File No. S360-50
Order GC28-2034-1

TSS

IBM Systems Reference Library

IBM Systems/360 Time Sharing System

Multiterminal Task Programming and Operation

This publication describes how to create a multiterminal (MTT) task, which will run under the IBM System/360 Time Sharing System (TSS/360). It also explains how to write an MTT application program, and how to connect it to a multiterminal task.

The publication is meant for use by TSS/360 installations as a reference book for persons responsible for administering and/or writing an MTT application program.

PREFACE

This publication is divided into four sections and two appendixes:

Section 1 contains an introduction to multiterminal tasks and MTT application programs.

Section 2 describes how to establish an MTT Administrator, use the MTT command, and create a multiterminal task.

Section 3 explains how to write an MTT application program.

Section 4 lists and describes the MTT macro instructions: CLEARQ, FINDQ, FREEQ, READQ, WRITEQ, and WAIT.

Appendix A: Terminal Control Table Entry

Appendix B: Macro Instruction Value Mnemonics

PREREQUISITE PUBLICATIONS

IBM System/360 Principles of Operation, GA22-6821
IBM System/360 Model 67 Functional Characteristics, GA27-2719
IBM System/360 Time Sharing System: Concepts and Facilities, GC28-2003

Second Edition (September 1970)

This is a revision of, and makes obsolete, Order No. GC28-2034-0 and Technical Newsletter GN28-3064. This edition contains the following updates:

- Programming notes, pertaining to concurrent execution of several copies of an MTT application program, have been added.
- Additional comments relating to the "MTT Administrator's Interaction With Application Programs" have been included.
- The examples, under FREEQ, READQ, and WRITEQ have been changed to make use of a simpler coding technique.
- A description of three return codes for the FREEQ macro instruction has been added.

A change to the text or a small change to an illustration is indicated by a vertical line to the left of the change; a changed or added illustration is denoted by the symbol(•) to the left of the caption.

This edition applies to Version 8, Modification 0, of IBM System/360 Time Sharing System, and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. Changes are periodically made to the specifications herein; before using this publication in connection with the operation of IBM systems, refer to the latest edition of IBM System/360 Time Sharing System: Addendum, Form C28-2043, for the editions of publications that are applicable and current.

This publication was prepared for production using an IBM computer to update the text and to control the page and line format. Page impressions for photo-offset printing were obtained from an IBM 1403 Printer using a special printing chain.

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

A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to IBM Corporation, Time Sharing System/360 Programming Publications, Department 643, Neighborhood Road, Kingston, N.Y. 12401

RELATED PUBLICATIONS

The following list of publications ca be used in conjunction with Multiterminal Task Programming and Operation. They will aid users of this manual in both establishing and programming an MTT application program and in providing further insight into the logical operation of multiterminal tasks. They are divided into two catagories; System Reference Library manuals, and Program Logic manuals.

System Reference Library manuals

System Generation and Maintenance, GC28-2010
Managers and Administrators Guide, GC28-2005
Assembler User's Macro Instructions, GC28-2004
System Programmer's Guide, GC28-2008
Command System User's Guide, GC28-2001
Quick Guide for Users, GX28-6400
Quick Guide for System Programmers, GX28-6401
Terminal User's Guide, GC28-2017

Program Logic Manuals

System Logic Summary, GY28-2009
System Control Blocks, GY28-2011
Resident Supervisor, GY28-2012
Access Methods, GY28-2016

SECTION 1: INTRODUCTION

MTT (multiterminal tasks) use several system control programs and tables that, in effect, transform a standard TSS/360 task, which services only one terminal, to a multiterminal task that can service many terminals. A specially designed program, executed as part of a multiterminal task, can provide unique services to those terminals; this program is an MTT application program.

WHAT IS AN MTT APPLICATION PROGRAM?

It is a program designed to perform some unique function for many concurrent users. These users are connected only to the MTT application program; they are not connected to TSS/360, at that time. However, the application program runs under TSS/360 and it can use any commands or macro instructions available to TSS/360 users. Also, use of a special set of MTT macro instructions (READQ, WRITEQ, FREEQ, FINDQ, CLEARQ, and WAIT), within the MTT application programs, allows them to perform I/O operations directly on the user's terminals.

WHY CREATE A MULTITERMINAL TASK?

One reason might be to allow a specific program to interface directly with many remote terminals. Users connected to an MTT application program can get quicker responses than those connected to TSS/360 tasks. MTT is not subject to the usual system overhead; the system performs normal overhead only on a single task, instead of one task for each terminal as in TSS/360 processing.

