destination(s) specified:

1. In the message header (handled by the ROUTE macro), or
2. By the DIRECT macro.

If CANCELM is used to test for an invalid destination code and the error has occurred, the message is canceled for any destinations that follow the invalid one in the message header as well as for the invalid destination.

If a message is not sent to its intended destination due to cancellation, it is important that some action be taken to notify a terminal operator or to perform some other corrective action. If no action is taken, there is no record of which messages have been lost because of cancellation. The ERRMSG macro instruction can be used to send a message to a terminal notifying its operator of the error; or the REROUTE macro instruction can be used to send the message in error to a selected terminal (see the descriptions of these macros). CANCELM must precede an ERRMSG or REROUTE macro instruction used to test for the same error condition in the End Receive subgroup.

The meaning of the bits in the error halfword tested is shown in Figure 15.

Use of CANCELM is optional. If used, it must appear within the End Receive subgroup of the LPS. Because all error types requiring message cancellation can be specified in the same error mask, only one CANCELM macro instruction is needed in the End Receive subgroup.

The CANCELM macro instruction must not be used in an LPS handling either messages to a PROCESS-EXPEDITE queue or multisegment messages in Initiate mode, since cancelled messages must be recalled from the DASD destination queue.

Operation	Operand
CANCELM	mask

mask
Is the hexadecimal representation of

the bit configuration used to test the error halfword for the communication line involved. The framing X and quotes must be coded.

COUNTER Macro Instruction

The COUNTER macro instruction enables the user to maintain four types of count:

1. Incoming message segments for each originating terminal, or incoming message blocks if ECB characters are received from IBM terminals and the ECB or ECBLC macro is used
2. Incoming complete messages for each originating terminal
3. Outgoing message segments for each destination terminal
4. Outgoing complete messages for each destination terminal or terminal component that has a single-terminal entry in the terminal table

The position of the COUNTER macro instruction within the LPS determines which of the four types of count will be maintained. COUNTER must appear in the Receive Segment subgroup to count incoming message segments or blocks, in the Receive Header subgroup to count incoming messages, in the Send Segment subgroup to count outgoing message segments, and in the Send Header subgroup to count outgoing messages. Any one, or all four counts, can be maintained by including the COUNTER macro instruction in the appropriate subgroups; within each subgroup, it may appear at any point.

For each COUNTER macro instruction issued, the user must define, by means of one OPTION macro instruction, a two-byte field (aligned on a halfword boundary) for each entry in the terminal table for which a count is to be maintained. This provides space for maintaining the messages per terminal or message segments per terminal count. The number of COUNTER macro instructions used in the LPS and the number of OPTION macro instructions for the count fields must each correspond to the number of counts being maintained. See the OPTION macro instruction description.

Use of COUNTER is optional. COUNTER must not be used in the Receive Segment subgroup for switched lines or for lines using Auto Poll. When used in the Receive Header subgroup, it must be preceded by a SOURCE macro instruction.

Note: If COUNTER is used to record incoming messages from a line group to which IBM 2260s are attached, each time a general poll is performed all message segments received from the various 2260s during that polling pass are counted as one message.

Operation	Operand
COUNTER	subfield

subfield
Is the name of a halfword field in the user's area of each single-terminal entry in the terminal table, as defined by an OPTION macro instruction. The field contains a binary count up to a maximum of 32,767. When maximum count has been reached, the count is reset to 1 for the next message or segment counted. The user may access the field at any time to determine and/or reset the count.

Date Stamp (DATESTMP) Macro Instruction

The DATESTMP macro instruction causes insertion of the date in the message header. DATESTMP can be included for incoming messages, outgoing messages, or both. The date is expressed as either bmm/dd/yy or bdd/mm/yy, where b is a blank, dd is the day of the month, mm is the month, and yy is the year. The format to be used is specified when the system is generated.

No operand is necessary in this macro instruction because the date field has a fixed length of nine. When DATESTMP is specified, the user must include the length of the inserted field (nine bytes) in his calculation of the value of the operand of the LPSTART macro instruction (see the LPSTART macro instruction description). The DATESTMP macro instruction causes the contents of the message header to be moved nine characters to the left, thus moving the idle characters out of the header. The date is then inserted in the header (see Figure 13).

Use of DATESTMP is optional. If used, it must appear in the Receive Header or Send Header subgroup. Its position within the subgroup must correspond to the relative position, within the header, of the field into which the date is inserted.

Operation	Operand
DATESTMP	

Note: The date inserted into the header will be only as accurate as the date entered into the system by the DATE job control statement or the operator SET command.

DIRECT Macro Instruction

The DIRECT macro instruction causes a message to be queued for the destination specified by the operand. Any destination for which there is an entry in the terminal table may be specified. DIRECT may be used in place of ROUTE when message headers do not contain destination codes. Either DIRECT or ROUTE must be specified to handle message routing; both cannot be used. Only one DIRECT macro may be used for each LPS or for each message type used within one LPS.

DIRECT may be used only within the Receive Header subgroup. If DIRECT is used, EOA may not also be used.

Note: If the TERM macro instruction specifies that the IBM 2260-2848 complex (Remote) is to be polled using the general poll feature, the DIRECT macro instruction must be used to send incoming messages to a message processing program. The processing program must then analyze the message, which consists of segments from different 2260s, and place each segment (via a PUT macro instruction) on the proper DASD destination queue or DASD process queue.

Operation	Operand
DIRECT	{=Cln'dest' {subfield }

dest
: Is the destination code, which may be the name of any entry in the terminal table. n must be equal to or greater than the longest such name appearing in the terminal table; or n may be 8 (the maximum allowable length).

subfield
: Is the name of an optional subfield in the terminal table entry for the originating terminal; this subfield contains the name of the terminal to which the message is to be sent. The name of the subfield specified by this operand must be the same as the name assigned to the subfield by an OPTION macro instruction. The contents of the subfield are specified by the TERM macro instruction that defines the terminal table entry for the originating terminal (see the OPTION and TERM macro instruction descriptions). If the originating terminal is on a switched line or on a line using Auto Poll, and the user wishes to use this operand, DIRECT must be preceded by the SOURCE macro instruction.

End-of-Address (EOA) Macro Instruction

The EOA macro instruction is required if the user wishes to provide multiple routing of incoming messages. The instructions generated by this macro instruction determine the end of the list of destination codes in the message header. The character sequence specified by the EOA macro instruction must appear in the header of each message after the last destination code, regardless of the number of destination codes in the header.

When used, this macro instruction must immediately follow the ROUTE macro instruction. EOA is not used if DIRECT is specified.

The EOA macro instruction must not be used in an LPS handling messages to a PROCESS-EXPEDITE queue.

Operation	Operand
EOA	eoa

eoa
: Is the EOA character or sequence of characters that must appear in the message header after the last destination code. If the destination codes all have the same length, and the optional operand in the ROUTE macro instruction is specified, no blank is required between the last code and the EOA character sequence. Otherwise, a blank must separate the two. One to eight nonblank characters may be specified as the EOA sequence. The EOA character(s) may be specified either as the character itself, or as the hexadecimal equivalent of the character(s). The framing C or X and quotes must be coded.

Examples: In an ECA macro instruction that specifies a # to be used as an ECA character, the # may be written:

1. As the character itself, or

2. In hexadecimal representation of the EBCDIC equivalent of that character:

Operation	Operand	
ECA	C'#'	(1)
ECA	X'7B'	(2)

End-of-Block (EOB) Macro Instruction

This macro instruction is not applicable to WTTA lines or to the IBM 2260-2848 Local. A longitudinal redundancy check (LRC) is performed each time an end-of-block (EOB) or end-of-text (ETX) character is encountered in message text. The check is made by the data adapter at the central processing location, for incoming messages, and by the terminal control unit, for outgoing messages. The EOB causes a positive response to be sent to the source of the message, if the data was received correctly.

For Incoming Messages: The EOB macro causes a positive response to be sent to the terminal if the message data was correctly received; this permits the terminal to send another message block. If the data was incorrectly received, no response is sent; the computer immediately ends message transmission by sending an end-of-transmission character. The terminal must resend the message block when contact with the computer is reestablished (by polling or dialing).

For Outgoing Messages: The EOB macro causes an EOB (or ETX), followed by an LRC character, to be sent to the terminal when an EOB (or ETX) character is encountered in message text. If the terminal receives the message data correctly, it returns a positive response; upon recognizing this response, the computer sends the next message block. If the terminal receives the data in error, it returns a negative response; upon receiving this response, the computer ends the message transmission by sending an end-of-transmission character.

The EOB macro must normally be specified, in both the End Receive and End Send subgroups of each LPS that handles messages to and from an IBM 1030, 1050, 1060, 2260 Remote, and 2740, Types 274D, 274E, 274F, and 274G. The EOB macro must be specified for a 2260-2848 Remote for which general polls are to be performed. It may be omitted only if: (1) all messages are one block long, and (2) possible errors are to be ignored (both conditions are required). If the EOB macro is not specified, the first EOB (or ETX) character encountered in incoming or outgoing message text is treated as an end-of-transmission (EOT) character, precluding transmission of any subsequent blocks of that message.

Either the EOB or the EOBLC macro instruction must be specified in both the End Receive and the End Send subgroups of the LPS that handles messages for the IBM 2740 Model 2 terminals.

This macro instruction is used only for the terminal types cited. In the case of the IBM 1050, 2260 Remote, 2740, Types 274D, 274E, 274F, and 274G, and 2740 Model 2, the EOBLC macro instruction (discussed in the following section) may be specified instead of the EOB macro.

Upon the occurrence of an error or condition for which the user has specified a CANCELM, ERRMSG, or REROUTE macro instruction, message transmission ends when the next EOB (or ETX) character is detected, if the error-handling macro instruction precedes the EOB macro.

Note: Error-handling macro instructions should not precede the EOB macro instruction because, if the specified error condition occurs, the EOB macro instruction will not be executed. The EOB character will be treated as an EOT, and the source terminal will not receive a response to the EOB-LRC sequence.

Within the coding subgroup in which the EOB macro appears, all functional macro instructions that precede the EOB macro are executed for all message blocks; all functional macros that follow the EOB macro are executed only at the end of the message (an EOB is treated as an end of message if a transmission error occurred during transmission of the block). The EOB macro instruction must not be used in an LPS handling messages to a PROCESS-EXPEDITE queue.

Operation	Operand
EOB	

End-of-Block and Line Correction (EOBLC) Macro Instruction

This macro instruction is not applicable to WTTA lines. ECBLC is an optional macro instruction used only for an IBM 1050, an IBM 2260-2848 Remote equipped with the checking feature, and IBM 2740, Types 274D, 274E, 274F, and 274G, and an IBM 2740 Model 2. It performs the same function and is used in the same manner as the EOB macro, but in addition returns a negative response to the message source if the data was incorrectly received, permitting the source to resend the erroneous message block.

Either the ECB or the ECBLC macro instruction must be specified in both the End Receive and the End Send subgroups of the LPS that handles messages for the IBM 2740 Model 2 terminals.

For a 2260-2848 local line group, this macro instruction is used only in the Send group of the LPS. If a data parity error is detected while sending a message to a 2260 (or 1053 Printer), ECBLC causes the message to be resent in an attempt to correct the error condition.

For Incoming Messages:

1. For an IBM 1050 not equipped with the line correction feature, resending is accomplished by rekeying the message block in error, or by repositioning the paper tape or card containing the erroneous block.

2. For an IBM 2260 Remote, the terminal automatically resends the message block.

3. For an IBM 2740, resending is accomplished by rekeying the message block in error.

4. For an IBM 1050 equipped with the line correction feature:

   a. If the erroneous message block originated from the paper tape reader or card reader, the device automatically repositions the tape or card and resends the block.

   b. If the erroneous message block originated from the keyboard, the operator must re-enter the message block in error.

For Outgoing Messages: For either of the above terminal types, the computer automatically resends the erroneous message block.

If ECBLC is specified, any message block whose transmission resulted in an error is retransmitted a maximum of two times. If the error persists after the second retry, an error flag is set in the error halfword for the line (see Figure 16).

The ECBLC macro instruction must not be used in an LPS handling either messages to a PROCESS-EXPEDITE queue or multisegment messages in Initiate mode.

Operation	Operand
ECBLC	

Error Message (ERRMSG) Macro Instruction

The ERRMSG macro instruction causes a user-written error message to be sent to a designated terminal when one of the errors specified by the error mask has occurred.

By means of the ERRMSG macro, the user specifies:

1. The bit configuration of the mask used to test the error halfword. The meaning of the bits in the error halfword tested is shown in Figure 15.

2. The destination to which the error message is to be sent.

3. The text that is to comprise the error message.

The error message includes the text written by the user and, optionally, the header of the message in error. The user specifies that the header is to precede the text by writing a period as the first character of the text. The length of the complete error message cannot exceed one segment (that is, one buffer).

If the ERRMSG macro does not specify that the message header is to be included with the error text, no LPS macros that refer to fields in the header may be used in the Send Header subgroup that is to process the error message, without some modification by the user.

It is assumed that for an ERRMSG macro that does not specify inclusion of the header of the message in error, the user will place the machine EOA character or sequence in the first character position(s) of the error text. (However, inclusion of the machine EOA character is not necessary for error messages to be transmitted to an IBM 1050, IBM 2740, IBM 2260 Remote, or IBM 2260 Local.) This is required by the terminal to receive the error message regardless of which LPS

macros are to process the error message. The scan pointer register, register 5, will be pointing to the first character ofthe error message; this character will be the ECA character (or the first character of the ECA sequence). If the user chooses to have a DATESTMP, TIMESTMP, or SEQCUT macro operate on an error message that does not contain the header of the erroneous message, he must reset the scan pointer register to point to the first character following the machine ECA. This may be done by incrementing the register by the number of characters comprising the ECA sequence.

Unless the MSGTYPE macro instruction is used to distinguish between different message types, the format of the header for an error message must be identical to the header format used for other outgoing messages. If the MSGTYPE macro instruction is used for this purpose, the formats of the respective message headers for the two types may differ after the message-type character. In either case, the correct machine ECA character for the destination terminal must be included.

If the incoming sequence number is invalid, and an error message is to be sent, ERRMSG will scan the error message. If the special character $ is encountered, the correct input sequence number is moved into the four bytes following the $, and the $ is overlaid with a blank. If a second $ is found before the end of the error message, the invalid sequence number is moved into the four bytes following the $, and this second $ is also overlaid with a blank. If this function is not desired, the special character $ should not appear in the error message text.

If the message is to be cancelled, the CANCEIM macro instruction must be used before the ERRMSG macro instruction.

This macro instruction, if used, must appear within the End Receive and/or End Send subgroup of the IPS; it can appear more than once in either subgroup.

Restriction: ERRMSG must not be used to send an error message on a message sent to a message processing program identified by a PROCESS macro instruction with the EXPEDITE operand.

Note: Messages generated by the ERRMSG macro may not be retrieved.

Operation	Operand
ERRMSG	mask, {=CLn'dest' / subfield / SOURCE}, {=C'message' / msgchar}

mask
: Is the hexadecimal representation of the bit configuration used to test the error halfword.

dest
: Is the destination code for the terminal to which the error message is sent; it may be the name of any entry in the terminal table except a distribution list entry. n must be equal to or greater than the longest such name appearing in the terminal table. The maximum value for n is 8.

subfield
: Is the name of a terminal table optional subfield that is associated with the name of the terminal from which the message in error originated. The error message is sent to the destination whose name appears in the optional subfield.

SOURCE
: Specifies that the error message is to be sent to the terminal from which the message in error originated. SOURCE may not be used if this ERRMSG macro is used for an illegal source code error (that is, if the mask contains a 1 in bit 6) originating from a switched terminal. SOURCE must not be used for lines using Auto Poll if the Receive Header subgroup does not contain the SOURCE macro instruction.

message
: Is the actual text of the error message.

msgchar
: Is the address of the first character of the error message text. This address must not be defined by an EQU statement.

Example: Shown below is an ERRMSG macro instruction used within the End Receive subgroup of an LPS to test for invalid destination codes or erroneous sequence numbers. The first operand is the hexadecimal representation of the bit configuration (1011000000000000) of the

mask that tests bits 0, 2, and 3 of the error halfword. The second operand indicates that the error message is to be sent to the terminal from which the message in error originated. The third operand is the address of the first character of the error message text.

```
┌---------┬-------------------------------┐
|Operation|Operand                        |
├---------┼-------------------------------┤
|ERRMSG   |X'E000',SOURCE,ERMSG023        |
└---------┴-------------------------------┘
```

Example: Shown below is an ERRMSG macro instruction used within the End Send subgroup of an IPS to test for transmission errors in outgoing messages to an IBM 2740 with station control. The first operand is the hexadecimal representation of the bit configuration (0000000010000000) of the mask that tests bit 8 of the error halfword. The second operand is the name of the terminal to which the error message is to be sent (all error messages are sent to the same terminal regardless of which destination terminals were to have received the message in error). The third operand is the text of the error message. The period as the first character causes the header of the message in error to precede the error text. If the error message is sent to an IBM terminal, it should, as any message should, end with an end-of-block character.

```
┌---------┬-------------------------------┐
|Operation|Operand                        |
├---------┼-------------------------------┤
|ERRMSG   |X'0080',=CL8'NYCSUPVR',        |
|         |=C'.TRANSM ERROR'              |
└---------┴-------------------------------┘
```

Intercept (INTERCPT) Macro Instruction

The INTERCPT macro instruction is used to permit messages that could not be transmitted to a terminal to be sent at a later time. It causes the suppression of all message transmission to a terminal when any of the errors specified by the mask has been detected. The untransmitted messages remain on the DASD destination queue for that terminal. If the INTERCPT macro instruction is to be used, the user must specify a 3-byte subfield in the optional user area of the terminal table; the name of this subfield must be specified by the INTRCPT operand of the LPSTART macro instruction (see the OPTION and LPSTART macro instruction descriptions). For each terminal for which message transmission is suppressed:

1. The relative disk address of the first intercepted message header is placed in the subfield reserved in the entry representing that terminal (allows rerouting and later transmissions).

2. The intercept bit in the IJLQTSTA byte of that entry is set to 1.

3. The send bit in the IJLQTSTA byte for that entry is set to 0.

No further messages are sent to the affected terminal until the user resets the intercept and send bits. This can be done by a message processing program using the RELEASEM or CHNGT macro instruction. If RELEASEM is used, all suppressed messages (those on the destination queue) and any new messages are sent. If CHNGT is used, only the new messages (those placed on the destination queue after CHNGT has been issued) are sent. In the latter case, the suppressed messages remain on the destination queue, and cannot be sent unless the user obtains them by a RETRIEVE macro instruction and reissues a PUT for each of them.

The meanings of the bits in the error halfword tested are shown in Figure 15.

Bit	Function and Explanation
0	Invalid destination code. The RCUTE macro instruction found a destination code in the message header for which there is no corresponding destination name in the terminal table. The message for the invalid destination is placed on the dead-letter queue. If a CANCELM macro instruction is given for this error condition, the message is canceled for any destinations whose codes follow the invalid one in the message header as well as for the invalid destination.
1	Terminal inoperative. The message was not sent to its destination because the send bit (bit 6 of the IJLQTSTA field) in the terminal table entry for that destination is off (i.e., a zero bit).
2	Sequence number high. The SEQIN routine found a message sequence number higher than the expected number for the next message originating from that terminal. When this error is detected, the expected sequence number is not changed.
3	Sequence number low. The SEQIN routine found a message sequence number lower than the expected number for the next message originating from that terminal. If the message is not canceled by the user, the same sequence number may appear in more than one message. When this error is detected, the expected sequence number is not changed.
4	Incorrect-length message. 1. An incorrect-length message has been received from an IBM 2260 Local terminal. This occurs when the terminal operator enters a message without having pressed the START key, or presses the ENTER key immediately after pressing the START key; that is, no text is entered. 2. It also indicates a buffer overflow condition detected on the 2740 Model 2 or 2848.
5	Incomplete header. The incoming message header did not terminate within the first message segment (or prior to the first end-of-block character).
6	Invalid source code. The SOURCE routine found that the source field in the incoming message header contained a code that: 1. did not correspond to the name of the terminal that was connected to the computer over a nonswitched line, or 2. did not correspond to any terminal name in the terminal table (applicable only to switched terminals).
7	Should-not-occur error. Error Recovery Procedures have found an error which should not occur.
8	Transmission error. An error in transmission (unit check or channel error [CSW bits 43-47]) occurred during the receiving or sending of a message (see also bit 12). In case of unit check, the field in the LCB labeled IJLQLSEN contains the sense information received from the device in error.
9	Timeout exceeded. The maximum allowable time interval between reception of successive characters of a message, or between polling/addressing of a terminal or component and receipt of a response from the terminal has been exceeded, indicating possible terminal or line failure (see also bit 12). Set from sense bits = timeout and/or intervention required.
10	Breakoff error. The BREAKOFF routine found an incoming message whose length exceeded the maximum allowable length, or one in which all of the characters in one of the buffers containing the message were identical (indicating line trouble).

Figure 15. Communication Line Error Halfword (Part 1 of 2)

11	Insufficient buffers. The QTAM buffer assignment routine was unable to provide buffers for an incoming message. This condition, when it occurs infrequently, may be corrected by requesting the originating terminal to resend the message. Frequent occurrences of this condition require that QTAM be redefined with a larger number of buffers.
12	Message not sent. The message was not sent because of a unit check or channel status error, a timeout, or a negative response. The specific reason is indicated by the presence of a 1 in bit 8 or 9 in combination with the 1 in bit 12: 1. A 1 in both the 8 bit and 12 bit indicates that a transmission error prevented message transfer. 2. A 1 in both the 9 bit and 12 bit indicates that the maximum allowable time interval between polling or addressing of a terminal and receipt of a response from that terminal has been exceeded. This indicates possible terminal or line failure. 3. A 1 in only the 12 bit indicates that a negative response to addressing has been received.
13	Control Unit Failure. A control unit failure occurred during the initial selection of a channel program. It is also set if an error recovery CCW initiated by ERP does not properly complete. It is intended for statistical use only and remains set for the users inspection even though subsequent retries of the operation may result in successful transmission.
14, 15	For internal use by QTAM.

Figure 15. Communication Line Error Halfword (Part 2 of 2)

After the first message has been intercepted for any condition specified, the send bit in the terminal table entry for the terminal is turned off. This causes all subsequent messages for that destination not to be sent, and setting of the terminal inoperative bit (bit 1) in the error halfword. These subsequent messages will not be intercepted unless the error mask to test the error halfword has the terminal inoperative bit specified.

It is recommended that the user include an INTERCPT macro in his LPS for IBM terminals on a switched network and in his LPS for TWX terminals to intercept messages for which an addressing error (bit 12) occurs. The addressing error occurs whenever QTAM attempts to dial a terminal and the call is not completed. If INTERCPT is not included and the addressing error occurs, QTAM flags the current message as though it had been transmitted.

Similarly, use of INTERCPT is recommended to handle possible addressing errors when messages are being sent from the CPU to a 1053 Printer attached to the 2260-2848 Display Complex (Remote). A negative response to addressing (bit 12) occurs when the printer is addressed and a previous message sent to the printer by either the CPU or a 2260 Display Station has not been completely printed.

Use of INTERCPT is optional. However, if it is not used, messages that could not be transmitted are considered as transmitted by QTAM, even though they did not reach their destination. If used, it must appear in the End Send subgroup of the LPS.

Operation	Operand
INTERCPT	mask,subfield

mask
Is the hexadecimal representation of the bit configuration used to test the error halfword for the communication line involved.

subfield
Is the name of an optional subfield in the terminal table entry for the terminal that has messages to be intercepted. If the error condition being tested has occurred, the relative disk address of the first

suppressed message is placed into this subfield.

Logging (LOGSEG) Macro Instruction

The LOGSEG macro instruction enables the user to log message segments (place them on an output device as a record of message traffic carried by the line group). The user may maintain any or all of four types of logs by appropriate placement of LOGSEG within the LPS. The four types of logs, and the corresponding coding subgroup in which LOGSEG must appear, are:

1. Incoming headers only (Receive Header)

2. All incoming segments, or incoming message blocks if EOB characters are received from IBM terminals and the EOB or EOBIC macro is used (Receive Segment)

3. Outgoing headers only (Send Header)

4. All outgoing segments (Send Segment)

If all segments of messages are logged, they are logged in the sequence in which they are received or sent. Therefore, segments of different messages are intermixed on the log, not grouped together as individual messages. If the PREFIX operand is specified, the last 24 bytes of a QTAM header prefix are recorded on the logging device. These bytes precede the header portion (and text portion, if any) of the first segment of a message. The last 14 bytes of a QTAM text prefix are recorded on the logging device. These bytes precede the text portion of a text message segment. In all cases, the prefix is expressed in nonprintable representation.

LOGSEG may appear at any point in the subgroup in which it is used. However, the results of any alteration of segments by functional macro instructions preceding LOGSEG will appear in the logged segment. For example, if LOGSEG is preceded by TIMESTMP, all logged headers will contain time-of-day information; if TIMESTMP follows LOGSEG, headers will be logged without time-of-day information.

The logging effected by LOGSEG is in addition to the logging associated with the queueing procedure of QTAM. Use of LOGSEG is optional.

Operation	Operand
LOGSEG	filename[,PREFIX]

filename
: Is the name of the DTF table the user must define to specify the parameters of the file used for logging the message segments. The DTP xx macro instruction for the file on which the messages are logged must specify WORKAREA=YES.

  The DTF table address may be loaded into parameter register 1 prior to execution of this macro instruction; filename may then be coded as (1).

PREFIX
: Specifies that the QTAM header or text prefix is to be included with the logged segment. If this operand is not coded, the prefix is not logged. The first 8 bytes of the prefix are not included. Thus a header segment is logged with a 24-byte prefix and a text segment is logged with a 14-byte prefix. The format of the QTAM prefixes is contained in Appendix A.

Message Mode (MODE) Macro Instruction

The MODE macro instruction causes execution of a designated function, either:

1. Unconditionally (the designated function is performed for all messages handled by this portion of the LPS), or

2. Conditionally, if the next nonblank character of the message header is the same as a character designated by the MODE macro instruction.

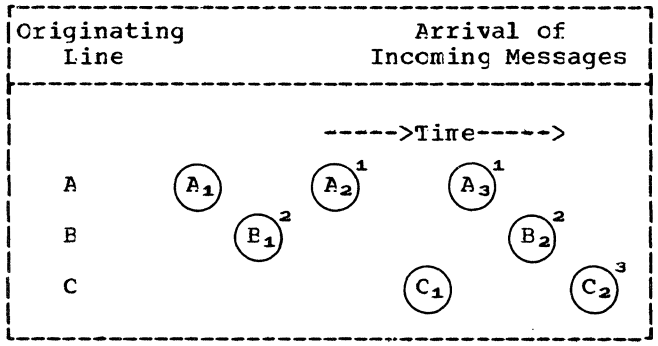
In the second case, if the characters are not the same, control returns to the next instruction in the LPS, and the scan pointer is reset to its position before the comparison.

MODE can cause the execution of any of four IBM-provided functions. The functions provided by IBM are discussed in the operand descriptions below.

The message priority scheme implemented by the MODE macro instruction with the PRIORITY operand is designed to permit a message from one of a group of lines sending to a common destination to be sent ahead of the other messages in the queue for that destination. The priority routine

compares the relative priority indicators of the last message sent from each line originating messages for the common destination. The highest-priority message is sent first, followed by the other messages, in order according to their priorities. The priority conferred on a message is valid only if that message is the last message to be sent from that line that is in the destination queue. If more than one message from a line is currently on the same destination queue, only the message last placed on that queue can have valid priority.

Example: Assume that lines A, B, and C are all sending messages to line D.



Messages being sent with priority are circled, (priority is indicated by superscript, 3 is highest priority). For the purpose of this example, a heavy circle indicates that the message was actually sent with priority; a dotted circle indicates that the message lost its assigned priority. This is explained as follows:

Assume that by time t, seven messages have arrived on the destination queue for line D, in the order A_1 B_1 A_2 C_1 A_3 B_2 C_2. Since messages A_3, B_2, and C_2 are the last messages received from their respective originating lines, only they will have priority. That is, because priority message A_3 arrived on the destination queue before priority message A_2, previously placed on the queue, was sent, message A_2 loses its assigned priority and is sent on a first-in-first-out basis, like all other nonpriority messages. Assuming no more messages arrive on the destination queue before all the messages are sent, the messages on the queue are sent in the order C_2 B_2 A_3 A_1 B_1 A_2 C_1.

Taking the example a step further, assume that at time t, message C_2 is sent. Then a new priority message, B_3, arrives on the queue before message B_2 is sent; this causes B_2 to lose its priority status to B_3. If no more messages arrive on the queue, the remaining messages are now sent in the order B_3 A_3 A_1 B_1 A_2 C_1 B_2. This case is not shown in the diagram.

Since the priority of some messages may be lost, due to later messages arriving on the disk for the same terminal, it is advisable that if PRIORITY is specified, sending should have priority over receiving (CPRI=S in DTFQT) in order to get messages off the disk as quickly as possible. This is particularly advisable for lines expected to have heavy volume.

Use of MODE is optional. If used, it must appear in the Receive Header or Send Header subgroup of the LPS, and may be used more than once in either subgroup. Its position within the subgroup must correspond to the point during header processing at which the function designated by its operand is to be performed. If used in the Receive Header subgroup of the LPS, the PRIORITY, CONVERSE, INITIATE, and condchar operands may be specified; if used in Send Header subgroup, the MOD 2260 and WRT60 operands may be used.

Operation	Operand
MODE	{PRIORITY / CONVERSE / INITIATE / MOD2260}
	[,condchar]
	[,WRT60=code]

PRIORITY
: Causes scanning of the header to locate the next nonblank character. This character is the priority value of the incoming message; it must be a letter or a digit. The priority sequence is A,...,0,...9 (9 is highest priority). If MODE designates a specific character by means of the second operand, scanning of the header for a priority character occurs only when the character designated in the second operand is found. If no specific character is designated, scanning always occurs, and all messages must have as the next nonblank character a priority character. This operand is effective only when used in the Receive group of the LPS. If multiple destinations are entered, only the first will be handled with priority. The PRIORITY operand may not be specified for IBM 2740 Model 2 terminals.

CONVERSE
: Causes the line on which the originating terminal is located to be

placed in conversational mode. The line is held open while an entire message from that terminal is sent to a message processing program and that program sends a response message to the same terminal. During this time, no incoming messages will be accepted from any other terminal on the line, and outgoing messages that have been queued for sending to any terminal on the line will not be sent. As the response message is sent, the line is taken out of conversational mode. After the response message is sent, the CPU polls the terminal.

Exception: During closedown, the terminal is not polled after the response message. The terminal operator may either:

1. Restore the line to conversational mode by sending another message; or

2. End transmission by entering an EOT character or allowing a timeout to occur.

If the originating terminal is a 2260 local, the entire line group is placed in conversational mode. If another 2260 local in the line group attempts to enter a message, the read request is recognized and queued for later servicing. However, the message will not be received from the second 2260 until a response message has been sent to the originating 2260.

The second operand can be specified for conditional use of CONVERSE. CONVERSE must be used only in the Receive Header subgroup of the IPS.

Note 1: The CONVERSE option of the MODE macro instruction must not be used when routing a message to a distribution LIST with more than one PROCESS entry.

Note 2: If a STOPIN has been issued to the line, the request for conversational mode is ignored.

INITIATE
Causes segments of a message to be sent from a destination queue to the destination as soon as they are placed on the queue (normally, segments are not sent to the destination until the complete message has accumulated on the queue). If a message has multiple destination codes specified in the header, the INITIATE function is performed only for the first destination. Sending to the remaining destinations will occur only after the complete message has been placed on the destination queue. The second operand can be specified for conditional use of INITIATE.

Restriction: The INITIATE function must not be used when sending a message to a message processing program.

Warning: INITIATE enables messages to be handled expeditiously. However, its use precludes the ability to perform error handling or to cancel the message.

MOD2260
Causes QTAM to modify the write operation for IBM 2260s (Remote or Local). The specific change must be indicated by the WRT60 keyword operand of this macro. MOD2260 may be specified only when the MODE macro is included in the Send Header LPS subgroup.

condchar
Is a character that, if found in the header before another nonblank character, causes execution of the function specified by the first operand. char can be any single nonblank character. If this second operand is omitted, the function is performed unconditionally. The character may be specified either as the character itself, or as the hexadecimal equivalent of the character. The framing C or X and quotes must be coded.

WRT60
Specifies the type of modification to be made to the write operation for the 2260 (Remote or Local).

WRT60=1 causes erasure of the 2260 screen before the header segment of the current message is displayed.

WRT60=2 causes a write-with-line-address address operation for the 2260. For a 2260 Remote, the user must place the desired line address in the first data byte of each outgoing message and in the data byte immediately following each STX character in the outgoing message. For the 2260 Local, the desired line address must be placed only in the first data position of the header segment. Figures 16 and 17 show the line address specifications for the two devices. The user can insert the line address by either:

1. Writing assembly language instructions to perform this in his LPS or in a message processing program; or

2. Using the PAUSE macro instruction.

This operand is specified only when the MOD2260 operand is specified.

Line Number	ASCII-8 Equivalent (Hex)	EBCDIC Equivalent (Hex)
1	50	F0
2	51	F1
3	52	F2
4	53	F3
5	54	F4
6	55	F5
7	56	F6
8	57	F7
9	58	F8
10	59	F9
11	5A	7A
12	5B	5E

Figure 16. Line Address ASCII and EBCDIC Equivalents for IBM 2260 Remote

Selected Line Number	Format of First Data Byte (HEX)
1	F0
2	F1
3	F2
4	F3
5	F4
6	F5
7	F6
8	F7
9	F8
10	F9
11	FA
12	FB

Figure 17. Line Address Specifications for IBM 2260 Local

Message Type (MSGTYPE) Macro Instruction

The MSGTYPE macro instruction enables the user to categorize incoming and outgoing messages into two or more message types, each of which he wishes to process in a different manner. A MSGTYPE macro instruction, encountered during processing of a message header, causes the next nonblank character or character sequence in the header to be compared with a character or character sequence specified by the operand of the MSGTYPE macro instruction. If they are identical, the instructions between this MSGTYPE macro and the next MSGTYPE macro or the next delimiter macro instruction are executed. If they are not identical, the instructions between the MSGTYPE macro performing the test and the next MSGTYPE macro or delimiter are not executed; the scan pointer is reset to its position prior to the scan. Instructions between a MSGTYPE macro instruction with no operand and the next delimiter are executed for messages that do not contain a message-type character or character sequence. These instructions are bypassed if the message was previously handled by a MSGTYPE macro instruction with a message-type character operand. If no operand is specified, the scan pointer is not advanced.

Use of MSGTYPE is optional, and it may be used any number of times. MSGTYPE may be used only within the Receive Header and Send Header subgroups.

Operation	Operand
MSGTYPE	typechar

typechar
Is the message-type code. It may consist of from one to eight nonblank characters. If this operand is omitted (that is, a blank is specified), the group of macro instructions that immediately follows this MSGTYPE macro instruction will process any message not handled by a preceding MSGTYPE macro instruction with a nonblank-character operand. If a MSGTYPE macro with a blank operand is used, it must be the last of the series of MSGTYPE macros. The message-type character sequence may be specified either as the characters themselves, or as the hexadecimal equivalent of the characters. The framing C or X and quotes must be coded.

Example: The beginning of a Line Procedure Specification section using MSGTYPE macro instructions is shown in Figure 18.

Name	Operation	Operand	Comments
LPS1	LPSTART	16	Reserves 16 bytes in the message header
	RCVSEG		Delimiter
	TRANS	RCVTWX	Macro instruction executed for all segments
	RCVHDR		Delimiter
	SEQIN SOURCE DATESTMP TIMESTMP COUNTER	4 3 6 MSGIN	Macro instructions executed for all header segments
	MSGTYPE	C'A'	Test for Type A messages
	- - DIRECT	 =CL8'CHI'	Macro instruction executed for all Type A messages
	MSGTYPE	C'B'	Test for Type B messages
	- DIRECT	 =CL8'NYC'	Macro instructions executed for all Type B messages
	MSGTYPE		Test for all other message types
	DIRECT	=CL8'PROCESSQ'	Macro instruction executed for all other message types
	ENDRCV		Delimiter
	- -		Remaining macro instructions of LPS

Figure 18. Use of MSGTYPE Macro Instruction in a nonaudio LPS

Operator Control (OPCTL) Macro Instruction

The operator control macro instruction designates one or two remote terminals as operator control terminals from which specified control messages may be entered into the system. If two terminals are specified, both terminals must use the same LPS. It is recommended that for an operator control terminal, receive have priority over send on the line. Otherwise, a large number of error messages may make it difficult or impossible to enter an operator control message at the terminal.

OPCTL must be specified in the Receive Header subgroup of the LPS for the line group that contains the operator control terminal(s). An IBM 1050, or an IBM 2740 with station control and checking, must be specified for this function. Only one OPCTL macro instruction may be specified. The message must be translated to EBCDIC before processing by OPCTL. When the OPCTL macro instruction is encountered in the LPS, the scan pointer must be set so that the next nonblank character is the first character in the control message identifier. For further information refer to the Operator Control Facility section.

Operator control messages are processed only by those macro instructions in the Receive group that precede the OPCTL macro instruction. Therefore, if the user wishes to insert the date or time of day information into the operator control message, the DATESTMP and TIMESTMP macro instructions must precede the OPCTL macro instruction.

Responses to operator control messages (sent in response to the COPYC and COPYT messages) are sent to the requesting terminal.

Restriction: operator control terminals receiving response messages or messages from error recovering procedures must have end-of-block checking feature.

Name	Operation	Operand
opname	OPCTL	CTLMSG=msgname, TERM=termname, [,ALTERM=termname] [,INTRCPT=symbol]

opname
The name of the macro instruction. It must be specified. It must be from one to eight nonblank characters.

CTLMSG
The control message identifier consisting of one to eight nonblank characters.

TERM
The name of the operator control terminal as it appears in the terminal table entry for this terminal.

ALTERM
If specified, the name of an alternate operator control terminal as it appears in the terminal table entry for that terminal.

INTRCPT
Is the name of a three-byte optional subfield in a terminal table entry and must be the same as the name assigned to the subfield by an CPTION macro instruction. This subfield contains the relative disk address of the first intercepted message header. This operand must be specified if the user wishes to use the Intercept Control Message (refer to Operator Control Facility section for further information on the Intercept Control Message). If the user attempts to use the Intercept Control Message when this operand has been omitted, the message is treated as an error.

PAUSE Macro Instruction

PAUSE causes automatic transmission of a user-specified sequence of characters on the communication line each time the LPS section containing the PAUSE encounters a user-specified character in the message segment currently being sent. The inserted characters are not placed in the outgoing message segment as contained in main storage. Rather, they become part of the segment as received at the terminal. To illustrate: If a message segment containing the characters

```
ABCDEF*GHI*ABCD*MNOPQ*RSTU**ABC
```

is handled by an LPS in which a PAUSE macro specifies insertion of the characters XY each time an asterisk is encountered, the segment as contained in main storage remains unchanged, but as received by the destination terminal becomes

```
ABCDEF*XYGHI*XYABCD*XYMNOPQ
      *XYRSTU*XY*XYABC
```

This facility has two main uses:

1. It permits the user to effectively modify outgoing message headers by inserting extra characters. The need for this arises when message headers received from certain terminal types are to be sent to other terminal types. In certain instances, extra control characters must be sent on the line during transmission of the header, in order for the message to be received properly.

2. It permits sending of nonprinting idle characters over the communication line, where necessary to prevent loss of message data. This usage is not required for the 2260-2848 Display Complex.

Characters in outgoing messages are sent continuously, even while the terminal device receiving the message is performing a mechanical positioning operation that interferes with correct recording of the incoming characters. For example, some terminal printers require more time for the carriage return operation than is available between printing of successive message characters: characters are printed randomly during the carriage return movement.

To avoid partial loss of a message from this cause, one or more nonprinting characters must be inserted into the message after each device control character (such as carriage return) that performs an operation otherwise resulting in loss of message characters. These nonprinting characters are referred to as idle characters, although the specific character to be used depends on the type of terminal that receives the message. The idle characters used by each type of device are shown in Figure 19.

The PAUSE macro can be used to cause insertion of idle characters each time a designated device control character appears in the message. (Device control characters can be inserted by a user-provided

subroutine or by the terminal that originates the message.) The specific control characters for which insertion is required, and the number of idle characters required for each, vary among terminal device types. For these requirements, see the reference manuals for the various terminal types.

The PAUSE macro instruction specifies:

1. The character that is to cause insertion.
2. The number of character sequences to be inserted.
3. The transmission code bit configuration of the characters to be inserted.

A separate PAUSE macro instruction must be specified for each control character for which insertion is required.

PAUSE, if used, must appear within the Send Header or Send Segment subgroups. If the TRANS macro instruction is used, the PAUSE must follow TRANS in the IPS.

If the PAUSE is used, the mmm operand of the BUFFER macro instruction may not be omitted and must be nonzero. Otherwise, the PAUSE macro instruction will not be effective.

Care should be taken in specifying the operands of the PAUSE macro instruction. Depending on the relative position of the PAUSE and TRANS macro instructions, the PAUSE operands should be in EBCDIC or in the transmission code for the terminal.

Operation	Operand
PAUSE	ctlchar,insertchar

ctlchar
: The actual transmission code bit configuration of the character for which insertion is required. It must be written in hexadecimal notation. This character cannot be an EOB or EOT.

insertchar
: The actual transmission code bit configuration of the character (or characters) to be inserted. It must be written in hexadecimal notation with the framing nX' ', where n is the number of character sequences to be inserted. (For example, 5X'E2E4' specifies that the sequence AB [in 1050 code] is to be sent five times.) This character cannot be an EOB or EOT.

Example: A PAUSE macro instruction to cause insertion of six idle characters into an outgoing message to an IBM 1050 each time a new line (NL) character is detected in that message by the message control program (5B and 5E are hexadecimal equivalents of 1050 device code new line and idle characters, respectively):

Operation	Operand
PAUSE	X'5B',6X'5E'

Terminal Type	Idle Character
IBM 1030, 1060	Idle (5E) (see note)
IBM 1050, 2740	Idle (5E) or delete (7F)
WU 33, 35	Rubout (FF)
AT&T 83B3, WU 115A	Figures shift (1B) or letters shift (1F)
WTTA	Figures shift (1B), letters shift (1F), or Mark (DF)

Note: The IBM 1033 Printer requires the insertion of three idle characters prior to each character transmitted to it.

Figure 19. Idle Characters

Polling Limit (POLLIMIT) Macro Instruction

This macro instruction is not applicable to WTTA lines or to the IBM 2260-2848 Local.

POLLIMIT is an optional macro instruction specifying a maximum number of messages to be accepted from a terminal during one polling pass. When this limit is reached, the next terminal is polled. If no polling limit is set (the POLLIMIT macro instruction is not used), each terminal is polled until it has no more messages to send during that polling pass.

The POLLIMIT macro instruction has no effect when used with a switched line. POLLIMIT may not be used for lines using Auto Poll if the Receive Header subgroup does not contain the SOURCE macro instruction. If used, POLLIMIT must appear at some point within the Receive Header or End Receive subgroup.

Note: For an IBM 2260 remote, the LPS must contain a POLLIMIT macro instruction that specifies a polling limit of one.

Operation	Operand
POLLIMIT	{nnn / subfield}

nnn
Is the maximum number of messages the user wishes to allow for each terminal in the line group. This option may be used only when the number of consecutive polls is to be the same for all terminals in the communication line group. The maximum value of nnn is 255; leading zeros are not required.

subfield
Is the name of a 1-byte optional subfield in a terminal table entry and must be the same as the name assigned to the subfield by an OPTION macro instruction. This subfield contains the limit of consecutive polls to be allowed for the originating terminal, as specified by a TERM macro instruction. This method of specifying the polling limit allows a different limit to be set for each terminal.

REROUTE Macro Instruction

The REROUTE macro instruction causes a message to be queued for an alternate destination (in addition to the destinations specified by the message header) when any of the errors specified by the mask have been detected.

The meaning of the bits in the error halfword tested is shown in Figure 15.

If the destinations specified by the message header are switched terminals, the SOURCE macro instruction must appear in the LPS prior to REROUTE, in order for the subfield operand to be specified. A distribution list cannot be specified as the alternate destination.

Use of REROUTE is optional. If used, it must appear in the End Receive or End Send subgroup. The REROUTE macro instruction must not be used to send messages to a PROCESS-EXPEDITE queue.

Operation	Operand
REROUTE	mask, {=CLn'dest' / subfield / SOURCE}

mask
Is the hexadecimal representation of the bit configuration used to test the error halfword in the line control block (LCB).

dest
Is the destination code for the alternate destination. The code may be the name of any entry that appears in the terminal table. If this option is selected, all messages with errors detected by REROUTE are sent to the same destination. n must be equal to or greater than the longest such name appearing in the terminal table. The maximum value for n is 8.

subfield
Is the name of an optional subfield in the terminal table that contains the name of the alternate destination. The name must be the same as the name assigned to the subfield by an OPTION macro instruction. If this option is selected, the alternate destination is the terminal specified in the option field of either:

1. the terminal table entry for the originating terminal, if REROUTE is used in the ENDRCV section of the LPS, or
2. the terminal table entry for the destination terminal, if REROUTE is used in the ENDSEND section of the LPS.

SOURCE
Specifies that the message in error is to be sent to the terminal from which it originated (in addition to the destination(s) specified by the message header). SOURCE may not be used for lines using Auto Poll if the Receive Header subgroup does not contain the SOURCE macro instruction, or if this REROUTE macro is used for an illegal source code (that is, if the mask contains a 1 in bit 6). SOURCE must not be used when the REROUTE macro instruction is in the End Receive subgroup if a should-not-occur error (bit 7), a transmission error (bit 8), a timeout exceeded error (bit 9) or a control unit failure (bit 13) has occurred.

Routing (RCUTE) Macrc Instructicn

The RCUTE macrc instruction causes scanning of the destination code field in the header of each incoming message. If the destination ccde is valid, RCUTE causes the message tc be queued for the specified destinations. If an invalid destination ccde (one not appearing in the terminal table) is detected:

1. Eit 0 of the error halfwcrd for the line containing the originating terminal is set to 1.

2. The message is placed on the dead-letter queue.

If further processing of messages placed on the dead-letter queue is required, a RERCUTE or ERRMSG macro instruction must be specified in the End Receive subgroup to nctify a terminal cperator cf the destination errcr.

Messages may be routed to multiple destinations in any cf three ways:

1. Mcre than cne destination code may be included in the message header. It is nct necessary tc indicate in the header the number of destinaticn codes included. When this method of routing to multiple terminals is used, the user must:

   a. Include an end-of-address (ECA) character after the last destinaticn code in the header of each incoming message.

   b. Specify an ECA macro instruction immediately following RCUTE in the IPS.

2. The message header may contain a single destinaticn code that identifies a distribution list in the terminal table. Each destinaticn in the distributicn list receives the message.

3. Where special machine features are available, group-code transmission may be used. Under this method, unique address characters cause the sending cf single messages simultaneously to a pre-specified group of terminals on the same line.

Either the RCUTE cr the DIRECT macro instruction must be specified tc handle message routing. Ecth cannot be used for the same message type. Only cne ROUTE macro may be used fcr each IPS cr for each message type used within one IPS (see the MSGTYPE macro instruction description).

RCUTE may be used only within the Receive Header subgroup.

Note: If the TERM macro instruction specifies that an IBM 2260-2848 complex is to be polled using the general poll feature, the DIRECT macro must be used. RCUTE cannot be used.

Operation	Operand
RCUTE	[n]

n

Is the number of characters in each destination code in the message header. n is specified only if the user chooses to make all destination codes the same length. The maximum value of n is 8. If this operand is omitted, destination codes are assumed to have varying lengths and a blank is required:

1. After a single destination code

2. Between multiple destination codes

3. Between the last destination code and the EOA character

If n is specified, the blanks are not required.

Sequence In (SEQIN) Macro Instruction

The SEQIN macro instruction causes scanning of the input sequence number field in the header of each incoming message. If the sequence number is not one higher than the sequence number of the last message received from the sending terminal, an error flag is set in bit 2 or bit 3 (depending on whether the number is high or low) of the error halfword for the line. When either error condition occurs, the sequence-in field in the terminal table entry remains unchanged.

The first message from a terminal must contain the same input sequence number as the sequence in (IJLQTSIN) field of the terminal table entry for that terminal. QTAM initially sets IJLQTSIN to 1. The user may at any time reset (by means of the CHNGT macro instruction) the contents of IJLQTSIN. If IJLQTSIN is reset before the maximum number (9999) is reached, the next incoming message must have the same number as IJLQTSIN. If IJLQTSIN is not reset before the maximum number is reached, the next incoming message after 9999 must be numbered zero.

In general, SEQIN causes the sequence-in field in the terminal table entry to be incremented for each message having a correct sequence-in number in the header. If, however, CANCEIM causes a message in error to be canceled, or if an EOBLC macro causes retransmission of the first block of a message, the input sequence number is not incremented. In the latter case, the number is incremented when the first block is successfully retransmitted.

Use of SEQIN is optional. For switched terminals or terminals using Auto Poll, the SEQIN macro instruction, if used, must be preceded by a SOURCE macro instruction. SEQIN may be used only within the Receive Header subgroup. Its position must correspond to the position of the sequence number field relative to other header fields.

Operation	Operand
SEQIN	[n]

n
: Is the number of character positions in the header field for the input-message sequence number. The maximum value of n is 4. If this operand is omitted, a variable-length field is assumed; in this case, the input-message sequence number must be followed by a blank used as a field delimiter. The value n does not include any blanks preceding or following the sequence number digits.

Sequence Out (SEQOUT) Macro Instruction

The SEQOUT macro instruction places an output sequence number in the header of each outgoing message. The LPS maintains a separate sequence count for each terminal and each terminal group (where group-code addressing is used). Each message for a terminal or terminal group is given a sequence number one greater than that of the preceding message for the same terminal or terminal group. A message in error rerouted via a REROUTE macro instruction or resent by the ECBIC macro instruction retains the output sequence number originally placed in it.

Use of SEQOUT is optional. If used, it must appear within the Send Header subgroup. Its position must correspond to the relative position, within the header, of the field into which the sequence number is inserted. Hence, it must perform the last editing of the header.

When SEQOUT is specified, the user includes the value of n in his calculation of the value of the operand of the LPSTART macro instruction (see the LPSTART macro instruction description).

Operation	Operand
SEQOUT	n

n
: Is the number of characters to be inserted in the header for the output sequence number. The first character is always a blank. The maximum value of n is 5; that is, the maximum field size is five characters, allowing for a sequence number range between 0001 and 9999. When the last available sequence number (99, 999, or 9999) has been issued to a message, the numbering cycle is repeated. The next message is numbered zero.

SKIP Macro Instruction

The SKIP macro instruction causes skipping of either a designated number of nonblank characters, or all characters up to and including a designated character or sequence of characters. This permits the user to skip fields in the message header during processing. SKIP macro instructions must appear among other functional macro instructions in the same relative order as fields to be skipped appear among other header fields.

Use of SKIP is optional. It may be used only within the Receive Header and Send Header subgroups.

Operation	Operand
SKIP	{n / skipchrs}

n
: Is the number of nonblank characters to be skipped. The maximum value of n is the number of characters remaining in the header.

skipchrs
: Is the character or sequence of characters designated to terminate the skip operation. The sequence must not exceed eight characters. The character or sequence of characters may be specified either as the characters themselves, or as the

hexadecimal equivalent of the characters.

Example: A SKIP macro instruction to cause skipping of five characters:

```
|Operation|Operand                    |
|---------|---------------------------|
|SKIP     |5                          |
```

Example: A SKIP macro instruction to skip characters up to and including #= may specify

1. the characters themselves, or
2. the hexadecimal representation of the characters:

```
|Operation|Operand                    |
|---------|---------------------------|
|SKIP     |C'#='                      |  (1)
|SKIP     |X'7B7E'                    |  (2)
```

SOURCE Macro Instruction

The SOURCE macro instruction causes scanning of the source terminal code field in the header of each incoming message to determine if the source code is valid. The validity check performed varies, depending on whether the source terminal is on a nonswitched or a switched line. Note that lines using Auto Poll are treated as switched rather than nonswitched lines.

If the source terminal is on a nonswitched line, SOURCE verifies that the header contains the symbolic name of the same terminal that was invited to send a message; that is, the source code field in the header is compared with the name of the terminal table entry for the terminal that was polled. If the names are not equal, an error flag is set in bit 6 of the error half-word for the line (see Figure 15). If the source terminal is a 2260 Local, SOURCE functions the same as for a terminal on a nonswitched line, with the exception that the 2260 Local is not polled.

If the source terminal is on a switched line, SOURCE can only verify that the source code field in the header contains a valid name (that is, the name of an entry in the terminal table, but not necessarily the name of the entry for the terminal that was polled). If a name that does not appear in the terminal table is detected, an error flag is set in bit 6 of the error half-word. Use of SOURCE is required if:

1. The SEQIN or COUNTER macro instruction is used in the Receive group of the LPS for switched terminals.
2. The DIRECT, ERRMSG, POLLIMIT, or REROUTE macro instruction containing the "subfield" operand is used in the Receive group of the LPS for switched terminals.
3. The name of a source terminal, attached to an autopolled line, is to be placed at the location specified by the TRMAD keyword operand of the DTFQT macro instruction when a GET is issued in the related Message Processing Program. (See DOS QTAM Message Processing Program Services, Form C30-5003.)

Note: On a switched line, if the connection has been established by the terminal, SOURCE must be issued in the LPS in order for messages on the destination queue for the terminal to be sent during this connection.

In either case, SOURCE must precede these macros in the LPS. SOURCE may be used only within the Receive Header subgroup. Its position within the subgroup must correspond to the position of the source terminal code field relative to other header fields.

```
|Operation|Operand                    |
|---------|---------------------------|
|SOURCE   |[n]                        |
```

n

Is the number of characters in the source terminal code field of the message header. The maximum value of n is 8. If this operand is omitted, a variable-length field is assumed. In this case, the source terminal code must be followed by a blank used as a field delimiter.

Time Stamp (TIMESTMP) Macro Instruction

TIMESTMP causes insertion of the time-of-day into the header portion of a message. This function can be specified for incoming messages, outgoing messages, or both. The time is expressed in the form bhh.mm.ss, where b is a blank, hh is the hours, mm the minutes, and ss the seconds. Nine character positions are required for the complete time information. However, the user may provide a shortened form (for example, omit the seconds) by reserving fewer than nine positions in the message header.

Use of TIMESTMP is optional. If used, it must appear in the Receive Header or Send Header subgroup. Its position within the subgroup must correspond to the relative position, within the header, of the field into which the time-of-day is inserted.

When TIMESTMP is specified, the user includes the value of nn in his calculation of the value of the operand of the LPSTART macro instruction (see the LPSTART macro instruction description).

Operation	Operand
TIMESTMP	nn

nn

Is the number of characters of time-of-day information to be inserted in the header portion of each message. The maximum value of nn is nine, and the value specified reflects the presence of the leading blank in the time information.

Note: TIMESTMP may be used only when the system includes the interval timer feature. However, it is not necessary that the interval timer be assigned to foreground 1. The time inserted into the header will be only as accurate as the time entered into the system by the operator.

Translate (TRANS) Macro Instruction

This macro instruction is not applicable to the IBM 2260-2848 Local, which is an EBCDIC device.

The TRANS macro instruction causes the characters of an incoming or outgoing message to be translated from one code to another. Incoming messages from a terminal are translated from the device code for that terminal type to EBCDIC. Outgoing messages are translated from EBCDIC to the transmission code for that terminal type. Translation is done character for character. TRANS specifies the transmission code from which or into which the message is to be translated.

TRANS is normally required in an LPS. It may be omitted if the source and destination terminals are of the same type and if the header analysis does not depend on the code. TRANS may be used in the Receive Header and/or Send Header subgroups to translate only the headers of incoming and outgoing messages, respectively (message switching to same terminal type). TRANS may be used in the Receive Segment and/or Send Segment subgroups to translate all segments, including header segments, of incoming and outgoing messages, respectively (inquiry processing and collection of data that is to be processed at a later time).

TRANS must not be used in the Receive Header subgroup if EOB or EOBLC is used in the End Receive subgroup. Similarly, TRANS must not be used in the Send Header subgroup if EOB or EOBLC is used in the End Send subgroup. For outgoing messages, TRANS must appear immediately preceding PAUSE (if used) or the ENDSEND delimiter.

TRANS is not required in a message switching application in which analysis of the header is not required of QTAM, provided that:

1. The originating and the destination terminals are of the same type.
2. The DIRECT macro, rather than the ROUTE macro, is used to send messages to destination terminals.

Code translation is normally accomplished through tables provided by QTAM, although the user may prepare and use his own tables, if desired. For each transmission code, QTAM provides two tables: one to translate from transmission code to EBCDIC, and one to translate from EBCDIC to transmission code. Exceptions are IBM 2740, IBM 1050, and WU 33/35 for which three tables are provided.

For WTTA lines, four translation tables are provided (two per 5-level code used). The user can modify these tables by using the four WTTA macro instructions RCVITA2, RCVZSC3, SNDITA2, and SNDZSC3, as explained above.

All of the characters in the character sets of each of the types of terminals capable of communicating with the System/360 CPU can be represented within the computer. However, some characters valid for one type of terminal device may not be valid for another type of terminal device. In a message switching application in which messages are exchanged between dissimilar terminal devices, the user should either:

1. Avoid placing in the message any characters that are not recognized by the destination terminal.
2. Employ a user-written translation table that converts such characters to other characters that are acceptable to the destination terminal.

The character sets of the IBM 1050 and of the IBM 2740 contain lowercase as well as uppercase alphabetic characters. When messages from either of these devices are sent to terminal devices or processing programs that do not recognize codes for lowercase characters, the user should either:

1. Use only the uppercase form of alphabetic characters, or

2. Employ the RCV1050F translation table (or the user's equivalent) when transmitting from the IBM 1050, or the RCV2740F translation table (or the user's equivalent) when transmitting from the IBM 2740. Each of these tables translates each incoming lowercase letter to the EBCDIC representation of that letter's uppercase equivalent.

When messages from an IBM 1050 or an IBM 2740 are sent to terminal devices or processing programs that do recognize codes for lowercase characters, the user may employ the RCV1050 translation table when transmitting from an IBM 1050, or the RCV2740 translation table when transmitting from an IBM 2740. Each of these tables translates incoming lowercase and uppercase letters into the corresponding EBCDIC lowercase and uppercase letters.

Note: All names for terminal table entries are assembled into the terminal table as uppercase EBCDIC characters. In order for source and destination code information in message headers to be recognized by the IPS macros as valid, such information must also appear to the IPS in uppercase EBCDIC form. For this reason, source and destination codes entered into message headers at an IBM 1050 must be entered in uppercase form, if the RCV1050 translate table is used. They may be entered in uppercase or lowercase if the RCV1050F table is used.

The SND2260 translate table converts lowercase alphabetic characters to uppercase so that the terminal receives only uppercase characters.

Two sending translate tables are provided for TWX terminals. The SNDTWXE translate table converts EBCDIC into 8-level code with even parity. The SNDTWXO translate table converts EBCDIC into 8-level code with no parity. If the user specified SNDTWX as an operand, the SNDTWXE (even parity) translate table is used, and an MNOTE generated at assembly time informs the user that this is being done.

Operation	Operand
TRANS	table-symbol

table
Is the name of the code translation table. Names of tables provided by QTAM are given in Figure 20.

Example: A TRANS macro instruction to translate messages sent from an IBM 1030 to the computer:

Operation	Operand
TRANS	RCV1030

Example: A TRANS macro instruction to translate messages from the computer to an AT&T 83B3 terminal:

Operation	Operand
TRANS	SND83B3

Table Name	Type of Conversion	Type of Terminal
For incoming messages:		
RCV1030	1030 code to EBCDIC	IBM 1030
RCV1050	1050 code to EBCDIC	IBM 1050
RCV1050F	1050 code to EBCDIC (converts lowercase alphabetic characters to uppercase)	IBM 1050
RCV1060	1060 code to EBCDIC	IBM 1060
RCV2260	2260 code to EBCDIC	IBM 2260 Remote
RCV2740	2740 code to EBCDIC	IBM 2740
RCV2740F	2740 code to EBCDIC (converts lowercase alphabetic to uppercase)	IBM 2740
RCV83B3	5-level (Baudot) code to EBCDIC	AT&T 83B3
RCV115A		WU 115A
RCVTWX	8-level code to EBCDIC	WU 33, 35
RCVARU	ARU code to EBCDIC	Audio terminals except IBM 3944
RCVITA2	ITA2 code to EBCDIC	WTTA
RCVZSC3	ZSC3 code to EBCDIC	WTTA
For outgoing messages:		
SND1030	EBCDIC to 1030 code	IBM 1030
SND1050	EBCDIC to 1050 code	IBM 1050
SND1060	EBCDIC to 1060 code	IBM 1060
SND2260	EBCDIC to 2260 code (converts lowercase alphabetic characters to uppercase)	IBM 2260 Remote
SND2740	2740 code to EBCDIC	IBM 2740
SND83B3	EBCDIC to 5-level (Baudot) code	AT&T 83B3
SND115A		WU 115A
SNDTWXE	EBCDIC to 8-level code (even parity)	WU 33/35
SNDTWXC	EBCDIC to 8-level code (non-parity)	WU 33/35
SNDITA2	EBCDIC to ITA2 code	WTTA
SNDZSC3	EBCDIC to ZSC3 code	WTTA

Figure 20. Names of Code Translation Tables Provided by QTAM

FUNCTIONAL MACRO INSTRUCTION DESCRIPTIONS (WTTA LINES)

The following macro instructions, RCVITA2, RCVZSC3, SNDITA2, SNDZSC3, and WRU, are applicable to WTTA lines only. The LPS for a WTTA line may also include functional macro instruction listed for nonaudio lines, except those whose use is specifically restricted. The macro instructions which may not be used in a WTTA LPS are BREAKOFF, EOB, ECBIC, and POLLIMIT, as well as the Audio functional macro instructions.

RCVITA2 AND RCVZSC3 MACRO INSTRUCTIONS

These macro instructions are applicable to WTTA terminals only. They allow the user to modify the two translation tables RCVITA2 and RCVZSC3, when necessary, and thus produce new tables which can be used by the TRANS macro instruction. These macro instructions can be placed anywhere in the message control program or can be assembled separately.

Name	Operation	Operand
symbol	RCVITA2	{Fx=hexchar,}...

Name	Operation	Operand
symbol	RCVZSC3	{Fx=hexchar,}...

symbol
Is the name of the translation table used in the TRANS macro instruction; its length cannot exceed four characters.

RCVITA2
Specifies that table RCVITA2 is to be modified and assembled.

RCVZSC3
Specifies that table RCVZSC3 is to be modified and assembled.

Fx=hexchar
Specifies a modification to the table concerned.

"F" means figureshift,

"x" represents the number of the code combination to be translated, and

"hexchar" is the hexadecimal representation of this character in EBCDIC.

The permissible values for "x" are:

For RCVITA2: 1, 2, 3, 6, 7, 8, 10 through 14, 19, 22, 24, 26, and 32.

For RCVZSC3: 1, 5, 8, 9, 11, 12, 14, 15, 17 through 20, 22, 24, 26, and 32.

Example: If a terminal operates in 5-bit International Telegraph Alphabet No. 2, combination 6 in figureshift representing the % character does not exist in table RCVITA2. Therefore, the user will create the required WTTA translation table (TBL) by writing:

```
TBL RCVITA2 F6=6C
```

where 6C is the hexadecimal representation of the % character in EBCDIC.

Note: These macro instructions can be used to create several translation tables in the same program, provided these tables are given different names. This enables several terminals using the same code, but with differences in their graphic arrangement, to operate in the same installation.

SNDITA2 AND SNDZSC3 MACRO INSTRUCTIONS

These macro instructions are applicable to WTTA terminals only. They allow the user to modify the two translation tables SNDITA2 and SNDZSC3, when necessary, and thus produce new tables which can be used by the TRANS macro instruction. These macro instructions can be placed anywhere in the message control program or can be assembled separately.

Name	Operation	Operand
symbol	SNDITA2	{Xyy=Fx,}...

Name	Operation	Operand
symbol	SNDZSC3	{Xyy=Fx,}...

symbol
Is the name of the translation table used in the TRANS macro instruction; its length cannot exceed four characters.

SNDITA2
Specifies that table SNDITA2 is to be modified and assembled.

SNDZSC3
Specifies that table SNDZSC3 is to be modified and assembled.

xyy=Fx
Specifies a modification to the table concerned.

"yy" is the hexadecimal representation in EBCDIC of the character to be translated.

"F" means figureshift, and

"x" is the number of the code combination to be translated.

The permissible values of "yy" are:

2A, 3F, 4A through 50, 5A through 61, 6A through 6F, 7A through 7F.

Example: If a terminal operates in 5-bit International Telegraph Alphabet no. 2, and if the user wishes to assign the hexadecimal value X'6C' (% character in EBCDIC) to combination 6 in figureshift (% character to be sent by the terminal), the required WTTA translation table (TBL) will be produced by writing:

```
TBL SNDITA2 X6C=F6
```

In the same way, the user can decide that the asterisk character (X'5C' in EBCDIC) is to be sent as a % character. The required WTTA translation table (TBL) will be produced by writing:

```
TBL SNDITA2 X5C=F6
```

And if the user decides that both the % and asterisk characters (X'6C' and X'5C' in EBCDIC, respectively) are to be sent as a % character, he will write:

```
TBL SNDITA2 X6C=F6,X5C=F6
```

Note: These macro instructions can be used to create several translation tables in the same program, provided these tables are given different names. This enables several terminals using the same code, but with differences in their graphic arrangement, to operate in the same installation.

WRU Macro Instruction

To request an identification exchange during transmission of an output message, a WRU macro instruction is written in either the Send Header or the End Send subgroups of the LPS. If the identification sent by the terminal is not the same as that specified by the ID parameter of the corresponding TERM macro instruction, the transmission-error bit (bit 8) and the message-not-sent bit (bit 12) of the error halfword for the line are set on as follows:

- Bit 8 is always set on.
- Bit 12 is set on only when an identification exchange has been requested by a WRU macro instruction written in the Send Header subgroup of the LPS.

The WRU macro instruction requires no operands and is effective provided either WRU=YES or IAM=YES is specified in the corresponding DTFQT macro instruction.

Operation	Operand
WRU	

INCLUDING A USER-WRITTEN ROUTINE WITHIN THE LPS

The design of an LPS section is such that a serially-reusable, user-written routine can be included. Linkage to a closed user-written routine can be included in any subgroup within the LPS There are several reasons why the user might include such a routine.

There may be no IBM-provided LPS routine to process particular information he wishes included in his message headers. Or, he may desire to expand the scope of an IBM-provided LPS routine (for example, to execute his own error-correction routines after the ERRMSG macro instruction indicates an error). A third case might be processing a header field in a manner entirely different from the way an IBM-provided LPS routine handles fields of this type.

To include a user-written, closed routine, the user must provide his own linkages. QTAM requires that the user-written routine save and restore the registers by standard register saving conventions. Register 13 contains the address of a QTAM-provided save area to be used for this purpose. If the user-written routine calls a second user-written routine, it must provide its own save area, etc. Figure 21 shows the control flow between an LPS and a closed, user-written routine.

Before entering the LPS section, the LPS control routine initializes certain registers with data that is used by many of the LPS macro instructions in performing their functions. For this reason, the registers must be preserved. The register contents are described in Appendix D and may be useful to the user in coding his own LPS routine. For example, if the user-written routine processes a field in the message header, the scan pointer register (register 5) may be used for scanning the field; if used, it must be updated to point to the end of the field (as described in Appendix D) upon return to the LPS.

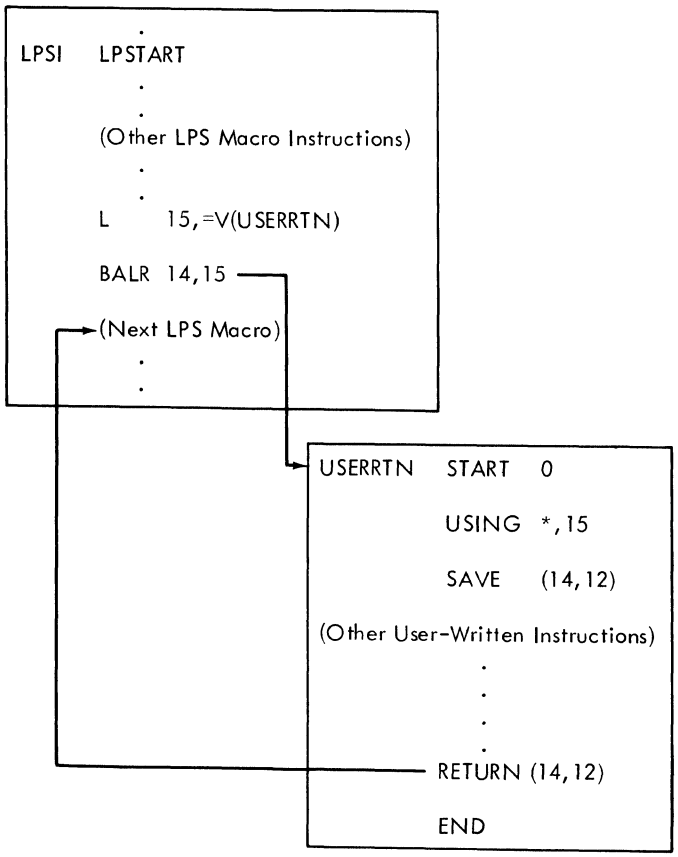


Figure 21. Activation of a Closed, User Written Routine in the LPS

The user also has the capability to include his own instructions as in-line code in the LPS. If any of the following registers are required for other than their intended purpose (see Appendix D), they must be saved and restored before execution of the next (in-line) LPS macro instruction: register 4, 5, 6, 7, 8, 9, and 13. If any of the following registers (work registers for the LPS macro instructions) are used, they must be saved before execution of the next LPS macro instructions and restored when needed: register 2, 3, 10, 11, and 12.

Note: Issuance of a Supervisor WAIT macro instruction from a user-written LPS routine (or in-line in the LPS) halts all processing of LPS macro instructions in the message control program until the condition being waited for is satisfied. WAIT, therefore, should either not be used, or should be used with extreme care. The ATTACH and DETACH macro instructions may not be used in a user-written LPS routine.

QTAM provides a variety of facilities which permit the user to examine and dynamically modify the status of the telecommunications network. These facilities (and some additional ones) are also available in a message processing program (refer to the DOS QTAM Message Processing Program Services publication).

Macro instructions are provided by QTAM to examine and modify the status of the system at the time the message control program initiates and activates the system. Execution of these macro instructions must occur between that of the OPEN and ENDREADY macro instructions. The user should be aware that as soon as execution of the OPEN macro instruction has completed, it is possible for a dialed line to be activated, in which case there may be some danger in attempting to change the terminal table, for instance.

Macro instructions which may be used to examine and modify the status of a ncnaudio system enable the user to dynamically:

- Activate a stopped line (STARTLN macro instruction).
- Examine the four cumultive counters for a line, and reset the threshold counters (COPYC macro instruction).
- Examine and modify the terminal table entries (COPYT and CHNGT macro instructions).
- Examine queue control blocks for DASD destination and process queues (COPYQ macro instruction).

When these macro instructions are used in the LPS, QTAM ensures that register 13 contains the address of an 18-word save area.

ACTIVATING A STOPPED LINE

QTAM provides means of activating a stopped line though the STARTLN macro instruction.

Start Line (STARTLN) Macro Instruction

STARTLN can be used to either:

1. Allow message transmission to resume on a particular line in a communication line group; or
2. Allow message transmission to resume on all lines in a communication line group.
3. Allow message transfer to resume for an IBM 2260-2848 Local line group.

The user must have previously opened the line group in the message control program.

If a line or line group is deactivated by a STOPLN macro instruction, issued in the message processing program, or if a line was opened idle, STARTLN must be issued before message transmission on that line or line group can resume.

In all the above cases, the presence of an active polling list is a prerequisite for message transmission. (An active polling list is one in which the second byte of the list is a nonzero character -- this character is initialized as a 1 and can be changed by the CHNGP macro instruction.) If STARTLN is used, polling of input lines or receipt of messages from 2260 Locals begins after the execution of that macro instruction. Initial polling of input lines in a line group and receipt of messages from 2260 Locals begins after the execution of the ENDREADY macro expansion in the message control program.

Name	Operation	Operand
[symbol]	STARTLN	filename, {rln / ALL}

symbol
Is the name of the macro instruction.

filename
Is the name of the DTF table for the line group containing the line to be reactivated. It must be identical to the name of the DTFQT macro instruction that generates the DTF table for the line group.

If register notation is used, the address of a location containing the name must be in the general register designated. The field that contains the name must be eight bytes (CL8) in length. (Padding with blanks is

required for names less than eight bytes.)

rln
Is the relative line number, within the line group, of the line to be reactivated. If register notation is used, the general register specified must contain the relative line number in binary form.

ALL
Specifies that all lines in the line group are to be reactivated.

Error returns from this macro instruction are contained in Appendix G.

EXAMINING AND MODIFYING LINE ERROR COUNTERS

Eight counters are provided for each line in the system: four threshold counters and four cumulative counters. Each cumulative counter corresponds to one of the threshold counters. The threshold counters keep a count of:

1. Transmissions
2. Data checks
3. Intervention required errors
4. Nontext timeouts

For further information, refer to the Error Recovery Procedures section.

Copy Error Counters (COPYC) Macro Instruction

This macro instruction does not apply to the IBM 2260-2848 local. CCPYC causes the four cumulative counters of the line specified to be placed into a designated work area. The threshold counters are reset to 0.

Name	Operation	Operands
[symbol]	CCPYC	termname,rln,workarea

symbol
The name of the macro instruction.

termname
The name of any terminal whose error counters are to be copied. If register notation is used, the general register designated must contain the address of a location containing the name of the terminal. The field that contains the name must be n bytes long, where n equals or exceeds the length of the longest name of any terminal table entry. If an invalid terminal name is specified, no data movement takes place. An error indicator of X'20' is returned in register 15. If no error is detected, register 15 contains 0.

rln
The relative line number, in the line group, of the line over which the terminal communicates with the computer. If this operand is not used, its absence must be indicated by a comma.

workarea
The address of the area into which the information is placed. The size of the work area must be ten bytes. If register notation is used, the general register designated must contain the address of the work area.

Bytes one through four of the workarea contain the number of transmissions; bytes five and six, the number of data checks; bytes seven and eight, the number of interventions required errors; and bytes nine and ten, the number of timeouts.

EXAMINING AND MODIFYING THE TERMINAL TABLE

QTAM provides macro instructions that enable the user to examine and change dynamically the control information contained in a terminal table entry.

The COPYT macro instruction causes the contents of a specified terminal table entry to be copied into a work area. This macro instruction can be used in conjunction with the CHNGT macro instruction, which substitutes a new terminal table entry for a superseded one. The user issues a COPYT, examines the information and changes it if necessary, and issues a CHNGT.

Copy Terminal Table Entry (COPYT) Macro Instruction

COPYT moves the information contained in a specified terminal table entry into a designated work area. The terminal table entry can be either a single terminal, group-code, distribution list, or process program entry. Formats for each of these entries are shown in Appendix A.

Name	Operation	Operand
[symbol]	COPYT	termname,workarea

symbol
Is the name of the macro instruction.

termname
Is the name of the terminal whose terminal table entry is to be copied. If register notation is used, the general register designated must contain the address of a location containing the name of the terminal; the field that contains the name must be n bytes long, where n equals or exceeds the longest name of any terminal table entry. If an invalid terminal name is specified, no data movement takes place; the routine linked to by the COPYT macro instruction returns an error code of X'20', right-adjusted in register 15. If no error is detected, register 15 contains zero.

workarea
Is the address of the area into which the information is placed. The first byte of the work area receives the first byte of data from the terminal table entry. The maximum size of the work area is 255 bytes (the maximum size of a terminal table entry). If register notation is used, the general register designated must contain the address of the work area.

Change Terminal Table Entry (CHNGT) Macro Instruction

CHNGT moves the information for a terminal table entry from a designated work area to the terminal table area allocated for that entry. CHNGT causes the entire contents of the superseded terminal table entry, except for the sequence-in (IJLQTSIN) and sequence-out (IJLQTSOT) fields, to be changed. The IJLQTSIN and IJLQTSOT fields are not changed because of the possibility that a message may be received between the moment the entry is copied and the moment it is changed. This would cause a sequence-number error to occur. In order to change the entire contents, including IJLQTSIN and IJLQTSOT, the user must precede the CHNGT macro with a STOPLN macro for the line to which the affected terminal is attached. CHNGT does not change the device access area (low order 24 bytes) of the terminal table entry for an IBM 2260 local.

CHNGT is normally preceded by the COPYT macro instruction and instructions to examine and modify the contents of the copied terminal table entry. The user must be certain that the new terminal table entry contains all the information required for proper execution of QTAM. The format of the terminal table entries and the information contained in each field are contained in Appendix A.

Name	Operation	Operand
[symbol]	CHNGT	termname,workarea

symbol
Is the name of the macro instruction.

termname
Is the name of the terminal whose terminal table entry is to be replaced. It must be the same as a name that appears in the name field of a TERM, PROCESS, or LIST macro instruction. If register notation is used, the address of a location containing the name must be in the general register designated; the field that contains the name must be n bytes long, where n equals or exceeds the longest name of any terminal table entry. If an invalid name is specified, no data movement takes place; the routine generated by CHNGT returns an error indicator of X'20', right-adjusted in register 15 (zero, if no error is detected). QTAM subsequently disregards the new terminal table entry and continues to use the old.

workarea
Is the address of the area from which the information is moved. If register notation is used, the general register specified must contain the address of the work area. If the new entry does not equal the size of the old entry, no data movement takes place. An error indicator of X'10' is returned in register 15 (zero, if no error is detected), and QTAM continues to use the old entry.

EXAMINING AND MODIFYING POLLING LISTS

QTAM provides macro instructions that enable the user to examine and modify the contents of the polling list for a line.

The COPYP macro instruction causes the contents of a specified polling list to be copied into a work area. This macro

instruction can be used in conjunction with the CHNGP macro instruction, which can substitute a new polling list for a superseded one. The user issues a COPYP, examines the information and changes it if necessary, and issues a CHNGP. CHNGP can also be used to stop or restart polling of the terminals on a line.

Copy Polling List (COPYP) Macro Instruction

COPYP causes the polling list for a specified line to be copied into a user-designated work area. The format of the polling list is shown in Appendix A.

Name	Operation	Operand
[symbol]	COPYP	filename,rln,workarea

symbol
Is the name of the macro instruction.

filename
Is the name of the DTF table for the line group containing the line whose polling list is to be copied. It must be identical to the name of the DTFQT macro instruction that defines the DTF table for the line group.

If register notation is used, the address of a location containing the name must be in the general register specified. The field that contains the name must be eight bytes (CL8) in length (padding with blanks is required for names less than eight bytes).

rln
Is the relative line number, within the line group, of the line whose polling list is to be copied. If register notation is used, the user must have previously placed the relative line number (in binary form) in the general register designated.

workarea
Is the address of the work area into which the polling list is to be copied. The first byte of the work area receives the first byte of data in the polling list. The size of the area necessary can be determined from the polling list format shown in Appendix A. If register notation is used, the general register specified must contain the address of the work area.

Error returns from this macro instruction are contained in Appendix G.

Change Polling List (CHNGP) Macro Instruction

CHNGP can either:

1. Place a new polling list in the polling list area for a specified line; or
2. Change the status of a polling list for a specified line.

CHNGP should be used to change only the status of a polling list associated with an IBM 2260-2848 Local line group. The terminal entries in the list are used only at Open and Close times and should not be modified. A CHNGP macro with the =C'0' operand can be used to temporarily stop receipt of messages from all 2260 Locals in the line group.

With autopolled terminals:

- To deactivate - use CHNGP and send a message to or from the terminal.
- To activate - use CHNGP and send a message to the terminal.

Name	Operation	Operand
[symbol]	CHNGP	filename,rln, {workarea / =C'0' / =C'1'}

symbol
Is the name of the macro instruction.

filename
Is the name of the DTF table for the line group containing the line whose polling list is to be modified. It must be identical to the name of the DTFQT macro instruction that defines the DTF table for the line group.

If register notation is used, the address of a location containing the name must be in the general register designated. The field that contains the name must be eight bytes (CL8) in length (padding with blanks is required for names less than eight bytes).

rln
Is the relative line number, within the line group, of the line whose

polling list is to be modified. If register notation is used, the user must have previously placed the relative line number (in binary form) in the general register specified.

workarea
Is the address of the area that contains the new polling list. The first byte of the polling list area receives the first byte of data in the work area.

If the new polling list is larger than the initial polling list for the line generated at assembly time, no data movement takes place. An error indicator of X'10' (zero, if no error is detected) is set in register 15. QTAM subsequently disregards the new polling list and continues to use the old.

=C'0'
Causes the second byte of the polling list to be changed to a zero. This results in the deactivation of the polling list; no further messages are received until the list is reactivated.

=C'1'
Causes the second byte of the polling list to be changed to a one. This results in the activation of the polling list. If the polling list is for a nonswitched line, QTAM begins polling the terminals on the line and accepting incoming messages, if the procedure is followed by either:

1. A STARTLN macro instruction for the line whose polling list was reactivated, or

2. Sending a message to a terminal on the line whose polling list was reactivated.

If the polling list is for a 2260-2848 local line group, QTAM automatically resumes receipt of messages from the 2260 Locals defined in the polling list.

If neither (1) or (2) occurs, no polling takes place on the line.

Other error returns from this macro instruction are contained in Appendix G.

EXAMINING QUEUE CONTROL BLOCKS

Each terminal table entry defined by a TERM or PROCESS macro instruction contains the address of the queue control block (QCB) for the DASD destination or DASD process queue on which outgoing messages to the destination(s) are placed. QTAM uses the QCB for:

1. Placing each message on its appropriate DASD queue.

2. Maintaining information on the status of the queue.

The COPYQ macro instruction enables the user to examine a QCB to ascertain the status of the DASD destination or DASD process queue associated with the QCB.

Figure 22 shows the contents and relative displacement of each field in the QCB that is of interest to the user. After issuing a COPYQ macro instruction to copy the QCB into a user-specified work area, the user can determine the contents of the fields from which he needs information. For example, he can determine the number of messages in the queue, or he can use the address of the queue on the disk to retrieve a message (see the RETRIEVE macro instruction description).

Copy Queue Control Block (COPYQ) Macro Instruction

COPYQ places the contents of a QCB into a specified work area. The user indicates the QCB desired by specifying the name of a terminal or the name of a DASD process queue. If the name of a terminal is specified, COPYQ places into the work area the QCB for the DASD destination queue associated with that terminal. If the name of a DASD process queue is specified, the QCB for the DASD process queue is placed into the work area. In both cases, the entire contents of the 32-byte QCB are provided. However, certain fields are used internally by QTAM routines and are not of interest to the user (see Figure 22).

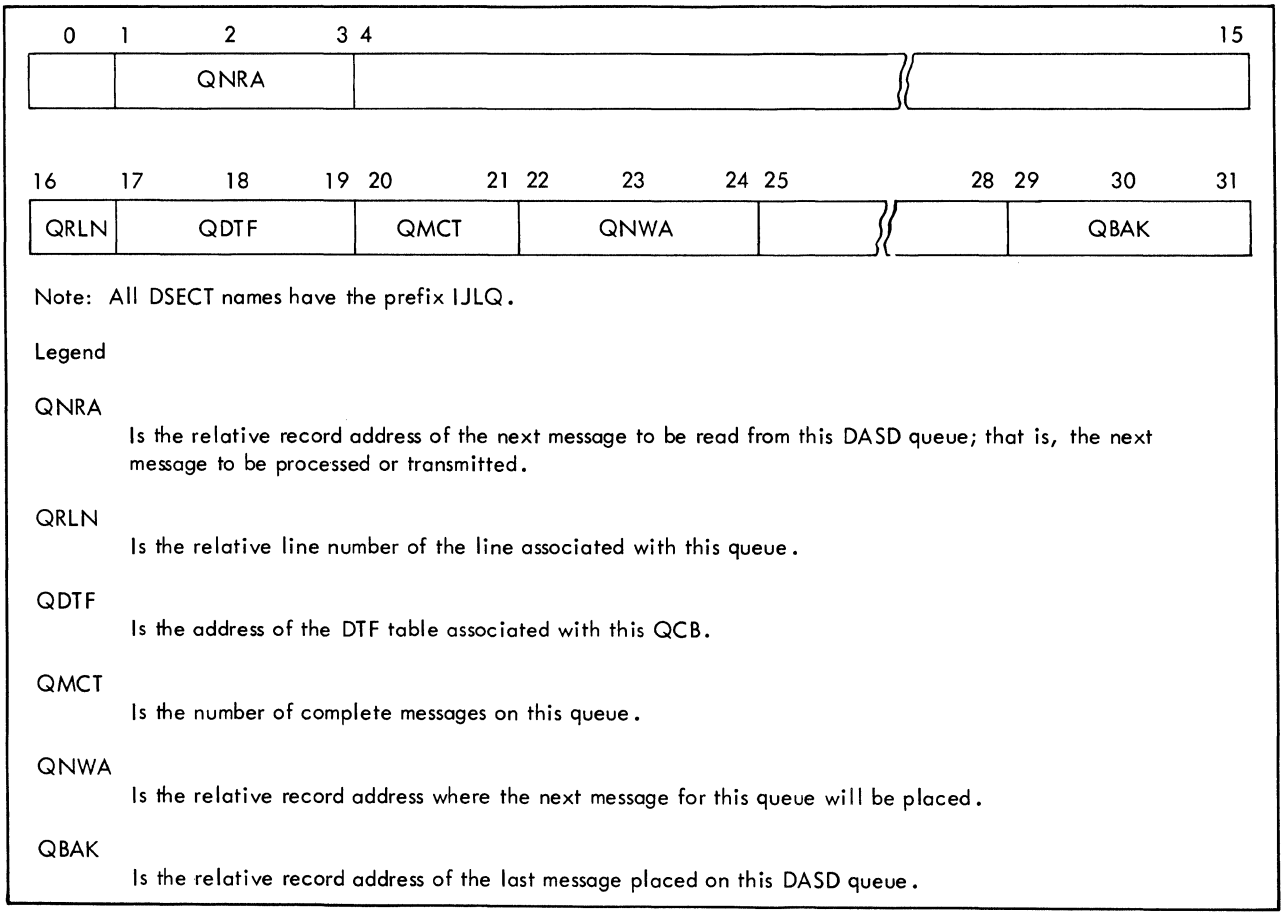
Name	Operation	Operand
[symbol]	COPYQ	termname,workarea

symbol
Is the name of the macro instruction.

termname
Is the name of the terminal or DASD process queue whose associated QCB is to be copied. Only the name of a single

terminal or process program terminal table entry can be specified (that is, the name specified in a TERM or PROCESS macro instruction). If an invalid name is specified, no data movement takes place. The routine linked to by the CCFYQ macro instruction sets X'20' in register 15 as an error indicator. If the name specified is valid, a 0 is placed in register 15. If register notation is used, the address of a location containing the name must be in the designated general register; the field that contains the name must be n bytes long, where n equals or exceeds the longest name of any terminal table entry.

workarea
Is the address of the area into which the contents of the QCB are placed. The area must be 32 bytes long (the size of the QCB). If register notation is used, the general register specified must contain the address of the work area.



Note: All DSECT names have the prefix IJLQ.

Legend

QNRA
Is the relative record address of the next message to be read from this DASD queue; that is, the next message to be processed or transmitted.

QRLN
Is the relative line number of the line associated with this queue.

QDTF
Is the address of the DTF table associated with this QCB.

QMCT
Is the number of complete messages on this queue.

QNWA
Is the relative record address where the next message for this queue will be placed.

QBAK
Is the relative record address of the last message placed on this DASD queue.

Figure 22. Format of Queue Control Block (QCB)

DEACTIVATING THE TELECOMMUNICATIONS SYSTEM

In order to terminate operation of the telecommunications system, the communication line group, the 7772 DCV vocabulary, and the DASD message queues files must be closed. However, before they may be closed, all message traffic in the system must cease, and all main storage process and destination queues opened in any message processing program must be closed. To accomplish this, the user must issue a CLOSEMC macro instruction in a message processing program, and then close all QTAM files opened in the message processing programs. During this procedure, QTAM controls line activity and monitors the closing of QTAM files in the message processing programs. When all QTAM files opened in the message processing programs have been closed and line activity has ceased, control is returned to the user to permit him to close the line group, the 7772 DCV vocabulary, and the DASD message queues. Deactivation of the system proceeds in the following manner.