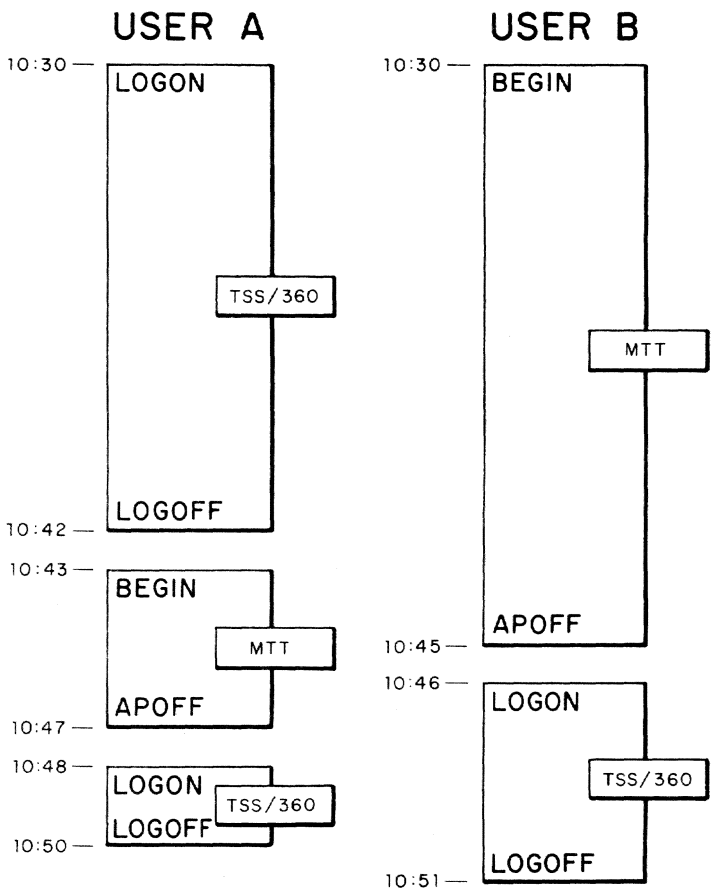
WHO CREATES A MULTITERMINAL TASK?

A multiterminal task can be created only by an MTT administrator who issues the MTT command (he is a user joined to TSS/360 with an O authority code and the T privilege class). Execution of the MTT command transforms the MTT administrator's task (previously established when he issued a LOGON command) to a multiterminal task and activates the MTT application program. The application program need not be coded by the MTT administrator, but only he can attach it to the system and execute it. Since the considerations in writing application programs will usually require that they be executed by the more experienced programmers, execution of the MTT macro instructions is restricted to programmers with O authority.

USING AN MTT APPLICATION PROGRAM

Connection to an application program can be made by switching the terminal to ON and immediately issuing the BEGIN command. If the terminal is already connected to TSS/360 (i.e., a LOGON had been issued), LOGOFF must be issued prior to BEGIN.

If two users switch on their terminals at 10:30, their sessions might proceed like this:



User A connects to TSS/360 as a standard TSS/360 task by issuing LOGON; User B connects to a multiterminal task application program by issuing BEGIN. User A can enter any TSS/360 commands at his terminal; User B can enter only commands defined by the MTT application program that he is using (e.g., APOFF, "application off," is such a command). When User A terminates his TSS/360 task and connects to the same MTT application program as User B, he may enter only the application program commands. When both subsequently complete their use of the MTT application program (by issuing APOFF), they may log on as standard TSS/360 tasks and issue TSS/360 commands.

The format of the BEGIN command will depend on specific application programs' requirements for input parameters (see also Command System User's Guide).

Subsequent communication between the application program and the terminal users is carried on by commands or procedures that will have been defined in the individual application program. Standard TSS/360 commands entered at an application program user's terminal are meaningless unless that application program was coded to accept that command.

Documentation Requirements

It is imperative that the creators of MTT application programs provide detailed documentation for the program's users. This documentation should define the functions of the application, any BEGIN command parameters required as input, and any other commands for user-communication with the application program.

APPLICATION PROGRAM PROCESSING

Each application program can provide unique processing for its users. It may operate as a desk calculator or as a message switching center for many remote points. It may simply retrieve specified records from their storage locations and display those records at the users' terminals.

Regardless of function, all application programs will have some similar processing steps. For example, all will locate terminals requesting service, define and process command input formats for user-interaction with the application program, and define users' disconnection procedures.

Since TSS/360 does not control MTT users' terminals, the application program must serve as the control. The command set that an application program might define to allow its users to perform simple calculating operations are demonstrated here:

Command	Operands
BEGIN	application name,userid,chargeno
ADD	expression1,expression2
SUBT	expression1,expression2
MULT	expression1,expression2
DIV	expression1,expression2
APOFF	none

(BEGIN and its first operand are processed by the system (see Command System User's Guide), but the userid and chargeno must be defined and processed by the MTT application program. The APOFF command must also be processed by the application program.)

Users of such a calculator program would have only these six defined commands. They enter BEGIN to start using the calculator functions of the application program at their terminals. Then they continue with their calculations by using ADD, SUBT, MULT, and DIV. When the application program performs the calculations and prints the results at the user's terminal, the user can issue the APOFF command and return to his desk with the results.

SECTION 2: CREATING AND ADMINISTERING MTT

INITIALIZATION

Before activating a multiterminal task (via the MTT command), a schedule table entry must be established for the table-driven scheduler (see the System Logic Summary) under which the multiterminal task is to start its execution. This entry should be designed and incorporated into the system either by the MTT administrator or by an experienced system programmer. It can be incorporated into the schedule using a delta data set during system startup (see System Generation and Maintenance).

The selected schedule table level will set the internal environment in which the multiterminal task will execute (although it may dynamically change during the task's execution) and it will specify if there is to be page stealing. Since different MTT applications have unique storage requirements (i.e., size, residency requirements, etc.), the page stealing gives the user greater flexibility in establishing the internal environment for operating his MTT application program. Page stealing permits a subset of the task's pages to be purged to a drum when the task has accumulated the maximum storage pages specified in the schedule table.

The multiterminal task will be subject to the same drum-to-disk migration algorithm that controls normal TTS/360 tasks.

THE MTT COMMAND

When he issues the MTT command, the MTT administrator must specify the number of users that will be allowed concurrent use of the application program, and the buffer requirements for each terminal. He must also specify the name of the application program and the schedule table entry level that was set up during initialization. The application program is activated when the MTT command is executed; the MTT administrator's terminal keyboard is then locked. Any further interaction between the MTT administrator and the application program must be predefined in the application program (see "MTT Administrator's Interaction with Application Programs").

Two unique tasks may issue the MTT command specifying the same application program. Thus, several copies of an MTT application program may exist concurrently. In this environment, connecting user terminals will be attached to the copy with the least number of active lines regardless of any user ID or charge number specified in a BEGIN command. When two copies of an application program are active, users should be aware that they have no control over which copy of the application program they will be connected to. For certain application programs, multiple copies can be helpful when the application is required to perform large quantities of VAM I/O and I/O overlap is desired.

MTT -- CREATE A MULTITERMINAL TASK

The MTT command converts a normal TSS/360 task to a multiterminal task that can service a large number of remote terminals. This task can be executed either conversationally or, if it is not necessary for the application program to be conversational, in the background as a batch task.

Operation	Operand
MTT	PROG=module name,MAXL=number,LEVEL=number [,BUFSIZ=number]

PROG
specifies the name of the application program to be executed. This parameter is required, it consists of one-to-eight alphameric characters.

MAXL
specifies the maximum number of lines (i.e., terminals) that may be simultaneously connected to the program. The number n must be $0 \leq n \leq 4095$. This parameter is required.

LEVEL
specifies the schedule table level at which this task is to start. The number n must be $0 \leq n \leq 255$. This parameter is required.

BUFSIZ
specifies the length in bytes of the buffer to be associated with each terminal line connected to an application program. The size of this buffer determines the maximum byte-length of an input message. The number n must be $16 \leq n \leq 4080$.

Default: 200

Functional Description

The MTT command processor verifies that the issuing user has an O authority code and a T privilege class. Then an attempt is made to load the application program. Input parameters are checked; any invalid parameters are prompted for and rechecked. When all parameters are validated, the application control blocks, necessary for the operation of a multiterminal task, are established. The blocks are the application terminal control table (TCT), the application multiterminal status control block (MTSCB), and its buffer pages. The MTT processor reserves virtual storage space needed for the application TCT and buffer pages, issues a CHANGE macro instruction to pass control to the specified schedule table level, invokes supervisor services (via a CONN macro instruction) to create the resident application MTSCB, and then explicitly calls the specified application program.