When deactivation of the system is to be initiated, a CLOSEMC macro instruction must be issued in a message processing program. A recommended procedure is to send a special message to a process queue from which a message processing program may obtain the message. The processing program recognizes the special message and enters a routine that issues a CLOSEMC. In addition, the partition issuing the CLOSEMC may have to signal any other processing partitions, via a PUT, to CLOSE its files and to issue its EOJ. It is necessary to signal each process queue in the other partitions that do not have the SYNCAD option.

When the CLOSEMC macro instruction is issued, the following action occurs. Outgoing message traffic continues on any lines that are not currently receiving messages. Meanwhile, incoming message traffic on each line is limited to the message currently being received over that line. When the last block of the current message is received, no more incoming messages are accepted (that is, the line is not repolled or reenabled). As each such line becomes free, any outgoing messages that have been queued for terminals on that line are sent. In this manner, incoming message traffic declines to nothing, while outgoing message traffic continues until all messages have been sent.

After CLOSEMC is completed, no further incoming messages are accepted over any line in the system. Thus all messages that entered the system from a terminal have either already been processed or are in queue waiting to be processed. At this point, the GET/PUT portion of the message processing program should be reentered to obtain and process any messages still remaining on the process queues. When a GET macro is issued for a process queue and the queue has been exhausted, return is made to a user-written termination routine, the address of which is specified in the FLUSHAD operand of the DTFQT macro used to define the MS process queue. This termination routine should consist of one or more CLOSE macro instructions to close the MS process queue and MS queue files and any other files opened in that message processing program. The user's termination routine should then issue an EOJ macro to end execution of the message processing program.

The user obtaining messages from multiple process queues in the same message processing program must include additional code in his termination routine to ensure that all messages have been processed. If all messages to be processed are from terminals, he need ensure only that the FLUSHAD return has been made for each process queue before closing the QTAM files. If messages are being transferred (via PUT macros) from this or any other message processing program to the process queues, he should ensure, additionally, that consecutive FLUSHAD returns have been made for all process queues. The latter is necessary because a message may arrive on a process queue (due to a PUT) after the FLUSHAD return has already been made for that particular queue.

The QTAM Close routine monitors the closing of the QTAM files opened in the message processing programs. When it finds that all of these files have been closed, and all outgoing message traffic has ended, the routine stops operations on each line in the system. When all lines have been stopped, control returns to the user at the address in the message control program specified in the EOJAD keyword operand of the DTFQT macro for the first file opened in the message control program. This address must begin a user-written routine that deactivates the message control program. This deactivation routine must issue CLOSE macro instructions for each of the files opened in the message control program (that is, the line group, message log, 7772 DCV vocabulary, DASD message

queues, and DASD checkpoint records files). This user written section of code must be assembled as part of the message control program.

The DASD checkpoint records file (if used) must be closed after the communication line group files and before the DASD message queues file. The DASD message queues file must be the last QTAM file to be closed. This is important, because closing this file constitutes deactivation of the telecommunications system. After the first-opened file has been closed, no further references can be made to queues, control blocks, terminal table, polling lists, etc. The deactivation routine should end with an EOJ macro instruction in order to end the message control job.

CLOSE Macro Instruction

The CLOSE macro instruction is used in the message control program to deactivate any line group, DASD message queues, and message log files the user has included in his telecommunications system. The DASD checkpoint records file (if used) must be closed after the nonaudio communication line group files and before the DASD message queues file. The first-opened QTAM file must be the last QTAM file closed.

This macro instruction, if used, must appear in a section of code the address of which is specified in the ECJAD keyword operand of the DTFQI for the first-opened QTAM file.

Name	Operation	Operand
[symbol]	CLOSE	filename[,...]

symbol
Is the name of the macro instruction

filename [,...]
Specifies the symbolic addresses of the DTF tables associated with the files being closed. All files can be closed with one CLOSE by including the addresses of their DTF tables as operands. If register notation is used, the addresses of the DTF table(s) must have previously been loaded into the general registers specified.

Telecommunications System Cancellation

At any time, the user may issue 1052 console requests to cancel his telecommunications system. However, in a QTAM application, the cancellation of the message control program also brings on the cancellation of the message processing programs. If a message processing program is cancelled in this way, that is, by program request, a dump of the message processing program may not be provided. To ensure that a dump of each QTAM partition is provided, the user must cancel each message processing program before cancelling the message control program.

This section describes the following services that QTAM provides to aid the user in network control, error recovery and testing:

- Operator Control Facility
- Checkpoint/Restart Facility
- On-line Terminal Testing
- Error Recovery Procedures
- Operator Awareness
- OBR/SDR Error Recording

Warning: The discussion in this section is not applicable to audio lines. For a discussion of these facilities for audio lines, refer to the appropriate section.

Warning: With multitasking, the maintask cannot use STIXIT IT if QTAM is attached with operator control, checkpoint by interval timer, or polling interval facilities in the application.

OPERATOR CONTROL FACILITY

The Operator Control facility allows the execution of QTAM macro instructions by special control messages received from specified operator control terminals. The QTAM macros supported under this facility are: COPYC, COPYT, CHNGT, INTERCPT, RELEASEM, STARTLIN, and STOPLN. Two additional functions, INTREI and SWITCH, are also provided. These messages are summarized in Appendix I.

The user has three choices for operator control support:

1. None,
2. Operator control facilities only, or
3. Operator control facilities with error messages sent to the operator control terminal(s).

The following table summarizes how each option may be specified.

Facility	TERMTBL Macro	OPCTL Macro
None	Not specified	Not defined
Operator control only	Not specified	Defined
Operator control with error message	OPCTI=chars	Defined

Error messages from the Error Recovery Procedures (but not error messages originating from the ERRMSG macro instruction) are normally sent to the IBM 1052 system console. Specifying the OPCTL=chars operand under the TERMTBL macro instruction and defining the OPCTL macro instruction causes these error messages to be sent to the operator control terminal. If this is done, error messages originating from errors on the operator control terminal are sent to the IBM 1052 system console.

The format of input control messages must be as shown in the individual message description below. Fields are delimited by one or more blanks.

If the control message (ctlmsg) field is incorrect, or if the source terminal is not an operator control terminal, the message is not processed by the operator control routine. If the function is invalid, the message is returned to the source terminal as entered. If any succeeding parameters are invalid, the results are unpredictable. If an error is detected in the data field, a shortened message which does not contain the data is returned.

MESSAGE FORMATS AND DESCRIPTIONS

The format of input control messages is discussed below. Output messages are limited to one buffer. Data is in hexadecimal format unless specified otherwise. The control message requesting the information is returned to the operator control terminal, followed by the information requested.

Copy Error Counters (COPYC)

The COPYC operator control message does not apply to the IBM 2260-2848 local.

This message causes the four cumulative counters of the line specified to be printed at the source terminal. The threshold counters are reset to 0. The data is in decimal format. If the control message and the counters copied exceed buffer size, only that portion of the output message that the buffer can contain is printed.

Identifier	Function	Operands
ctlmsg	COPYC	termname rln

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
The name of any terminal for which counters are desired.

rln
The relative line number of the line in the line group. It must be in decimal.

Copy Terminal Table Entry (COPYT)

This message causes the terminal table entry beginning at IJLQTSIN to be printed on the source terminal. The information is printed in hexadecimal. If the control message and the part of the terminal table entry copied exceed buffer size, only that portion of the entry that the buffer can contain is printed.

Identifier	Function	Operands
ctlmsg	COPYT	termname

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
The name of the terminal table entry for the terminal.

Change Terminal Table Entry (CHNGT)

This message causes the data entered to replace the terminal table entry specified beginning at IJLQTSIN up to the end of data but not exceeding the entry itself.

Identifier	Function	Operands
ctlmsg	CHNGT	termname data

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
The name of the terminal table entry for the terminal.

data
The actual data that is to replace the terminal table entry. It must be entered continously, in hexadecimal format, ending with a blank, EOB, or EOT, and beginning with the IJLQTSIN field of the terminal table entry. The sequence fields are changed only if the line is stopped.

Intercept Message (INTERCPT)

This message causes the suppression of all message transmission to the terminal specified. The terminal remains intercepted until a RELEASEM operator control message is entered for that terminal (or a RELEASEM macro instruction is issued for that terminal in a message processing program). If the INTERCPT control message is used, the user must:

1. specify an INTERCPT macro instruction in the End Send subgroup of the LPS for the terminal specified in termname with a mask which includes "terminal inoperative;" and
2. specify the INTRCPT keyword operand in the OPCTL macro instruction.

Identifier	Function	Operands
ctlmsg	INTERCPT	termname

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
The name of the terminal table entry for the terminal.

Intercept and Release Messages (Line Test) (INTREL)

This message causes the suppression of all message transmission to and from the terminal specified for a two-minute interval. After two minutes, messages are released until an unrecoverable error occurs, at which time the cycle begins again with intercept. This continues until a RELEASEM is issued. If the INTREL control message is used, the user must specify an INTERCPT macro instruction in the End Send subgroup of the IPS for the terminal specified in termname with a mask which includes "terminal inoperative." Note that if this message is to be used, the interval timer feature is required and must be assigned to the foreground-one partition.

If INTREL is specified for a line that has been deactivated by STOPLN, the operator awareness message INTREL NCT DONE is issued.

Identifier	Function	Operands
ctlmsg	INTREL	termname

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
The name of the terminal in the terminal table.

Release Messages (RELEASEM)

This message causes all intercepted messages with that terminal as the destination to be sent as well as new messages. That is, it resets the INTERCPT and INTREL condition for this terminal. If the RELEASEM control message is to be used, the user must specify the INTRCPT keyword operand in the OPCTL macro instruction.

Identifier	Function	Operands
ctlmsg	RELEASEM	termname

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
The name of the terminal table entry for the terminal.

Stop Line (STOPLN)

This message removes a nonaudio communications line from active use.

Identifier	Function	Operands
ctlmsg	STOPLN	termname, [ALL / rln]

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
The name of any terminal on the line that is to be stopped.

ALL
Specified that all lines in the line group are to be stopped.

rln
Specifies in decimal form the relative line number within the line group of the line to be stopped.

Start Line (STARTLN)

This message causes transmission to begin or resume on a line or all lines in a line group, provided the line group is opened and has an active polling list.

Identifier	Function	Operands
ctlmsg	STARTLN	termname, [ALL / rln]

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTI macro instruction.

termname
The name of any terminal on the line that is to be started.

ALL
Specifies that all lines in the line group are to be started.

rln
Specifies in decimal form the relative line number within the line group of the line to be started.

Switch Control Terminals (SWITCH)

This message causes error information to be sent to the control terminal that is not presently receiving these messages instead of the present terminal. This message is only valid if the ALTERM operand was specified in the OPCTI macro instruction.

Identifier	Function	Operands
ctlmsg	SWITCH	

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTI macro instruction.

CHECKPOINTING AND RESTARTING THE MESSAGE CONTROL PROGRAM

QTAM provides optional checkpoint and restart facilities for the message control program. Checkpoint causes records to be written either at:

1. User-specified intervals, or
2. A certain point in one or more message processing programs

on a checkpoint records file maintained on a direct access storage device (DASD). These records contain the information necessary to record the status of the queues and the teleprocessing network. In particular, the checkpoint record includes the polling lists, the terminal table, disk pointers and status information associated with each queue, and disk pointers and status information associated with each line. Note that the data in the buffers is not included in the checkpoint record. Two such checkpoint records are maintained in the checkpoint file along with a pointer to the most recent record.

Should a system failure occur, the Restart facility uses the data in the checkpoint records to reinitialize the system to the status it had at the time the last checkpoint record was written except for audio lines. This avoids excessive loss of time or message data.

CHECKPOINTING THE MESSAGE CONTROL PROGRAM

In order to use the Checkpoint facility, the user must:

1. Allocate space on the DASD for the checkpoint records file. Space must be allocated on the DASD only the first time the file is used. To calculate the number of tracks required, refer to the section Direct Access Checkpoint Records File.

2. Preformat the area allocated for the file. He must preformat each extent with a dummy record four bytes long, the first byte of which must be zero. The checkpoint routine formats the other records needed.

   This formatting is necessary only the first time this area is used, not each time the message control program is initiated. However, if, because of a system failure, the file has not been closed and the user does not wish to restart, he must reinitialize (reformat) this file before initiating the message control program. An access method or utility other than QTAM must be used to format this area.

   Each extent formatted must have a format 1 label. The Clear Disk utility is recommended for formatting.

3. Define the checkpoint records file with a DTFQT macro instruction. Refer to the section Direct Access Checkpoint Records File.

4. Open and close the checkpoint records file. Refer to the description of the OPEN and CLOSE macro instructions.

5. Define the mode of checkpointing by either:

   a. Specifying the checkpoint interval in the TERMTBL macro instruction, or

b. Issuing CKREQ in the message processing program. (This macro instruction is described in the DOS QTAM Message Processing Program Services).

If the CPINTV keyword operand is specified in the TERMTBL macro instruction, the message control program will be checkpointed at the given interval of time. The first checkpoint interval is initiated at the time of execution the ENDREADY macro expansion. For further information, refer to TERMTBL Macro Instruction.

If CKREQ is issued in the message processing program, the message control program is checkpointed whenever all message processing partitions have issued the CKREQ macro instruction. For example, if there are two message processing partitions, and only one message processing partition has issued a CKREQ macro instruction, that partition enters a wait state until the other partition also issues a CKREQ macro instruction. At that time the checkpoint is taken. This puts all message processing partitions in synchronization with the message control program. Steps can be taken by the processing programs at the checkpoint to guard against duplicate messages following a restart.

CPINTV and CKREQ are mutually exclusive. If both are specified, CPINTV takes precedence. If the CPINTV operand is specified and a processing program issues a CKREQ macro instruction, the CKREQ macro instruction is ignored and the checkpoints are taken at the intervals specified in the CPINTV operand. An error code of X'08' is returned to the processing program in register 15.

RESTARTING THE MESSAGE CONTROL PROGRAM

When operating with the Checkpoint Restart facility, the user may restart the message control program at any time. Restart reestablishes the queues and the telecommunications network to the status it had at the time of the most recent checkpoint (except for audio lines, which will have the same status as in an initial program load). Restart is accomplished by reloading the program. The checkpoint records file is examined and, if the file has been properly closed, normal operation takes place. If the file has not been closed, a restart operation is performed.

After a restart has been performed, messages queued for sending to a terminal on a switched (dial) line are not automatically sent. The terminal must send a new message; after the message is received, the line is turned around and the messages on the destination queue are then sent.

Note: Upon restart, the checkpoint file should be opened before any processing by the message processing program is resumed.

SYSTEM DESIGN CONSIDERATIONS

- Checkpoint does not terminate incoming message traffic before taking the checkpoint record. Some of the messages at checkpoint time may therefore be partially received. If a system failure does occur, these partially recieved records, as well as the records received after checkpoint time, must be reentered from the terminal after restart.
- Messages on the DASD at restart that had not been completely sent to their destinations before the most recent checkpoint record was taken are sent, starting with the header segment whether or not the header segment had been sent before the checkpoint.
- Lines in initiate or conversational mode at checkpoint time are in normal mode upon restart.
- Lines stopped at checkpoint time remain stopped upon restart.

ON-LINE TERMINAL TESTING

The on-line terminal test facility provides tests that can be used by the terminal operator as a startup procedure, and by the IBM customer engineer for terminal checkout and diagnosis of terminal failure.

The tests provided operate on-line with the user's problem program and in no way hinder user operation except for the line time required by the terminal tests to perform their function on the selected line.

Tests requested from a terminal can be returned to that terminal, to any other terminal on the same line, or to any other terminal in the system. The tests allow message switching, comparison of incoming data to a stored pattern in core storage, all characters to be sent to the specified terminal, and test patterns for diagnosis

of failures in the SELECTRIC R typing element of the terminal.

Requests for the various tests are entered from a remote terminal and are identified by a test activation code of 99999. The individual tests and terminal addresses are selected by secondary activation codes.

Tests are not provided for non-IBM terminals, for terminals associated with audio response units, or for the IBM 2260-2848 Local.

The tests available are described in Appendix K.

ERROR RECOVERY PROCEDURES

The error recovery procedures are a comprehensive set of procedures for dealing with all kinds of input/output errors that may occur within the telecommunications system.

Upon occurrence of an input/output error, the error recovery procedures examine the sense bits and CSW status bits to determine which type of error has occurred. Depending upon the type of error, the following functions may be performed:

1. The failing action is retried two times, and on the third occurrence of the error, an operator message is provided,
2. The failing action is not retried, and an operator message is immediately provided,
3. The type of failing action is counted in a threshold counter, or
4. A combination of 1, 2, and 3, depending upon the type of terminal on which the error was detected.

   Note: if the error is counted in a threshold counter, it is counted every time that it occurs, including both of the two retry attempts. Messages to the operator are normally sent to the IBM 1052 system console. However, if the OPCTL=chars operand is specified in the TERMTBL macro instruction, they are sent to the operator control terminal. For further information, refer to the Operator Control Facility and TERMTBL Macro Instruction sections.

Eight counters are provided for each line in the system: four threshold counters and four cumulative counters. Each cumulative counter corresponds to one of the threshold counters. Threshold and cumulative counters are not provided for the IBM 2260-2848 Local.

The threshold counters keep a count of the number of:

1. Transmissions.
2. Data checks.
3. Intervention required errors.
4. Nontext timeouts.

Whenever any one of the three error threshold counters (but not the transmissions threshold counter) reaches a specified threshold value, the following action is performed:

1. A message is provided to the operator showing:
   a. The threshold value specified for each threshold counter, and
   b. The value in each threshold counter at this time. (The error count could exceed its threshold value if another error occurs after threshold is reached but before the message is written.)
2. The threshold counters are reset to 0.

Whenever the transmissions threshold counter reaches its threshold value, the threshold counters are reset to 0, but no message is provided.

The threshold values represent the number of the three types of errors considered excessive within a certain number of total transmissions. These values are specified in the THRESH operand under the DTF macro instruction for the line group in which the line is located. The threshold value for any of the four threshold counters must not be more than 255. However, the threshold values specified for any of the three error counters should be enough less than that specified for the number of transmissions to allow an error message to be provided. If the THRESH operand is omitted, the following values are assumed:

No. of transmissions: 255.
No. of data checks: 10.
No. of intervention required errors: 5.
No. of nontext timeouts: 5.

Error recovery procedures also provide the COPYC macro instruction. For further

information, refer to COPYC Macro Instruction.

OPERATOR AWARENESS

Operator awareness messages are provided for all permanent and unrecoverable errors and excessive temporary errors as determined by the line error threshold counters. These messages are printed on the 1052 system console unless operator control (OPCTL macro instruction) is specified. When operator control is specified, all operator awareness messages pertaining to the operation of the data links are sent to the telecommunications system control terminal. An operator awareness message will be truncated if it cannot be contained in one buffer. (See the DCS Operating Guide for specific messages.)

```
4QnnI text SYSnnn=cuu CCW=xxxxxxxxxxxxxxxx
TI=xxxx (or DC=xxxxxxxxxx)
CSW17=xxxxxxxxxxxxxx SN=xxyy LCB=xxxxxx
```

(When CBR/SDR is included in the system, the last line of this message is omitted.)

where:

4QnnI
: Is the standard message code for the operator. The internal component name is 4Q, the serial is nn, and the action code is I for informational (immediate operator action is not required).

text
: Type of error detected.

SYSnnn=cuu
: Is the symbolic unit assignment of the device, and cuu is the actual unit assignment of the device.

LCB=xxxxxx
: Is the address of the line control block for the terminal or of the Audio line control block for the Audio line.

TI=xxxx
: Is the terminal ID (polling or addressing characters) in hexadecimal format. If only one polling character is used, it will be left justified in this field. This field is not included in Audio messages.

DC=xxxxxxxxxx
: Is the hexadecimal representation of the dial characters for the terminal. Maximum number of digits is eight. This field is not included in Audio messages.

CSW17=xxxxxxxxxxxxxx
: Is bytes 1 through 7 of the channel status word as specified in the channel command block (CCB) in hexadecimal format.

CCW=xxxxxxxxxxxxxxxx
: Is the failing channel command word (CCW) in the channel program in hexadecimal format.

SN=xxyy
: For all terminals except 2740 Model 2
  - xx is the sense byte of the failing command in hexadecimal format.
  - yy is the sense byte of the error recovery CCW (if any) in hexadecimal format.

  For 2740 Model 2 terminal
  - xxyy is the hexadecimal representation of the 2-byte response received from the terminal (see the 2740 SRL for possible responses and their meanings).

The format of the error count threshold message is:

```
4Q00I LINE ERROR THRESHOLD REACHED
SYSnnn=cuu TR=aaa/bbb DC=ccc/ddd
IR=eee/fff TO=ggg/hhh
```

where:

TR=aaa/bbb
: aaa is the threshold value for the number of transmissions specified in the DTF/keyword THRESH, and bbb is the number of transmissions attempted up to the time an error threshold was reached. This information is in decimal format.

DC=ccc/ddd
: ccc is the threshold value for the number of data checks specified in the THRESH parameter, and ddd is the number of data checks that occurred, in decimal format, in the past bbb transmissions.

IR=eee/fff
: eee is the threshold value for the number of intervention required errors specified in the THRESH parameter, and fff is the number that occurred, in decimal format, in the past bbb transmissions.

TC=ggg/hhh
ggg is the threshold value for the number of nontext timeout errors specified in the THRESH parameter, and hhh is the number that occurred, in decimal format, in the past bbb transmissions.

CBR/SDR ERROR RECORDING

QTAM provides optional OBR (Outboard Recorder) and SDR (Statistical Data Recorder) facilities for recording error information. CBR/SDR helps to reduce the time the system is inoperative by providing more information for the diagnosis of line and terminal problems. OBR/SDR is not supported for audio devices.

The CBR facility writes one record on disk for each permanent error (exceptions -- time-out and intervention required on responses to polling and addressing, and time-out on a read test command; these are considered to be operational errors). The error recorded in an CBR record is the last error which occurred within a nonrecovering series of retries. Each OBR record contains the following information:

- Date
- Time
- Program ID (always shown as "QTAM MCP")
- First CCW
- Failing CCW
- Channel and unit
- CSW (key field will always be shown as X'30', regardless of its actual value)
- Sense data
- Device type
- "Polling characters"

  Actual data in this field will be:

  1. Polling and addressing characters for leased-line 1050, 1030, 1060, 2848, 2740C, 2740D, 83B3, and 115A terminals.
  2. Zeros for 2260-local, WTTA, 2740A, and 2740F terminals.
  3. Dial digits (up to a maximum of 8) for switched lines when the actual terminal is known.
  4. Zeros for switched lines when the terminal identity is not known by the program.

- Logical unit

The SDR facility maintains an 8-byte set of counters in main storage for each line or terminal (whenever the specific terminal identity is known to the program). A count of "total transmissions" is kept in a 1-byte counter, and counts of all the errors are kept in 1/2-byte counters. The following counters are kept:

For Devices on a 2701, 2, or 3	Byte-Halfbyte A=1st halfbyte B=2nd halfbyte
Total transmissions	0
Unit exception on a write	1-A
Time-out on prepare or nontext read	1-B
Time-out on dial, enable, or disable	2-A
Intervention required	2-B
Overrun	3-A
Bus-out check	3-B
Data check on write	4-B
Data check on read	5-A
Data check on poll	5-B
2740-2 terminal electronic error	6-A
2740-2 terminal I/O error	6-B
2740-2 transmitting parity error	7-A
2740-2 receiving parity error	7-B

For 2848-2260 Local	Byte-Halfbyte
Total transmissions	0
Bus-out check	3-B
Equipment check	6-A

Transmissions and errors are counted for SDR only during the first attempt at an I/O operation, not during any retries. When any counter reaches its maximum value (255 for total transmissions; 15 for errors), the entire 8-byte set of counters is added to a corresponding set of larger numbers on disk. The counters in main storage are then reset to zero.

Each SDR record on disk contains the following information:

- Channel and unit
- "Polling characters" (see CBR record for actual data in this field)
- Kind of device
- SDR counters

Note that there will be one SDR record on disk for each combination of line address and "polling characters" data used. For example, a switched terminal on line X'039' with phone number 1234 could have its transmissions and errors divided between two different SDR counter sets:

- line 039, "polling characters" 00000000 when the terminal's ID was not known to the program (counting by line)
- line 039, "polling characters" 12340000 when the terminals's ID is known (counting by terminal)

During normal closedown or cancel procedures, QTAM automatically adds all SDR counters in main storage to their corresponding disk counters.

If a permanent wait state occurs, the SDR counters in main storage are not added to their corresponding disk counters. These counts will be lost unless they are manually looked up in the dump of main storage. A table of SDR counter sets is generated at the label IJLQOTBL in the TERMTBL macro expansion. A 2-byte offset into the SDR counter table can be found in the TERM entry's IJLQTSDR field for counts by terminal, and in the last halfword of the LCB for counts by line. The first counter set has an offset of 8; zero is not used.

The OBR and SDR records are kept in a special recorder file (SYSREC) on disk, and are converted into printed form by a utility program, EREP, running independent of QTAM (for operating instructions see DOS System Control and System Service Programs, Form C25-5036).

To include the OBR/SDR option in the QTAM Message Control Program:

- Include the OBR/SDR option in the DOS system at system generation time;
- Specify the OBRSDR operand in the TERMTBL macro; and
- Include two extra bytes (for SDR counter assignment) in the ACLOC=integer operand of each line group DTFQT macro instruction.

The remainder of this document, except for the appendices, is concerned with programming for audio message control. All the information needed to control an audio-only system is provided in the following sections. In addition, all necessary considerations for programming for a mixed audio and nonaudio configuration are provided. Wherever possible and necessary, nonaudio considerations are included with the audio discussion wherever there is overlap. This section thus provides a complete documentation of programming for audio-only or mixed configurations.

AUDIO MESSAGE FORMATS

Audio messages include input and output messages that consist of a complete nonsegmented text, without any header.

Each line used for the transmission of audio messages can operate in one of the three following modes:

- Information mode. The computer receives no data, but sends an invariable audio output message (called an informational message), to the audio terminal connected on line.
- Inquiry mode. As soon as the calling terminal is connected on line, the computer either reads the input message, called the inquiry message, if the user has requested an initial read operation; or sends an invitational message to the calling terminal to signal that the computer is ready to read the inquiry message, if the user has requested an initial write operation. In both cases, the inquiry message is received by the computer, stored into an input buffer, and processed by the message processing program to produce the appropriate response message. From this response message, the message control program issues an audio output message which, through the audio response unit, becomes the audio answer received by the calling terminal.
- Conversation mode. This mode is an extension of the inquiry mode. It enables a sequence of inquiry messages and audio answers, called a conversation, to be set up within the same telephone communication.

Audio Input Messages

An audio input message consists of a series of alphameric characters which have been keyed or dialed on an audio terminal.

The input messages received by an audio response unit are transferred, character by character, to System/360 main storage and assembled in input buffers. The messages dialed on IBM 3944 Dial Terminals are directly assembled in EBCDIC representation; but all other messages are assembled in ARU code which is a 8-bit byte representation of the transmission code used by the audio terminals.

One input buffer is associated with each line. The length of an input message must exceed neither 255 bytes nor the length of the input buffer into which this message will be stored.

The end of an input message is indicated by one of the following:

- An EOT (end-of-transmission) character sent by the audio terminal.
- A timeout: a character has not been keyed or dialed in due time on the audio terminal.
- An overlength: the length of the input message exceeds the size of the corresponding input buffer.

The message control program automatically recognizes the end of a conversation or the cancellation of an inquiry at its beginning, when the first character of an input message is an EOT character or when a time out has occurred and no character has been sent by the audio terminal.

However, the first character of an input message can be recognized by the message control program as a code, if it has been specified as such by the user in an appropriate macro instruction. This code can be :

- A predetermined message-type indicator which identifies a message to be processed in a special way by the message control program. This indicator is passed to the message control program with the remainder of the text of the message.

- A repeat code, used in conversation mode only, which enables the last audio answer to be repeated. An invitational message cannot be repeated.

Audio Output Messages

The response message produced in a work area by the message processing program consists of an ordered sequence of addresses (address chain) specifying the location of each audio word to be sent to the calling terminal and preceded by two leading bytes indicating the length of this address chain.

One address chain buffer is associated with each line. The address chain is transferred to the address chain buffer, and its length must exceed neither 255 bytes nor the length of the address chain buffer.

When the line is attached to an IBM 7770, the address chain is made up of addresses on a magnetic drum from which this audio response unit fetches the words comprising the audio answer and sends them to the calling terminal. In this case, the address chain is identical to the audio output message sent by the computer.

When the line is attached to an IBM 7772, the address chain includes one or two types of DCV word address. The first type indicates the location of a DCV word on DASD. The second type indicates the relative position of a DCV word in a word table permanently kept in main storage. When at least one line attached to an IBM 7772 transmits DCV words dynamically retrieved from DASD, the user must define DCV buffers (DCV buffer pool); these DCV buffers will receive the DCV words from which the audio response unit will send the audio answer. In this case, the audio output message sent by the computer is made up of DCV words.

AUDIO MESSAGE FLOW

This section describes the flow of an audio message through a system operating under DOS QTAM from its receipt at the computer to the audio transmission to the calling terminal. Figure 23 illustrates this flow. When one of the numbers of the computer is dialed on an audio terminal, this terminal is connected to the computer if the corresponding audio line is enabled.

The input message is entered via the audio terminal.

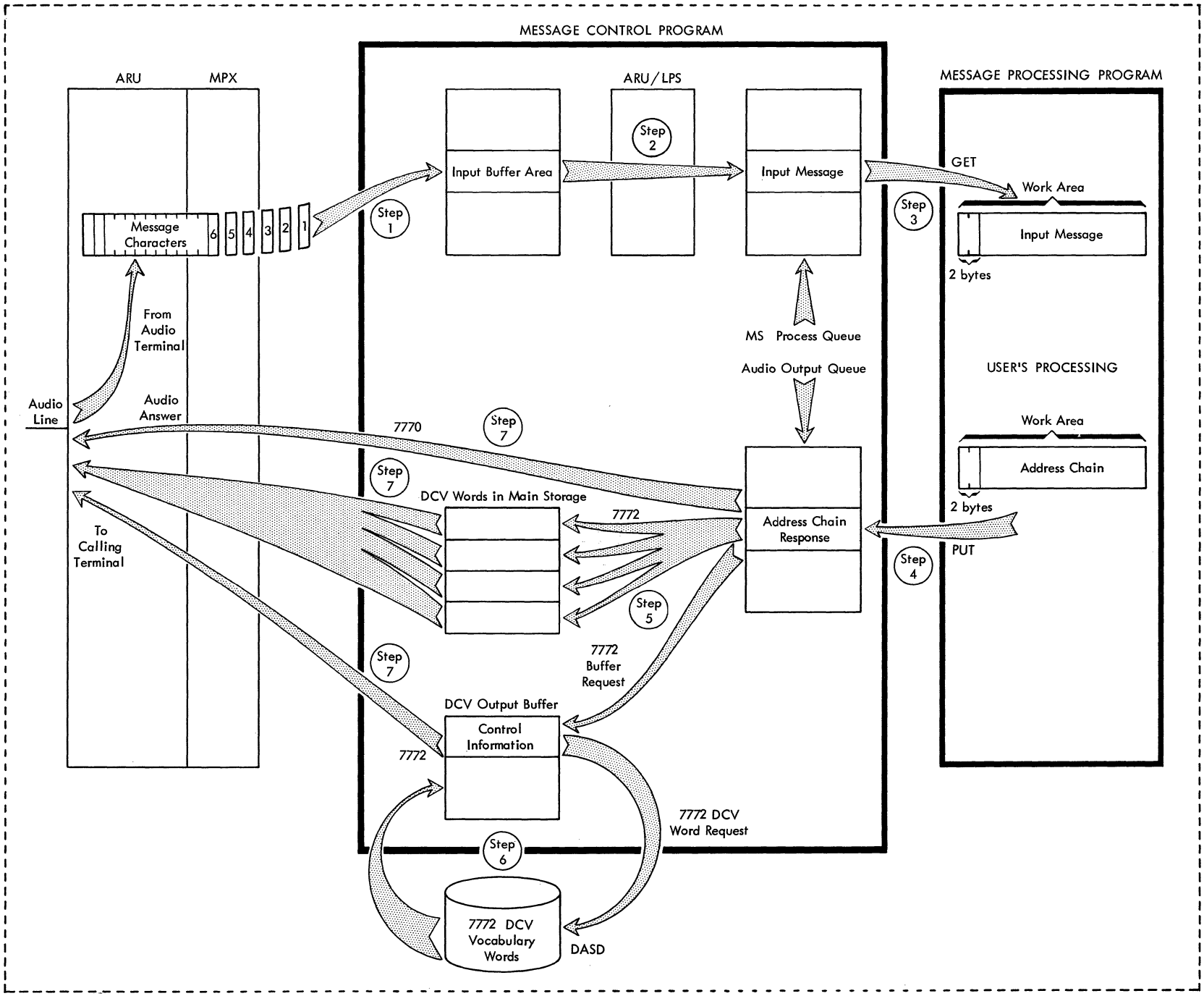


Figure 23. Audio Message Flow

The following are to explain the seven steps shown in Figure 23.

Step 1: The input message passes through an audio response unit (IBM 7770 or 7772) and the multiplexer channel, and is stored into an input buffer.

The user defines the sizes of his input and address chain buffers as parameters of the DTFQT macro instruction required for the communication line group. Thus, the corresponding buffer areas are reserved for each line of the line group.

Step 2: When the input buffer has been filled with the input message, an audio line procedure specification (LPS) performs user-selected functions such as message repetition, code conversion, logging with or without time-received information, and input message checking. Then, the LPS routes the input message to the MS-process queue. This routing is possible if a DTF table has been defined for the MS-process queue and opened by the message processing program; if not, the routing is deferred to the opening of the message processing program.

If the audio line group is in information mode, there is neither input message nor LPS; an interrupt signals the connection on line of the calling terminal which receives an invariable audio answer. Figure 5 does not illustrate this case.

Step 3: Whenever the message processing program issues a GET macro instruction, QTAM transfers the message from the MS-process queue to a user-specified work area in the message processing program. The length of the audio input message is placed in the first two bytes of this work area. Then, the message processing program processes the input message as required by the user's application.

Step 4: When the message processing program issues a PUT macro instruction, QTAM transfers the address chain from the user-specified work area to an address chain buffer. The user must specify the length of this address chain in the first two bytes of the work area, define and open in the message processing program a DTF table for the audio output queue to which the address chain is to be routed.

For the 7770 lines, message flow proceeds to step 7 since steps 5 and 6 only concern the 7772 lines.

Step 5: If the audio line transmits DCV words permanently kept in main storage, message flow bypasses step 5 and proceeds to step 6. If the audio line transmits DCV words dynamically retrieved from DASD, QTAM obtains a DCV buffer from a DCV buffer pool defined by the user.

Step 6: Each address in the address chain is analyzed to determine where a DCV word is to be fetched. If this DCV word is permanently kept in main storage, message flow proceeds directly to step 7. If not, the word is fetched from the 7772 DCV vocabulary file and loaded in the DCV buffer allocated to the line. Then, message flow proceeds to step 7.

Step 7: For the 7770 lines, the address chain enables the 7770 audio response unit to produce the audio answer. Message flow then returns to step 1.

For the 7772 lines, the DCV word is sent to the calling terminal. If this DCV word is the last word of the message, message flow returns to step 1. If not, the return is to step 6.

A telecommunications system operating under DOS/QTAM can be designed for a wide variety of applications including message switching, collecting data, standard audio answering, processing collected data, and inquiry processing. The audio applications are described here briefly.

MESSAGE CONTROL APPLICATIONS

One application for a message control program is to standard audio answering, which is only applicable to audio lines working in information mode.

When an audio terminal is connected to the CPU on a line working in information mode, there is no input message, but the connection itself represents an inquiry which requires an invariable audio answer. Therefore, the sending of this audio answer can be entirely accomplished within the message control program, but a message processing program must be loaded and initiated to terminate the execution of the message control program.

MESSAGE PROCESSING APPLICATIONS

Among a variety of message processing applications is that of inquiry processing.

An inquiry application involves receiving messages from terminals (performed by the message control program), processing the data contained in the messages (performed by the message processing program), and sending replies to the originating terminals (message control program).

The routines called by the message processing program to process the messages need not reside in main storage. For example, in an inquiry processing application that requires processing of many different types of inquiries, it may not be economical to have all of the required processing routines in main storage. The message processing program can contain an analysis routine that determines the type of the message and loads the routine required to process it from the core image library (via a FETCH or LOAD macro instruction). The routines fetched dynamically in this manner must have previously been linkage-edited onto the core image library at a specified address in an available area in the partition in which the message processing program is executing.

An optional feature of the inquiry application is operation in a conversational mode.

When an audio line is in conversational mode, the connection is maintained by the message control program during the successive inquiry/audio answer series. The end of a communication is notified either by an inquiry expressed by one EOT character or by no inquiry (timeout). Conversational mode is specified by a keyword operand of the DTFQT macro instruction associated with the audio communication line group. The input messages are routed by the audio LPS of the message control program to the MS process queue of the message processing program, if this program has been loaded and initiated. When the message processing program is not initiated, the input messages are queued in a waiting chain. In both cases, the ARU operand must be specified in the PROCESS macro instruction associated with this message processing program.

MESSAGE CONTROL PROGRAM (AUDIO LINES)

In every telecommunications system using QTAM, there must be one message control program. The message control program must be executed in the foreground 1 partition of the Disk Operating System. With multitasking, the message control program must be executed as the highest priority task in the foreground-one partition.

For audio lines, message control includes those functions that:

1. Control the flow of the audio messages between the computer and remote terminals.
2. Prepare the input messages for processing and route them to their appropriate message processing program.
3. Prepare the IBM 7772 output messages from the user's output messages and send them to the calling terminal.
4. Provide the user with statistical information relating to message traffic.
5. Provide the user with a copy of messages received from audio terminals.

The message control program includes both audio communication line QTAM routines and audio message handling QTAM routines. Input messages coming from audio terminals are translated into ARU code by the audio response unit before arriving at the computer. A facility is provided to convert this ARU code into EBCDIC representation to simplify message analysis. Such a conversion is not required for incoming messages from an IBM 3944 Dial Terminal attached to an IBM 7772 ARU with Dial Terminal feature, which are already in EBCDIC representation.

The input messages are normally sent to the MS process queue of the corresponding message processing program on a first-in-first-out basis.

To construct his message control program, the user must select, place into order, and assemble macro instructions provided by QTAM. There are four major sections in the message control program and each can be wholly defined by QTAM macro instructions. The four major sections and the order in which they will be discussed are:

1. File definition.
2. Control information.
3. File initialization and activation.
4. Line procedure specification.

File definition and control information macro instructions generate the tables, lists, and buffer areas needed in the system. Initialization and activation macro instructions ready the system for operation.

For audio lines, the LPS section is a minor part of the message control program which concerns only the audio input messages. It is here that the user specifies the LPS macro instructions establishing the linkage to QTAM routines that perform the message code translating, message repetition, error checking, logging, and routing functions.

The information specified in the file definition and control information sections is also used by the LPS routines in performing their functions. There must be one LPS for each communication line group that requires different message handling functions.

QTAM provides DSECTs (see Figure 24) that enable the user to refer symbolically to the various fields in the tables and control blocks used by QTAM. Two, three, or four such DSECTs are automatically generated in the message control program:

1. Terminal table entry DSECT (includes names assigned to an optional subfield by the user in OPTION macro instructions).
2. Audio line control block DSECT.

Note: Two DSECTs are generated when processing audio messages. Four DSECTs are generated when processing audio and nonaudio messages.

If any other DSECT listed in Figure 24 is desired in the message control program, it can be included at assembly time by including a COPY statement with the name in the specify column of Figure 24 in the operand field. If any DSECT is required in a message processing program, it can be included in the same manner (none of the DSECTs are automatically generated in a message processing program). It should be noted that all QTAM DSECT names have the four-character prefix, IJLQ; therefore,

theuser should refrain from using names beginning with these four characters.

Table or Control Block	DSECT Name	Specify (for COPY)
Terminal Table Entry	IJLQTBL0	IJLQTBLD
7772 DCV Buffer	IJLQBAB0	IJLQBABD
Audio Line Control Block	IJLQLAB0	IJLQLABD
Queue Control Block	IJLQQCB0	IJLQQCBD
DTF Table	IJLQDTF0	IJLQDTFD

Figure 24. QTAM DSECTS

FILE DEFINITION

A file definition macro instruction must be specified for each file referred to by the message control program. Four types of files are normally used for audio or combined systems:

- Communication line group files.
- Direct access checkpoint records file.
- Message log files.
- 7772 DCV vocabulary file.

COMMUNICATION LINE GROUP FILES

A communication line group file consists of messages transmitted via communication lines, or between the computer and a locally attached IBM 2260-2848 Display complex. One or more files of this type are required. The user must specify one DTFQT macro instruction to define a DTF table for each line group in the system.

For audio lines, a line group can consist of one to thirty-one 7770 or of one to eight 7772 audio lines with the following common characteristics:

- Association with the same functional type of audio terminals (for example, all the lines connect IBM 1001s and 1092s to the system; or, when an IBM 7772 is used, all the lines connect IBM 3944s to the system).
- Use of the same audio response unit (IBM 7770 or 7772).
- Use of the same message processing program, if any.
- Use of the same operating mode (information, inquiry, or conversation).
- Use of the same length for input buffers.
- Use of the same length for address chain buffers.
- Use of the same invariable audio answer when the line group is working in information mode (in this case, the address chain and input buffers are not required).
- Use of the same 7772 DCV buffer pool, if any.
- Use of the same Audio LPS.

One additional requirement is that no line within the line group can be defined as a part of another line group.

MESSAGE LOG FILE

A message log file consists of messages that are stored and maintained sequentially on secondary storage for accounting purposes. A message log can be produced as a byproduct of normal message handling. The appropriate DTFxx macro instruction must be specified for each message log required by the user. The DTFxx used depends on the secondary storage device employed. Magnetic tape (DTFMT) is the storage medium generally used. (For a detailed discussion of the appropriate DTFxx macro instruction, refer to the Supervisor and Input/Output Macros publication listed in the Preface.) The QTAM message control program employs logical IOCS to record the messages on the log.

7772 DCV VOCABULARY FILE

One 7772 DCV vocabulary file is required by all the 7772 Audio Response Units. This vocabulary is made up of all the DCV words, in any language, previously loaded on the IBM 2311 Disk Storage Drive used (for detailed information, refer to the Vocabulary File Utility Program for the IBM 7772 Audio Response Units publication

listed in the Preface). The user must specify one DTFQT macro instruction to define a DTF table for this DCV vocabulary file.

FILE DEFINITION MACRO INSTRUCTIONS

DTFQT Macro Instruction

One DTFQT macro instruction must be specified for the DASD message queues file, the 7772 DCV vocabulary file, and for each communication line group file. If the Checkpoint/Restart feature is desired, a DTFQT macro instruction must be provided for the DASD checkpoint records file. At assembly time, DTFQT causes the allocation of main storage for a DTF table. Parameters based on the keyword operands specified in the macro instruction are included in the DTF table.

Name	Operation	Operand
dtf	DTFQT	keyword operands

dtf
: Is the name of the macro instruction. It is also the name of the DTF table generated by the expansion of the macro instruction. The name is specified by from one to seven nonblank characters. The eighth byte of the name field is used by QTAM to indicate a 2311 or 2314 direct access device.

keyword operands
: Are the operands that can be included. The operands for each type of file are described in Figures 25 and 26. Operands for the message log files and checkpoint records that may be used in combined applications are described in figures 6 and 7.

Keyword Operand	Value Description
TYPE = AV	AV Identifies the file organization as that of the 7772 DCV vocabulary on an IBM 2311 or 2314.
DEVADDR = SYSnnn	SYSnnn If multiple volumes are used, only the symbolic unit for the first volume is specified. Actual units and channels are assigned to the file at job execution with appropriate ASSGN statements. The file is expanded to accomodate up to 16 extents.
[SEPASM = YES] [SEPASM = NO]	YES Specifies that this DTFQT will be assembled separately from the rest of the user's code. NO Specifies that this DTFQT will be assembled with the user's code. NO is assumed if this operand is omitted or illegal.
[WCRDTBL = YES] [WCRDTBL = NO]	YES Specifies that the user wishes to keep permanently some 7772 DCV words in main storage. NO Specifies there are no 7772 DCV words permanently kept in main storage. NO is assumed if this operand is omitted.
[BUFPL = (bufpool1,...)]	bufpool$_1$ Specifies via a sublist the symbolic names of the DCV buffer pools used by the 7772 line groups. A DCV buffer pool is required by a 7772 line group when messages sent over its lines are made up of DCV words dynamically retrieved from an IBM 2311 or 2314. The DCV buffer pool name must be identical to the name specified in the name field of the BUFARU macro instruction used to define the DCV buffers. This operand is omitted when no DCV words are retrieved from the IBM 2311 or 2314: in this case, the WCRDTBL operand must include YES.
[EOJAD = relexp]	relexp Is the name of the instruction starting a user-written section of code that closes all files opened in the message control program and performs other termination functions. Refer to the section, Deactivating the Telecommunications System, for a discussion of the functions required in this user-written section of code. Note: This operand will be omitted if it has already been specified in a DTFQT macro instruction for DASD message queues.

Figure 25. Keyword Operands for the 7772 DCV Vocabulary DTFQT Macro Instruction

Keyword Operand	Value Description
TYPE=LG	LG Identifies the file organization as that of a communication line group.
[CLPS=lpsname]	lpsname Is the name of the line procedure specification (LPS) section for this audio line group. This name must be identical to the name specified in the name field of the LPSTART macro instruction that begins the LPS section for this line group. This operand will be omitted for an audio line group working in information mode.
LINELST= (nnn-integer,...)	nnn Specifies via a sublist the correspondence between symbolic unit (SYSnnn) and relative line number. In the sublist, the user codes one 3-digit number for each line in the line group. The 3-digit number is interpreted as the "nnn" of SYSnnn. The order of coding the 3-digit numbers determines which symbolic units are associated with the individual lines in the line group. As many as thirty-one 3-digit numbers from 000-244 may be coded in the sublist. Example: LINELST=(005,010,007). This results in associating: SYS005 with relative line number 1, SYS010 with relative line number 2, SYS007 with relative line number 3, in a line group constituted by three lines.
[SEPASM=YES] [SEPASM=NO]	YES Specifies that this DTFQT is to be assembled separately from the rest of the user's code. NO Specifies that this DTFQT is to be assembled with the rest of the user's code.
[TYPEFLE={CMBND / CUTPUT}]	CMBND Indicates that the audio line group is to be used for both input and output operations. (Assumed, if necessary, when the line group is not working in information mode.) CUTPUT Indicates that the audio line group is to be used for output operations. (Assumed, if necessary, when the line group is working in information mode.)
CU={7770 / 7772}	7770 or 7772 Defines the type of audio response unit (7770 or 7772) associated with the audio line group.
[EUFIN=absexp]	absexp Is the length in bytes of each input buffer, which must not exceed 255 bytes. This operand will be omitted when the audio line group is working in information mode.

Figure 26. Keyword Operands for the Audio Communication Line Group DTFQT Macro Instruction (Part 1 of 3)

Keyword Operand	Value Description
[CMODE=INF] [CMODE=IQR] [CMODE=IQW] [CMODE=CVR] [CMODE=CVW]	INF Indicates that the audio line group will be used in information mode. IQR Indicates that the audio line group will be used in inquiry mode with initial read operation. IQW Indicates that the audio line group will be used in inquiry mode with initial write operation. CVR Indicates that the audio line group will be used in conversation mode with initial read operation. CVW Indicates that the audio line group will be used in conversation mode with initial write operation. Note: If this keyword operand is omitted, IQR is assumed.
[BUFAC=absexp]	absexp Is the length in bytes of each address chain buffer, which must not exceed 255 bytes. This operand will be omitted when the audio line group is working in information mode.
[DCVBUF=relexp] [DCVBUF=NO] (only applicable to the 7772 ARU's)	relexp Is the name of the DCV buffer pool specified in the name field of the BUFARU macro instruction. NO is assumed if this operand is omitted.
[PROCESS=relexp]	relexp Is the name specified in the name field of the PROCESS macro instruction used to refer to the message processing program that will process the input messages received from the lines of this line group. This operand will be omitted if the audio line group is working in information mode.
[PUTAC=relexp]	relexp Is the name or the absolute address of the address chain representing an invariable audio output message. This type of message is required for a line group working in information mode, or in inquiry or conversation mode with initial write operation. The user must define this address chain in the message control program, and place its length in its first two bytes. This length must not exceed 255 bytes. Refer to IBM System/360 Disk Operating System, QTAM Message Program Services, Form C30-5003, for complete specifications of audio address chains.
LOGTIME=YES LOGTIME=NO	YES Specifies that the input messages to be logged must contain a time-of-day indication. NO Specifies that the messages to be logged must not contain a time-of-day indication. Note: If the keyword operand is omitted or illegal, NO is assumed.

Figure 26. Keyword Operands for the Audio Communication Line Group DTFQT Macro instruction (Part 2 of 3)

Keyword Operand	Value Description
[EOJAD=relexp] (only applicable to the 7770 ARU's, when stand-alone)	relexp Is the name of the instruction starting a user-written section of code that closes all files opened in the message control program and performs other termination functions. Refer to the section, Deactivating the Telecommunications System, for a discussion of the functions required in this user-written section of code. Note: When the 7770 ARU's are stand-alone, there is neither DASD message queue file nor 7772 DCV vocabulary file, and this operand must be specified in the DTFQT macro instruction of each 7770 line group file.
THRESH=(absexp$_1$, absexp$_2$, absexp$_3$, absexp) THRESH=(255,255,5,5)	Provides the threshold values to be used in determining excessive number of errors (both temporary and permanent) for a specified number of transmissions for each line of this line group. absexp$_1$ Is the threshold value for the number of transmissions (must be from 1 to 255 inclusive). If the number of transmissions on any line in this line group reaches this threshold before any of the error counters reach their thresholds, the four threshold counters for this line are added to the four cumulative counters for this line, and the threshold counters are reset. absexp$_2$ Is the threshold value for the number of hang-up operations (must be from 1 to 255 inclusive). If the number of hang-up operations on any line in this line group reaches this threshold before the number of transmissions reaches its threshold value, a message is provided (to the 1052 system console or the telecommunication system control terminal if the OPCTL macro instruction is included), the four threshold counters for this line are added to the four cumulative counters for this line, and the threshold counters are reset. absexp$_3$ Is the threshold value for the number of data checks on read operations (must be from 1 to 255 inclusive). Same action is taken as in absexp$_2$. absexp Is the threshold value for the number of data checks on write operations (must be from 1 to 255 inclusive). Same action is taken as absexp$_2$. If this operand is omitted, the threshold values 255, 255, 5, 5 are assumed.

Figure 26. Keyword Operands for the Audio Communication Line Group DTFQT Macro Instruction (Part 3 of 3)

Examples:

1. A DTFQT macrc instruction that defines the DTF table representing an audio communication line group:

```
| Name  | Operaticn | Operand                                                    Col. 72 |
|-------|-----------|-------------------------------------------------------------------|
| ARU1  | DTFQT     | TYPE=LG,CLPS=ARULPS,PRCCESS=CPU,LINELST=(011,012),          *     |
|       |           | TYPEFLE=CMBND,CU=7772,CMODE=CVR,BUFIN=40,BUFAC=100                |
```

2. A DTFQT macro instruction that defines the DTF table representing a 7772 DCV vccabulary file:

```
| Name  | Operaticn | Operand                                                           |
|-------|-----------|-------------------------------------------------------------------|
| AUDIC | DTFQT     | TYPE=AV,DEVADDR=SYS010,WORDTBL=YES                                |
```

CONTRCI INFCRMATICN

In constructing the audio section of the message ccntrcl program for his telecommunications system, the user must provide certain ccntrol inforrnation. This data includes:

- A terminal table that contains all of the terminal ccdes as well as ccmplete information atcut the terminals ccnnected tc the system. These terminals include the message prccessing prcgrams, which are the only terminals to re specified fcr an audio applicatien.
- An audio line table that contains information about audio communication lines.
- A 7772 word table that contains inforrnatien atcut the DCV wcrds permanently kept in main stcrage.
- 7772 DCV buffer specificaticns that define the size and the maximum number of DCV buffers constituting a DCV buffer pccl.

The IBM-prcvided logic that supports the message control program uses this control infcrmaticn in perfcrming the message handling functions specified by the user. Macro instructicns are provided that allow the user to define the terminal table, the audio line table, the 7772 word table, polling lists, and buffer areas in accordance with the requirements of his application.

TERMINAL TABLE

A telecommunications system using QTAM requires one terminal table. For audio lines, the information required in the terminal table consists of a table control field defining the length of the table, and blocks of information (process program entries) about each message processing program that processes the audio messages. This field is described in Appendix A and in the following paragraphs.

For audio lines, the size, structure, and contents of the terminal table are based on information provided by the user through the TERMTEL, and PROCESS macro instructions.

- TERMTBL is specified once and defines the limits of the table.
- Process creates a process program entry.

Process Program Entry

A process program entry is required for each message processing program whatever the type of application:

- Audio application
- Combined application

Audio Applications: The terminal table must include one process program entry for each MS process queue. The input messages are directly routed to the MS process queue associated with the message processing

program, after it has been initiated (if not, they are queued on a waiting chain).

The structure of a process program entry is the following:

Field	Description
IJLQTSZE	Size of the entry.
IJLQTQAD	Pointer to the address of the queue control block for the MS process queue.
IJLQTTID	Name assigned to the process program entry.
IJLQTWCH	Address of the first element in the waiting chain.

Combined Applications: The same message processing program can process audio and nonaudio messages. In this case, only one process program entry must be included in the terminal table. The elements constituting this entry are:

IJLQTSZE
IJLQTQAD
IJLQTTID
IJLQTSCT
IJLQTSTA
IJLQTWCH

For the three types of application, the PROCESS macro instruction provides the initial contents for all fields in the process program entry. Detailed information on this entry can be found in Appendix A.

Terminal Table (TERMTBL) Macro Instruction

The TERMTBL macro instruction causes a table control field to be created for the terminal table and defines the length of the table. Depending upon the type of application, the expansion of this macro instruction also generates up to four DSECTs that enable the user's symbolic references to communicate with those of QTAM. One DSECT provides names for the fields in each terminal table entry, and another supplies names for the fields in an audio line control block (ALCB). QTAM maintains an LCB for each communication line, and an ALCB for each audio communication line attached to the system. Each ALCB contains control information about its associated line. The fourth DSECT provides names for the buffer prefix.

One TERMTBL macro instruction is required, and it must precede all other macro instructions used in creating the terminal table.

Name	Operation	Operand
	TERMTBL	entry[,n] [,OPCTL=chars] [,CPINTV=integer] [,CONF={ARU | MIXED}]

entry
Is the name of the last entry in the terminal table.

The entry name can be TERMTBL itself for compatibility when the audio response units are stand-alone and working in information mode only (no message processing program). In this case, n can be omitted.

n
Is the number of characters in the longest terminal name. This operand is not necessary if the lengths of all terminal names are the same.

OPCTL
The name of the OPCTL macro instruction in the LPS. This operand specifies that error messages are to be sent to the operator control terminal specified in the OPCTL macro instruction, rather than to the 1052 system console. When this operand is specified, error messages that originate from errors on the operator control terminal receiving error messages are printed on the 1052 system console. This operand must not be specified unless the OPCTL macro instruction is specified in the LPS. When this operand is not specified, the error messages continue to be sent to the 1052 system console.

CPINTV
The number of 15-second intervals between checkpoints. It must be an integer from 1 (15 seconds) to 60 (15 minutes) inclusive.

CONF
CONF=MIXED must be coded if there are both audio and nonaudio lines in the system.

CONF=ARU must be coded if there are only audio lines in the system.

This operand may not be omitted for audio or combined systems.

Note: The name field is ignored. IJLQTTBL is the name generated for

the terminal table by the macro instruction expansion.

Terminal Table Process (PROCESS) Macro Instruction

The PROCESS macro instruction causes the name and associated information of a DASD process queue (combined applications) or of an MS process queue (audio applications) to be included as an entry in the terminal table. The entry produced is a process program entry. It differs from other terminal table entries in that it does not have an optional area or device access area, the IJLQTSIN field is not used, and an additional waiting chain field is required by the audio response units. A message processing program, like a terminal, can be a destination for a message. However, unlike a terminal, the processing program is not associated with a communication line and does not need addressing and polling characters.

One PROCESS macro instruction must be included for each MS process queue that is defined by a DTFQT macro instruction in a message processing program.

For nonaudio lines, the EXPEDITE operand permits the user to speed the processing of messages by the message processing program associated with the process program entry. This function is valuable for an application requiring immediate transmission of data to a message processing program. A combination of the EXPEDITE function and conversational mode (subsequently discussed) provides the most rapid response to an inquiry.

For audio lines, the ARU operand permits the user to create an additional field in the process program entry (to queue the input messages waiting for the MS process queue chaining).

Name	Operation	Operand
symbol	PROCESS	[EXPEDITE][,ARU]

symbol
Is the name of the process program entry in the terminal table. The name must be specified, and must be the same as the PROCESS keyword operand specified in the MS-process queue DTFQT macro instruction in a message processing program. The name can contain from one to eight nonblank characters.

EXPEDITE
This optional operand (not applicable to the audio lines) specifies that messages are to be routed directly to the message processing program's MS-process queue, bypassing the normal intermediate step of placing the messages on a DASD-process queue. EXPEDITE should not be specified if multisegment messages are expected, because segments from different messages may be intermixed as they are delivered to the queues. This restriction does not apply for multisegment messages received from an IBM 2260 Local.

ARU
This operand must be specified when the process program entry is defined for a message processing program working on audio lines.

Note: If this operand is omitted, the process program entry is considered as defined for nonaudio lines.

AUDIO LINE TABLE

A telecommunications system with audio lines requires one audio line table. Two control information macro instructions produce, at assembly time, an audio line table tailored to the audio configuration desired by the user. The audio line table is made up of a table control field, which defines the length of the table, and of blocks of appropriate information for each audio line; these blocks are called audio line table entries, or line entries.

The size, structure, and contents of the audio line table are based on information provided by the user through two macro instructions, LINETBL and LINE. LINETBL is specified once, to define the limits of the table. LINE creates a line entry in the line table.

Each line entry is made up of four fields, whose contents are as follows:

- Size of the line entry
- Address of the DTF table for the audio line group in which the line is included
- Relative line number of the line within the audio line group
- Name assigned to the audio line by the user. This field is present in each line entry; its length is the same in each entry, and is based on information provided by the user. If the number of characters in each line name varies,

the number of bytes in this field is equivalent to the number of characters in the longest line name. If the number of characters in each line name does not vary, the number of bytes in this field is equivalent to the fixed number of characters. In any case, this field can be a maximum of eight bytes.

Audio Line Table (LINETBL) Macro Instruction

The LINETBL macro instruction creates the table control field of the line table, and defines the length of the table. This instruction must precede all the LINE macro instructions used to create the line table.

Name	Operation	Operand
	LINETBL	entry [,n]

entry
Is the name of the last entry in the line table.

n
Is the number of characters in the longest line name. This operand is not necessary when all line names have the same length.

Note: The name field is ignored. IJLQLTBL is the name generated for the line table by the expansion of the macro instruction.

Audio Line Table Entry (LINE) Macro Instruction

The LINE macro instruction includes a line name and its associated line information as an entry in the line table. One LINE macro instruction must be specified for each audio line and all LINE macro instructions must be grouped together.

Name	Operation	Operand
symbol	LINE	filename,rln

symbol
Is the line name, made up of one to eight nonblank characters; it must be specified. This name must also be present in the source or destination field defined by the user, when a GET or PUT macro instruction concerns a message using the audio line.

filename
Is the name of the DTF table for the audio line group in which the line is included.

rln
Is the relative line number of the line within the audio line group.

Example -- Audio Line Table Definition

Figure 27 shows the coding sequence to be used to create an audio line table. The audio terminals are IBM 1001's attached to an IBM 7770 ARU by four switched audio communication lines operating in inquiry mode.

No.	Name	Operation	Operand
1		LINETBL	LONDON,6
2	PARIS	LINE	LGONE,1
3	ROMA	LINE	LGONE,2
4	BERLIN	LINE	LGTWO,1
5	LONDON	LINE	LGTWO,2

Figure 27. Example of Coding Sequence Used to Create an Audio Line Table

Instruction 1 (LINETBL): Identifies the last entry in the audio line table. The second operand specifies the length of the longest line name.

Instructions 2 through 5 (LINE): Defines the line entries for the lines in two audio line groups. The first operand of each LINE macro instruction specifies the name of the DTF table for the audio line group in which the line is included. In this case, there are two audio line groups and two DTF tables. The second operand of each LINE macro instruction is the relative line number of the line within its audio line group. In the audio line group associated with the DTF table named LGONE, PARIS and ROMA audio lines have the line numbers 1 and 2, respectively. These relative line numbers correspond to those given to the ALCB's generated in the expansion of the DTFQT macro instruction with LINELST operand. The user assigns each audio line at system generation (this assignment can be modified by an ASSGN statement).

7772 WORD TABLE

A telecommunications system with at least one IBM 7772 ARU using DCV words permanently kept in main storage requires one 7772 word table. Two control information macro instructions produce, at assembly time, a 7772 word table tailored to the DCV words specified by the user. The 7772 word table is made up of a table control field, which defines the length of the table, and of blocks of information for each DCV word permanently kept in main storage. Each block is called a 7772 word table entry, or word entry.

The size, structure, and contents of the word table are based on information provided by the user through two macro instructions, WORDTBL and WORD. WORDTBL is specified once, to define the limits of the table. WORD creates a word entry in the word table. The word table must contain a word entry for each DCV word required in main storage.

Each word entry is made up of four fields, whose contents are as follows:

- Disk address of the DCV word representation: this address is expressed in the form TTR, where TT is the track number and R the record number.
- Size of the word entry
- Length of the DCV word representation
- DCV word field: an area in which the DCV word representation is read from the 7772 DCV vocabulary file.

7772 Word Table (WORDTBL) Macro Instruction

The WORDTBL macro instruction creates the table control field of the 7772 word table, and defines the length of this table. When the WORDTBL macro instruction is specified, it must precede all the WORD macro instructions used to create the word table.

Name	Operation	Operand
	WORDTBL	entry

entry
Is the name of the last entry in the word table.

Note: The name field is ignored. IJLQWTBL is the name generated for the word table by the expansion of the macro instruction.

7772 Word Table Entry (WORD) Macro Instruction

The WORD macro instruction creates an entry in the word table which requires a DCV word representation to be included in the DCV word field of this entry. The DCV word representation is extracted from the 7772 DCV vocabulary file, and read in the DCV word field at open time of this file. One WORD macro instruction must be specified by the user for each DCV word required in main storage. All WORD macro instructions must be grouped together.

Name	Operation	Operand
[symbol]	WORD	diskaddr-X'hex',lgth

symbol
Is the word name, consisting of one to eight nonblank characters. It must be specified for the last WORD macro instruction used to create the word table.

diskaddr
Is the disk address of the DCV word representation. It must be written in hexadecimal notation.

length
Is the length of the DCV word representation.

Note: The values of "diskaddr" and "length" are indicated in the listing of the DCV operative vocabulary file, which results from the execution of the OVF list function of the vocabulary file utility program (see Vocabulary File Utility Program for the IBM 7772 Audio Response Units publication listed in the Preface).

Example - 7772 Word Table Definition

Figure 28 shows the coding sequence to be used to create a 7772 word table.

No.	Name	Operation	Operand
1		WCRDTBL	WORD3
2	WORD1	WORD	X'004801',240
3	WORD2	WORD	X'005306',384
4	WORD3	WORD	X'01020B',127

Figure 28. Example of Coding Sequence Used to Create a 7772 Word Table

Instruction 1 (WCRDTBL): Identifies the last entry in the word table, used by the opening routine of the 7772 DCV vocabulary file to load the DCV word representations in main storage.

Instructions 2 through 4 (WORD): Define the word entries for three DCV words required in main storage.

AUDIO BUFFER SPECIFICATION

An audio application may require three types of buffer:

- Input buffers to store the input messages.
- Address chain buffers to store the chains of addresses corresponding to the words which constitute the audio output messages.
- Output buffers to store the DCV representations of each word dynamically retrieved from the 7772 vocabulary file, as required by the address chain content.

The input and address chain buffers are specified through the keyword operands BUFIN and BUFAC of the DTFQT macro instruction defined for the audio line group.

When at least one audio line attached to an IBM 7772 ARU uses DCV words dynamically retrieved from the 7772 DCV vocabulary file, the user must specify the size and number of main storage areas required by QTAM for output buffering. This information is provided through the BUFARU macro instruction in the message control program . The main storage areas defined above form a DCV buffer pool dynamically used by QTAM.

All buffers in a DCV buffer pool have the same length. This length must be at least equal to twice the length of the longest DCV representation of the words to be dynamically retrieved from the 7772 DCV vocabulary file.

The minimum number of buffers in a DCV buffer pool is one. The optimum number is equal to the number of lines likely to be simultaneously used for output transmission (refer to the IBM System/360 Component Description, IBM 7772 Audio Response Unit publication listed in the Preface). The number cannot exceed eight (the maximum number of lines).

BUFARU Macro Instruction

This macro instruction is only applicable to the 7772 audio lines.

A BUFARU macro instruction enables the user to define a DCV buffer pool for one or several line groups associated with an IBM 7772 ARU to be specified, when required by QTAM for an audio application.

Name	Operation	Operand
symbol	BUFARU	n,length

symbol
: Is the name of the DCV buffer pool. This name is always required and must appear in the DCVBUF keyword operand of the DTFQT macro instructions defined for the line groups associated with this ARU, when the messages transmitted on their lines are made up of DCV words to be dynamically retrieved from the 7772 DCV vocabulary file.

n
: Is the number of DCV buffers to be reserved.

length
: Is the length, in bytes, of each DCV buffer. This length must be equal to twice the length of the longest DCV word to be retrieved from the 7772 DCV vocabulary file.

FILE INITIALIZATION AND ACTIVATION

The file initialization and activation section of the message control program begins with an OPEN macro instruction and ends with the ENDREADY macro instruction. Within the message control program, this section must precede the LPS section.

whenthe instructicns in this section have been executed, the system is ready to handle message traffic.

The CPEN macro instruction completes the initialization for and activation of the DASD message queues, 7772 DCV vocabulary, communication line group, message log, and checkpoint records files. The files used by the message control program can be opened by separate CPEN macro instructions, or they can all be opened with one OPEN. However, regardless of which method is used, the user must open the DASD message queues file, if any, before any other file used by QTAM. If there is no DASD message queue file, but there is a 7772 DCV vocabulary file, the latter must be opened before any other file used by QTAM, and always before the communication line group files. Opening a communication line group file causes all lines in the line group to be prepared for operation. The lines are activated automatically for message transmission.

The ENDREADY macro instruction must be the last instruction in the file initialization and activation section. When ENDREADY has been executed, the system is ready to handle message traffic. The expansion of this macro instruction causes a branch to the IBM-provided logic that supports the message control program, where procurement of the first message is awaited (for nonaudio lines, the first message procured can be either a message coming in from a terminal, or a message being sent to a terminal by a message processing program; for audio lines, the awaited message is always a message coming in from an audio terminal. When the first message is procured, control is returned to the LPS section of the message control program for handling of the message. For this reason, no executable code may be included between ENDREADY and the LPSTART macro instruction that begins the LPS section.

Once the LPS is initially entered via the expansion of the ENDREADY macro instruction, execution in the message control program is restricted to the LPS section; that is, the LPS is continually reentered to handle messages entering and, except for the audio messages, leaving the computer as long as the message control program is active. Deactivation of the message control program and the telecommunications system is accomplished by closing the message control program files. The section, Deactivating the Telecommunications System, contains a discussion of the procedures necessary to terminate the message control program.

For audio lines, the STARTARU and STOPARU macro instructions may be used in the initialization and activation section, in the user-written routines of the LPS section, or in the message processing program.

OPEN Macro Instruction

OPEN is used in the message control program to complete the initialization and activation of the DASD message queues, 7772 DCV vocabulary, communication line group, message log, and checkpoint records files. All these files can be opened separately or with one OPEN. However, the user must open the DASD message queues file, if any, before any other file used by QTAM. If the Checkpoint Restart feature is used, the DASD checkpoint records file must be opened after the DASD message queues file: If there is no DASD message queues file, but there is a 7772 DCV vocabulary file, this file must be opened before any other file used by QTAM and always before any communication line group file. The user specifies, as the operand(s) of the OPEN macro instruction, the name(s) of the DTF table(s) for the file(s).

If the DTF table for a 7772 DCV vocabulary file is specified, the OPEN routine completes the initialization for use of all the DCV buffer pools, and loads the DCV words required in main storage if a 7772 word table is present in the message control program.

If the DTF table for a communications line group is specified, the OPEN routine completes the initialization for all lines in the line group and automatically activates the lines for message transmission. For the audio lines, enabling is initiated. If a line does not have an active polling list with terminal entries, or if TYPEFLE=OUTPUT is specified, polling is not initiated. If the OPEN is for a switched line group with TYPEFLE=INPUT or CMBND, the OPEN routine issues commands to enable each access line in the line group.

If the Checkpoint/Restart feature is used, the status of audio lines is not restored to that of the previous checkpoint, since a restart for audio lines is the same as an initial program load.

The formats for the message queues file and for the checkpoint records file are checked when the particular file is opened. If the file is formatted incorrectly, an error message is provided, and the job is cancelled.

If a message log DTF table is specified, initialization is completed for placement of messages on the device to be used for logging.

Name	Operation	Operand
[symbol]	OPEN	filename,...

symbol
Is the name of the macro instruction. The name is optional.

filename
Specifies the name of the DTF table associated with the file being opened. Several files may be opened with one OPEN by entering their DTF names as operands.

If register notation is used, any general register in the range 2 through 12 may be specified. The programmer must load the address in the register before issuing the OPEN.

Note 1. Several files may be opened with one OPEN.

Note 2: It is permissible to intermix register notation and relocatable expressions in the same OPEN. For example:

Name	Operation	Operand
OPENER	OPEN	DISKFILE,(3),(4),LOG

Note (only applicable to audio lines): Assuming the DTF table of the DASD message queue file is named DISKFILE, and the DTF table of the 7772 DCV vocabulary file is named VOCFILE, the opening sequence of all the files must be one of the following:

- If DISKFILE and VOCFILE are specified:

Name	Operation	Operand
	OPEN	DISKFILE,VOCFILE, INGRCUP1,...,INGROUPN, LOGFILE

- If only VOCFILE is specified:

Name	Operation	Operand
	OPEN	VOCFILE,INGRCUP1,..., INGRCUPN,LOGFILE

- If neither DISKFILE nor VOCFILE is specified:

Name	Operation	Operand
	OPEN	LNGROUP1,...,LNGROUPN, LOGFILE

ENDREADY Macro Instruction

The file initialization and activation section must be ended by an ENDREADY macro instruction. ENDREADY is essentially a wait-type instruction; the event awaited is the procurement of the first message. Only one ENDREADY macro instruction can be included, and it must be the last in the group of file initialization/activation instructions. Prior to issuing ENDREADY, the user must ensure that register 13 contains the address of an 18-word save area. ENDREADY saves the user's registers in this area by standard register saving conventions. When control returns to the user at the address specified in the EOJAD keyword operand of the DTFQT for the first file opened in the message control program, the registers (except for register 14) are restored. No operand is required.

Name	Operation	Operand
	ENDREADY	

AUDIO LINE PROCEDURE SPECIFICATION

The procedure followed by a message control program in operating upon messages being received from or, except for audio lines, sent to terminals is defined by a line procedure specification (LPS). An LPS is a sequence of IBM-provided statements called LPS macro instructions. The user must provide an LPS for each communication line group in his system. However, one LPS may be used by more than one line group if each of the groups requires identical message control procedures. For audio lines, the LPS (or Audio LPS) is only concerned with audio input messages.

For audio lines, the purpose of the LPS is to define macro-introduced routines that:

1. Examine and process control information, if any, contained in the first byte of the input message.

2. Perform functions necessary to send the input message to the message processing program.

Preparing an LPS consists primarily of selecting the appropriate IBM-provided macro instructions and writing them in a particular sequence, according to the requirements of the installation and of the line group. Considerations such as the message header format(s), the processing requirements for various types of messages (if messages having different handling requirements are directed to the same LPS), the type of terminal in the line group, and whether the LPS is operating on messages from switched or nonswitched lines must be carefully analyzed.

Two major types of macro instructions are used in the LPS: functional macro instructions and delimiter macro instructions. In general, the functional macro instructions perform the specific operations required on messages directed to the LPS. Delimiter macro instructions for audio lines identify a sequence of functional macro instructions.

COMPONENTS OF THE AUDIO LPS

The Audio LPS consists of one group of macro instructions handling the audio input messages. Within this group, the user can build different subgroups of coding sequence by using an ARUMGTYP macro instruction. This macro instruction examines the message-type indicator in the first byte of the input message, and directs this message to the appropriate subgroup; the user can include his own routines in this subgroup to perform special functions in place of or in addition to the functions provided by QTAM macro instructions. It should also be noted that any user code within the ARU/LPS should be re-entrant. An Audio LPS must contain at least the LPSTART and POSTARU delimiter macro instructions to link the Audio LPS with the IBM-provided logic that supports the message control program. If all the audio lines are operating in information mode, no Audio LPS is required.

The following shows the different macro instructions which can be included in the Audio LPS. The functional macro instructions are listed in alphabetical order.

Delimiter	Functional	Remarks
LPSTART		required
	ARUMGTYP CHECKARU LOGSEG REPEAT TRANS	
POSTARU		required

There are two major types of Audio LPS macro instructions: delimiter and functional macro instructions. The user is cautioned against transferring control between macro instructions within the LPS. A user-written branch to a macro instruction may require that the user perform certain functions (such as register saving and restoring) which are normally provided by QTAM. Since user-written branches are the exception rather than the rule, the name fields in the macro instruction formats for the audio LPS macro instructions (with the exception of LPSTART) have been omitted. The user may supply a name with an equate statement if a name is needed.

DELIMITER MACRO INSTRUCTIONS (AUDIO LINES)

Line Procedure Specification Start (LPSTART) Macro Instruction

The LPSTART macro instruction provides an initialization procedure for the Audio LPS. LPSTART is required and must be the first macro instruction in every Audio LPS.

Name	Operation	Operand
lpsname	LPSTART	TERM=(termcode$_{1}$,...)

lpsname
Is the name of the macro instruction and is required. It must be the same as lpsname specified in the CLPS keyword operand of the DTFQT macro instruction for the line group.

termcode$_{1}$,...
The identifying code for the types of terminals that utilize this LPS. The following values can be included in the sublist:

7770=IBM 7770 audio response units;

7772=IBM 7772 audio response units.

Post Audio LPS (PCSTARU) Macro Instruction

The PCSTARU macro instruction identifies the end of the sequence of instructions which processes the input messages coming in from the audio terminals.

One PCSTARU macro instruction is required in each Audio LPS, in which it must be the last instruction. No operand is required.

Operation	Operand
PCSTARU	

FUNCTIONAL MACRO INSTRUCTION DESCRIPTIONS (AUDIO LINES)

The functional macro instructions available for audio lines perform specific functions on input messages. The macro instructions and their fuctions are:

ARUMGTYP -Routes input messages to a user-specified sequence of instructions.

CHECKARU -Checks for errors in message reception and, if found, sends the user an error message.

LOGSEG -Maintains logs of input messages on an auxiliary storage device (and inserts times).

REPEAT -Checks for a request for a repetition of the previous audio answer.

TRANS -Translates input messages.

The following functional macro instructions are applicable only to audio lines. No other QTAM macro instructions may be coded for audio lines.

ARU Message Type (ARUMGTYP) Macro Instruction

The ARUMGTYP macro instruction classifies incoming messages into types depending on the processing required. The first character of an input message is compared with a type character specified in the operand of the ARUMGTYP macro instruction. If the two characters are the same, the user-written instructions placed between the ARUMGTYP macro instruction and the next ARUMGTYP macro instruction are executed. If the two characters are different, the user-written instructions are not executed.

The last ARUMGTYP macro instruction in the audio LPS is coded with no operand. The user-written instructions between this ARUMGTYP macro instruction and the POSTARU delimiter are executed if the first character of the input message is not the same as the message-type character operand of a previous ARUMGTYP macro instruction. If the character is the same, the message has already been handled by a previous ARUMGTYP macro instruction and the user instructions between the final ARUMGTYP and the POSTARU macro instructions are not executed.

The ARUMGTYP macro instruction may be used as frequently as desired or may be omitted.

Operation	Operand
ARUMGTYP	typchar

typechar
Is the message-type character which may be any nonblank character. This operand must be omitted if the ARUMGTYP macro instruction which contains it is the last ARUMGTYP macro instruction; in this case, the following macro instructions or user instructions process any input message which has not been handled by a previous ARUMGTYP macro instruction with a nonblank operand. The message-type character may be specified either as the character itself or as the hexadecimal representation of the character. The framing C or X and quotes must be coded.

Example: An Audio LPS using ARUMGTYP macro instruction is shown in figure 30.

Name	Operation	Operand	Comments
LPSARU	LPSTART	TERM=(7770)	Delimiter
	TRANS REPEAT	RCVARU X'FO'	Macro instructions executed for all input messages
	ARUMGTYP	C'A'	Test for A-type messages
	- - -		LOGSEG or CHECKARU macro instructions, or user-written routines executed for all A-type messages
	ARUMGTYP	C'B'	Test for B-type messages
	-		LOGSEG or CHECKARU macro instructions, or user-written routines executed for all B-type messages
	ARUMGTYP		Test for all other types of messages
	- - -		LOGSEG or CHECKARU macro instructions, or user-written routines executed for all other types of messages
	POSTARU		Delimiter

Figure 29. Use of ARUMGTYP Macro Instructions in an Audio LPS

Audio Error Message (CHECKARU) Macro Instruction

The CHECKARU macro instruction sends an audio error message to the audio terminal on the line, when one of the errors specified in the error mask has been found in the error byte.

The user must specify in the CHECKARU macro instruction:

- The bit configuration of the mask used to test the error byte associated with the audio line
- The symbolic address of the address chain representing the user's error message, or the address chain itself

Each bit in the line error byte is set on and takes the value 1 whenever an error has been detected. It is set off and takes the value 0 after an audio answer has been sent on the line to the calling terminal. The user can set bits 4 through 7 (user-reserved bits) by using any user-written routine which performs a check specific to the awaited input message (see the section Including a User-written Routine within the LPS or the Audio LPS). Figure 30 summarizes the functions of the line error byte.

The length of the address chain representing the user's error message must exceed neither 255 bytes nor the length of the address chain buffer defined for the audio line. Refer to IBM System/360 Disk Operating System QTAM Message Processing Program Services, Form C30-5003, for a complete specification of audio address chains.

Bit	Function	Explanation
0	Overlength message	An audio terminal has attempted to send an incoming message whose length exceeds the length of the input buffer area.
1	Repeat error	A repeat code is the first character in an input message which is the first message in a conversation.
2	Read error	An error has been detected in a read operation.
3		Reserved for IBM use only.
4-7		The four user-reserved bits which can be set on (value 1) and tested in any user-written routine.

Figure 30. Communication Line Error Byte (audio lines)

The CHECKARU macro instruction may be issued anywhere in the Audio IPS and used as frequently as desired.

Operation	Operand
CHECKARU	mask, {=X'message'} {msgchar}

mask
Is the hexadecimal representation of the bit configuration of the error byte attached to an audio line.

message
Is the address chain representing the user's audio error message. The framing X and quotes must be coded. When this operand is used, the length of the address chain, including the length representation, must not exceed 61 bytes. The first two bytes of the address chain must contain the length of the chain.

msgchar
Is the address of the area defined by the user which contains the address chain representing the user's audio error message. The first two bytes of the address chain must contain the length of the chain.

Logging (LOGSEG) Macro Instruction

For audio messages, the LOGSEG macro instruction with the ARU operand enables the user to log input messages (to place them on an output device as a record of the input message traffic carried by the audio lines). The input messages are logged in the sequence in which they are received preceded by the symbolic assignment of the receiving audio line (3 bytes), the length of the input message (3 bytes), and the date stamp information (8 bytes). If the input message is coming in via an audio line of a line group defined by a DTFQT macro instruction with the LOGTIME=YES operand, automatic time stamping will be performed in an area reserved via this LOGTIME operand. In this case, the logged input message includes the symbolic assignment of the audio line (3 bytes), the length of the input message (3 bytes), the date stamped information (8 bytes), the time stamped information (8 bytes), and the input message itself. In either case, these successive fields are separated by a one-byte separator (minus sign). Note that the unused part of the input buffer is reset to X'00'.

The LOGSEG macro instruction with the ARU operand may appear in any place and more than once in the Audio LPS, if the ARUMGTYP macro instruction is used to define different types of input messages.

The logging effected by LOGSEG is in addition to the logging associated with the queueing procedure of QTAM. Use of LOGSEG is optional.

Operation	Operand
LOGSEG	filename [,PREFIX ,ARU]

filename
Is the name of the DTF table that the user must define to specify the parameters of the file used for logging the audio input messages. TheDTFxx macro instruction for the

file on which the messages are logged must specify WORKAREA=YES.

(1) may be used to specify that the address of the DTF table is in parameter register 1. The address must be loaded into register 1 prior to execution of this macro instruction.

PREFIX
Specifies that the QTAM header or text prefix is to be included with the logged segment. If this operand is not coded, the prefix is not logged. The first 8 bytes of the prefix are not included. Thus a header segment is logged with a 24-byte prefix and a text segment is logged with a 14-byte prefix. The format of the QTAM prefixes is contained in Appendix A.

ARU
Specifies that the message to be logged is an audio input message. This operand is required for each ICGSEG macro instruction issued in an Audio LPS.

Audio Answer Repetition (REPEAT) Macro Instruction

The REPEAT macro instruction can only be used in the Audio LPS of line groups operating in conversation mode. It enables the user to obtain a repetition of the previous audio answer, if any. (Conversation mode may be specified in the CMODE operand of the DTFQT macro instruction for the line group.)

The first character of the input message is compared with the character specified in the operand of the REPEAT macro instruction. If these two characters are different, the REPEAT macro instruction produces no effect on the input message. If these two characters are the same, or if there is no operand in the REPEAT macro instruction, the input message is tested to determine whether it is the first input message in the conversation (the repetition request cannot be considered as the first input message). If the input message is the first input message in the conversation, the repeat bit in the line error byte is set on by the REPEAT macro instruction. In this case, if the user tests the repeat bit by using a CHECKARU macro instruction, an error message is sent to the calling terminal. If the repeat bit is not tested, the REPEAT macro instruction has no effect on the input message. If the input message is not the first input message in the conversation, the REPEAT macro instruction results in repetition of the last audio answer.

The REPEAT macro instruction may be issued more than once, anywhere in the Audio LPS, depending on the use of the ARUMGTYP macro instruction.

Operation	Operand
REPEAT	[codechar]

codechar
Is the repeat code, and may be any character. It must be specified as either the character itself or the hexadecimal representation of this character. The framing C or X and quotes must be coded.

Translate (TRANS) Macro Instruction

For audio messages, the TRANS macro instruction results in the translation of the input messages, character by character, from ARU code into EBCDIC representation (the input messages are in ARU code when they are transferred to main storage). The TRANS macro instruction must not be used for the messages coming in from IBM 3944 Dial Terminals via a 7772 ARU with Dial Terminal Features, because these messages are already in EBCDIC representation.

Code translation is accomplished either through a RCVARU table provided by QTAM, or through a user provided table. The user may omit the TRANS macro instruction and work with ARU code; in this case, the operands of the ARUMGTYP, CHECKARU, or REPEAT macro instructions must be specified in ARU code representation.

The TRANS macro instruction, when used, must precede any ARUMGTYP, CHECKARU, or REPEAT macro instruction.

Operation	Operand
TRANS	table

table
Is the name of the code translation table. It may be RCVARU or the name of a user-provided table

Example: A TRANS macro instruction to translate input messages sent from an audio terminal other than an IBM 3944 Dial Terminal to the computer:

Operation	Operand
TRANS	RCVARU

INCLUDING A USER-WRITTEN ROUTINE WITHIN THE AUDIC LPS

The design of an LPS section is such that a serially-reusable, user-written routine can be included. Linkage to a closed user-written routine can be included in any subgroup within the Audic LPS. There are several reasons why the user might want to include such a routine.

For audio messages: The user may wish to process particular information included in the input messages and change the contents of these messages. Or, he may desire to execute his own error-checking routines on the input messages (for example, checking the length and nature of the contents). If the length or the nature of the information is not in accordance with the length or the nature of the expected input messages, the user can set on (value 1) a user-reserved bit in the line-error byte. By testing this bit with a subsequent CHECKARU macro instruction, the user may issue an error message to the calling terminal. The user may have access to the following information contained in each audio line control block (ALCB) labeled:

- IJLQLIML input message length (half-word)
- IJLQLIBA input message address (full-word)
- IJLQLERB line-error byte

Before passing control to the LPS, QTAM places the address of the ALCB associated with the line involved in register 4. The expansion of the LPSTART macro instruction establishes this register as the base for the ALCB DSECT generated by the expansion of the TERMTBL macro instruction.

For all messages: To include a user-written, closed routine, the user must provide his own linkages. QTAM requires that the user-written routine save and restore the registers by standard register saving conventions. Register 13 contains the address of a QTAM-provided save area to be used for this purpose. If the user-written routine calls a second user-written routine, it must provide its own save area, etc. Figure 31 shows the control flow between an LPS and a closed, user-written routine.

Before entering the LPS section, the LPS control routine initializes certain registers with data that is used by many of the LPS macro instructions in performing their functions. For this reason, the registers must be preserved. The register contents are described in Appendix D and may be useful to the user in coding his own LPS routine. For example, if the user-written routine processes a field in the nonaudio message header, the scan pointer register (register 5) may be used for scanning the field; if used, it must be updated to point to the end of the field (as described in Appendix D) upon return to the LPS.

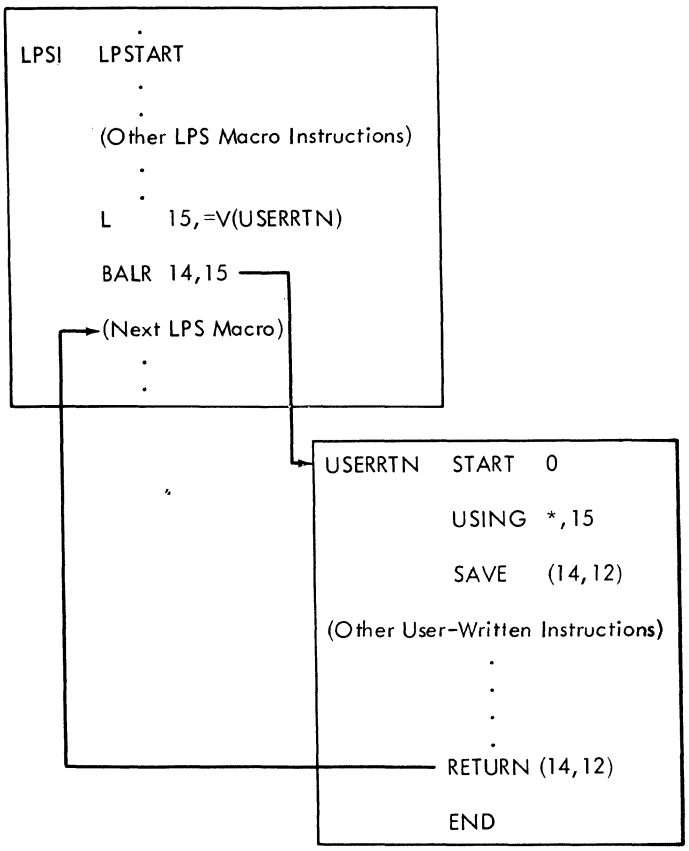


Figure 31. Activation of a Closed User Written Routine in the Audio LPS

The user also has the capability to include his own instructions as in-line code in the LPS. If any of the following registers are required for other than their intended purpose (see Appendix D), they must be saved and restored before execution of the next (in-line) LPS macro instruction: register 4, 5, 6, 7, 8, 9, and 13. If any of the following registers (work registers for the LPS macro instructions) are used, they must be saved before execution of the next LPS macro instructions and restored when needed: register 2, 3, 10, 11, and 12.