If a task is already multiterminal, when the MTT command is issued, the task is disconnected from its current application program by the FREEQ ALL and DCON macro instructions. CONN is then reissued to connect the task to the currently specified application program. Thus, if an MTT administrator issues two MTT commands at his terminal, only the last will remain active. If the MTT administrator hits the attention key, and then issues a second MTT command for the same application program, thereby reactivating the application program, any external interruption servicing routine previously established -- via SIR and SEEC macro instructions -- due to the initial activation of the application program, is still in effect.

When a normal return (i.e., via a RETURN macro instruction) is made from the application program, the MTT processor unloads the program and issues a FREEQ (ALL, message-addr). This releases the application's TCT and buffer slots, and informs all terminals connected to the application program's task: "APPLICATION TASK TERMINATED OPERATION" and "TERMINAL LOGICALLY DISCONNECTED, RECONNECT OR HANG UP." Each terminal line is placed in a read-state similar to the status of an initial connection.

After getting such a message, every terminal user has two minutes to enter BEGIN or LOGON, or hang up the phone; if there is no action in that time, the terminal will be physically disconnected.

Then a DCON SVC is issued to clean up storage; the task is returned to the original schedule table level; the remaining application TCT page and buffer pages are released; the application MTSCB is erased; and control returns to the command system.

Then the MTT administrator's SYSIN/SYSOUT terminal receives an underscore; the administrator may LOGOFF or proceed as a normal TSS user.

ABEND AND LOGOFF ACTIONS

If a multiterminal task goes through an ABEND or LOGOFF procedure, warning messages will be sent to any active users of the MTT application program:

For ABEND

"APPLICATION TASK ABEND"

"TERMINAL LOGICALLY DISCONNECTED, RECONNECT OR HANG UP"

For LOGOFF

"APPLICATION TASK LOGOFF"

"TERMINAL LOGICALLY DISCONNECTED, RECONNECT OR HANG UP"

In either case, the terminal users have two minutes, as in a normal application return, to enter LOGON or BEGIN, or hang up the phone.

EXAMPLE: An MTT administrator logs on to TSS/360, causing a standard task status index (TSI) to be created. When he subsequently issues the MTT command, his TSI is converted to a multiterminal task TSI (i.e., the TSIMTT flag is set on); control is passed to the specified application program. The MTT administrator's terminal keyboard remains locked while the application program is processing (see "MTT Administrator's Interaction with Application Programs").

```
User:    LOGON   USEREJB
System:  B001  LOGON  TASKID00014  11/14/69  13:08
System:
User:    MTT     MODULE,300,130,255
```

MTT ADMINISTRATOR'S INTERACTION WITH APPLICATION PROGRAMS

Interaction between an MTT application program and its MTT administrator must be defined and provided in the application program. For conversational multiterminal tasks, the MTT administrator's keyboard is locked when the MTT application program is activated; interaction can only be initialized by: the MTT administrator hitting his attention key, or the application program issuing macro instructions that pass control to the administrator's terminal (e.g., GATRD, GTWSR, CLIC, CLIP, etc.).

When the attention key is pressed on the administrator's SYSIN/SYSOUT terminal, while an application program is running, an asynchronous interruption will be queued on the multiterminal task's TSI. It is processed by normal TSS/360 interruption-handling facilities. The application program can process this interruption by creating an asynchronous interruption-servicing routine, using SIR, SAEC, and USATT macro

instructions, or by use of the AETD macro instruction. The interruption-servicing routine can then prompt the MTT administrator's terminal for what he wants the application program to do. That program can then interpret the administrator's reply accordingly. The administrator might want to shut down the application, deactivate particular users, or write messages to all connected terminals. Note that while the application program is communicating with the MTT administrator, via the SYSIN/SYSOUT terminal, it appears dormant to its users; it will not be able to perform processing for them. If the MTT application program is running as a batch task, it can provide combined administrator/operator functions through one of the connected terminals. This allows concurrent communication between the application program, the administrator, and the application program's users.

If no attention-handling routine has been defined in the application program, the attention will be processed by TSS/360, and the MTT administrator's terminal will be prompted for additional input via the standard TSS/360 prompting characters (_ ! or ¬). The MTT administrator can then enter TSS/360 commands or resume application program processing.

SECTION 3: WRITING AN MTT APPLICATION PROGRAM

When a multiterminal task is initially dispatched by the system, because the MTT administrator issued the MTT command, its application program receives control. At this time, no users should have been connected to the program. Thus the application program might begin by going through an initialization procedure (e.g., opening all data sets that comprise the application service program). When initialization has been completed, the application program should search for users who may subsequently have been connected (by their use of the BEGIN command). The program locates initially connected users by one of two processing techniques: the polling mode or interruption mode.