Note: Issuance of a Supervisor WAIT macro instruction from a user-written LPS routine (or in-line in the LPS) halts all processing of LPS macro instructions in the message control program until the condition being waited for is satisfied. WAIT, therefore, should either not be used, or be used with extreme care. The ATTACH and DETACH macro instructions may not be used in a user-written LPS routine.

Examination and modification of the status of the telecommunications system is principally done in the message processing program. For a complete discussion, refer to the publication, QTAM Message Processing Program Services.

Macro instructions are provided by QTAM which enable the user to dynamically:

- Activate or deactivate a particular line in an audio communications line group (STARTARU and STOPARU macro instructions).
- Add the four threshold counters for a line to the four cumulative counters for that line, examine the cumulative system, and reset the threshold counters (COPYC macro instruction).

LINE ACTIVATION AND DEACTIVATION

The lines in a line group are automatically activated for message transmission when the line group is opened in the message control program. When issued in a message processing program, the STOPARU and STARTLN (or STARTARU) macro instructions enable the user to dynamically deactivate and reactivate a specific line (or all the lines) within the line group at any point during the operation of the system.

STOPARU is used to effect a temporary deactivation of a specific line when the line is expected to be reactivated by a subsequent STARTLN (or STARTARU) macro instruction. STOPARU can also be used in the message control program to defer the activation of a specific line. For example, if traffic is not expected on a line at the time the line group is initially opened in the message control program, that line can be deactivated by a STOPARU macro instruction. The line can be activated later by a STARTLN macro instruction in a message processing program.

If STARTLN is used in the message control program, it must be issued after the line group is opened and before the ENDREADY macro instruction.

If STARTARU is used during the initialization of the message control program, it must be issued after a corresponding STOPARU. If STOPARU is used, it must be issued after the opening of the corresponding audio line group and before the ENDREADY macro instruction. STOPARU and STARTARU may be used anywhere in the Audio LPS section of the message control program.

Stop Audio Line (STOPARU) Macro Instruction

The STOPARU macro instruction removes an audio communication line from active use, but the line is disabled only after completion of any initialized transaction on line. A stop flag is set on for the designated audio line and control then returns to the program which issued STOPARU. This stop flag is tested whenever the audio line must be enabled to allow the reception of the next transaction. If it is on, the line remains disabled; if it is off, the line becomes enabled.

If the STOPARU macro instruction is issued when the audio line is already enabled, the line will be deactivated only after the processing of the next transaction awaited on this line.

Name	Operation	Operand
[symbol]	STOPARU	filename, {rln / ALL}

symbol
Is the name of the macro instruction.

filename
Is the name of the DTF table for the line group containing the line to be deactivated. It must be identical to the name of the DTFQT macro instruction that generates the DTF table for the line group.

If register notation is used, the address of a location containing the name must be in the general register designated. The field that contains the name must be eight bytes (CL8) in length (padding with blanks is required for names shorter than eight bytes).

rln
Is the relative line number, within the line group, of the line to be

deactivated. If register notation is used, the general register specified must contain the relative number in binary form.

ALL
Specifies that all lines in the line group are to be deactivated.

If a specification error is detected, an error code is returned in register 15. The error conditions tested for, and the return code for each, are contained in Appendix G.

Start Audio Line (STARTARU) Macro Instruction

The STARTARU macro instruction can be used to allow message transmission to resume on one line, or on all lines in the same audio communication line group. Normally, these lines have been previously disabled by the STOPARU macro instruction, and the user must have previously opened the audio line group in the message control program.

The stop flag of each specified audio line is tested. If it is off, the corresponding audio line is already in active use, and the STARTARU macro instruction produces no effect. If it is on, it is set off, and the corresponding audio line is enabled with the initial condition specified in the CMODE operand of the DTFQT macro instruction issued for the concerned audio line group. Control is then returned to the program which issued the STARTARU macro instruction.

Initial enabling of lines in an audio line group starts when the line group is opened in the message control program.

If a disable switch on an IBM 7772, a "make busy" switch on an IBM 7770, or the meter switch on either device has been turned to disable, a STARTARU macro instruction must be issued to enable the ARU lines for use after these switches have been turned back to enable.

Name	Operation	Operand
[symbol]	STARTARU	filename, {rln / ALL}

symbol
Is the name of the macro instruction.

filename
Is the name of the DTF table for the line group containing the line to be reactivated. It must be identical to the name of the DTFQT macro instruction that generated the DTF table for the line group.

If register notation is used, the address of a location containing the name must be in the general register designated. The field that contains the name must be eight bytes (CL8) in length (padding with blanks is required for names shorter than eight bytes).

rln
Is the relative line number, within the line group, of the line to be reactivated. If register notation is used, the general register specified must contain the relative line number in binary form.

ALL
Specifies that all lines in the line group are to be reactivated.

Error returns from this macro instruction are contained in Appendix G.

EXAMINING AND MODIFYING LINE ERROR COUNTERS

Eight counters are provided for each line in the system: four threshold counters, and four cumulative counters. Each cumulative counter corresponds to one of the threshold counters.

For audio lines, the threshold counters keep a count of the number of:

1. Transmissions.
2. Hang-up operations.
3. Data checks for read operations.
4. Data checks for write operations.

For further information, refer to the Error Recovery Procedures section.

Copy Error Counters (COPYC) Macro Instruction

This macro instruction does not apply to the IBM 2260-2848 Local. COPYC causes the four cummulative counters of the line specified to be placed into a designated work area. The threshold counters are reset to 0.

Name	Operation	Operands
[symbol]	COPYC	linename,[rln], workarea

symbol
The name of the macro instruction.

linename
The name of the audio line whose error counters are to be copied. If register notation is used, the general register designated must contain the address of a location containing the name of the line. The field that contains the name must be n bytes long, where n equals or exceeds the length of the longest name of any audio line table entry. If an invalid audio line name is specified, no data movement take place. An error indicator of X'20' is returned in register 15. If no error is detected, register 15 contains 0.

rln
The relative line number, in the line group of the line over which the terminal communicates with the computer. If this operand is not used, its absence must be indicated with a comma.

workarea
The address of the area into which the information is placed. The size of the work area must be ten bytes. If register notation is used, the general register designated must contain the address of the work area.

Bytes one through four of the work area contain the number of transmissions; bytes five and six, the number of hang-up operations; bytes seven and eight, the number of data checks for read operations; and bytes nine and ten, the number of data checks for write operations.

This section discusses the applicability to audio lines of the following services that QTAM provides to aid the user in network control, error recovery and testing:

- Operator Control Facility
- Error Recovery Procedures
- Operator Awareness

The On-line Terminal Test Facility is not applicable to audio lines and cannot be used. The effect of the Checkpoint/Restart Facility is discussed below.

OPERATOR CONTROL FACILITY

The Operator Control facility allows the execution of QTAM macro instructions by special control messages received from specified operator control terminals. The QTAM macros supported under this facility for audio lines are: COPYC, STARTARU, and STOPARU.

The user has three choices for operator control support:

1. None,
2. Operator control facilities only, or
3. Operator control facilities with error messages sent to the operator control terminal(s).

The following table summarizes how each option may be specified.

Facility	TERMTBL Macro	OPCTL Macro
None	Not specified	Not defined
Operator control only	Not specified	Defined
Operator control with error message	OPCTL=chars	Defined

Error messages from the Error Recovery Procedures (but not error messages originating from the ERRMSG macro instruction) are normally sent to the IBM 1052 system console. By specifying the OPCTL=chars operand under the TERMTBL macro instruction and defining the OPCTL macro instruction causes instruction and defining the OPCTL macro instruction causes these error messages to be sent to the operator control terminal. If this is done, error messages originating from errors on the operator control terminal are sent to the IBM 1052 system console.

INPUT CONTROL MESSAGE DESCRIPTIONS

Copy Error Counters (COPYC)

This message causes the four cummulative counters of the line specified to be printed at the source terminal. The threshold counters are reset to 0. The data is in decimal format. If the control message and the counters copied exceed buffer size, only that portion of the output message that the buffer can contain is printed.

Identifier	Function	Operands
ctlmsg	COPYC	{termname rln} {linename }

ctlmsg
: The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

termname
: The name of any terminal for which counters are desired

linename
: The name of the line for which counters are desired.

rln
: The relative line number of the line in the line group. It must be in hexadecimal. If specified, indicates that all lines in the line group are to be stopped.

Start Audio Line (STARTARU)

This message causes transmission to begin or resume on a line or on all lines in an audio line group, provided the line group is opened.

Identifier	Function	Operands
ctlmsg	STARTARU	linename [ALL]

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

linename
The name of the audio line that is to be started.

ALL
If specified, indicates that all lines in the line group are to be started.

Stop Audio Line (STOPARU)

This message removes an audio communications line from active use. If transmission is on the line, the line is not stopped until the transmission has completed.

Identifier	Function	Operands
ctlmsg	STOPARU	linename [ALL]

ctlmsg
The control message identifier. It must be the same as that specified in the OPCTL macro instruction.

linename
The name of the audio line to be stopped.

ALL
If specified, indicates that all lines in the line group are to be stopped.

CHECKPOINTING AND RESTARTING THE MESSAGE CONTROL PROGRAM

This facility does not apply to a system consisting of audio lines only. If a system including audio lines is restarted, the audio lines will have the same status as in the initial program load.

ERROR RECOVERY PROCEDURES

The error recovery procedures are a comprehensive set of procedures for dealing with all kinds of input/output errors that may occur within the telecommunications system.

Upon occurrence of an input/output error, the error recovery procedures examine the sense bits and CSW status bits to determine which type of error has occurred. Depending upon the type of error, the following functions may be performed:

1. The failing action is retried two times, and on the third occurrence of the error, an operator message is provided,
2. The failing action is not retried, and an operator message is immediately provided,
3. The type of failing action is counted in a threshold counter, or
4. A combination of 1, 2, and 3, depending upon the type of terminal on which the error was detected.

   Note: if the error is counted in a threshold counter, it is counted every time that it occurs, including both of the two retry attempts. Messages to the operator are normally sent to the IBM 1052 system console. However, if the OPCTL=chars operand is specified in the TERMTBL macro instruction, they are sent to the operator control terminal. For further information, refer to the Operator Control Facility and TERMTBL Macro Instruction sections.

Eight counters are provided for each line in the system: four threshold counters and four cumulative counters. Each cumulative counter corresponds to one of the threshold counters.

For audio lines, the threshold counters keep a count of the number of:

1. Transmissions.
2. Hang up operations.
3. Data checks on read operations
4. Data checks on write operations.

Whenever any one of the three error threshold counters (but not the transmissions threshold counter) reaches a specified threshold value, the following action is performed:

1. A message is provided to the operator showing:
   a. The threshold value specified for each threshold counter, and
   b. The value in each threshold counter at this time.
2. The threshold counters are reset to 0.

Whenever the transmissions threshold counter reaches its threshold value, the threshold counters are reset to 0, but no message is provided.

The threshold values represent the number of the three types of errors considered excessive within a certain number of total transmissions. These values are specified in the THRESH operand under the DTF macro instruction for the line group in which the line is located. The threshold value for any of the four threshold counters must not be more than 255. However, the threshold values specified for any of the three error counters should be enough less than that specified for the number of transmissions to allow an error message to be provided. If the THRESH operand is omitted, the following values are assumed:

No. of Transmissions: 255.
No. of Hang-up operations: 255.
No. of Data checks on:
    Read operations: 5.
    Write operations: 5.

Error recovery procedures also provide the CCPYC macro instruction. For further information, refer to CCPYC Macro Instruction.

OPERATOR AWARENESS

Operator awareness messages are provided for all permanent and unrecoverable errors and excessive temporary errors as determined by the line error threshold counters. These messages are printed on the 1052 system console unless operator control (OPCTL macro instruction) is specified. When operator control is specified, all operator awareness messages pertaining to the operation of the data links are sent to the telecommunications system control terminal.

```
4QnnI text SYSnnn=cuu LCB=xxxxxx
TI=xxxx DC=xxxxxxxxxx
CSW17=xxxxxxxxxxxxxxxx
CCW=xxxxxxxxxxxxxxxx SN=xxxx
```

where:

4QnnI
: Is the standard message code for the operator. The internal component name is 4Q, the serial is nn, and the action code is I for informational (immediate operator action is not required).

text
: Type of error detected.

SYSnnn=cuu
: Is the symbolic unit assignment of the device, and cuu is the actual unit assignment of the device.

LCB=xxxxxx
: Is the address of the line control block for the terminal or of the Audio line control block for the Audio line.

TI=xxxx
: Is the terminal ID (polling or addressing characters) in hexadecimal format. If only one polling character is used, it will be left justified in this field. This field is not included in Audio messages.

DC=xxxxxxxxxx
: Is the hexadecimal representation of the dial characters for the terminal. This field is not included in Audio messages.

CSW17=xxxxxxxxxxxxxx
: Is bytes 1 through 7 of the channel status word as specified in the channel command block (CCB) in hexadecimal format.

CCW=xxxxxxxxxxxxxxxx
: Is the failing channel command word (CCW) in the channel program in hexadecimal format.

SN=xxxx
: Is the sense byte as specified in the channel command block (CCB) in hexadecimal format.

The format of the error count threshold message for audio lines is:

```
4Q00I LINE ERROR THRESHOLD REACHED
SYSnnn=cuu TR=aaa/bbb HU=jjj/kkk
RDC=lll/mmm WDC=nnn/ppp
```

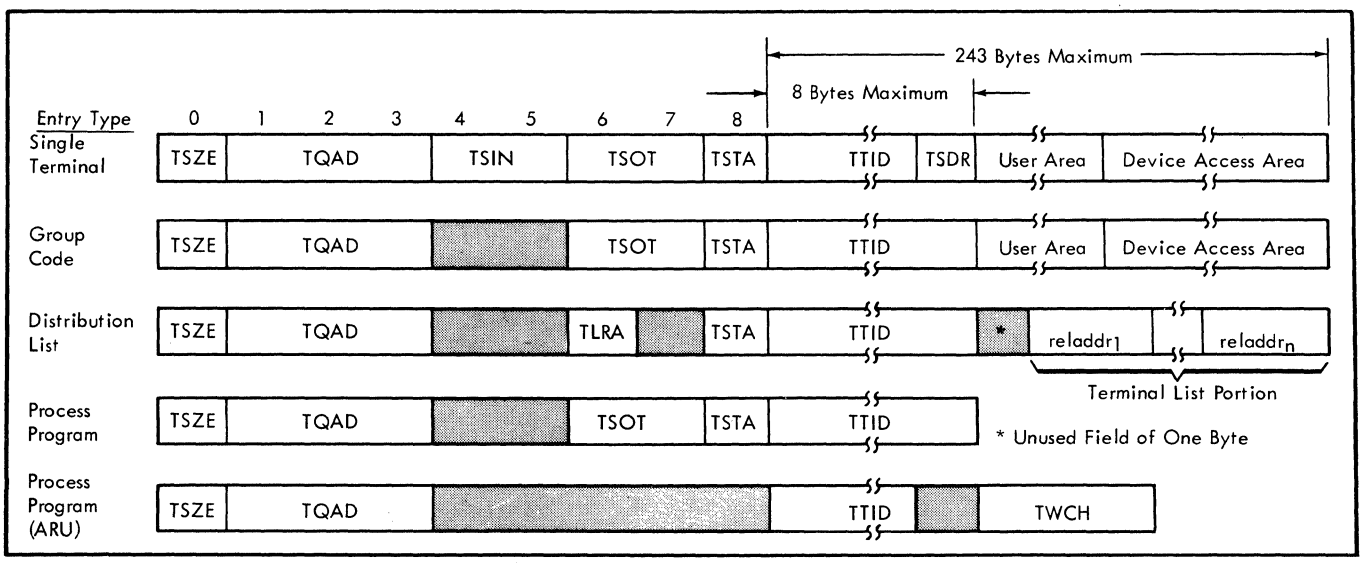
where:

TR=aaa/bbb
aaa is the threshold value for the number of transmissions specified in the DTF/keyword THRESH, and bbb is the number of transmissions attempted up to the time an error threshold was reached. This information is in decimal format.

HU=jjj/kkk
jjj is the threshold value for the number of hang-up operations specified in the THRESH parameter, and kkk is the number that occurred, in decimal format, in the past bbb transmissions.

RDC=lll/mmm
lll is the threshold value for the number of errors on read operations specified in the THRESH parameter, and mmm is the number that occurred, in decimal format, in the past bbb transmissions.

WDC=nnn/ppp
nnn is the threshold value for the number of errors on write operations specified in the THRESH parameter, and ppp is the number that occurred, in decimal format, in the past bbb transmissions.



Note: All DSECT names have the prefix IJLQ.

Field Name	Function	Start Location	Field Length	Form	Value provided by	Value Range	Initial Value
IJLQTSZE	All entry types: Specifies, entry size (in bytes) and provides access to the next higher terminal table entry.	byte 0	1 byte	binary number	macro-generator	1-255	--
IJLQTQAD	All entry types: Contains the address of the QCB for the queue on which outgoing messages to the destinations are placed. Using this address identification, the queueing routine places each message on its appropriate queue. Note: For audio messages, this field contains a pointer to the MS process queue.	byte 1	3 bytes	binary address	macro-generator	--	--

Figure 32. Terminal-Table Entry Formats (Part 1 of 5)

IJLQTSIN	Single-terminal entries: Stores and maintains a sequence number for incoming messages from the terminal represented by this entry. The SEQIN macro instruction uses this value as a check against the sequence number appearing in the incoming message header.	byte 4	2 bytes	binary count	QTAM	0-9999	1
IJLQTSCT	Single-terminal, group-code, process-program entries: Provides and maintains a sequence number for outgoing messages to the destination(s) represented by this entry. The SEQOUT macro instruction obtains the current value from IJLQTSCT and places it in the outgoing message header. The sequence number is incremented by 1 each time the number is placed in a message. If a terminal is represented by both a single-terminal entry and a group-code entry, only the IJLQTSCT field in the group-code entry is incremented when a message is transmitted via group-code addressing.	byte 6	2 bytes	binary count	QTAM	0-9999	1
IJLQTLRA	Distribution-list entry: Contains the address of the beginning of the terminal list portion of this entry, relative to the address of byte 0 of the entry. The terminal list portion consists of subfields reladdr$_1$, ... reladdr$_n$.	byte 6	1 byte	binary relative address	macro-generator	--	--
IJLQTSTA	All entry types: Indicates various communication conditions associated with the terminal(s) represented by this entry.	byte 8	1 byte	binary status	macro-generator	--	see specific bits

Figure 32. Terminal-Table Entry Formats (Part 2 of 5)

	Bit 0: Attention bit; initially set to 0. This bit is set to 1 when an Attention interrupt (read request) is received from the 2260 Local represented by this entry. It is reset to 0 when the read request is serviced. This bit applies only to a 2260 local. Bit 1: Channel queue bit; when the CCB for this 2260 Local is on the DCS channel scheduler queue, this bit is set to 1. Otherwise, it is set to 0. This bit applies only to a 2260 Local. Bit 2: Input allowed bit; initially sent to 0. This bit is set to 1 by the Open routine if the 2260 local represented by this entry appears in the polling list for the line group. This bit applies only to a 2260 Local. Bit 3: Printer bit; this bit is initially set to 1 if the terminal represented by this entry is a 1053 Printer in a 2260-2848 local line group, or if it has been specified as an IBM 2740 Model 2 terminal. Otherwise, it is set to 0. Bit 4: RELEASEM pending bit; initially set to zero. Set to 1 when RELEASEM has been requested but has not yet completed. Reset to zero when RELEASEM completes (i.e., when the next message for a terminal is read from DASD). Bit 5: Intercept bit; initially set to 0. This bit is set to 1 upon issuance of an INTERCPT macro instruction to indicate that a message on the queue was not transmitted. It may be reset to 0 by a CHNGT or RELEASEM macro instruction when transmission can be resumed.						

Figure 32. Terminal-Table Entry Formats (Part 3 of 5)

	Bit 6: Send bit; initially set to 1. This bit is set to 0 upon issuance of an INTERCPT macro instruction, indicating that messages on the queue for the destination are withheld from transmission. It may be reset to 1 by a CHNGT or RELEASEM macro instruction when transmission can be resumed. Bit 7: Receive bit; initially set to 1. This bit is set to 1 to indicate that the terminal represented by this entry is being polled. It may be set to 0 to prevent polling of the terminals. Setting to 0 or 1 is achieved by means of the CHNGT macro instruction. Note: Bit 7 is not applicable to, and is not used by, group-code, distribution-list and process-program entries.						
IJIQTTID	All entry types: Contains the name of the terminal(s) that this entry represents, in the form of a terminal code (or code for the process-program). This code is the same code that can appear in the source or destination code field of the message header. Note: For audio messages, this code must appear in the DTF table defined for the audio line groups using a process-program entry.	byte 9	1 to 8 bytes	EBCDIC characters (upper-case)	User	--	--
IJIQTSDR (optional)	Single-terminal entries: Contains the offset of the SDR counter set assigned to this terminal. This field is included only if the OER/SDR option is in use.	Immediately following IJIQTTID	2 bytes	binary count	QTAM	--	0

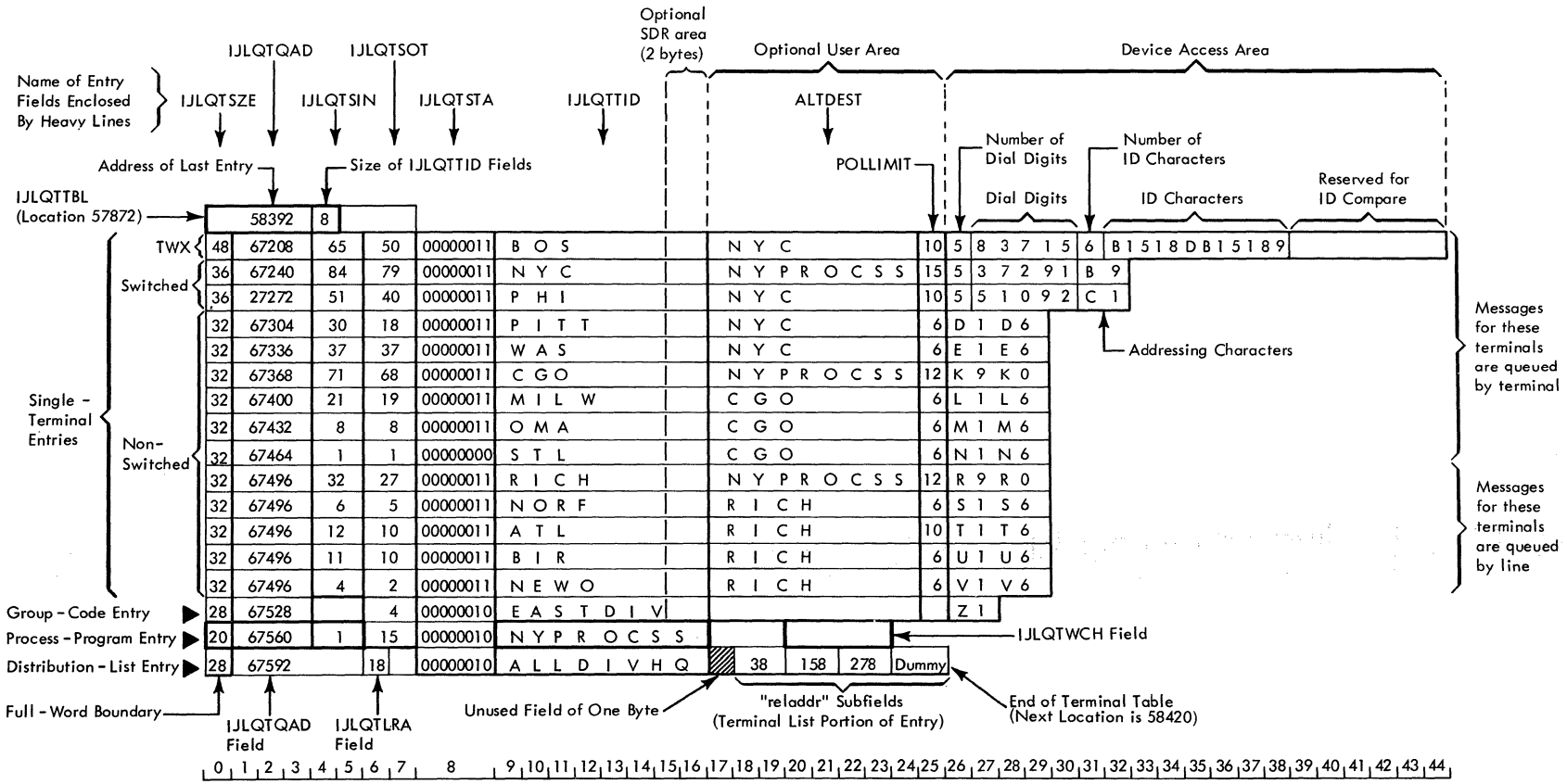
Figure 32. Terminal-Table Entry Formats (Part 4 of 5)

User area	Single-terminal and group-code entries: Contains such data as the particular application may require, e.g., alternate destination codes, polling limit parameters, and diagnostic information. The user area consists of a contiguous series of subfields whose form, length, and contents are specified by means of OPTION and TERM macro instructions (see the descriptions of these macros.	(see Note 1)	Cumulative length of all subfields in entry	As specified by OPTION macros	User	--	--
Device access area	Single-terminal and group-code entries: For nonswitched terminals, contains the polling and addressing characters for the terminal(s) represented by this entry. These characters are specified by the "addressing" operand of the TERM macro instruction. For a 2260 or 1053 in a 2260-2848 Local line group, this field contains the CCB and other information required for the terminal. For WTTA terminals this field contains the terminal identification sequence.	Immediately following user area	Cumulative length of all subfields in entry	Device code	User	--	--
reladdr	Distribution-list entries: Contains the address of a single-terminal entry relative to the address of the terminal table (i.e., IJLQTTBL). The last "reladdr" subfield is set to 0, indicating the end of the terminal list portion of the distribution-list entry.	Immediately following IJLQTTID subfield	2 bytes per subfield	Binary relative address	User	--	--
IJLQTWCH	Audio process-program entries: Contains the address of the first input message to be transferred to the MS-process queue. This field is adjusted on a full-word boundary.	Following IJLQTTID subfield after adjustment	4 bytes	Binary address	QTAM	--	--

Note: Start location immediately follows the IJLQTTID or IJLQTSDR subfield. Symbolic references may be made to the optional subfields named by OPTION macro instructions.

Figure 32. Terminal-Table Entry Formats (Part 5 of 5)

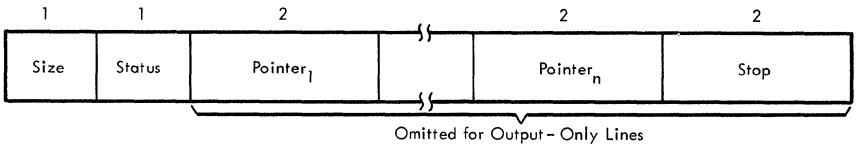
Figure 33. Example of Terminal Table



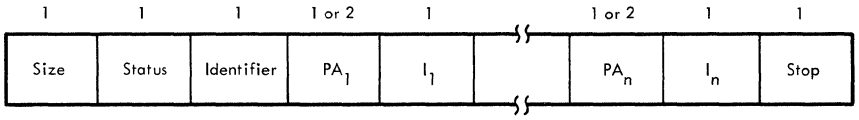
NOTES:

1. ALTDEST and POLLIMIT are optional fields specified by OPTION macro and "opdata" operands of TERM macro instructions.
2. Fields and bytes in which no characters appear contain binary zeros or blanks.
3. Fields and bytes used by the audio lines are shown on the squares drawn in thick lines.

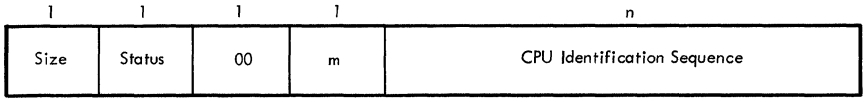
Polling List for Nonswitched Lines and for IBM 2260-2848 Local:



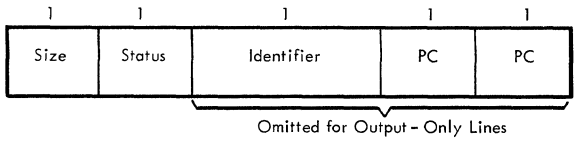
Polling List for Nonswitched Lines (Autopolled Lines).



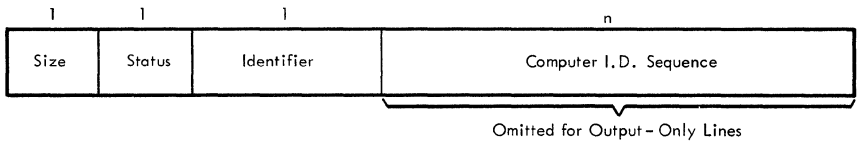
Polling List for Nonswitched Lines (WTTA Terminals):



Polling List for Switched IBM 1050:



Polling List for TWX (AT&T 33, 35) Lines:



LEGEND:

Size: Indicates total length of polling list.

Status: Indicates current status of the polling list: if non-zero, the list is active (the line can be polled [nonswitched line] or enabled [switched line]). The status byte is initialized to X'01' (active) when the POLL macro-instruction is assembled.

$Pointer_1$ through $Pointer_n$: Represent the addresses, relative to the terminal table address, of the first byte of the terminal table entries associated with this list.

Stop: Identifies the end of the polling list for a nonswitched line. Stop is X'0000' for polled lines and X'FE' for autopolled lines.

Identifier: Identifies the polling list as being for a switched or autopolled line. The identifier is X'00' for switched lines, X'02' for autopolled lines with IBM 1030, and X'03' for autopolled lines with IBM 1050, 1060, or 2740.

PA_1 through PA_n: Represents the polling addresses of the terminals attached to an autopolled line. Each address is made up of one character for IBM 1030 and of two characters for IBM 1050, 1060, or 2740.

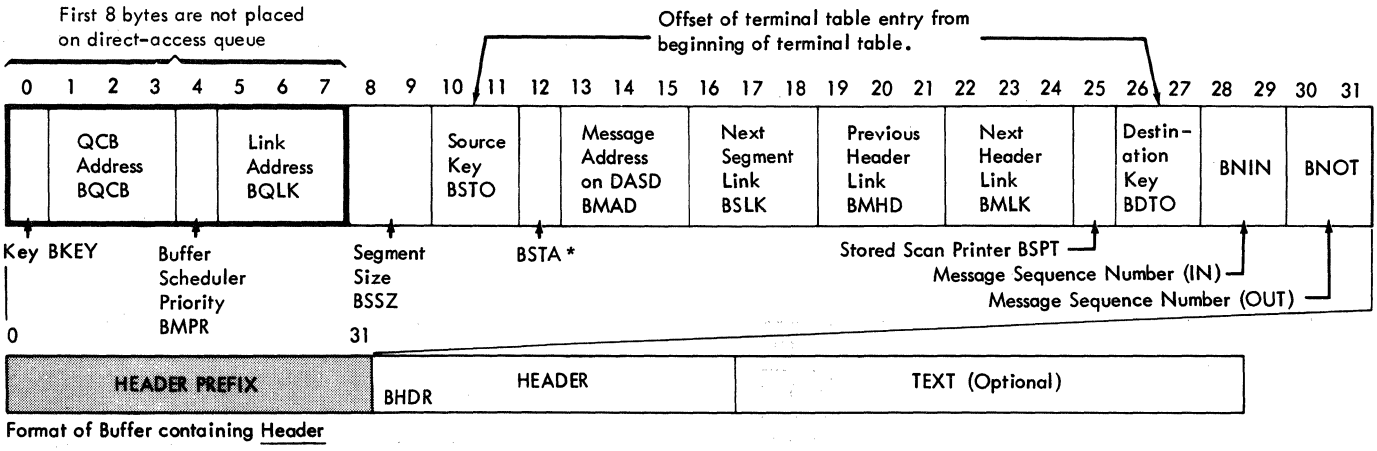
I_1 through I_n: Represents the index characters associated with the polling addresses.

PC: Is a polling character to be sent to an IBM 1050 terminal that calls the computer on the line represented by this polling list.

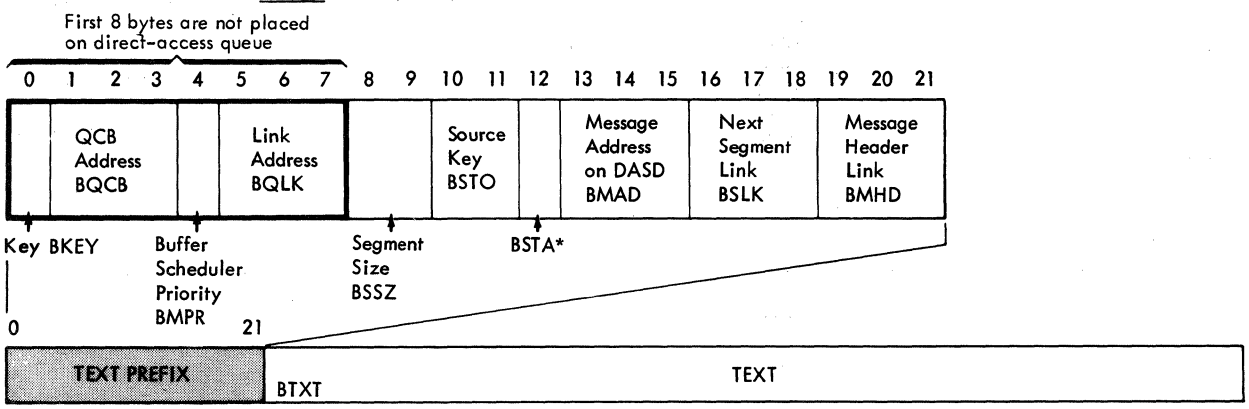
m : Is the number of characters in the CPU identification sequence.

Computer I.D. Sequence: Is the sequence of characters to be sent to any TWX terminal that calls the computer on the line represented by this polling list. For WTTA terminals, CPU Identification Sequence is the sequence of characters to be sent to a WTTA terminal during an identification exchange.

Figure 34. Polling List Format



Format of Buffer containing Header



Format of Buffer containing Text

Note: All DSECT names have the prefix IJLQ.

* Significance of the bits in the BSTA byte is as follows:

Bit Position	Bit Name	
0	CANCEL	0 = Send or process message 1 = Do not send or process message
1	REROUTE	0 = Original copy of header 1 = Duplicate copy of header
2	EOB	0 = No EOB is present in any buffer position except the last 1 = An EOB is present in some buffer position other than the last (Presence or lack of an EOB in the last position does not affect setting of bit)
3	SRVCD	0 = Message was not previously serviced 1 = Message was previously serviced
4	TRUNC	0 = Normal message 1 = Message is generated thru ERRMSG marco
5	PRIORITY	0 = Message sent without priority 1 = Message sent with priority
6-7	SEGTYP	00 = Header segment (Not last segment) 01 = Text segment (Not last segment) 10 = Header segment (Last segment) 11 = Text segment (Last segment)

Figure 35. Formats of filled Buffers (nonaudio lines)

Name	Operation	Operand	Function of Macro Instruction: Buffer Assignment	Polling List Definition	Terminal Table Definition	Line Table Definition	Word Table Definition
Symbol	BUFARU	n,length					•
	BUFFER	integer, length [,mmm] [,brb]					•
Symbol	LINE	filename, rln		•			
	LINETBL	entry [,n]		•[3]			
Symbol	LIST	entry			•		
Subfield	OPTION	typelength			•[1]		
Pollname	POLL	{ [entry] [,AUTOPOL = {1 | 2}] polladr | nid | 2740 }				•	
Symbol	PROCESS	[EXPEDITE][,ARU]			•		
Symbol	TERM	qtype, filename, rln, [adchars (opdata, ...). [CALL = integer | CALL = NONE], [ID = hexchars][, DEVADDR = SYSnnn] [, PRNTR = YES]			•		
	TERMTBL	enter,[n] [, OPCTL = chars][, CPINTV = integer] [,CONF = {ARU | MIXED}]			•[2]		
Symbol	WORD	diskaddr, length	•				
	WORDTBL	entry	•[4]				

Notes: 1. OPTION must directly follow TERMTBL macro instruction.
2. TERMTBL must be the first of the macro instructions used to create a terminal table
3. LINETBL must be specified before all LINE macro instructions used to create the audio line table
4. WORDTBL must be specified before all WORD macro intructions used to create the 7772 DCV word table

Figure 36. Summary of Control Information Macro Instructions

Operation	Operand	RCVSEG	RCVHDR	ENDRCV	SENDSEG	SENDHDR	ENDSEND	LPSTART
ARUMGTYP	[typechar]							•
BREAKOFF	nnnnn	•						
CANCELM	mask			•				
CHECKARU	mask							•
COUNTER	subfield	•	•		•	•		
DATESTMP			•			•		
DIRECT	{=CLnn'dest'} {subfield}		•					
ECA	eca		•					
ECB				•			•	
ECBIC				•			•	
ERRMSG	mask, {=CLn'dest"} {subfield} {source} , {=C'message'} {msgchar}			•			•	
INTERCPT	mask,subfield						•	
LOGSEG	filename[,{ARU} {PREFIX}]	•	•		•	•		•
MODE	{PRICRITY} {CONVERSE} {INITIATE} {MCL2260} [,condchar][,wrt60=code]		•			•		
MSGTYPE	[typechar]		•			•		
OPCTL	CTLMSG=msgname,TERM=termname [,ALTERM=termname],[INTRCPT={YES} {NO}]		•					
PAUSE	ctlchar,insertchar				•			
PCLILIMIT	{nnn} {subfield}		•	•				
REPEAT	codechar							•
RERCUTE	mask,{=CLn'dest'} {subfield} {SOURCE}			•			•	
RCUTE	[n]		•					
SEQIN	[n]		•					

Figure 37. Functional Macro Instructions and the Delimiter Each May Follow (Part 1 of 2)

Operation	Operand	RCVSEG	RCVHDR	ENDRCV	SENDSEG	SENDHDR	ENDSEND	LPSTART
SEQCUT	n		•			•		
SKIP	[n skipchrs]		•			•		
SOURCE	[n]		•					
TIMESTMP	nn		•			•		
TRANS	table	•	•		•	•		•
WRU						•	•	

Note: Restrictions governing usage are explained in individual macro instruction descriptions.

Figure 37. Functional Macro Instructions and the Delimiter Each may follow (Part 2 of 2)

Name	Operation	Operand	Restrictions
	ENDRCV		
	ENDSEND		
lpsname	LPSTART	[nn,] TERM= (termname$_1$,..)	Required delimiter; must be the first macro instruction in an LPS.
	POSTAUD		Required delimiter in an Audio LPS; must appear only once.
	POSTRCV		Required delimiter; must immediately follow the last of the sequence of macro instructions that handle incoming messages. Only one POSTRCV may appear in an LPS.
	POSTSEND		Required delimiter; must immediately follow the last of the sequence of macro instructions that handle outgoing messages. Only one POSTSEND may appear in an LPS.
	RCVHDR		
	RCVSEG		
	SENDHDR		
	SENDSEG		

Figure 38. Summary of Line Procedure Specification Delimiter Macro Instructions

APPENDIX C: SAMPLE MESSAGE CONTROL PROGRAMS

Two sample message control programs are provided: the first concerns a nonaudio application, the second concerns an audio-only application.

NONAUDIO SAMPLE MESSAGE CONTROL PROGRAM

The following sample message control program is included to illustrate a method of arranging the source statements required to generate a message control program. The sample program performs the simple function of switching messages between two remotely located terminals. The following equipment and options are required to support the program:

1. Disk: IBM 2311 preformatted for QTAM (SYS005).
2. Terminals: two IBM 1050s with the line correction feature.
3. Telecommunications Control Unit: IBM 2701 connected to the multiplexor channel of the IBM System/360.
4. Line configuration: one nonswitched line (SYS006) with two data sets (data path between terminals and the 2701).

Input: Any data that can be sent by a 1050 terminal is acceptable as input. Messages may be of any length and may consist of any number of blocks. The last block of a message should be followed by an EOT character for maximum efficiency (otherwise a timeout will occur). No message header is necessary because the DIRECT macro is used to route messages, and no LPS macro instructions that process header fields are included.

Output: The message data received from one terminal is sent to the other terminal unchanged.

Two methods may be used to furnish the necessary DTF tables to the message control program:

1. DTF tables and message control program assembled together.
2. DTF tables assembled separately from the remainder of the message control program.

Both methods are illustrated in this appendix.

ASSEMBLING THE DTF TABLES AND MESSAGE CONTROL PROGRAM TOGETHER

If the DTF tables and the message control program are assembled together, the assembly source deck is as follows:

```
QTAMMCP  START 0
BASEREG  EQU   12
         BALR  BASEREG,0              INITIALIZE BASE REGISTER.
         USING *,BASEREG              ESTABLISH ADDRESSABILITY.

*    THE FILE INITIALIZATION AND ACTIVATION SECTION
*    CONSISTS OF THE FOLLOWING MACRO INSTRUCTIONS

         OPEN  DISK,GRP               OPEN DASD-MESSAGE QUEUES AND
*                                     LINE GROUP FILES.
         LA    13,SAVEAREA            ADDR OF USER'S SAVE AREA.
         ENDREADY                     BRANCH TO QTAM LOGIC TO AWAIT
*                                     1ST MSG. CONTROL RETURNS TO
*                                     INSTRUCTION LABELED LPS.
SAVEAREA DS    18F                    USER SAVE AREA.

*    THE CONTROL INFORMATION SECTION CONSISTS OF
*    THE FOLLOWING MACRO INSTRUCTIONS

         BUFFER 3,100,5,3             PROVIDES THREE BUFFERS OF 100
*                                     BYTES EACH FOR THE QTAM BUFFER
*                                     POOL. ALSO PROVIDES STORAGE FOR
*                                     FIVE CCWS THAT QTAM GENERATES
*                                     FOR SENDING IDLE CHARACTERS.
         TERMTBL NYCX                 SPECIFIES THE EXTENT OF THE
*                                     TERMINAL TABLE.
DEST     OPTION CL4                   THE NAME AND LENGTH OF AN
*                                     OPTIONAL SUBFIELD TO CONTAIN
*                                     THE DESTINATION CODE FOR MSGS
*                                     RECEIVED FROM EACH TERMINAL.
BOST     TERM  L,GRP,1,E202E20B,(NYCX) DEFINES THE TERMINAL TABLE
*                                     ENTRY FOR THE BOSTON TERMINAL.
NYCX     TERM  L,GRP,1,E402E40B,(BOST) ENTRY FOR NYC TERMINAL.
POLLIST  POLL  (BOST,NYCX)            SPECIFIES THE ORDER IN WHICH
*                                     THE TERMINALS WILL BE POLLED.
*
*    THE LPS SECTION CONSISTS OF THE FOLLOWING MACRO INSTRUCTIONS
*
LPS      LPSTART TERM=(1050)          IDENTIFIES BEGINNING OF LPS.
         RCVHDR                       IDENTIFIES RECEIVE HEADER
*                                     SUBGROUP.
         DIRECT DEST                  CAUSES INCOMING MESSAGES TO BE
*                                     ROUTED TO THE DESTINATION SPECI-
*                                     FIED IN THE OPTIONAL SUBFIELD
*                                     OF THE TERMINAL TABLE ENTRY FOR
*                                     THE ORIGINATING TERMINAL. MSGS
*                                     FROM BOST ARE SENT TO NYCX
*                                     THOSE FROM NYCX ARE SENT TO BOST
         ENDRCV                       END RECEIVE SUBGROUP DELIMITER.
         EOBLC                        ALLOWS THE 1050 TERMINAL TO
*                                     CONTINUE SENDING AFTER AN EOB.
*                                     ALSO PROVIDES FOR UP TO TWO
*                                     RETRIES IF A TRANSMISSION ERROR
*                                     IS DETECTED.
         POSTRCV                      END OF THE RECEIVE GROUP OF
*                                     THE LPS.
         SENDSEG                      SEND SEGMENT SUBGROUP DELIMITER.
         PAUSE X'5E',13X'5E'          CAUSES 13 IDLE CHARACTERS
*                                     (X'5E') TO BE INSERTED EACH
*                                     TIME A NL CHARACTER IS DETECTED
*                                     IN AN OUTGOING MSG.
```

```
         ENDSEND                        END SEND SUBGROUP DELIMITER.
         ECBLC                          INFORMS QTAM TO CONTINUE SEND-
*                                       ING UPON DETECTING AN EOB.
*                                       ALSO SPECIFIES UP TO TWO RETRIES
*                                       IF A TRANSMISSION ERROR IS
*                                       DETECTED.
         POSTSEND                       END OF THE SEND GROUP OF THE
*                                       LPS.
*
ENDMCP   CLOSE GRP,DISK                 CLOSE FILES.

         EOJ                            TERMINATES MSG CONTROL PROGRAM.

*    THE FILE-DEFINITION SECTION CONSISTS OF THE
*    FOLLOWING MACRO INSTRUCTIONS AND PARAMETERS
                                                                  Col. 72
DISK     DTFQT TYPE=DA,                                              -
               DEVADDR=SYS005,                                       -
               ECJAD=ENDMCP

GRP      DTFQT TYPE=IG,                                              -
               LINELST=(006),                                        -
               SWITCH=NO,                                            -
               DEVICE=1050,                                          -
               CPCLI=(PCLILIST),                                     -
               BUFNO=3,                                              -
               ACICC=17,                                             -
               CLPS=LPS,                                             -
                                                                  Col. 72
               CPRI=S,                                               -
               CU=2701,                                              -
               TYPEFLE=CMBND
         END   QTAMMCP
```

ASSEMBLING THE DTF TABLES SEPARATELY

If the DTF tables are assembled separately from the main part of the message control program, it is the user's responsibility to include the necessary ENTRY and EXTRN statements. The main program is assembled as follows:

```
QTAMMCP  START   0
BASEREG  EQU     12
         ENTRY   IPS,PCLLIST,ENDMCP
         EXTRN   DISK,GRP
         BALR    BASEREG,0
         USING   *,BASEREG
         OPEN    DISK,GRP
         LA      13,SAVEAREA
         ENDREADY
SAVEAREA DS      18F
         BUFFER  3,100,5,3
         TERMTEL PHIL
DEST     OPTION  CI4
BOST     TERM    I,GRP,1,E202E20B,(PHIL)
PHIL     TERM    I,GRP,1,E402E40B,(BOST)
PCLLIST  PCLL    (BOST,PHIL)
LPS      LPSTART TERM=(1050)
         RCVHDR
         DIRECT  DEST
         ENDRCV
         EOBIC
         POSTRCV
         SENDSEG
         PAUSE   X'5E',6X'5E'
```

```
          ENDSEND
          EOBIC
          POSTSEND
ENDMCP    CLOSE     GRP,,DISK
          ECJ
          END       CTAMMCP
```

The DTF table for the DASD message queues file is assembled as follows:

```
                                        Col. 72
DISK  DTFQT   TYPE=DA,                       X
              DEVADDR=SYS005,                X
              ECJAD=ENDMCP,                  X
              SEPASM=YES
      END
```

The DTF table for the line group file is assembled as follows:

```
                                        Col. 72
GRP   DTFQT   TYPE=LG,LINELST=(006),,        X
              SWITCH=NO,,DEVICE=1050,,       X
              CPOLL=(POLLIST),BUFNO=3,       X
              ACICC=17,CLPS=LPS,CPRI=S,      X
              CU=2701,TYPEFLE=CMBND,,        X
              SEPASM=YES
      END
```

ENTRY statements are required in the main program for the names specified in the CPOLL and CLPS operands of the line group DTFQT macro, and in the ECJAD operand of the DASD message queues DTFQT macro. This enables the linkage editor to resolve these address constants at linkage edit time. EXTRNs are not required for these names in the DTF's because V-type address constants are generated for them in the macro expansion if SEPASM=YES is specified.

An EXTRN statement must be included in the main program for the name of any separately assembled DTF referenced by the main program. An ENTRY statement is not necessary in the DTF assembly because the macro expansion generates it as a CSECT name. Only the DTF sections can be assembled separately. The remainder of the user-written message control program must be assembled together.

AUDIO SAMPLE MESSAGE CONTROL PROGRAM

The following sample message control program illustrates the source statements required to generate an audio message control program. The following equipment is necessary:

1. Audio Response Unit: IBM 7770 connected to the multiplexor channel of the IBM System/360.

2. Audio Line configuration: two switched lines (SYS007 and SYS008) with two data sets (data path between switched network and the 7770).

Input: Any data that can be sent by an IBM 1001 is acceptable as input. The input message representing an inquiry may be of any length within the input buffer area specified for a line.

Output: The output message representing the audio answer is sent to the same terminal connected on line.

ASSEMBLING THE DTF AND MESSAGE CONTROL PROGRAM TOGETHER

In this case, the assembly source deck is as follows:

```
ACTAMMCP START 0
BASEREG  EQU   12
         BALR  BASEREG,0              INITIALIZE BASE REGISTER.
         USING *,BASEREG              ESTABLISH ADDRESSABILITY.

*    THE FILE INITIALIZATION AND ACTIVATION SECTION
*    CONSISTS OF THE FOLLOWING MACRO INSTRUCTIONS

         OPEN  ARULG                  OPEN 7770 LINE GROUP FILE.
         LA    13,SAVEAREA            ADDR OF USER SAVE AREA.
         ENDREADY                     BRANCH TO QTAM LOGIC TO AWAIT
*                                     1ST MESSAGE. CONTROL RETURNS TO
*                                     INSTRUCTION LABELED ARULPS.
SAVEAREA DS    18F                    USER SAVE AREA.

*    THE CONTROL INFORMATION SECTION CONSISTS OF
*    THE FOLLOWING MACRO INSTRUCTIONS

         TERMTBL AMPP,CONF=ARU        SPECIFIES THE EXTENT OF THE
*                                     TERMINAL TABLE.
AMPP     PROCESS ,ARU                 DEFINES THE PROCESS-PROGRAM
*                                     ENTRY USED BY THE AUDIO LINES.

         LINETBL LINE2                SPECIFIES THE EXTENT OF THE
*                                     AUDIO LINE TABLE.
LINE1    LINE  ARULG,1                DEFINES THE LINE TABLE ENTRY
*                                     FOR THE FIRST LINE.
LINE2    LINE  ARULG,2                ENTRY FOR THE SECOND LINE.

*    THE ARU/LPS SECTION CONSISTS OF THE
*    FOLLOWING MACRO INSTRUCTIONS

ARULPS   LPSTART TERM=(7770)          IDENTIFIES BEGINNING OF ARU/LPS.
         TRANS RCVARU                 CAUSES AUDIO INPUT MESSAGES
*                                     TO BE TRANSLATED FROM ARU CODE
*                                     INTO EBCDIC.
         REPEAT C'A'                  CAUSES THE REPETITION OF THE
*                                     PREVIOUS ANSWER, IF ANY, WHEN
*                                     THE FIRST CHARACTER OF THE
*                                     INPUT MESSAGE IS A.
         POSTARU                      END OF THE ARU/LPS.

ENDAMCP  CLOSE ARULG                  CLOSE 7770 LINE GROUP FILE.
         EOJ                          TERMINATES MESSAGE CONTROL
*                                     PROGRAM.

*    THE FILE-DEFINITION SECTION CONSISTS OF THE
*    FOLLOWING MACRO INSTRUCTIONS
                                                              Col. 72
ARULG    DTFQT TYPE=LG,                                          -
               LINELST=(007,008),                                -
               CU=7770,                                          -
               CLPS=ARULPS,                                      -
               PRCCESS=AMPP,                                     -
               ECJAD=ENDAMCP,                                    -
               BUFIN=20,                                         -
               BUFAC=50,                                         -
               CMCDE=CVR,                                        -
               TYPEFLE=CMBND

         END   ACTAMMCP
```

ASSEMBLING THE DTF SEPARATELY

If the DTF table is assembled separately from the main part of the message control program, it is the user's responsibility to include the necessary ENTRY and EXTRN statements. The main program is assembled as follows:

```
ACTAMMCP  START     0
BASEREG   EQU       12
          ENTRY     ARULPS,AMPP,ENDAMCP
          EXTRN     ARULG
          BALR      BASEREG,0
          USING     *,BASEREG
          OPEN      ARULG
          LA        13,SAVEAREA
          ENDREADY
SAVEAREA  DS        18F
          TERMTBL   AMPP,CONF=ARU
AMPP      PROCESS   ,ARU
          LINETBL   LINE2
LINE1     LINE      ARULG,1
LINE2     LINE      ARULG,2
ARULPS    LPSTART   TERM=(7770)
          TRANS     RCVARU
          REPEAT    C'A'
          POSTARU
ENDAMCP   CLOSE     ARULG
          EOJ
          END       ACTAMMCP
```

The DTF table for the 7770 line group is assembled as follows:

```
                                        Col. 72
ARULG DTFQT   TYPE=LG,                          *
              LINELST=(007,008),                *
              CU=7770,CLPS=ARULPS,              *
              PROCESS=AMPP,                     *
              ECJAD=ENDAMCP,                    *
              BUFIN=20,BUFAC=50,                *
              CMODE=CVR,TYPEFLE=CMBND,          *
              SEPASM=YES
      END
```

ENTRY statements are required in the main program for the names specified in the CLPS, PROCESS and ECJAD keyword operands of the 7770 line group DTFQT macro. This enables the linkage editor to resolve these address constants at linkage edit time. EXTRNs are not required for these names in the DTF because V-type address constants are generated for them in the macro expansion if SEPASM=YES is specified.

An EXTRN statement must be included in the main program for the name of any separately assembled DTF referenced by the main program. An ENTRY statement is not necessary in the DTF assembly because the macro expansion generates it as a CSECT name.

Only the DTF sections can be assembled separately. The remainder of the user-written message control program must be assembled together.

APPENDIX D: QTAM REGISTER USAGE IN THE LPS (OR AUDIO LPS)

Certain of the registers used by QTAM may be of value to the user who writes closed routines (or in-line instructions) to be included in an LPS or an ARU/LPS. The usage of each of these registers is explained below. For an Audio LPS only registers 4, 7, and 13 are significant.

Register 4 -- LCB or ALCB Address Register

Contains the address of the line or audio line control block for the line over which the current message segment was received (input LPS or Audio LPS processing) or sent (output LPS processing).

Register 5 -- Scan Pointer Register

The address of either the last character of the last header field scanned, or the first blank character following the field.

1. If the operand of the last macro that referenced a header field either does not permit any variation in the length of the field (for example, EOA C'A'), or specifies explicitly the length of the field (for example, SOURCE 6), then register 5 contains the address of the last character of the field.

2. If the operand of the last macro that referenced a header field does not explicitly specify the length of the field (for example, SOURCE [no operand]), register 5 contains the address of the first blank character that follows the field.

Register 6 -- Buffer Address Register

Contains the address of the buffer currently being processed by the LPS. If the buffer contains a header segment, the first data character in the header is located 32 bytes beyond the buffer address. If the buffer contains a text segment, the first data character is located 22 bytes beyond the buffer address. Both offsets are relative to register 6 as the base register.

Register 7 -- LPS or Audio LPS Base Register

Contains the address of the beginning of the LPS or the ARU/LPS currently being executed (that is, the address of the first instruction of the LPSTART macro expansion). LPSTART establishes this register as the base register for this LPS.

Register 8 -- Terminal Table Register

Contains the address of an entry in the terminal table. The particular entry depends on whether the Receive or Send group of the LPS is being executed.

1. Receive Group. The address of the terminal table entry for the terminal from which the message segment currently being processed was received if the terminal is on a nonswitched line. If the terminal is on a switched line, register 8 points to the beginning of the terminal table unless a SOURCE macro is included in the LPS. After SOURCE (if used) is executed, register 8 contains the address of the terminal table entry for the originating terminal.

2. Send Group. The address of the terminal table entry for the terminal to which the current message segment is being sent.

Register 9 -- End-of-Segment Address Register

Contains the address of the last character position in the buffer currently being read into or out of.

Register 13 -- Save Area Register

Contains the address of a QTAM-provided, 18-word save area to be used for saving the registers by a user-written routine.

This appendix contains charts which define the character sets and transmission code bit patterns used by the various terminals supported by QTAM. Charts are also provided which facilitate reading the terminal code found in storage.

QTAM CHARACTER SET AND CODE CORRESPONDENCE CHART

This chart shows the character set and bit patterns for the Extended Binary Coded Decimal Interchange Code (EBCDIC) and the character sets and transmission code bit patterns for each of the terminal types supported by DOS QTAM.

The chart may be used to determine the bit patterns (as contained in main storage bytes) for each of the various characters sent or received by a specified terminal type; and to determine the relationship (as established by the arrangement of the IBM-provided translate tables) among the character sets for the various terminal types.

For convenience in referring to particular chart locations, the chart's columns and rows are given reference numbers. Combined, these numbers enable reference to a particular chart location. For example, location 21/17, the intersection of row 21 and column 17, contains NL.

Arrangement of Chart

The chart contains a group of three columns for the EBCDIC character set and a group for each of the various terminal character sets. Within the EBCDIC group, column 3 contains the 256 bit patterns comprising the code. For those bit patterns to which characters are currently assigned, the characters appear in column 1 (graphics) and column 2 (line controls and device controls). (All currently assigned characters are shown, regardless of whether they are in the character sets of any of the terminal types represented in the remainder of the chart).

Each of the remaining groups (columns 4-36) contains the characters comprising the character set of a specific terminal type, along with the transmission code bit patterns. Column 37 repeats the EBCDIC code presented in column 3, for ease of reference.

In the EBCDIC group, the bit patterns and characters are arranged in collating sequence, from hexadecimal 00 to hexadecimal FF. In the remainder of the chart, the locations of bit patterns and characters are determined by the arrangement of the translate tables.

Terminal Character Sets

This chart shows only the characters comprising the commonly used character set options. The options represented in the chart are:

IBM 1030: Standard and H options
IBM 1050: System/360 option
IBM 1060: Standard option
IBM 2260: Standard option
IBM 2740: Standard option
IBM 7770/7772: Standard option
AT&T 83B3: A and C options
WU 115A: A and C options
WU TWX: Standard option
WTTA: Standard option

IBM 1030 graphics and AT&T 83B3/WU 115A graphics that differ for the respective options are indicated in the chart by S and H, and A and C, respectively. Graphics not so marked are the same in both options.

Transmission Codes

The notations in the code columns of the chart for the various terminal types represent the System/360 byte bit pattern equivalents of the applicable transmission codes. The applicable transmission codes are:

IBM 1030: Perforated Tape and Transmission Code

IBM 1050: Perforated Tape and Transmission Code

IBM 1060: Perforated Tape and Transmission Code

IBM 2260: IBM 2260 Transmission Code

IBM 2740: Perforated tape and transmission code

IBM 7770/7772: Audio Response Unit Code (ARU code)

AT&T 83B3: 5-level Baudot Code

WU 115A: 5-level Baudot Code

WU TWX: 8-level TWX Code

WTTA: 5-level ITA2 code
5-level ZSC3 code

Representation of Characters and Bit Patterns

Appearance of a character and its associated bit pattern in a character set signifies that the appropriate IBM-provided translate tables effect incoming translation (that is, translation of that character to the corresponding EBCDIC character), outgoing translation (that is, translation of the corresponding EBCDIC character to that character), or both. How the bit pattern appears indicates which of these cases applies:

1. Where the hexadecimal representation of the bit pattern appears in brackets, only incoming translation is performed.

2. Where in parentheses, only outgoing translation is performed.

3. Where the bit pattern is not enclosed by brackets or parentheses, both incoming and outgoing translation is performed.

Because each unique bit pattern for a terminal character can be represented in an incoming translate table only once, the character associated with the bit pattern can be translated to only one EBCDIC character. The converse is not true, however. Any one transmission code bit pattern can be placed any number of times within an outgoing translate table. Therefore, any number of EBCDIC characters can be translated to the terminal character represented by that bit pattern.

Appearance of two bit patterns opposite a single character signifies that the character has both an uppercase and a lowercase bit pattern, and that both forms of the character are translated to the same EBCDIC character.

Example: The bit pattern of the NL character appears in location 21/9. Both the lowercase and uppercase bit patterns of this character are translated to the EBCDIC NL character when they appear in an incoming message. When an EBCDIC NL character appears in an outgoing message, QTAM translates it to the lowercase form of the NL character.

Where more than one EBCDIC character requires translation to the same character in a terminal character set, the terminal character appears an equivalent number of times in the column. (For example, locations 0/26, 6/26, 7/26, 23/26 and 50/26 all contain the LTRS character.)

Where a character appears in both the graphics and the controls columns for a terminal type, its function depends on whether it is sent when the line is in control mode or in text mode. Depending on the type of terminal and the mode, the character may perform a control function, print as a graphic, or both. For details, see the reference manuals for the various terminal types.

Nonequivalent Characters

Designing the system to accommodate terminal types having different character sets and control functions has resulted in several instances where dissimilar characters have been equated in translate tables. This accounts for the appearance in certain rows of this chart of nonequivalent characters; for example, in rows 3, 38, and 50.

In other instances, the same or similar functions have different names among the various terminal types; for example, HT and Tab in row 5 are equivalent, as are DEL and Rubout, in row 7.

In a few instances, terminals using the same transmission code have different meanings assigned to the identical bit pattern; for example, bit pattern 79 in the transmission code has the meaning PF for an IBM 1050, and Subtract, for an IBM 1060.

Substitutions

Where blank positions appear in the terminal character set portion of the chart, there is no equivalent character for the EBCDIC character or bit pattern at the left of the chart. Where these blanks appear, the SUB character is to be assumed (they were omitted to make the chart more readable). That is, in each translate table that handles incoming messages, each posi-

tion representing an invalid transmission code bit pattern (that is, one not used by a character in the terminal's character set) contains the EBCDIC code (3F) for the SUB character. In each translate table that handles outgoing messages,

1. each position that represents an invalid EBCDIC bit pattern (a pattern to which no EBCDIC character has been assigned), and

2. each position that represents a bit pattern for a character having no equivalent in the destination terminal's character set

contains the transmission code bit pattern for a substitute graphic. For the IBM 1050 and 2260, and the AT&T 83B3 and WU 115A, this substitute character is a colon (:). For the IBM 1030 and 1060, the WTTA terminals and the WU TWX, it is a slash (/).

General Notes

1. Standard abbreviations are used to represent the control characters. The full names of the characters are given. See the reference manuals for the various terminals for descriptions of these characters.

2. "Circle" characters (B , D , etc.) in the chart are alternate names for the characters after which they appear.

3. Notes pertaining to specific characters or bit patterns are indicated by superscript numerals next to the character or bit pattern. The notes follow, and indicate the chart locations to which they apply.

4. Most of the characters in the S and H character set options (1030) and in the A and C character set options (83B3, 115A) are identical. Where they differ between the options, the translate tables favor the S option and the A option, as illustrated in the chart. If messages from an H option 1030 are sent only to another H option 1030, the translate table may be used as is, and similarly, for the 83B3/115A, with respect to the C option. If messages from terminals with the H or C option are to be exchanged with other terminal types, the user may wish to modify the tables.

5. Some TWX terminals send even parity transmission code bit patterns. Others send nonparity bit patterns. All bit patterns sent by nonparity machines have a 1 in the low-order bit position (that is, the position that serves as the parity bit in even parity machines). The RCVTWX translate table translates either a nonparity or an even parity bit pattern to the EBCDIC bit pattern for the corresponding character. For those characters whose even parity and nonparity bit patterns are identical, a single bit pattern appears in Column 30 of the chart. For example, a single pattern, X'C3', appears in location 195/30. For those characters whose even parity and nonparity bit patterns differ by the setting of the low order bit, two bit patterns appear, as for example, in location 193/30. Where two bit patterns appear, the one enclosed in [] is the nonparity bit pattern. The [] indicates that the nonparity bit patterns are only received from TWX terminals. In outgoing message transmission, the SNDTWXE translate table sends even parity bit patterns, while the SNDTWXO translate table sends nonparity bit patterns.

Notes

[1]Left bracket translates to EBCDIC hex 79. No EBCDIC character has been assigned to this bit pattern (location 121/3, 121/28).

[2]No graphic prints in the A character set option (location 90/25).

[3]Backslash translates to EBCDIC hex E1. No EBCDIC character has been assigned to this bit pattern (locations 225/3, 225/28).

[4]IBM 1030 sends the numeric 0 as a hex 20. 1033 receives the numeric 0 as a hex 15 (location 240/4).

[5]Right bracket translates to EBCDIC hex 49. No EBCDIC character has been assigned to this bit pattern (location 73/3, 73/28).

Control Characters

ACK	Acknowledge
Ⓔ	End-of-Block (same as EOB)
BEL	Bell
BS	Backspace
BYP	Bypass
Ⓒ	End-of-Transmission (same as ECT)
CAN	Cancel
CC	Cursor Control
CR	Carriage (carrier) Return
Ⓐ	Machine End-of-Address (same as ECA)
DC1	
DC2	Device Controls
DC4	
DEL	Delete
DLE	Data Link Escape
DS	Digit Select
EM	End of Medium
ENQ	Enquiry
ECA	End-of-Address
ECB	End-of-Block
ECC	End-of-Card
ECFC	End-of-First-Card
ECM	End-of-Message
ECT	End-of-Transmission
ETB	End-Transmission-Block
ETX	End-of-Text
FF	Forms Feed
FIGS	Figures shift
FS	(EBCDIC hex 22) Field Separator
HT	Horizontal Tabulate
IFS	Interchange File Separator
IGS	Interchange Group Separator
II	Idle
IRS	Interchange Record Separator
IUS	Interchange Unit Separator
LC	Lowercase Shift
LF	Line Feed
LF-CR	Line Feed-Carriage Return
LTRS	Letters shift
MZ	Minus Zero
Ⓝ	Negative Response to polling, addressing, or LRC/VRC
NAK	Nonacknowledge
NL	New Line
NUL	Null
PF	Punch Off
PN	Punch On
PRE	Prefix
PZ	Plus Zero
RES	Restore
RM	Record Mark
RS	(EBCDIC hex 35) Reader Stop
Ⓢ	Start-of-Address
SI	Shift In
SM	Set Mode
SMI	Start Manual Input
SO	Shift Out
SOH	Start-of-Header
SMM	Start Manual Message
SOS	Start-of-Significance
SP	Space
STX	Start-of-Text
SUB	Substitute
SYN	Synchronous Idle
Tab	Tabulate (horizontal)
TM	Tape Mark
TpAuxOff	Tape Auxiliary Off
TpAuxOn	Tape Auxiliary On
UC	Uppercase Shift
VT	Vertical Tabulate
WRU	Who Are You?
X-Off	Transmitter Off
X-On	Transmitter On
Ⓨ	Positive Response to polling, addressing, or LRC/VRC

Ref.	EBCDIC Character Graphic	EBCDIC Character Control	EBCDIC Code (Hex)	IBM 1030 Character Graphic	IBM 1030 Character Control	IBM 1030 Code (Hex)	IBM 1050 Character Graphic	IBM 1050 Character Control	IBM 1050 Code (Hex)	IBM 1060 Character Graphic	IBM 1060 Character Control	IBM 1060 Code (Hex)	IBM 2260 2260 Character Graphic	IBM 2260 2260 Character Control	IBM 2260 2260 Code (Hex)	IBM 2260 1053 Character Graphic	IBM 2260 1053 Character Control	IBM 2260 1053 Code (Hex)	IBM 2740 Character Graphic	IBM 2740 Character Control	IBM 2740 Code (Hex)	IBM 7770/7772 Character Graphic	IBM 7770/7772 Character Control	IBM 7770/7772 Code (Hex)	AT&T 83 B3 W U 115A Character Graphic	AT&T 83 B3 W U 115A Character Control	AT&T 83 B3 W U 115A Code (Hex)	AT&T TWX Character Graphic	AT&T TWX Character Control	AT&T TWX Code (Hex) Even	AT&T TWX Code (Hex) Non	WTTA (1TA2) Character Graphic	WTTA (1TA2) Character Control	WTTA (1TA2) Code (Hex)	WTTA (ZSC3) Character Graphic	WTTA (ZSC3) Character Control	WTTA (ZSC3) Code (Hex)	EBCDIC Code (Hex)	Ref.
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	
0		NUL	00		Pad	(DF)		IL	(5E)		IL	(5E)								IL	(5E)					LTRS	(1F)		Rubout	(FF) (FF)		LTRS	(1F)		LTRS	(1F)	00	0
1		SOH	01	s # H =	Ⓢ	(37)								SOH	01		SOH	(01)		Ⓢ	(37)															01	1
2		STX	02		EOA(Ⓓ)	(16)	#	EOA(Ⓓ)	(16)	#	EOA(Ⓓ)	(16)		STX	02		STX	(02)	#	EOA(Ⓓ)	(16)					CR	(02)		CR	(B1) (B1)		CR	(02)		CR	(02)	02	2
3		ETX	03		EOB(Ⓑ)	(3D)			(3D)		EOB(Ⓑ)	(3D)		ETX	03		ETX	(03)		EOB(Ⓑ)	(3D)															03	3
4		PF	04					PF	79 [F9]		Subtr	(79)																TpAuxOff	28 29							04	4
5		HT	05		HT	(7A)		Tab	7A [FA]		Tab	(7A)								HT	7A [FA]								HT	90 91							05	5
6		LC	06		Pad	(DF)		Dwnshft	7C [FC]		IL	(5E)								Dwnshft	7C [FC]					LTRS	1F [3F]					LTRS	(1F)		LTRS	(1F)	06	6
7		DEL	07		EOC DEL	7F		DEL	7F [FF]		DEL	7F								DEL	7F [FF]					LTRS	(1F)		Rubout	FF FF		LTRS	(1F)		LTRS	(1F)	07	7
8			08																																	08	8
9			09																																	09	9
10		SMM	0A										▶	Start MI	[BD]	¢		(BD)																		0A	10
11		VT	0B																									VT	01 01							0B	11
12		FF	0C																									FF	30 31							0C	12
13		CR	0D		LF-CR	(5B)		NL	(5B)		CR	(5B)		NL	(0A)		NL	(0A)		NL	(5B)					CR	(02)		CR	(B1) (B1)		CR	(02)		CR	(02)	0D	13
14		SO	0E					BYP	(3B)																			SO	71 71							0E	14
15		SI	0F					RES	(5B)																			SI	F0 F1							0F	15
16		DLE	10																																	10	16
17		DC1	11																									X-On	88 89							11	17
18		DC2	12																																	12	18
19		TM	13																																	13	19
20		RES	14					RES	5B [DB]	*		(5B)								NL	5B [DB]					LF	(08) [28]									14	20
21		NL	15		LF-CR	(5B)		NL	5B [DB]		CR	(5B)	◢	NL	0A		NL	(0A)		BS	5D [DD]								LF	[50] [51]			CR LF		02 [22]	15	21
22		BS	16					BS	5D [DD]											IL	5E [DE]															16	22
23		IL	17		Pad	(DF)		IL	5E [DE]		IL	(5E)													LTRS	(1F)		Rubout	(FF)			LTRS	(1F)				17	23
24		CAN	18											CAN	[18]								CU1	[22]												18	24
25		EM	19										■	Check	42	"		(42)																		19	25
26		CC	1A																																	1A	26
27		CU1	1B																																	1B	27
28		IFS	1C																																	1C	28
29		IGS	1D																																	1D	29
30		IRS	1E																																	1E	30
31		IUS	1F																																	1F	31
32		DS	20																																	20	32
33		SOS	21																																	21	33
34		FS	22																																	22	34
35			23																																	23	35
36		BYP	24					BYP	3B [BB]											LF	3B [BB]					LF	(08)		LF	(50) [51]			LF	08 [28]		LF	08 [28]	24	36
37		LF	25		LF	(3B)		LF	3B [BB]		LF	(3B)								EOB	3D [BD]					CR	[02] [22]		CR	B1			LF			LF		25	37
38		ETB (EOB)	26		EOB(Ⓑ)	3D		EOB	3D [BD]		EOB(Ⓑ)	3D		ETX	(03)		ETX	(03)					EOB	[24]													26	38
39		ESC (PRE)	27					PRE	3E [BE]																											27	39
40			28																				CU2	[21]												28	40
41			29																																	29	41
42		SM	2A																																	2A	42
43		CU2	2B																									Bell	3A					Bell	2C	2B	43
44			2C																																	2C	44
45		ENQ	2D																									WRU	A0 A1							2D	45
46		ACK	2E											ACK	06		ACK	06																			2E	46
47		BEL	2F																						c'	ABell	3A		Bell	E1 E1							2F	47
48			30																																	30	48
49			31																																	31	49
50		SYN	32		Pad	(DF)		IL	(5E)		IL	(5E)								IL	(5E)					LTRS	(1F)		Rubout	(FF) (FF)		LTRS	(1F)		LTRS	(1F)	32	50
51			33																																	33	51
52		PN	34					PN	19 [99]																			TpAuxOn	48 49							34	52
53		RS	35					RS	1A [9A]																											35	53
54		UC	36					Upshift	1C [9C]											Upshft	1C [9C]					FIGS	1B [3B]						FIGS	1B [3B]				36	54
55		EOT	37		EOT(©)	1F		EOT(©)	1F		EOT(©)	(1F)		EOT (©)	04		EOT(©)	04		EOT(©)	1F		EOT	[23]			(25)		EOT	21 21			LTRS	(1F)		LTRS	(1F)	37	55
56			38																				CU3	[1C]												38	56
57			39																																	39	57
58			3A																																	3A	58
59		CU3	3B																																	3B	59
60		DC4	3C																									X-Off	C9 C9							3C	60
61		NAK	3D											NAK	15		NAK	[15]																			3D	61
62			3E																																	3E	62
63		SUB	3F	/		(23)	:		(88)	/		(23)	:		(5A)	:		(5A)	:		(88)						(37)			(5C) (5D)						(2A)	3F	63
64		SP	40		SP	01		SP	01 [81]		SP	01		SP	40		SP	(40)		SP	01 [81]		SP	[0C]		SP	04 [24]		SP	05 05			SP	04 [24]		SP	04 [24]	40	64
65			41																																	41	65
66			42																																	42	66
67			43																																	43	67
68			44																																	44	68
69			45																																	45	69
70			46																																	46	70
71			47																																	47	71

Ref.	EBCDIC Character Graphic	EBCDIC Character Control	EBCDIC Code (Hex)	IBM 1030 Character Graphic	IBM 1030 Character Control	IBM 1030 Code (Hex)	IBM 1050 Character Graphic	IBM 1050 Character Control	IBM 1050 Code (Hex)	IBM 1060 Character Graphic	IBM 1060 Character Control	IBM 1060 Code (Hex)	IBM 2260 (2260) Character Graphic	IBM 2260 (2260) Character Control	IBM 2260 (2260) Code (Hex)	IBM 2260 (1053) Character Graphic	IBM 2260 (1053) Character Control	IBM 2260 (1053) Code (Hex)	IBM 2740 Character Graphic	IBM 2740 Character Control	IBM 2740 Code (Hex)	IBM 7770/7772 Character Graphic	IBM 7770/7772 Character Control	IBM 7770/7772 Code (Hex)	AT&T 83B3 W U 115A Character Graphic	AT&T 83B3 W U 115A Character Control	AT&T 83B3 W U 115A Code (Hex)	AT&T TWX Character Graphic	AT&T TWX Character Control	AT&T TWX Code (Hex) Even	AT&T TWX Code (Hex) Non	WTTA (ITA2) Character Graphic	WTTA (ITA2) Character Control	WTTA (ITA2) Code (Hex)	WTTA (Z5C3) Character Graphic	WTTA (Z5C3) Character Control	WTTA (Z5C3) Code (Hex)	EBCDIC Code (Hex)	Ref.
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30		31	32	33	34	35	36	37	
72			48																																		48	72
73			49[9]																								]5										49	73
74	¢		4A				¢	Ⓨ	A0	.	Ⓨ	(76)	.		4E	.		(4E)	¢	Ⓨ	A0								BB	BB						4A	74
75	.		4B		Ⓨ EOFC	76	·		76										·		76					27			74	75	.		2F	.		21	4B	75
76	<		4C				<		84				<		5C	<		(5C)	<		84								3C	3D							4C	76
77	(		4D				(		93				(		48	+		(48)	(		93				A(c1/2		3E			14	15	(		3E	(		3E	4D	77
78	+		4E				+		E1				+		48			(48)	+		E1									D4	D5	+		31	+		38	4E	78
79	|		4F				|		B7				|		FE			(FE)	|		B7								7B	7B							4F	79
80	&		50	s& H+		61	&		61	+		(61)	&		46	&		(46)	&		61	&	(7770 only)	[CO]	&		2B	&		65	65				&		2B	50	80
81			51																																		51	81
82			52																																		52	82
83			53																																		53	83
84			54																																		54	84
85			55																																		55	85
86			56																																		56	86
87			57																																		57	87
88			58																																		58	88
89			59																																		59	89
90	!		5A				!		D7										!		D7				A2 c1/4		36	!		B4	85						5A	90
91	$		5B	$		57	$		57	$		(57)	$		44	$		(44)	$		57				$		32	$		24	25						5B	91
92	*		5C				*		90				*		4A	*		(4A)	*		90							*		55	55						5C	92
93	)		5D				)		95				)		49	)		(49)	)		95				A) c3/4		29	)		95	95	)		29	)		29	5D	93
94	;		5E				;		87				;		5B			(5B)			87				A2 c3/8		2F	;		DD	DD						2F	5E	94
95	¬		5F				¬	Ⓨ	F6				¬		FC	¬		(FC)	¬	Ⓨ	F6																	5F	95
96	-		60	-	Ⓝ	40	-	Ⓝ	40	-	Ⓝ	(40)	-		4D	-		(4D)	-	Ⓝ	40	-	(7770 only)	[80]	-		38	-		B4	B5	-		38	-		30	60	96
97	/		61	/		23	/		23	/		23	/		4F	/		(4F)	/		23	/		[49]	/		37	/		F5	F5	/		37	/		2A	61	97
98			62																																		62	98
99			63																																		63	99
100			64																																		64	100
101			65																																		65	101
102			66																																		66	102
103			67																																		67	103
104			68																																		68	104
105			69																																		69	105
106		EOM	6A										▬	EOM	41	!		(41)																			6A	106
107	,		6B		Ⓢ	37	,		37	,		(37)	,		4C	,		(4C)	,	Ⓢ	37				A, c7/8		26	,		35	35	,		26	,		26	6B	107
108	%		6C				%		8B				%		45	%		(45)	%		8B							%		A5	A5						6C	108
109	—		6D				—	Ⓝ	C0				—		BF	—		(BF)	—	Ⓝ	C0							←		FA	FB						6D	109
110	>		6E				>		8E				>		5E	>		(5E)	>		8E							>		7D	7D						6E	110
111	?		6F				?		A3				?		5F	?		(5F)	?		A3				A? c5/8		33	?		FC	FD	?		33	?		25	6F	111
112			70																																		70	112
113			71																																		71	113
114			72																																		72	114
115			73																																		73	115
116			74																																		74	116
117			75																																		75	117
118			76																																		76	118
119			77																																		77	119
120			78																								[1		DB	DB						78	120
121			79[3]																								:		5C	5D						79	121
122	:	EOA	7A	s# H=	EOA(Ⓓ)	16	:	EOA(Ⓓ)	88	#	EOA(Ⓓ)	16	:		5A	:		(5A)	:	EOA (Ⓓ)	88				A: c1/8	CR	2E	#		C5	C5	:		2E	,		23	7A	122
123	#		7B	s@ H'		(20)	#		16		Add	(20)	#		43	#		(43)	#		16						(02)										7B	123
124	@		7C				@		20				@		E0	@		(E0)	@	EOA (Ⓓ)	20							@		03	03						7C	124
125	'		7D				'		8D				'		47	'		(47)	'		8D				A'	c Bell	34	'		E4	E5	,		34	,		34	7D	125
126	=		7E				=		82				=		5D	=		(5D)	=		82							=		BD	BD	=		2F	=		2F	7E	126
127	"		7F				"	EOA (Ⓓ)	96				"			"			"		96				"		31	"		44	45							7F	127
128			80																																		80	128
129	a		81	A		(62)	a		62	A		(62)	A		(A1)	a		(A1)	a		62				A		(18)	A		(82)	(83)	A		(18)	A		(18)	81	129
130	b		82	B		(64)	b		64	B		(64)	B		(A2)	b		(A2)	b		64				B		(13)	B		(42)	(43)	B		(13)	B		(13)	82	130
131	c		83	C		(67)	c		67	C		(67)	C		(A3)	c		(A3)	c		67				C		(0E)	C		(C3)	(C3)	C		(0E)	C		(0E)	83	131
132	d		84	D		(6B)	d		68	D		(68)	D		(A4)	d		(A4)	d		68				D		(12)	D		(22)	(23)	D		(12)	D		(12)	84	132
133	e		85	E		(6B)	e		6B	E		(68)	E		(A5)	e		(A5)	e		6B				E		(10)	E		(A3)	(A3)	E		(10)	E		(10)	85	133
134	f		86	F		(6D)	f		6D	F		(6D)	F		(A6)	f		(A6)	f		6D				F		(16)	F		(63)	(63)	F		(16)	F		(16)	86	134
135	g		87	G		(6E)	g		6E	G		(6E)	G		(A7)	g		(A7)	g		6E				G		(0B)	G		(E2)	(E3)	G		(06)	G		(08)	87	135
136	h		88	H		(70)	h		70	H		(70)	H		(A8)	h		(A8)	h		70				H		(05)	H		(12)	(13)	H		(05)	H		(05)	88	136
137	i		89	I		(73)	i		73	I		(73)	I		(A9)	i		(A9)	i		73				I		(0C)	I		(93)	(93)	I		(0C)	I		(0C)	89	137
138			8A																																		8A	138
139			8B																																		8B	139
140			8C																																		8C	140
141			8D																																		8D	141
142			8E																																		8E	142
143			8F																																		8F	143

	EBCDIC			IBM 1030			IBM 1050			IBM 1060			IBM 2260						IBM 2740			AT&T 83B3 W U 115A			AT&T TWX			WTTA (ITA2)			WTTA (ZSC3)			EBCDIC	
													2260			1053																			
	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Code	
	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	(Hex)	
Ref.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	Ref.
144			90																														90	144
145	i		91	J		(43)	i		43	J		(43)	J		(AA)	J		(AA)	i		43	J		(1A)	J		(53)	J		(1A)	J		(1A)	91	145
146	k		92	K		(45)	k		45	K		(45)	K		(AB)	K		(AB)	k		45	K		(1E)	K		(D2)	K		(1E)	K		(1E)	92	146
147	l		93	L		(46)	l		46	L		(46)	L		(AC)	L		(AC)	l		46	L		(09)	L		(33)	L		(09)	L		(09)	93	147
148	m		94	M		(49)	m		49	M		(49)	M		(AD)	M		(AD)	m		49	M		(07)	M		(B2)	M		(07)	M		(07)	94	148
149	n		95	N		(4A)	n		4A	N		(4A)	N		(AE)	N		(AE)	n		4A	N		(06)	N		(72)	N		(06)	N		(06)	95	149
150	o		96	O		(4C)	o		4C	O		(4C)	O		(AF)	O		(AF)	o		4C	O		(03)	O		(F3)	O		(03)	O		(03)	96	150
151	p		97	P		(4F)	p		4F	P		(4F)	P		(B0)	P		(B0)	p		4F	P		(0D)	P		(0A)	P		(0D)	P		(0D)	97	151
152	q		98	Q		(51)	q		51	Q		(51)	Q		(B1)	Q		(B1)	q		51	Q		(1D)	Q		(B8)	Q		(1D)	Q		(1D)	98	152
153	r		99	R		(52)	r		52	R		(52)	R		(B2)	R		(B2)	r		52	R		(0A)	R		(4B)	R		(0A)	R		(0A)	99	153
154			9A																														9A	154
155			9B																														9B	155
156			9C																														9C	156
157			9D																														9D	157
158			9E																														9E	158
159			9F																														9F	159
160			A0																														A0	160
161			A1																														A1	161
162	s		A2	S		(25)	s		25	S		(25)	S		(B3)	S		(B3)	s		25	S		(14)	S		(CA)	S		(14)	S		(14)	A2	162
163	t		A3	T		(26)	t		26	T		(26)	T		(B4)	T		(B4)	t		26	T		(01)	T		(2B)	T		(01)	T		(01)	A3	163
164	u		A4	U		(29)	u		29	U		(29)	U		(B5)	U		(B5)	u		29	U		(1C)	U		(AA)	U		(1C)	U		(1C)	A4	164
165	v		A5	V		(2A)	v		2A	V		(2A)	V		(B6)	V		(B6)	v		2A	V		(0F)	V		(6A)	V		(0F)	V		(0F)	A5	165
166	w		A6	W		(2C)	w		2C	W		(2C)	W		(B7)	W		(B7)	w		2C	W		(19)	W		(EB)	W		(19)	W		(19)	A6	166
167	x		A7	X		(2F)	x		2F	X		(2F)	X		(B8)	X		(B8)	x		2F	X		(17)	X		(1B)	X		(17)	X		(17)	A7	167
168	y		A8	Y		(31)	y		31	Y		(31)	Y		(B9)	Y		(B9)	y		31	Y		(15)	Y		(9A)	Y		(15)	Y		(15)	A8	168
169	z		A9	Z		(32)	z		32	Z		(32)	Z		(BA)	Z		(BA)	z		32	Z		(11)	Z		(5A)	Z		(11)	Z		(11)	A9	169
170			AA																														AA	170
171			AB																														AB	171
172			AC																														AC	172
173			AD																														AD	173
174			AE																														AE	174
175			AF																														AF	175
176			B0																														B0	176
177			B1																														B1	177
178			B2																														B2	178
179			B3																														B3	179
180			B4																														B4	180
181			B5																														B5	181
182			B6																														B6	182
183			B7																														B7	183
184			B8																														B8	184
185			B9																														B9	185
186			BA																														BA	186
187			BB																														BB	187
188			BC																														BC	188
189			BD																														BD	189
190			BE																														BE	190
191			BF																														BF	191
192		PZ	C0					PZ	75		Restore	(75)																					C0	192
193	A		C1	A		62	A		E2	A		62	A		A1	A		(A1)	A		E2	A		18	A		82 [B3]	A		18	A		18	C1	193
194	B		C2	B		64	B		E4	B		64	B		A2	B		(A2)	B		E4	B		13	B		42 [43]	B		13	B		13	C2	194
195	C		C3	C		67	C		E7	C		(67)	C		A3	C		(A3)	C		E7	C		0E	C		C3	C		0E	C		0E	C3	195
196	D		C4	D		68	D		E8	D		(68)	D		A4	D		(A4)	D		E8	D		12	D		22 [23]	D		12	D		12	C4	196
197	E		C5	E		6B	E		EB	E		68	E		A5	E		(A5)	E		EB	E		10	E		A3	E		10	E		10	C5	197
198	F		C6	F		6D	F		ED	F		6D	F		A6	F		(A6)	F		ED	F		16	F		63	F		16	F		16	C6	198
199	G		C7	G		6E	G		EE	G		(6E)	G		A7	G		(A7)	G		EE	G		0B	G		E2 [E3]	G		0B	G		0B	C7	199
200	H		C8	H		70	H		F0	H		(70)	H		A8	H		(A8)	H		F0	H		05	H		12 [13]	H		05	H		05	C8	200
201	I		C9	I		73	I		F3	I		73	I		A9	I		(A9)	I		F3	I		0C	I		93	I		0C	I		0C	C9	201
202			CA																														CA	202
203			CB																														CB	203
204			CC																														CC	204
205			CD																														CD	205
206			CE																														CE	206
207			CF																														CF	207
208		MZ	D0					MZ	54		Message	(54)																					D0	208
209	J		D1	J		43	J		C3	J		43	J		AA	J		(AA)	J		C3	J		1A	J		53	J		1A	J		1A	D1	209
210	K		D2	K		45	K		C5	K		45	K		AB	K		(AB)	K		C5	K		1E	K		D2 [D3]	K		1E	K		1E	D2	210
211	L		D3	L		46	L		C6	L		(46)	L		AC	L		(AC)	L		C6	L		09	L		33	L		09	L		09	D3	211
212	M		D4	M		49	M		C9	M		(49)	M		AD	M		(AD)	M		C9	M		07	M		82 [B3]	M		07	M		07	D4	212
213	M		D5	N		4A	N		CA	N		(4A)	N		AE	N		(AE)	N		CA	N		06	N		72 [73]	N		06	N		06	D5	213
214	O		D6	O		4C	O		CC	O		(4C)	O		AF	O		(AF)	O		CC	O		03	O		F3	O		03	O		03	D6	214
215	P		D7	P		4F	P		CF	P		(4F)	P		B0	P		(B0)	P		CF	P		0D	P		0A [0B]	P		0D	P		0D	D7	215

	EBCDIC			IBM 1030			IBM 1050			IBM 1060			IBM 2260						IBM 2740			AT&T 83B3 W U 115A			AT&T TWX			WTTA (ITA2)			WTTA (ZSC3)			EBCDIC	
													2260			1053																			
	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code	Character		Code (Hex)		Character		Code	Character		Code	Code
Ref.	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	(Hex)	Graphic	Control	Even Non		Graphic	Control	(Hex)	Graphic	Control	(Hex)	(Hex)	Ref.
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27		28	29	30	31	32	33	34	
216	Q		D8	Q		51	Q		D1	Q		(51)	Q		B1	Q		(B1)	Q		D1	Q		1D	Q		8B 8B		Q		1D	Q		1D	D8	216
217	R		D9	R		52	R		D2	R		(52)	R		B2	R		(B2)	R		D2	R		0A	R		4B 4B		R		04	R		0A	D9	217
218			DA																																DA	218
219			DB																																DB	219
220			DC																																DC	220
221			DD																																DD	221
222			DE																																DE	222
223			DF																																DF	223
224		RM	E0					RM	34																										E0	224
225			E1[3]																						[3]		3A 3B								E1	225
226	S		E2	S		25	S		A5	S		(25)	S		B3	S		(B3)	S		A5	S		14	S		CA CB		S		14	S		14	E2	226
227	T		E3	T		26	T		A6	T		(26)	T		B4	T		(B4)	T		A6	T		01	T		2B 2B		T		01	T		01	E3	227
228	U		E4	U		29	U		A9	U		(29)	U		B5	U		(B5)	U		A9	U		1C	U		AA AB		U		1C	U		1C	E4	228
229	V		E5	V		2A	V		AA	V		(2A)	V		B6	V		(B6)	V		AA	V		0F	V		6A 6B		V		0F	V		0F	E5	229
230	W		E6	W		2C	W		AC	W		(2C)	W		B7	W		(B7)	W		AC	W		19	W		EB EB		W		19	W		19	E6	230
231	X		E7	X		2F	X		AF	X		(2F)	X		B8	X		(B8)	X		AF	X		17	X		1B 1B		X		17	X		17	E7	231
232	Y		E8	Y		31	Y		B1	Y		(31)	Y		B9	Y		(B9)	Y		B1	Y		15	Y		9A 9B		Y		15	Y		15	E8	232
233	Z		E9	Z		32	Z		B2	Z		(32)	Z		BA	Z		(BA)	Z		B2	Z		11	Z		5A 5B		Z		11	Z		11	E9	233
234			EA																																EA	234
235			EB																																EB	235
236			EC																																EC	236
237			ED																																ED	237
238			EE																																EE	238
239			EF																																EF	239
240	0		F0	0[4]		(15) [20]	0		15	0		15	0		50	0		(50)	0		15	0		2D	0		DC 0D		0		2D	0		28	F0	240
241	1		F1	1		02	1		02	1		02	1		51	1		(51)	1		02	1		3D	1		8D 8D		1		3D	1		3C	F1	241
242	2		F2	2		04	2		04	2		04	2		52	2		(52)	2		04	2		39	2		4D 4D		2		39	2		3A	F2	242
243	3		F3	3		07	3		07	3		07	3		53	3		(53)	3		07	3		30	3		CC CD		3		30	3		39	F3	243
244	4		F4	4		08	4		08	4		08	4		54	4		(54)	4		08	4		2A	4		2D 2D		4		2A	4		36	F4	244
245	5		F5	5		08	5		0B	5		0B	5		55	5		(55)	5		0B	5		21	5		AC AD		5		21	5		35	F5	245
246	6		F6	6		0D	6		0D	6		0D	6		56	6		(56)	6		0D	6		35	6		6C 6D		6		35	6		33	F6	246
247	7		F7	7		0E	7		0E	7		0E	7		57	7		(57)	7		0E	7		3C	7		ED ED		7		3C	7		27	F7	247
248	8		F8	8		10	8		10	8		10	8		58	8		(58)	8		10	8		2C	8		1D 1D		8		2C	8		2E	F8	248
249	9		F9	9		13	9		13	9		13	9		59	9		(59)	9		13	9		23	9		9C 9D		9		23	9		2D	F9	249
250			FA																																FA	250
251			FB																																FB	251
252			FC																																FC	252
253			FD																																FD	253
254			FE																																FE	254
255			FF																																FF	255

TERMINAL CODE TRANSLATION CHART

This chart may be used in reading the terminal code found in dumps of storage. The hexadecimal representation of the terminal code, as found in a dump, is shown at the side of each section of the chart. Beneath the terminal type is found the desired character to which the terminal code translates; also shown is the EBCDIC translation. The programmer must determine if the hexadecimal code in main storage represents EBCDIC (translated) or terminal code (untranslated).

Example: In order to translate

```
1601E4CC A5011515 150201CA B1E70190
```

as found in a dump, the characters are first separated into pairs:

```
16 01 E4 CC A5 01 15 15
15 02 01 CA B1 E7 01 90
```

If the terminal is an IBM 1050, the chart shows that the characters in storage translate to

```
EOA SP B O S SP 0 0

0 1 SP N Y C SP *
```

so that the message entered at the terminal was, in part,

```
BOS 0001 NYC *
```

S/360 Byte (Hex)	EBCDIC	IBM 1030	IBM 1050	IBM 1060	IBM 2260 R		IBM 2740	AT&T 8383 WU 115A	AT&T TWX	WTTA		S/360 Byte (Hex)
					2260	1053				ITA2	ZSC3	
	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	
00	NUL											00
01	SOH	SP	SP	SP		SOH	SP			T	T	01
02	STX	1	1	1	STX	STX	1			CR	CR	02
03	ETX				ETX	ETX			@	O	O	03
04	PF	2	2	2	EOT	EOT	2			SP	SP	04
05	HT								SP	H	H	05
06	LC				ACK	ACK				N	N	06
07	DEL	3	3	3			3			M	M	07
08		4	4	4			4			LF	LF	08
09										L	L	09
0A	SMM				◢ NL	NL			P	R	R	0A
0B	VT	5	5	5			5		P	G	G	0B
0C	FF											0C
0D	CR	6	6	6			6			I	I	0D
0E	SO	7	7	7			7		0	P	P	0E
0F	SI									C	C	0F
										V	V	
10	DLE	8	8	8			8			E	E	10
11	DC1									Z	Z	11
12	DC2								H	D	D	12
13	TM	9	9	9			9		H	B	B	13
14	RES								(	S	S	14
15	NL	0	0	0			0		(	Y	Y	15
16	BS	# EOA	# EOA	# EOA	NAK	NAK	# EOA			F	F	16
17	IL									X	X	17
18	CAN				CAN					A	A	18
19	EM									W	W	19
1A	CC		PN							J	J	1A
1B	CU1		RS						X	FIGS	FIGS	1B
1C	IFS									U	U	1C
1D	IGS		Upshift				Upshift			Q	Q	1D
1E	IRS								8	K	K	1E
1F	IUS	EOT	EOT	EOT			EOT			LTRS	LTRS	1F
20	DS											20
21	SOS									5		21
22	FS								D	CR	CR	22
23									D	9	:	23
24	BYP											24
25	LF									SP	SP	25
26	ETB (EOB)											26
27	ESC(PRE)											27
28												28
29									TpAuxOff	LF	LF	29
2A	SM								TpAuxOff	)	)	2A
2B	CU2								T	4	0	2B
2C												2C
2D	ENQ										Bell	2D
2E	ACK								4	8	9	2E
2F	BEL									0	8	2F
30												30
31												31
32	SYN								FF	3	-	32
33									FF	+		33
34	PN											34
35	RS		RM									35
36	UC											36
37	EOT	, (S)	,				, (S)		,	6	5	37
38												38
39			BYP									39
3A												3A
3B	CU3	LF	LF	LF			LF			Bell	FIGS	3B
3C	DC4											3C
3D	NAK											3D
3E		EOB	EOB	EOB			EOB					3E
3F	SUB		PRE							LTRS	LTRS	3F

S/360 Byte (Hex)	EBCDIC	IBM 1030	IBM 1050	IBM 1060	IBM 2260 R		IBM 2740	AT&T 8383 WU 115A	AT&T TWX WU 115A	WTTA		S/360 Byte (Hex)
					2260	1053				ITA2	ZSC3	
	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	Gr Ctl	
40												40
41		SP	SP	SP	SP	SP						41
42					EOM				B			42
43					Check				B			43
44									"			44
45					$	$			"			45
46		K	k		%	%	k					46
47		L	l		&	&	l					47
48					'	'						48
49					(	(			TpAuxOn			49
4A	¢	M	m		)	)	m		TpAuxOn			4A
4B	.	N	n		*	*	n					4B
4C	<				+	+			R			4C
4D	(	O	o		,	,	o					4D
4E	+				-	-						4E
4F	!	P	p		.	.	p		2			4F
50	&				/	/						50
51		Q	q		0	0	q		LF			51
52		R	r		1	1	r		LF			52
53					2	2			J			53
54			MZ	Message	3	3						54
55					4	4						55
56					5	5			*			56
57		$	$	$	6	6	$					57
58					7	7						58
59			RES	*	8	8						59
5A	!				9	9			Z			5A
5B	$	LF-CR	NL	CR	:	:	NL		Z			5B
5C	*				;	;			:			5C
5D	)		BS		<	<	BS		:			5D
5E	;		IL	IL	=	=	IL					5E
5F	¬				>	>						5F
60	-				?	?						60
61	/											61
62												62
63		A	a	A			a		F			63
64												64
65		B	b	B			b		&			65
66												66
67		C	c	C			c					67
68												68
69		D	d	D			d					69
6A	EOM								V			6A
6B	,	E	e	E			e		V			6B
6C	%								6			6C
6D	_	F	f	F			f		6			6D
6E	>	G	g	G			g					6E
6F	?											6F
70		H	h	H			h					70
71									SO			71
72		I	i	I			i		N			72
73									N			73
74									.			74
75			PZ	Restore					.			75
76		. (Y) EOFC	. (Y)	(Y)			(Y)					76
77												77
78												78
79			PF	Subtr								79
7A	:	HT	Tab	Tab			HT		↑			7A
7B	# EOA											7B
7C	@		Dwnshft				Dwnshft		>			7C
7D	'											7D
7E	=											7E
7F	"	EOC DEL	DEL	DEL			DEL					7F

S/360 Byte (Hex)	EBCDIC		IBM 1030		IBM 1050		IBM 1060		IBM 2260 R				IBM 2740		AT&T 83B3 W U 115A		AT&T TWX		WTTA				S/360 Byte (Hex)
									2260		1053								ITA2		ZSC3		
	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	
80																							80
81	a					SP								SP									81
82	b				=								=				A						82
83	c																A						83
84	d				<								<				!						84
85	e																!						85
86	f				;								;										86
87	g																						87
88	h				:								:				X-On						88
89	i																X-On						89
8A																							8A
8B					%								%				Q						8B
8C																							8C
8D					'								'				]						8D
8E					>								>										8E
8F																							8F
90					*								*				HT						90
91	j																HT						91
92	k																						92
93	l				(								(				[						93
94	m																						94
95	n				)								)				)						95
96	o																						96
97	p				"	EOA							"	EOA									97
98	q																						98
99	r					PN																	99
9A						RS											Y						9A
9B																	Y						9B
9C						Upshift								Upshift			9						9C
9D																	9						9D
9E																							9E
9F																							9F
A0					¢								¢				WRU						A0
A1									A		A						WRU						A1
A2	s								B		B												A2
A3	t				?				C		C		?				E						A3
A4	u								D		D												A4
A5	v				S				E		E		S				%						A5
A6	w				T				F		F		T										A6
A7	x								G		G												A7
A8	y								H		H												A8
A9	z				U				I		I		U										A9
AA					V				J		J		V				U						AA
AB									K		K						U						AB
AC					W				L		L		W				5						AC
AD									M		M						5						AD
AE									N		N												AE
AF					X				O		O		X										AF
B0									P		P						CR						B0
B1					Y				Q		Q		Y										B1
B2					Z				R		R		Z				M						B2
B3									S		S						M						B3
B4									T		T						-						B4
B5									U		U						-						B5
B6									V		V												B6
B7									W		W												B7
B8						BYP			X		X												B8
B9									Y		Y												B9
BA									Z		Z												BA
BB						LF								LF			]						BB
BC						EOB								EOB			=						BC
BD						PRE																	BD
BE																							BE
BF									_		_												BF

S/360 Byte (Hex)	EBCDIC		IBM 1030		IBM 1050		IBM 1060		IBM 2260 R				IBM 2740		AT&T 83B3 W U 115A		AT&T TWX		WTTA				S/360 Byte (Hex)
									2260		1053								ITA2		ZSC3		
	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	Gr	Ctl	
C0		PZ				Ⓝ								Ⓝ									C0
C1	A				_								_										C1
C2	B																						C2
C3	C				J								J					C					C3
C4	D																						C4
C5	E				K									K				#					C5
C6	F				L									L									C6
C7	G																						C7
C8	H																						C8
C9	I				M									M				X-Off					C9
CA					N									N			S						CA
CB																	S						CB
CC					O									O			3						CC
CD									▶Start MI		¢						3						CD
CE																							CE
CF					P									P									CF
D0		MZ																					D0
D1	J				Q									Q				VT					D1
D2	K				R									R			K						D2
D3	L																K						D3
D4	M																+						D4
D5	N																+						D5
D6	O																						D6
D7	P				!									!									D7
D8	Q					RES																	D8
D9	R																						D9
DA																							DA
DB						NL								NL			[						DB
DC						BS								BS			0						DC
DD						IL								IL									DD
DE																	;						DE
DF			Pad																				DF
E0		RM							"		"												E0
E1					+								+					Bell					E1
E2	S				A								A				G						E2
E3	T																G						E3
E4	U				B								B				'						E4
E5	V																'						E5
E6	W																						E6
E7	X				C								C										E7
E8	Y				D								D										E8
E9	Z																						E9
EA																							EA
EB					E								E				W						EB
EC					F								F				7						EC
ED					G								G										ED
EE																							EE
EF																							EF
F0	0				H								H										F0
F1	1																	SI					F1
F2	2																						F2
F3	3				I								I				O						F3
F4	4																						F4
F5	5				⌐	Ⓨ							⌐	Ⓨ			/						F5
F6	6																						F6
F7	7																						F7
F8	8																						F8
F9	9					PF								HT			←						F9
FA						Tab											←						FA
FB																							FB
FC						Dwnshft			⌐		⌐			Dwnshft			?						FC
FD																	?	SI					FD
FE									I		I												FE
FF						DEL								DEL				Rubout					FF

APPENDIX F: EXCHANGING MESSAGES BETWEEN IBM AND NON-IBM TERMINALS

This appendix is not applicable to audio message handling.

Certain line and device control functions are implemented differently for IBM terminals and non-IBM terminals. Generally, no difficulties arise when messages are exchanged between IBM terminals of the same or different types, or between non-IBM terminals of the same type. For applications in which messages are to be exchanged between non-IBM terminals of dissimilar types, or between IBM and non-IBM terminals, the user should be aware of the considerations explained here, and plan his message headers accordingly. In some cases it is necessary to edit certain characters or character sequences out of incoming messages and edit certain characters or sequences into outgoing messages. The functions concerned are carriage return, line feed, end of address, end of block, end of transmission, and who are you? (the latter function applies to TWX and WTTA terminals).

End-of-Address

All QTAM-supported IBM terminals employ a single machine end-of-address (EOA) character, known as a D . Of the non-IBM terminals, the 83B3 represents EOA by the sequence CR LF LTRS; the 115A represents it by a single space character; the TWX terminals have no EOA sequence; and WTTA terminals have no machine EOA sequence.

The first character in the message header sent to a Western Union Plan 115A terminal must be an EOA character (X'40' before a TRANS macro instruction or X'04' after a TRANS macro instruction). There must not be any excess idle characters preceding the EOA character. The QTAM user may ensure this by:

1. reserving one more idle character in the message header, via the LPSTART macro instruction, in addition to any idle characters reserved for time-of-day, current-date, or output sequence number information; and

2. moving an EOA character into the first data byte of the header (the thirty-second byte of the buffer) in the Send Header subgroup of the LPS. If used before a TRANS macro instruction, this may be done as follows:

```
MVI IJLQBHDR,X'40'
```

If used after a TRANS macro instruction, X'04' should be inserted.

If messages are to be switched from a non-IBM terminal to an IBM terminal, the user must edit out the received EOA character or sequence and insert the proper sequence for the receiving terminal. Figure 35 provides the code representations for the EOAs for each terminal type.

For all IBM terminals except the 2740 Types A and F , there may be two EOAs (two STXs for 2260) as the first two characters transmitted to the terminal. The first one is sent by the access method which sends the message to the terminal, while the second appears in core as the first character in the buffer (following idle characters specified in the LPSTART macro-instruction). This second EOA was sent by the terminal as the first character in the message.

The EOA character transmitted by the access method will perform its normal function of putting the terminal in text mode, but the second EOA will print as a text mode character at the beginning of the message. The user may wish to insert the following code to delete this character before transmitting the message:

```
   CLI  1(5),X'7B' IS THIS EOA CHARACTER
   BNE  NO         NO IT IS NOT
   MVI  1(5),X'17' YES, REPLACE WITH IDLE
   LA   5,1(0,5)   INCREMENT SCAN POINTER
NO .
   .
   .
```

Carriage Return, Line Feed, New Line, and End-of-Block

For non-IBM terminals, the carriage return and line feed functions are performed by two separate characters, CR and LF. For IBM terminals, the functions are performed by the single character, new line (NL). A NL character in a message sent from an IBM terminal to a non-IBM terminal cannot be translated to two separate characters; that is, to both the CR and LF characters. To compensate for this, QTAM takes advantage of the usual practice of sending an EOB character at the end of each line of text printed on the printer of an

IBM terminal (that is, sending an EOB followed by a NL). Standard QTAM translate tables effect conversion of the EOB and NL characters to LF and CR, respectively, for messages sent from an IBM terminal to a non-IBM terminal. Conversely, CR and LF characters sent by a non-IBM terminal are converted to EOB and NL characters, when the message is sent to an IBM terminal. Thus, as long as any messages originating from an IBM terminal always use the EOB and NL characters in combination, the carriage return and line feed functions at the receiving terminal (non-IBM) are effected, just as if the originating terminal had entered CR and LF into the message.

End-of-Transmission and WRU

All IBM terminals employ a single end-of-transmission (EOT) character, called a C . TWX terminals also employ a single EOT character. The 83B3 and 115A terminals represent EOT by the sequence FIGS H LTRS.

An EOT in a message sent from an IBM or TWX terminal to an 83B3 or 115A is translated by QTAM to the two-character sequence FIGS H -- (the LTRS character is not sent), so the EOT sequence is not complete. The sequence is completed when QTAM deselects the terminal before polling the line or addressing a terminal on the line. When QTAM sends the EOT character that always begins a polling or addressing operation, the TCU first sends the LTRS character, completing the EOT sequence. The TCU then sends the complete EOT sequence FIGS H LTRS again. The EOT sequence thus appears on the receiving line twice, but this has no ill effect.

The EOT sequence FIGS H LTRS sent from an 83B3 or 115A terminal to an IBM terminal appears in main storage as an upshift H (transmission code X'25'). The TCU has deleted the LTRS character from the incoming data stream and converted the characters FIGS H to the single character, upshift H. The upshift H is treated as an invalid character by the QTAM translate table; it is translated to a substitute character (X'3F', in EBCDIC). The user should edit this substitute character out of the message. When the message is sent to the destination IBM terminal, the user should edit into it the appropriate EOT character. Figure 39 provides the code representations for the EOTs for each terminal type.

For messages sent from a TWX terminal to an IBM terminal, the user must edit out an X-off character and replace it with an EOT character (so the terminal will not remain in text mode).

The user must edit out all WRU characters appearing in message destined for TWX terminal (DOS/QTAM does not support the WRU function in outgoing messages to TWX terminals). The user should also edit out EOT characters appearing in messages destined for TWX terminals, because EOT will cause the terminal to disconnect from the line prematurely (that is, while QTAM is preparing to send additional messages to the same terminal). See the section, Management of Nonaudio Switched Lines.

Terminal Type		EOA Sequence			EOT Sequence		
		Characters	Trans Code (hex)	EBCDIC (hex)	Characters	Trans Code (hex)	EBCDIC (hex)
IBM	2260	STX	02	02	EOT (©)	04	37
	All others	EOA (Ⓔ)	16	7B	EOT (©)	1F	37
Non-IBM	83B3	CR LF LTRS	02081F	0D2506	FIGS H LTRS	1B051F	368806
	115A	Space	04	40	FIGS H LTRS	1B051F	368806
	TWX				EOT	21	37
	WTTA				EOM EOT	Note Note	37 37

Note: Any character assigned by the user.

Figure 39. EOA and EOT Characters and Sequences

Upon return to the routine that issued the macro instruction, the following return codes are set in the low-order byte, right-adjusted, in register 15. All numbers in Figure 40 appear in hexadecimal notation.

Macro	Normal Return	Unopened DTF	Invalid Disk Address	Line not Intercepted	Invalid Relative Line Number	Invalid Count	Invalid Terminal Table Entry or DTF Name	Invalid Sequence Number
CHNGP	X'00'	X'01'			X'08'	X'10'	X'20'	
CHNGT	X'00'					X'10'	X'20'	
COPYC	X'00'	X'01'					X'20'	
CCPYP	X'00'	X'01'			X'08'		X'20'	
CCPYQ	X'00'						X'20'	
CCPYT	X'00'						X'20'	
STARTARU	X'00'	X'01'			X'08'		X'20'	
STARTLN	X'00'	X'01'			X'08'		X'20'	
STOPARU	X'00'	X'01'			X'08'		X'20'	

Figure 40. Return Codes for Macro Instructions Used to Modify and Examine System Status

A format illustration accompanies each macro instruction description in this publication. The illustrations indicate which operands must be coded exactly as shown, which are required, which are variable, etc. The conventions stated to describe the operands are as follows:

1. Keyword operands are described by a three-part structure that consists of the keyword, followed by an equals sign (both of which must be coded exactly as shown), followed by either:

   - an uppercase keyword which must be coded as shown, or
   - a lowercase term which must be replaced by an allowable expression as indicated in the chart or in the macro description, or
   - a combination of the above.

Examples: TYPE=PQ, CPCTL=chars, DEVADDR=SYSnnn. Note that there are no blanks in the three-part structure.

2. Positional operands are described either by:

   - a single uppercase term which must be coded exactly as shown, or
   - a single lowercase term, which must be replaced by an allowable expression as indicated in the chart or in the macro description.

Example: SOURCE, filename

3. Upper-case letters and punctuation marks (except as described in these conventions) represent information that must be coded exactly as shown.

4. Lower-case letters and terms represent information that must be supplied by the programmer. More specifically, n indicates a decimal number, nn a decimal number with at most two digits, nnn a decimal number with at most three digits, etc.

5. An ellipsis (a comma followed by three periods) indicates that a variable number number of items may be included.

6. { } Options contained within braces represent alternatives, one of which must be chosen.

7. [ ] Information contained within brackets represents an option that can be included or omitted, depending on the requirements of the program.

8. [A / B / C] Underlined elements represent an assumed value in the event a parameter is omitted.

Abbreviations Used in Foldout Chart

Abbreviation	Meaning
SYM	An X in this column indicates that the operand may be any symbol valid in the Assembler Language
DEC DIG	An X in this column indicates that the operand may be any decimal digits, up to the value indicated in the associated macro instruction description
REGISTER	An X in this column indicates that the operand specifies a general register, always coded within parentheses, as follows: (2-12)- one of general registers 2 through 12, previously loaded with the right-adjusted value or address indicated in the macro instruction description. The unused high-order bits must be set to zero. The register may be designated symbolically or with an absolute expression. (1)- general register 1, previously loaded as indicated above. The register can be designated only as (1).
RX type	An X in this column indicates that the operand is coded as any address that is valid in an RX-type instruction (e.g., LA).
REL EXP	An X in this column indicates that the operand is coded as a relocatable expression (acceptable as an A-type or V-type address constant by the assembler).
CHAR	An X in this column indicates that the operand is coded as a character string, without framing characters and quotes. An F in this column indicates that framing characters, such as C' ', CLn' ', etc., are coded. Refer to the macro description.
HEX CHAR	An X in this column indicates that the operand is coded as a hexadeciaml character string, without framing characters and quotes. An F in this column indicates that framing characters, such as X' ', are coded. Refer to the macro description.
W/S	An X in this column indicates that the operands are to be coded exactly as shown in the OPERANDS column.

MACRO INSTRUCTION	OPERANDS	WRITTEN AS									
				Register							
		Sym	Dec Dig	(2-12)	(1)	RX Type	Rel Exp	Abs Exp	Char	Hex Char	W/S
ARUMGTYP	typechar								F	F	
BREAKOFF	nnnnn		X								
BUFARU	n		X								
	length		X								
BUFFER	nnn		X								
	length		X								
	mmm		X								
	brb		X								
CANCELM	mask									F	
CHECKARU	mask									F	
	=X'message'								F		
	msgchar					X					
CHNGP	filename			X			X				
	rln		X	X							
	workarea			X		X					
	=C'0'										X
	=C'1'										X
CHNGT	termname			X			X				
	workarea			X		X					
CLOSE	filename			X			X				
COPYC	termname (nonaudio)			X			X				
	rln (nonaudio)		X								
	linename (audio)			X					X		
	workarea			X		X					
COPYP	filename			X			X				
	rln		X	X							
	workarea			X		X					
COPYQ	termname			X			X				
	workarea			X		X					
COPYT	termname			X			X				
	workarea			X		X					
COUNTER	subfield	X									

MACRO INSTRUCTION	OPERANDS	WRITTEN AS									
				Register							
		Sym	Dec Dig	(2-12)	(1)	RX Type	Rel Exp	Abs Exp	Char	Hex Char	W/S
DIRECT	=CLn'dest'								F		
	subfield	X									
DTFQT	all operands	Refer to Macro Description									
EOA	eoa								F	F	
ERRMSG	mask									F	
	CLn'dest'								F		
	subfield	X									
	SOURCE										X
	message								F		
	msgchar					X					
INTERCPT	mask									F	
	subfield	X									
LINE	filename	X									
	rln		X								
LINETBL	entry						X				
	n		X								
LIST	entry						X				
LOGSEG	filename				X		X				
	,PREFIX (nonaudio)	Refer to Macro Description									
	,ARU (audio)	Refer to Macro Description									
LPSTART	nn (nonaudio)		X								
	TERM=	X									
	INTRCPT=	X									
MODE	PRIORITY										X
	CONVERSE										X
	INITIATE										X
	MOD2260										X
	condchar								F	F	
	WRT60=code	Refer to Macro Description									
MSGTYPE	typechar								F	F	
OPCTL	CTLMSG=								X		
	TERM=								X		

MACRO INSTRUCTION	OPERANDS	WRITTEN AS									
				Register							
		Sym	Dec Dig	(2-12)	(1)	RX Type	Rel Exp	Abs Exp	Char	Hex Char	W/S
OPCTL (CONT)	,ALTERM=								X		
	,INTRCPT	X									
OPEN	filename			X			X				
OPTION	typelength							X			
PAUSE	ctlchar									F	
	insertchar	Refer to Macro Description									
POLL	entry						X				
	AUTOPOL=	Refer to Macro Description									
	polladdr									X	
	nid									X	
	2740										X
POLLIMIT	nnn		X								
	subfield						X				
PROCESS		Refer to Macro Description									
REPEAT	codechar								F	F	
REROUTE	mask									F	
	=CLn'dest'								F		
	subfield						X				
	SOURCE										X
ROUTE	n		X								
SEQIN	n		X								
SEQOUT	n		X								
SKIP	n		X								
	skipchrs								F	F	
SOURCE	n		X								
STARTARU	filename			X			X				
	rln		X	X							
	ALL										X
	ctltbl						X				

MACRO INSTRUCTION	OPERANDS	WRITTE				
				Register		
		Sym	Dec Dig	(2-12)	(1)	RX Ty
STARTLN	filename			X		
	rln		X	X		
	ALL					
	ctltbl					
STOPARU	filename			X		
	rln		X	X		
	ALL					
	ctltbl					
TERM	qtype	Refer to Macr				
	filename					
	rln		X			
	adchars					
	opdata	Refer to Macr				
	CALL=	Refer to Macr				
	ID=					
	DEVADDR=SYSnnn					
	PRNTR=YES					
TERMTBL	entry					
	n		X			
	OPCTL=					
	CPINTV=		X			
	CONF=	Refer to Macr				
TIMESTMP	nn		X			
TRANS	table	X				
WORD	diskaddr					
	length					
WORDTBL	entry					
WTTA MACROS		Refer to Macr				

	WRITTEN AS									
			Register							
	Sym	Dec Dig	(2-12)	(1)	RX Type	Rel Exp	Abs Exp	Char	Hex Char	W/S
								F	F	
		X								
		X								
		X								
		X								
		X								
		X								
		X								
									F	
									F	
								F		
					X					
			X			X				
		X	X							
			X		X					
										X
										X
			X			X				
			X		X					
			X			X				
udio)			X			X				
		X								
)			X					X		
			X		X					
			X			X				
		X	X							
			X		X					
			X			X				
			X		X					
			X			X				
			X		X					
	X									

MACRO INSTRUCTION	OPERANDS	WRITTEN AS									
				Register							
		Sym	Dec Dig	(2-12)	(1)	RX Type	Rel Exp	Abs Exp	Char	Hex Char	W/S
DIRECT	=CLn'dest'								F		
	subfield	X									
DTFQT	all operands	Refer to Macro Description									
EOA	eoa								F	F	
ERRMSG	mask									F	
	CLn'dest'								F		
	subfield	X									
	SOURCE										X
	message								F		
	msgchar					X					
INTERCPT	mask									F	
	subfield	X									
LINE	filename	X									
	rln		X								
LINETBL	entry						X				
	n		X								
LIST	entry						X				
LOGSEG	filename				X		X				
	,PREFIX (nonaudio)	Refer to Macro Description									
	,ARU (audio)	Refer to Macro Description									
LPSTART	nn (nonaudio)		X								
	TERM=	X									
	INTRCPT=	X									
MODE	PRIORITY										X
	CONVERSE										X
	INITIATE										X
	MOD2260										X
	condchar								F	F	
	WRT60=code	Refer to Macro Description									
MSGTYPE	typechar								F	F	
OPCTL	CTLMSG=								X		
	TERM=								X		

MACRO INSTRUCTION	OPERANDS	WRITTEN AS									
				Register							
		Sym	Dec Dig	(2-12)	(1)	RX Type	Rel Exp	Abs Exp	Char	Hex Char	W/S
OPCTL (CONT)	,ALTERM=								X		
	,INTRCPT	X									
OPEN	filename			X			X				
OPTION	typelength							X			
PAUSE	ctlchar									F	
	insertchar	Refer to Macro Description									
POLL	entry						X				
	AUTOPOL=	Refer to Macro Description									
	polladdr									X	
	nid									X	
	2740										X
POLLIMIT	nnn		X								
	subfield						X				
PROCESS		Refer to Macro Description									
REPEAT	codechar								F	F	
REROUTE	mask									F	
	=CLn'dest'								F		
	subfield						X				
	SOURCE										X
ROUTE	n		X								
SEQIN	n		X								
SEQOUT	n		X								
SKIP	n		X								
	skipchrs								F	F	
SOURCE	n		X								
STARTARU	filename			X			X				
	rln		X	X							
	ALL										X
	ctltbl						X				

MACRO INSTRUCTION	OPERANDS	WRITTEN AS									
				Register							
		Sym	Dec Dig	(2-12)	(1)	RX Type	Rel Exp	Abs Exp	Char	Hex Char	W/S
STARTLN	filename			X			X				
	rln		X	X							
	ALL										X
	ctltbl						X				
STOPARU	filename			X			X				
	rln		X	X							
	ALL										X
	ctltbl						X				
TERM	qtype	Refer to Macro Description									
	filename						X				
	rln		X								
	adchars									X	
	opdata	Refer to Macro Description									
	CALL=	Refer to Macro Description									
	ID=									X	
	DEVADDR=SYSnnn										X
	PRNTR=YES										X
TERMTBL	entry						X				
	n		X								
	OPCTL=								X		
	CPINTV=		X								
	CONF=	Refer to Macro Description									
TIMESTMP	nn		X								
TRANS	table	X									
WORD	diskaddr									F	
	length							X			
WORDTBL	entry						X				
WTTA MACROS		Refer to Macro Description									

APPENDIX I: SUMMARY OF OPERATOR CONTROL MESSAGES

CONTROL MESSAGE	OPERANDS	WRITTEN AS			
		Dec Dig	Rel Exp	Hex Char	W/S
CHNGT	termname		X		
	data			X	
COPYC*	termname (nonaudio)		X		
	rln (nonaudio)			X	
	linename (audio)		X		
COPYT*	termname		X		
INTERCPT	termname		X		
INTREL	termname		X		
STARTARU	linename		X		
	ALL				X
STARTLN	termname		X		
	ALL				X
STOPARU	linename		X		
	ALL				X

* A response message is returned to the operator control terminal.

Note: The Control Message ID (ctlmsg) is the same for all operator control messages.

Figure 41. Summary of Operator Control Messages

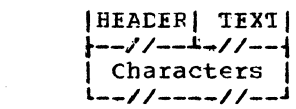
This section discusses certain device-dependent aspects of programming. Some of this material may also be found elsewhere in this document; it is repeated here for the convenience of the programmer.

Warning: This discussion is not necessarily exhaustive: the programmer must still consider all aspects of his application carefully.

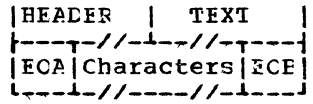
IBM 1030 DATA COLLECTION SYSTEM

Message Formats

Format of message entered at terminal:



Format of message received by message control program:



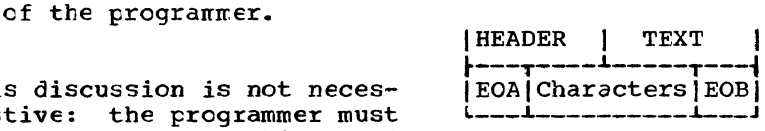
Notes:

1. EOA and EOB are automatically sent by the 1031 control unit when the terminal is polled and has requested to transmit. Because the terminal is polled for each block of a multi-block message, all blocks received by the message control program contain both EOA and EOB.

2. For autopolled terminals, the EOA character is replaced by a 1-byte index.

Format of message sent by message control program must be:

First block:



Other blocks:

Notes:

1. EOA is required only in the first block of a multi-block outgoing message.

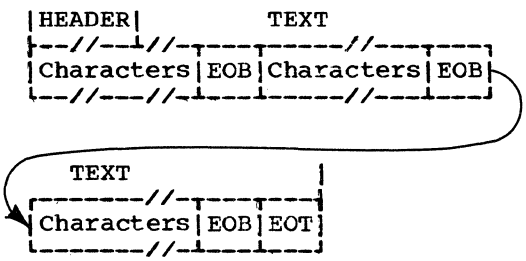
2. All printable characters in an outgoing message toa 1033 printer must be separated by three Write Mark characters (transmission code, X'DF')

Idle Characters: The IBM 1033 Printer requires the insertion of three idle characters (Write Mark) prior to each character transmitted to it.

IBM 1050 DATA COMMUNICATION SYSTEM

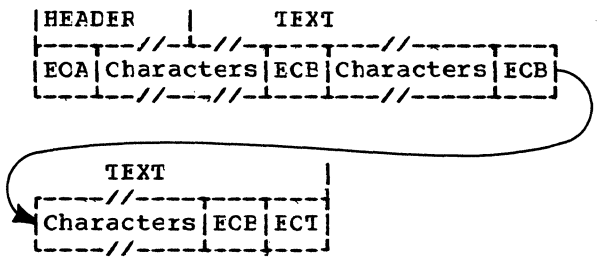
Message formats (switched or nonswitched terminals):

Format of message entered at terminal:



Note: It is usual practice to send EOB followed by NL when communicating with non-IBM terminals, since QTAM translates EOB, NL to CR,LF in sending to non-IBM terminals.

Format of message received by message control program:

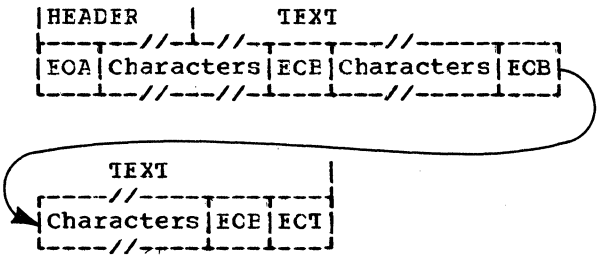


Notes:

1. ECA is automatically sent by the terminal when it is polled and has requested to transmit. Because the terminal is polled for only the first block of a multi-block message, only the first block begins with EOA.

2. For autopolled terminals, the ECA character is replaced by a 1-byte index.

Format of message sent by message control program must be:



Note: ECA is generated by the channel program for this device.

Calls from the Computer to a Switched IBM 1050

After the computer establishes contact with an IBM 1050 on a switched line, QTAM sends the addressing characters specified in the terminal table entry for that terminal. When the terminal returns a positive response, QTAM sends all messages in the queue for that terminal. QTAM then sends the polling characters specified in the polling list for that line and accepts an incoming message from the terminal if it has one ready.

By turning the line around in this manner, QTAM allows the terminal to send during this connection all messages it may have ready. After each incoming message terminated by an EOT (not to be confused with the EOT in a null message), QTAM sends any further messages that may have arrived on the destination queue for the terminal and then polls the terminal again. The procedure of accepting a message from the terminal and then sending any messages that arrive on the destination queue continues until:

1. The computer receives a negative response to the poll; or

2. The terminal sends a null message. That is, a single EOT is sent following the positive response to the poll.

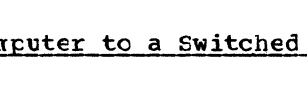
QTAM recognizes either of these as an indication that the terminal has sent its last message (or has no message to send). Note that one terminal is being repeatedly polled instead of a list of terminals being polled in turn.

After all incoming messages have been received from the terminal, QTAM makes a final check to determine if further messages have arrived on the destination queue for the terminal. If so, the procedures for sending the messages and then polling the terminal for incoming messages are repeated during this line connection. When no further messages have arrived on the destination queue, QTAM breaks the line connection and reenables the line for its next use.

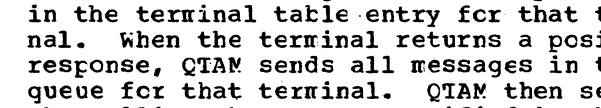
IBM 1060 DATA COMMUNICATION SYSTEM

Message Formats:

Format of message entered at terminal:



Format of message received by the message control program:



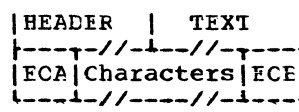
Notes:

1. EOA and EOB are automatically sent by the terminal when it is polled and has requested to transmit. Because the terminal is polled for each block of a multi-block message, all blocks

received by the message control program contain both EOA and EOB.

2. For autopolled terminals, the ECA character is replaced by a 1-byte index.

Format of message sent by the message control program must be:

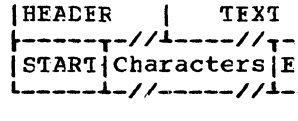


Note: Each outgoing message block must begin with EOA.

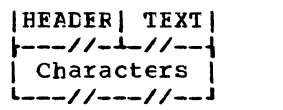
IBM 2260-2848 DISPLAY COMPLEX (LOCAL)

Message Formats

Format of messages entered at the terminal:

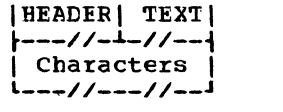


Format of message received by the message control program:



Note: A message received from an IBM 2260 Local through a Read consists entirely of text. The transmitted text consists of all characters displayed between the START and ENTER symbols, excluding any characters to the right of the first NL symbol on a line. There are no line control characters for this device.

Format of message sent by the message control program must be:



Note: A message sent to an IBM 2260 Local through a Write consists entirely of text, except when Write-with-line-address is specified in the MODE macro instruction, in which case the first byte must contain a line address code. The line address code is transmitted but not displayed. A zero length message is allowed and may be used with the ERASE/WRITE to effect erasure of the screen.

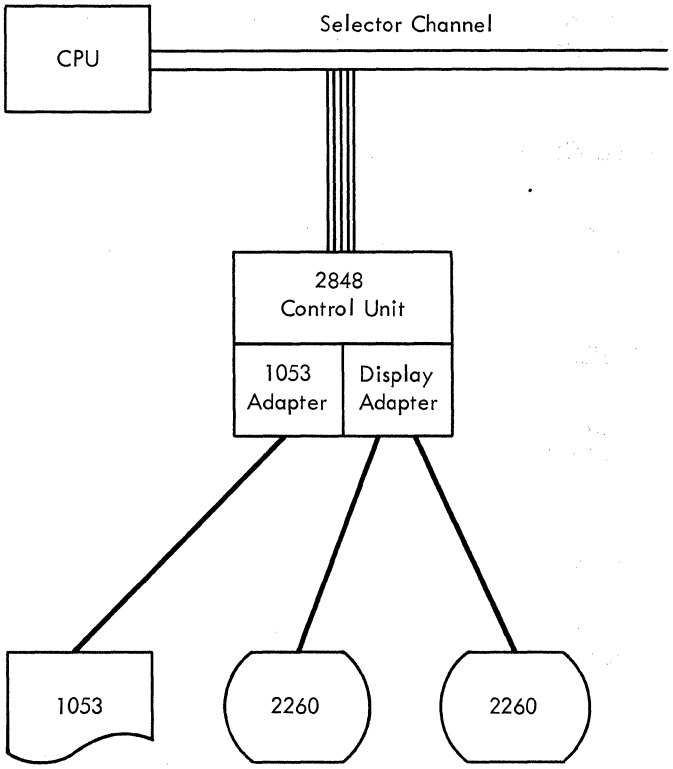
Line Group Definition

The requirements for defining an IBM 2260-2848 Local line group are:

- A DTFQT macro must be specified for each IBM 2848 Display Control. More than one 2848 cannot be defined in the same line group. However, the same 2848 can be defined in more than one line group.
- No IBM 2260 DISPLAY Station or 1053 Printer within the line group can be defined as part of another line group.
- The same number of buffers must be assigned in advance for each transfer of data from a 2260 to the computer.
- The same type of Read operation must be used Read Display Station]DS[Manual Input]MI[or Short Read DS MI for each 2260 in the line group.
- The same LPS must be used for each device in the line group.
- The relative position of the device access area in a terminal table entry must be the same for all terminal table entries associated with the line group (see the section, The Terminal Table).

IBM 2260-2848 Local Operations

QTAM supports the IBM 2260-2848 Display Complex (Local) attached directly either to the multiplexer or a selector channel. Figure 42 shows the configuration of a 2260-2848 Local.



Note: The number of Display Adapters which may be included in a single 2848 Display Control varies.

Figure 42. IBM 2260 Local Configuration

The 2260 Local terminal differs in operation from the other QTAM-supported devices. Because it is locally attached, the 2260 Local is neither polled nor addressed on input/output operations. Instead, when the operator at the 2260 desires to send a message to the CPU, he keys in the START symbol and text and then presses the ENTER key. Pressing the ENTER key results in a CPU I/O interrupt with the Attention bit on in the status. This I/O interrupt is referred to as an Attention interrupt or read request in this publication.

When an Attention interrupt occurs at the CPU, a Command Control Block (CCB) for the 2260 initiating the read request must be in the DOS channel scheduler queue for the Attention interrupt to be honored. The CCB is a part of the terminal table entry for the 2260. If the CCB is not in the queue, the Attention interrupt is ignored.

When a 2260-2848 Local line group is opened, QTAM causes the CCB for each 2260 from which messages can be received to be placed on the channel scheduler queue. Each such 2260 in the line group must be defined in the list of terminals defined by the POLL macro instruction. After the line group has been opened, read requests from the 2260s are serviced automatically on a first-come first-served basis. A message is read into buffers obtained from the QTAM buffer pool and then passed to the Receive Group of the user's LPS section. Starting at this point, processing and further routing of the message is the same as for a message received from a remote terminal. The procedures for sending a message to a 2260 Local terminal (or a 1053 Printer attached to the 2848) are the same as for sending one to a remote terminal except that addressing is not performed.

Each time a transfer of data is completed between the CPU and a 2260 Local, QTAM causes the CCB to remain on the channel scheduler queue if the 2260 is defined for input operations. This action anticipates a subsequent read request from the terminal.

Data transfer can occur between the CPU and only one 2260 Local at a time. For this reason, only one Line Control Block (LCB) is generated for a 2260-2848 Local line group. This LCB contains control information required for I/O operations in the line group and for LPS processing of messages. Read requests (Attention interrupts) from 2260s may occur while the LCB is not available because of LPS processing of a previous message or because QTAM is preparing to send a message to a terminal in the line group. In such cases, QTAM queues the read requests and services them as the LCB becomes available for receiving.

Sending has priority over receiving for the 2260-2848 Local. That is, any time a message appears on the destination queue for the terminals in the group, it is sent before another receiving operation is initiated.

The 2260 Local terminal is designed primarily for inquiry type applications, where the operator enters an inquiry and then waits for a message processing program to construct and return a response. It also may be used for input only or output only applications. Message switching between two or more 2260s or between 2260s and other terminal types is not encouraged. If the operator of a 2260 attempts to enter a message while QTAM is preparing to send a message to the same terminal, his read request is ignored. The message sent by the CPU may obliterate the message the operator has prepared. When this occurs,

the operator must key the message in again and reenter it via an ENTER. This condition may occur frequently in a message switching application where message traffic is heavy. If message switching is necessary, it should be used infrequently and with caution.

It has already been noted that only one ICB is generated for a 2260-2848 Local line group. QTAM treats a 2260-2848 Local line group in basically the same way as it treats a one-line nonswitched line group. When a function is being described that pertains to the 2260-2848 Local, it must be remembered that the word line applies to the entire line group. For example, the description of the STOPLN macro instruction states that "STOPLN removes a line from active use." When this macro is used for a 2260-2848 Local, the entire line group is stopped.

DTFQT Macro Instruction

The INTVL, OPRI, LINELST, and THRESH operands should be omitted.

TERM Macro Instruction

The DEVADDR operand for the TERM macro instruction must be specified (applies also to IBM 1053 terminals in a 2260-2848 Local line group). PRNTR=YES must be specified if the terminal is an IBM 1053 terminal in a 2260-2848 Local line group.

Polling List

One and only one polling list must be specified for a 2260-2848 Local line group; it must contain the names of all IBM 2260 Display Stations in the line group from which messages are to be received. A terminal must not appear in the list more than once. Messages are not accepted from 2260 terminals not defined in the list. If the operator of an undefined terminal attempts to enter a message by pressing the ENTER key, the subsequent Attention interrupt is ignored.

QTAM uses the polling list only at OPEN time, when the CCB for each 2260 Local terminal in the list is queued on the DCS channel scheduler queue in anticipation of receiving a read request (Attention interrupt) from the device. Thereafter, the CCB is always maintained on the channel scheduler queue except when QTAM is sending a message to the terminal.

MODE Macro Instrction

Note the applicable discussion under the MODE macro instruction.

Translation

The 2260 is an EBCDIC device; no translation is necessary.

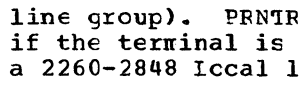
Idle Characters

The insertion of idle characters is not required.

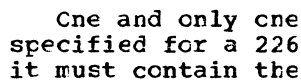
IBM 2260-2848 OR 2265-2845 DISPLAY COMPLEX (REMOTE)

Message Formats

Format of messages to be entered at terminal:

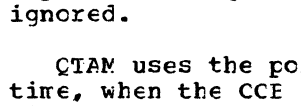


Format of messages received by the message control program:



Note: Characters displayed to the right of the first NL symbol on a line are not transmitted.

Format of messages sent by the message control program must be:



Notes:

1. For write-with-line-address, the first byte following the STX must contain a

line address code, which is transmitted but not displayed.

2. An STX character is generated by the 2260 Remote channel program for the beginning of a message.

Operating the IBM 2260

The high rate of data transfer from the CPU to the 2260 Display Station can cause the display screen to be filled several times before the terminal operator has had time to read the initial display. Additionally, a message sent to the Display Station obliterates any message being formatted by the operator but not yet entered.

If a 1053 Printer is attached to the 2260-2848 Dispaly Complex (remote), messages can be transferred directly from the display screen to the printer by simply pressing the Print key. However, if messages are to be sent from the CPU to the printer, the user should either:

1. Ensure that the previous message sent from the CPU to the printer and any message sent directly from a 2260 to the printer have been completely printed before sending another message from the CPU to the printer; or

2. Control the sending of messages from the CPU to the printer through the use of the INTERCPT and RELEASEM macros described in this publication.

IBM 2740 COMMUNICATION TERMINAL

IBM 2740 Type Codes.

IBM 2740 terminal types are identified to the system (DEVICE operand of DTFQT macro instruction) by means of the following codes. These codes are also used in this section to identify the type of 2740 terminal.

274A: Basic 2740, nonswitched

274B: Basic 2740, switched

274C: Basic 2740 with station control, nonswitched

274D: Basic 2740 with station control and checking, nonswitched.

274E: Basic 2740 with transmit control and checking, switched.

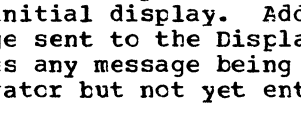
274F: Basic 2740 with checking, nonswitched.

274G: Basic 2740 with checking, switched.

274H: Basic 2740 with transmit control, switched.

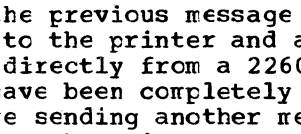
Message Formats (274A and 274B)

Format of message entered at terminal:



Note: The first EOB character encountered in incoming text is treated as an EOT character, precluding transmission of any subsequent blocks of that message.

Format of message received by message control program:



Note: No EOA character is received.

Format of message sent by message control program must be:

Notes:

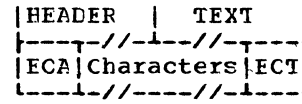
1. EOA is generated by the channel program for this device.

2. The first EOB (or ETX) character encountered in outgoing message text is treated as an EOT character, precluding transmission of any subsequent blocks of that message.

Message Formats (274C)

Format of message entered at terminal: - Same as for 274A and 274B

Format of message received by message control program:

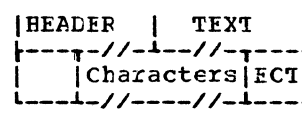


Notes:

1. ECA is automatically sent by the terminal when it is polled and has requested to transmit. Because the terminal is polled for only the first block of a multi-block message, only the first block begins with EOA.

2. For autopolled terminals, the ECA character is replaced by a 1-byte index.

Format of message sent by message control program must be:

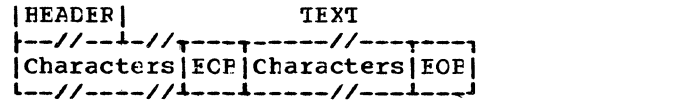


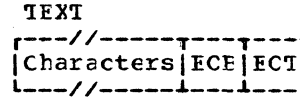
Note: The first ECB (or ETX) character encountered in outgoing message text is treated as an ECT character, precluding transmission of any subsequent blocks of that message.

Message Formats (274D, 274E, 274F, and 274G)

The formats of the messages for these terminals are the same as for the IBM 1050, with the following exception for the 274F and 274G:

Format of message received by message control program:





Note: No EOA character is received.

Message Formats (274H).

The format of the messages for this terminal is the same as for the 274A and 274B for messages entered at the terminal, the same as for the 274C for messages received by the message control program, and the same as for the 274A and 274B for messages sent by the message control program.

DTFQT Macro Instruction

For IBM 2740 terminals, types 274A and 274F, the CPRI operand of the DTFQT macro instruction should be omitted.

Calls from Computer to a Switched IBM 2740

After the computer establishes contact with an IBM 2740 on a switched line, QTAM sends a D and 15 idle characters followed by all waiting messages. QTAM then turns the line around to receive an incoming message in one of the following ways:

1. For the IBM 2740 with transmit control, dial, and checking or transmit control and dial, the terminal is polled with a / and a blank character.

2. For the IBM 2740 basic with dial or dial and checking, the computer writes C C C followed by a Prepare.

After each incoming message terminated by an EOT, QTAM sends any further messages that may have arrived on the destination queue for the terminal, and then receives from the terminal again. The line turnaround procedure continues until the terminal operator presses the Dial Disconnect key.

IBM 2740 MODEL 2 COMMUNICATION TERMINAL

DTFQT Macro Instruction

The IBM 2740 Model 2 terminal is identified to the system by the DEVICE operand of the DTFQT macro instruction as follows:

DEVICE=274D if the terminal has the record checking feature;

DEVICE=274C if the terminal is not equipped with the record checking feature.

TERM Macro Instruction

The qtype operand of the TERM macro instruction should specify queuing by terminal for the 2740 Model 2 in order to reduce unnecessary disk activity.

INTERCPT Macro Instruction

If the terminal is equipped with the Buffer Receive option, the user must specify the INTERCPT macro instruction in the End Send section of the LPS with a mask including the 'message not sent' bit. This is necessary because the terminal may be addressed while a message is being entered at the terminal, resulting in a negative response to addressing. If this happens, the terminal bell rings and the attention light is turned on. The intercepted message will be retransmitted when a positive response to addressing is received.

Multi-block Messages

The IBM 2740 Model 2 is a single message block terminal. If the record checking feature is installed, the terminal expects to receive an ECB at the end of the message block transmission. The terminal then sends a positive response to the Transmission Control Unit and resets the terminal buffer address register to zero. The terminal then expects to receive an EOT, which initiates printing. Therefore, if a multi-block message is sent to the terminal, only the last block is printed and no error condition is returned.

Message Exceeding Buffer Size

If a message sent to the IBM 2740 Model 2 exceeds the size of the terminal buffer, the terminal returns an ECT to indicate a buffer overflow condition. The transmitted message is not printed at the terminal and an error message appears on the IBM 1052 printer-keyboard or at the operator control terminal. The error halfword for the line has the 'incorrect message length' and 'should not occur' bits set.

EOB or EOBLC Macro Instruction

If the 2740 Model 2 is equipped with the record checking feature, the transmission of EOB is a hardware function. Accordingly, either the EOB or the EOBLC macro instruction must be issued in the End receive and End send sections of the LPS.

AUTOPOLLED TERMINALS - GENERAL

The 1-byte autopoll index, received in place of an EOA character, should be replaced by an idle character (X'17') in the Receive Header portion of the user's LPS after any code which uses that byte has been executed.

If this is not done, the index byte may be identical to a control character causing an error in the QTAM Get routine or in a later sending operation.

When receiving from an autopolled line, the source terminals identity is not known to the program. If knowledge of the source terminal is required, the user must use a source macro in the Receive Header section.

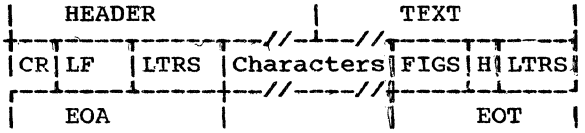
IBM 3944 DIAL TERMINAL

Output of messages from an IBM 3944 is in EBCDIC representation and requires no translation.

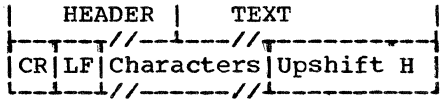
AT&T 83B3 SELECTIVE CALLING STATIONS

Message Formats.

Format of messages entered at the terminal:

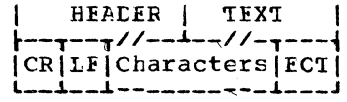


Format of messages received by the message control program:



Note: The TCU has deleted the ITRS character from the input stream and converted FIGS H ITRS to the single character, Upshift H (transmission code X'25').

Format of messages sent by the message control program must be:



Notes:

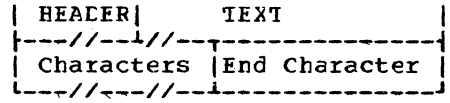
1. The user should complete the EOA sequence by inserting CR LF LTRS.

2. An EOT in a message sent from an IBM or TWX terminal to an 83B3 is translated by QTAM to FIGS H. Since the LTRS character is not sent, so the ECT sequence is not complete. The sequence is completed when QTAM deselects the terminal before polling the line or addressing a terminal on the line. When QTAM sends the ECT character that begins a polling or addressing operation, the TCU first sends the LTRS character, completing the ECT sequence. The TCU then sends the complete ECT sequence, FIGS H LTRS.

TWX TERMINALS

Message Formats

Format of messages entered at the terminal:

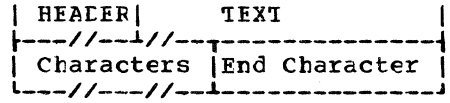


Notes:

1. A CR in an incoming message from a non-IBM terminal is translated to ECB; thus these terminals can in effect transmit multi-block messages.

2. An X-off character will serve as an end character.

Format of messages received by the message control program:

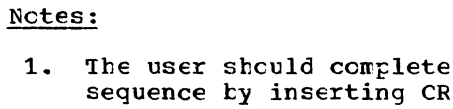


Notes:

1. If X-off is used as an end character and the message to be sent to an IBM terminal, the user should change this character to an IBM EOT character.

2. An EOT character in a message received by a TWX terminal will cause the TWX line to disconnect.

Format of messages sent by message control program must be:



Note: An EOT in a message sent by a TWX terminal will cause the TWX line to disconnect.

Calls from the Computer to a TWX Terminal

After the computer establishes contact with a TWX terminal by dialing its telephone number, the terminal sends its identification sequence. QTAM checks this sequence against the sequence specified in the terminal table entry for the terminal. If they do not match, QTAM sets the MESSAGE NOT SENT bit (bit 12) in the error halfword for the line to indicate an addressing error, and breaks the line connection to the terminal (evidently, a wrong number was reached).

If the two sequences do match, QTAM sends all messages currently on the destination queue for that terminal, then sends the CPU identification sequence (defined in the polling list for that line) to the terminal. The terminal then sends to the computer any messages it may have ready. Each of these messages should end with a transmitter off (X-off) character. Each time a message terminated by this character is received, any further messages that have arrived on the destination queue for the terminal are sent. After these messages are sent (or if no further messages have arrived on the destination queue), QTAM again sends the CPU identification sequence and receives another message from the terminal. When the terminal has sent its last message, it should send an X-off character in response to the CPU identification sequence. When this character is the only character received from the terminal after sending the CPU identification sequence, QTAM recognizes it as an indication that the terminal has no more messages to send. QTAM then makes a final check to determine

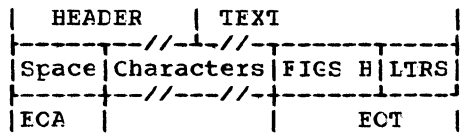
if further messages have arrived on the destination queue for the terminal. If so, the procedures for sending these messages and then accepting incoming messages are repeated during this line connection. When no further messages have arrived on the destination queue, QTAM breaks the connection and reenables the line for its next use.

Restriction: There is a possibility that some message on the destination queue for a TWX terminal will not be sent unless the line connection between the computer and the terminal is terminated by the computer. To avoid this possibility, an ECT (or any other character that causes the connection to be broken prematurely) should not be sent by the terminal nor appear in a message being sent to the TWX terminal.

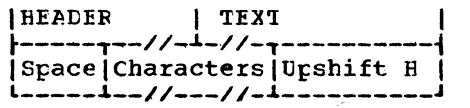
WESTERN UNION 115 A OUTSTATIONS

Message Formats

Format of message entered at terminal:

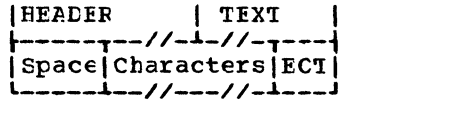


Format of message received by message control program:



Note: The TCU has deleted the LTRS character from the input stream and converted FIGS H LTRS to the single character Upshift H (transmission code X'25').

Format of message sent by message control program must be:



Note: An ECT in a message sent from an IBM or TWX terminal to a 115 A is translated by QTAM to FIGS H. Since the LTRS character is not sent, the ECT sequence is not complete. The sequence is completed when QTAM deselects the terminal before polling the line or addressing a terminal on the line. When QTAM sends the ECT character that begins a polling or addressing operation, the TCU first sends the LTRS character, completing the EOT sequence. The TCU then sends the complete EOT sequence, FIGS H LTRS.

WTTA TERMINALS

WTTA Codes

The WTTA adapter deletes all incoming LTRS or FIGS characters and updates a shift bit (S) which is added to each character transferred to main storage. The adapter examines the shift bit of each outgoing character and automatically generates a LTRS or FIGS character, whenever necessary. Figure 43 illustrates the configurations of a System/360 byte and of a WTTA character:

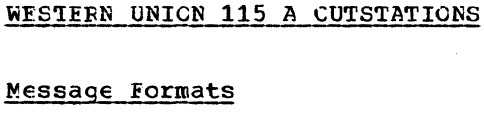


Figure 43. System/360 byte and WTTA-Character Configurations

WTTA terminals use either the International Telegraph Alphabet No.2 (referred to as ITA2) or the Figure Protected Code ZSC3 (referred to as ZSC3). These two codes are illustrated in Figure 44 .

Optional Features

Normally, the motor of a WTTA terminal is off and the first LTRS character sent or received by the terminal starts the motor. The motor needs 1.5 seconds to reach nominal speed. During this interval, the terminal cannot correctly send or receive a character. The motor stops when no character has been transmitted during a period of from 10 to 30 seconds. The terminal is said to be operating in Motor-Off mode. Optionally, the terminal can be equipped with a heavy-duty motor that is never switched off; in this case, the terminal is said to be operating in Motor-on mode.

Most WTTA terminals can be equipped with another optional feature called the Automatic Answerback Unit. This feature enables a string of up to 20 identification characters, generated by a mechanical drum, to be sent over the WTTA line by either pressing the IAM key of the terminal or receiving the character FIGS D (combination

No. 4). For terminals connected to a 2703 Control Unit, the character string must be a multiple of four significant characters (i.e. excluding FIGS and LTRS). The last character of the identification character-string can be any character except LTRS or FIGS. This last character is not a significant part of the terminal identification.

When a WTTA terminal is operating in Motor-off mode, the MCNDLY operand of the DTFQT macro instruction (refer to the section "DTFQT Macro Instruction") enables the user to specify the number of Mark characters corresponding to the 1.5-second interval mentioned above. When QTAM builds a Write channel program, it recognizes the motor mode of the terminal (motor-on or motor-off) and generates a LTRS character (that can be followed by a user-specified number of Mark characters) which precedes the data to be sent over the WTTA line.

Exchange of Identification

Either the CPU or a WTTA terminal can ask for the identification sequence of the other. When an identification exchange is performed, the CPU sends its identification sequence to the terminal (IAM=YES must be specified in the DTFQT macro instruction), and the terminal sends its identification sequence to the CPU (WRU=YES must be specified in the DTFQT macro instruction). An identification exchange can be performed during either:

1. Receiving operations, each time the terminal operator sends the WRU signal to the CPU, or

2. Sending operations, at the beginning of an output message (if the WRU macro instruction is in the Send Header subgroup of the IPS) or at the end of an output message (if the WRU macro instruction is in the End Send subgroup of the IPS).

When the CPU receives the terminal identification sequence, QTAM compares this sequence with that specified in the TERM macro instruction (refer to the section "TERM Macro Instruction").

End-of-Message, End-of-Transmission, and WRU for WTTA Terminals

The World Trade Telegraph Adapter recognizes two end conditions which are set in the hardware at the time the control unit is installed. These are FIGS x and FIGS y LTRS, where x and y are characters assigned by the user of a specific system.

For a terminal equipped with the Automatic Answerback Unit, FIGS x must be the code combination No. 4 (FIGS D) sent by the terminal WRU key. FIGS D is referred to as the WRU signal. For terminals not equipped with the Automatic Answerback Unit, any other code combination can be selected.

Note 1: x and y must not be the same character.

Note 2: The FIGS y LTRS sequence causes a Read operation to end. Therefore, FIGS y can be sent by a terminal as data only if it is not followed by LTRS.

The above termination signals can be used as EOM signals. Either the FIGS y LTRS sequence (if not yet used as an EOM signal) or two consecutive EOM signals can represent the EOT signal.

Code Tables for World Trade Telegraph Terminals

The following figures show, for each character in each of the code sets used by World Trade telegraph terminals, the transmission code bit pattern corresponding to the character, the code combination number representing that bit pattern, and the System/360 byte representation (including shift bit) of that bit pattern.

Letters Shift			Figures Shift				
Character		S/360 Byte Bit Pattern (Hex)	Character		S/360 Byte Bit Pattern (Hex)	Trans-mission Code Elements 12 345	Code Combin-ation
Graphic	Control		Graphic	Control			
A		18	-		38	11 000	1
B		13	?		33	10 011	2
C		0E	:		2E	01 110	3
D		12		WRU	32	10 010	4
E		10	3		30	10 000	5
F		16			36	10 110	6
G		0B			2B	01 011	7
H		05			25	00 101	8
I		0C	8		2C	01 100	9
J		1A		Bell	3A	11 010	10
K		1E	(		3E	11 110	11
L		09	)		29	01 001	12
M		07	.		27	00 111	13
N		06	,		26	00 110	14
O		03	9		23	00 011	15
P		0D	0		2D	01 101	16
Q		1D	1		3D	11 101	17
R		0A	4		2A	01 010	18
S		14	'		34	10 100	19
T		01	5		21	00 001	20
U		1C	7		3C	11 100	21
V		0F	=		2F	01 111	22
W		19	2		39	11 001	23
X		17	/		37	10 111	24
Y		15	6		35	10 101	25
Z		11	+		31	10 001	26
	CR	02		CR	22	00 010	27
	LF	08		LF	28	01 000	28
	LTRS	1F		LTRS	3F	11 111	29
	FIGS	1B		FIGS	3B	11 011	30
	SP	04		SP	24	00 100	31
		00			20	00 000	32

Where no character appears opposite a code combination, no character has been assigned for that combination

Figure 44. International Telegraph Alphabet No. 2 (ITA2)

Letters Shift		S/360 Byte Bit Pattern (Hex)	Figures Shift		S/360 Byte Bit Pattern (hex)	Trans-mission Code Elements 12 345	Code Combin-ation
Character			Character				
Graphic	Control		Graphic	Control			
A		18	+		38	11 000	1
B		13	6		33	10 011	2
C		0E	8		2E	01 110	3
D		12		WRU	32	10 010	4
E		10	-		30	10 000	5
F		16	4		36	10 110	6
G		0B	0		2B	01 011	7
H		05	?		25	00 101	8
I		0C			2C	01 100	9
J		1A	2	Bell	3A	11 010	10
K		1E	(		3E	11 110	11
L		09	)		29	01 001	12
M		07	7		27	00 111	13
N		06	,		26	00 110	14
O		03	:		23	00 011	15
P		0D	9		2D	01 101	16
Q		1D			3D	11 101	17
R		0A	/		2A	01 010	18
S		14	'		34	10 100	19
T		01	.		21	00 001	20
U		1C	1		3C	11 100	21
V		0F	=		2F	01 111	22
W		19	3		39	11 001	23
X		17			37	10 111	24
Y		15	5		35	10 101	25
Z		11			31	10 001	26
	CR	02		CR	22	00 010	27
	LF	08		LF	28	01 000	28
	LTRS	1F		LTRS	3F	11 111	29
	FIGS	1B		FIGS	3B	11 011	30
	SP	04		SP	24	00 100	31
		00			20	00 000	32

Where no character appears opposite a code combination, no character has been assigned for that combination

Figure 45. Figure Protected Code (ZSC3)

APPENDIX K: ON-LINE TERMINAL TEST

Nine tests are provided for IBM 1030, 1050, 1060, 2740, and 2260 (remote) devices. The integer associated with each test description is the code to be entered in the test field to select that test for use.

1 Message Switching. This test receives a message from the requesting terminal, and returns it to the same terminal or to any other terminal as specified in the addr field.

Note: The number of characters that can be switched is directly dependent on the buffer length that the user specifies in the BUFFER macro instruction. If the total length of the test request message exceeds this length, the surplus characters in the test request message are ignored.

2 Tilt. The tilt test is sent to the terminal specified in the addr field. This test is designed to check the SELECTRIC typing element.

3 Rotate. The rotate test is sent to the terminal specified in the addr field. This test is designed to check the SELECTRIC typing element.

4 Twist. The twist test is sent to the terminal specified in the addr field. This test is designed to check the SELECTRIC typing element.

Note: The inability of the SELECTRIC typing element to perform correctly the tilt, rotate, and twist tests is normally detected by observing partially printed characters within the pattern printed during the test.

5 Stored Compare. The text transmitted from the requesting terminal is compared with a stored message in the CPU. The message in storage is compatible with the transmitting capabilities of the terminal(s) involved. The compare message sent from the terminal consists of the numbers 0 through 9 followed by the alphabet (A through Z). The alphabet is entered in lowercase from an IBM 1050 or an IBM 2740.

Exceptions:

1. When transmittal is from an IBM 2740 terminal with station control, a space character must not precede the comparison data. When transmittal is from an IBM 2740 terminal without station control, two space characters must precede the comparison data.

2. The stored compare test for an IBM 1060 is requested by entering the following message:

```
999996534210  TELLER A
              TELLER B
```

Comparison is then made to this message. Response to this request is printed only at the requesting terminal.

The number of characters that can be compared depends directly upon the data length of the buffer that the user specifies in the BUFFER macro instruction. The total length of the test request message must not exceed this length.

If the comparison to the stored message is valid, the following message is sent to the terminal specified in the addr field.

```
CMP VLD-n
```

where n is the last character against which a comparison could be made. If the data length of the buffer as specified in the BUFFER macro instruction is not great enough to hold all the message transmitted, the message is truncated after one buffer is filled and comparison is made only to the contents of that buffer. So long as the text content of that buffer is valid, the comparison is considered valid. However, if the buffer length is so limited that no characters can be compared, n is a slash (/).

Exception: The message sent to an IBM 1060 after a valid comparison is:

```
CMP VLD
```

If the comparison to the stored message is invalid, the data received is

message switched to the terminal specified in the addr field.

Note: The stored compare test is not applicable to the IBM 1030 badge reader or manual entry.

6 All Characters. This is a standard all-characters test for customer engineer terminal checkout and for a "good morning" message for the user. Special characters are not used in this test. Characters received at the terminal are:

1. For IBM 1030, 1060, and 2848 remote (2260 and 1053): numbers 0-9 and alphabet A-Z.
2. For IBM 1050 and 2740: numbers 0-9, alphabet a-z (lowercase), and alphabet A-Z (uppercase).

7 Carriage Mechanism Analyzer. A defined message in storage is used to exercise the terminal specified in order to analyze the capability of the typewriter carriage mechanism to perform within defined specifications. This test is not applicable to an IBM 1053 printer attached to a remote 2848 control unit.

8 Write Line Address (2260 remote only). This is a line selectivity test that uses the first two characters after the unit field (format 0) or the addr field (format 1) as a new line code. These characters can be followed by data that is to be switched to the terminal and written on the line specified on the display station screen. The following characters are used to select the line on the display station screen:

Characters	Line Number
01	1
.	.
.	.
09	9
10	10
11	11
12	12

9 Request Address (2260 remote only). The addr and unit fields are not used in this test. ETX can be sent immediately after the type field. The message returned to the requesting display station is in the following format:

DC + DV dcaddr dvaddr

where:

dcaddr
Is the predefined code necessary to select this display control unit (two bytes).

dvaddr
Is the predefined code necessary to select this display control unit (two bytes).

FORMAT OF TEST REQUEST MESSAGE

All fields of the test request message are consecutive. That is, they are not separated by blanks. The test message should not be preceded with a carriage return, space, or any other character.

If the total length of the test request message exceeds the size of a buffer as specified in the BUFFER macro instruction, the surplus characters in the test request message are ignored.

The format of the test control message is:

```
99999 format-integer
test-integer type-integer
[addr-char(s)] [unit-char(s)]
[text-chars] end-char
```

where:

99999
Is the primary action code used to identify this message to the system as a test request message. This field must always appear exactly as shown in every test request message.

format
Defines the test header format. It is either 0 or 1. Format 0 uses actual line addresses and can be used to address any terminal on the same line. Format 1 uses symbolic addresses and can be used to address any terminal within the system.

test
Specifies the test to be executed. It is always one integer (1 through 9).

type
Specifies the type of terminal for which the test is being requested. Type codes that may be used are shown in the following table:

Terminal Type	Type Code
IBM 1030	1
IBM 1050	2
IBM 1060	3
IBM 1740	4
IBM 1030 Badge reader or Manual Entry	5
IBM 2260 Remote	6

Exception: Type code 5 is used only with format 0. It defines the type of terminal requesting the test (as well as the type of terminal for which the test is being requested).

addr

The address of the terminal

1. To which a test message is to be sent (tests 1, 6, and 8),
2. At which a device to be tested mechanically is located (tests 2, 3, 4, and 7), or
3. To which a response message from the terminal-test facility is sent (test 5). (Test 9 does not utilize the address field.)

Note: For the IBM 1050, 1060, and 2260-2848 remote, two addressing characters are specified in the TERM macro instruction. For these devices, the first of the two addressing characters is the actual address of the terminal, and is therefore the character to be specified in the addr field. The second of the two addressing characters specifies the particular device at the terminal, and is specified in the unit field.

When used with format 0, this is one character or two characters depending on the type of device from which the test request message is being entered. It is a one-character field for the IBM 1030 card reader, IBM 1050 devices, and IBM 1740 devices, and is the addressing character for the selected terminal. Only one character is necessary because these devices are capable of transmitting the actual alphabetic terminal address character.

For the IBM 1030 Badge Reader or manual entry, IBM 1060 devices, and IBM 2260 remote devices, this field must consist of two characters. The address is selected by transmitting a predefined code as follows.

1. IBM 1060:

Terminal Address	Code Entered
A	01
B	02
C	03
.	.
.	.
.	.
Z	26

2. IBM 1030 Badge Reader or manual entry:

Terminal Address	Code Entered
B	02
C	03
D	04
.	.
.	.
.	.
Z	26

Note: If 10 is entered as the addr field, the message is considered an invalid request because the corresponding address (J) is the address for the IBM 1032 Digital Time Unit exclusively, for which no tests are provided.

3. IBM 2260 remote devices: The addr field is used to select the IBM 2848 Display Control Unit. The address of a display control unit can be any ASCII noncontrol character. Therefore, there are 96 possible display control unit addresses.

Terminal Address in ASCII	Code Entered
0100000	01
0100001	02
0100010	03
.	.
.	.
.	.
1111111	96

Note: The predefined code applicable to a particular display control unit can be determined from a display station by utilizing the request address test (test 9).

When used with header format 1, this field is variable in length (from one to nine characters). The first character is a digit defining the number of following characters that constitute the symbolic address name. This symbolic address name defines a terminal in the terminal table.

Examples:

a. 4CHII (four-character symbolic name).

b. 7CHICAGO (seven-character symbolic name).

c. 0 (a zero indicates that the test is to be returned to the requesting terminal).

Note that terminal CHII could request a test for itself by using either example a or c.

unit

Specifies the particular unit at the terminal specified in the addr field. This field is used only when the format 0 is specified. When format 1 is used both the terminal and the unit at the terminal are defined by the symbolic name in the addr field.

For IBM 1050 and IBM 1060 devices, one character is specified. The appropriate code can be determined from the publication describing the type of terminal being addressed.

Note: This field is not applicable to IBM 1030 and IBM 2740 devices. Therefore, text can start in this position.

For IBM 2848 devices, two characters are specified. IBM 2260 Display Stations and IBM 1053 Printers are selected by transmitting a predefined code. The device selection character can be any of 25 ASCII noncontrol characters.

Device Selection Character in ASCII	Code Entered
1000000	01
1000001	02
1000010	03
.	.
.	.
.	.
1011000	25

Note: The predefined code applicable to a particular display station can be determined from a display station by utilizing the request address test (test 9).

text

The text of the message sent as a part of the terminal test. Text is included only when the message switching test (test 1), stored compare test (test 5), or write line address test (test 8) are used.

end

The end character for the device from which the test request message is being transmitted.

Device	End Character
IBM 1030	EOB
IBM 1050	EOT
IBM 1060	EOB
IBM 2740	EOT
IBM 2260	ETX

Note: The header transmitted from an IBM 1060 device is entered by use of the data and transaction keys. The EOB character is entered by pressing the Teller A or Teller B key.

TERMINAL TEST RESTRICTIONS

1. The length of the buffer as specified in the BUFFER macro instruction must be at least 56 bytes in order to contain all the test request header (that is, all the test request message before the text field) plus the header prefix inserted by QTAM.

2. To request a test from an IBM 1030 Badge Reader, the badge reader must be wired to read out the entire ten columns of the badge. (Refer to the

appropriate publication on IBM 1030 devices).

3. The transaction code received from IBM 1030 devices is not included as part of the test request.

4. When header format 0 is used, all IBM 1030 tests require an IBM 1033 Printer on the same line as the requesting terminal. The printer is specified in the addr field.

5. The terminal tests will not test IBM 1035 Badge Readers or IBM 1030 Badge Readers in a 1035 environment.

6. When switching messages from one terminal type to another, the sending terminal must conform to the character set of the receiving terminal.

7. A maximum of 39 characters can be switched to an IBM 1033 Printer.

8. To return a test to the requesting terminal on a dial line, format 0 must be used and EOT must be sent within the first buffer.

9. On an IBM 2740 basic terminal or terminals on a line using Auto Poll, format 1 must not be used with a 0 in the addr field.

10. All on-line terminal tests must be completed before a closedown procedure is initiated (that is, before a CLOSEMC macro instruction is issued).

11. Test messages cannot be canceled. If the test message has been entered incorrectly, the user should enter EOT and begin again.

12. If format 0 is used on a line using Auto Poll, EOT must be sent within the first buffer.

CTAM indicates the occurence of an error by setting bits in the error halfword, for nonaudio lines, or the error byte, for audio lines. The following defines the errors indicated by each of the bits of these two areas. A detailed explanation can be found:

On page 83 , for nonaudio lines
On page 140 , for audio lines.

The mask used to test for each bit is indicated for each bit. Masks may also be used to test for a combination of errors (for example, a mask of x'3000' coded in the ERRMSG macro instruction tests for an indication of sequence number low or high).

Bit	Nonaudio Lines		Audio Lines	
	Mask	Function	Mask	Function
0	X'8000'	Invalid destination	X'80'	Overlength message
1	X'4000'	Terminal inoperative	X'40'	Repeat error
2	X'2000'	Sequence number high	X'20'	Read error
3	X'1000'	Sequence number low		(Reserved for IBM use only)
4	X'0800'	Zero-length messages	X'08'	For user use
5	X'0400'	Incomplete header	X'04'	For user use
6	X'0200'	Invalid source	X'02'	For user use
7	X'0100'	Should-not-occur	X'01'	For user use
8	X'0080'	Transmission error		
9	X'0040'	Timeout exceeded		
10	X'0020'	Breakoff		
11	X'0010'	Insufficient buffers		
12	X'0008'	Message not sent		
13	X'0004'	Control Unit failure		
14	X'0002'	(Reserved for IBM use)		
15	X'0001'	(Reserved for IBM use)		

Figure 46. Line Error Indicators

addressing: a procedure in which the computer transmits identifying characters to a terminal preparatory to sending a message to that terminal.

address chain: an ordered sequence of drum or disk addresses which is the user's response message.

addressing characters: a set of characters peculiar to a terminal and the addressing operation; response to the transmission of these characters indicates whether or not the terminal can receive a message.

answering: a procedure by which a called party completes a connection (for switched lines).

ARU code: a 8-bit byte representation of the transmission code used by audio terminals other than IBM 3944 Dial Terminals.

audio communication line (or audio line): a communication line attached to an audio response unit.

audio line control block: an area of main storage containing control data for operations on an audio line. Abbreviated, ALCB.

Audio Line Procedure Specification: a line procedure specification (LPS) which only concerns the audio input messages. Abbreviated, Audio LPS.

Audio Response Unit: an IBM 7770 or 7772 control unit. Abbreviated, ARU.

audio terminal: a terminal attached to an audio line via a switched network. An audio terminal accepts keyed or dialed data as input to be sent to the computer, and/or produces an audio response as output received from the computer.

buffer: a storage device or area used to compensate for a difference in the rate of flow of information, or the time of occurrence of events. Buffers consist of main-storage areas; size of the areas is designated by the user.

calling: a procedure by which a first party attempts to establish a connection with a second party through a central office exchange. Also, dialing.

chain: the part of a queue consisting of an ordered arrangement of items. The items are related to each other by links. One or more chains may exist in each queue.

closed routine (or subroutine): a routine or subroutine that is not inserted as a block of instructions within a main routine but is entered by basic linkage from the main routine.

communication line group: a group of lines with similar characteristics (such as association with the same type of terminal device).

component: a point in a communications network at which data can enter or leave; an input/output device. A component is always attached to a terminal control unit.

data collection: a telecommunications application in which data from several locations is accumulated at one location before processing.

destination code: the name of a terminal or processing program to which a message is directed.

destination queues, DASD: a group of queues in which the queue control block for each queue resides in main storage, and the message-segment chain for each queue resides on a direct access storage device. The queues contain message segments that are transmitted to terminals.

dead-letter queue: a queue containing messages that could not be placed in the appropriate destination or process queue. The dead-letter queue begins in relative record address 0 on the direct access storage device used.

DCV: abbreviation for Digitally Coded Voice.

DCV vocabulary: an input vocabulary, supplied by IBM, made up of DCV words and to be used by an IBM 7772 Audio Response Unit.

DCV word: a word in DCV representation.

delimiter macro instructions: LPS macro instructions that group functional macros into various coding subgroups.

direct access queues: a group of queues, or, more specifically, message-segment chains of queues, residing on a direct access storage device. The group can include destination and process queues.

direct-connection line: a line that carries messages directly to their destination point (that is, does not go through a central office exchange); a leased line.

distribution-list entry: a terminal table entry containing information on a group of terminals, each of which is to successively receive any message directed to the group. The information in the entry includes relative addresses that locate the single terminal entries for each terminal in the group.

DTF table: an area of main storage that serves as a logical connector between the user's problem program and a file. The DTF table can also be used to provide control information for any transfer of data.

end-of-address character (machine): a control character transmitted but not included with message data at the receiving location; the character indicates the end of non-message-data characters (for example, addressing characters). Abbreviated, EOA.

end-of-address character (program): a QTAM 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. Abbreviated, EOA.

exchange, common-carrier: the location of a common carrier's communication equipment for interconnecting subscribers' lines.

functional macro instructions: LPS macro instructions that operate on message segments and perform functions such as message editing, checking validity of codes used in the header, routing messages to specified destinations, maintaining logs of messages, and checking for errors in transmission or specification.

group-code entry: a terminal table entry containing information on a prespecified group of terminals with the group code feature; this feature facilitates simultaneous transmission of a message to all members of the group through the specification of a single set of unique address characters.

header: a part of the first segment of a message containing information necessary for directing the message to its destination, and other control information.

line control block (LCB): an area of main storage containing control data for operations on a line. The LCB can be divided into several groups of fields; most of these groups can be identified as generalized control blocks. QTAM maintains an LCB for each line in the system.

Line Procedure Specifications: a sequence of user-selected macro instructions that:

1. specifies the manner in which control information in the message header is to be examined and processed; and

2. Specifies other functions (such as translating) to be performed.

Abbreviated, LPS.

log: a collection of messages that provides a history of message traffic.

logging: the process of recording messages on a storage medium for purposes of maintaining a history of message traffic.

LPS control routine: a QTAM routine that:

1. performs initialization functions; and

2. obtains the address of the LPS line group routine to be used for processing a particular message segment.

LPS line group routine: a user-defined routine comprised of subroutines necessary to prepare a message segment for processing, and examine and process the control information in the message segment. The functions performed are based on the user-selected macro instructions that determine the configuration of each line group routine. One line group routine must be provided for each line group included in the system.

message: a combination of letters, digits, and symbols whose termination point is marked by an end-of-transmission (EOT) character.

message data: transmitted characters that are recorded as part of a message. A message data area is the area in a buffer that receives message data. In QTAM, a message data area begins with either the thirty-second byte of a buffer (if the message data includes a message header), or with the twenty-second byte of the buffer (if the message data consists of text only).

message segment: that portion of a message that fits in the message-data area of a buffer.

message switching: a telecommunications application in which a message is received at a central location, stored on a direct-access device until the proper

outgoing line is available, and then transmitted to the appropriate destination.

polling: a flexible, systematic, centrally-controlled method of permitting terminals on a multiterminal line to transmit without contending for the line. The computer contacts terminals according to the order specified by the user; each terminal contacted is invited to send messages.

polling characters: a set of characters peculiar to a terminal and the polling operation; response to these characters indicates to the computer whether or not the terminal has a message to send.

polling list: a list containing control information and names of entries in the terminal table; the order in which the names are specified determines the order in which the terminals are polled.

process-program entry: a terminal table entry containing information on a processing program as the destination for a message.

process queue, DASD: a queue in which the queue control block resides in main storage, and the message-segment chain resides on a direct access storage device. These queues contain message segments that are sent to a message processing program.

processing collected data: an application in which the data accumulated through a data collection application is processed.

processing inquiries: a telecommunications application involving receipt of a message from a remote terminal, processing of the message, generation of a response message, and transmission of the response message to the originating terminal.

queue: an item system consisting of:

1. a queue control block; and
2. one or more ordered arrangements of items (chains).

queue control block: an area in main storage containing control data for a queue. Abbreviated, QCB.

relative line number: a number assigned by the user to a communications line at system generation.

resource: any facility of the computing system or operating system required by a job and including main storage, input/output devices, the central processing unit, files and control and processing programs.

single terminal entry: a terminal table entry containing information on a single terminal.

telecommunications: A general term expressing data transmission between a computing system and remotely located devices via a unit that performs the necessary format conversion and controls the rate of transmission.

Teleprinter: a trade name used by Western Union to refer to its telegraph terminal equipment.

Teletype: a trademark of the Teletype Corporation. A system used for transmitting messages to remote points; the system employs keyboard or paper tape sending and printed receiving.

Teletypewriter: a trade name used by AT&T to refer to its telegraph terminal equipment.

Teletypewriter Exchange Service (TWX): a switched network providing the means for interconnecting AT&T subscribers.

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 (see component).

terminal name: the symbolic name for a terminal, as assigned by the user.

terminal table: an ordered collection of information consisting of a control field for the table and blocks of information on each terminal from which a message can originate, and each terminal, group of terminals, and processing program to which a message can be sent.

terminal table entry: a block of information on a terminal, group of terminals, or processing program; one of the units that comprise the terminal table.

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).

Note 1. Page references in this type refer to nonaudio topics.

Page references in this type refer to audio only or mixed audio and nonaudio topics.

Note 2. When more than one page reference is given (in the same type), the major reference appears first.

Note: Page references in this type refer to nonaudio topics. *Page references in this type refer to audio topics.*

Note: Page references in this type refer to nonaudio topics. *Page references in this type refer to audio topics.*

Note: Page references in this type refer to nonaudio topics. *Page references in this type refer to audio topics.*

Note: Page references in this type refer to nonaudio topics. *Page references in this type refer to audio topics.*

Note: Page references in this type refer to nonaudio topics. *Page references in this type refer to audio topics.*

READER'S COMMENT FORM

IBM System/360
Disk Operating System
Message Control Program
SRL

Order No. GC30-5004-3

- How did you use this publication?

  As a reference source ☐
  As a classroom text ☐
  As a self-study text ☐

- 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 . . .

This publication is one of a series that serves as a reference source for systems analysts, programmers, and operators of IBM systems. 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 System/360 DOS QTAM Message Control Program (S360-30) Printed in U.S.A. GC30-5004-3

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

IBM System/360 DOS QTAM Message Control Program (S360-30) Printed in U.S.A. GC30-5004-3

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]