INTERRUPTION MODE PROCESSING

Because initial connections are unsolicited, the system automatically queues an external interruption with message-number 127 (see "Processing External Interruptions") on the application task. When the interruption is subsequently processed, control is passed to any interruption-servicing routine defined in the application program. If no routine has been defined, the interruption will be ignored; then any initial connections can be detected only by the polling mode.

POLLING MODE PROCESSING

A slot in the application TCT is assigned to each connecting terminal. A flag in that slot will be set to denote initial connections; this is done whether or not an interruption-servicing routine has been established. The FINDQ macro instruction can be issued to poll the application TCT (see "Polling the TCT"). If no terminal has been connected to the application program, or if no connected terminal desires to do any work, the program should issue the WAIT macro instruction, after which it will remain idle until there are subsequent requests for services.

SUBSEQUENT COMMUNICATION WITH USER TERMINALS

After locating the initially connected terminals, an application program must communicate with those terminals to determine which of its services the terminal user wants to use. The READQ, WRITEQ, and CLEARQ macro instructions should be used for this communication. The application program can use READQ and WRITEQ to elect the polling mode or interruption mode.

I/O Processing

When application programs issue read or write requests to user terminals, buffer areas are allocated and I/O buffer slots are created for each request in the buffer page tables. These buffer areas and slots are later released. Buffers for read operations can be released, following completion of the input operation, by issuing a CLEARQ macro instruction or another I/O operation; buffers for write operations are released automatically on completion of the write operation.

Upon completion of any I/O operation, the TCT slot for that device will be posted to indicate work completion. Also, if the READQ/WRITEQ interruption option was selected, completion of an I/O operation will cause an external interruption to be queued on the application task.

For completion of read operations, the virtual storage address of the input buffer slot is passed to the application program in a parameter list (i.e., either in the FINDQ parameter list, DSECT CHAFNQ, or the terminal interruption parameter list, DSECT CHATII).

Note that issuing READQ and WRITEQ causes the requested I/O operation to be initialized and control to be returned to the application program before completion of the operation.

I/O Completion

If processing of an application program depends upon the completion of a previous I/O operation, the program must verify that the I/O operation has been completed before continuing the processing. The verification can be done in either the interruption mode or the polling mode, similar to intial connection processing.

1. If the READQ or WRITEQ interruption option was selected, the application program can write an external interruption-processing routine (via SIR and SEEC macro instructions) that will automatically receive control upon completion of a terminal I/O operation (see "Processing External Interruptions").

2. A FINDQ macro instruction can be issued to poll the application TCT table for terminal slots that have the work-completion indications set in the work flag (see "Polling the TCT").

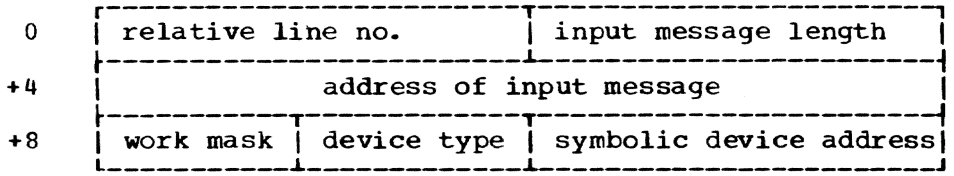
PROCESSING EXTERNAL INTERRUPTIONS

The application program can process external interruptions by establishing an interruption-processing routine through issuance of the SIR and SEEC macro instructions. All external interruptions generated by the system for processing by an application program will have external message-number 127 associated with them; when the application program issues a SEEC to assume control, it must specify the external message number 127 to identify the type of external interruption to be processed.

This interruption-servicing routine will automatically receive control whenever an external interruption with message number 127 is generated for a terminal connected to the application task. When the routine receives control, the address of an interruption control block (ICB), containing the address of the interruption processor's communication area, will be in register 1 (see the SIR and SEEC macro instructions in Assembler User Macro Instructions for more on these parameter areas).

The message area address, in the second word of the communication area, points to the message control block (MCB) that contains terminal interruption information. The three-word interruption message can be located at a displacement of 16 from the beginning of the MCB and has this format:

DSECT CHATII



Work mask	Meaning
0	message in, complete
1	message out, complete
2	attention
3	initial connection
4	unrecoverable error
5	negative response to polling for 1050 (component not ready)
6	buffer overflow

Device type (binary no.)	Meaning
1	1050
2	2741 (correspondence keyboard)
3	2741 (PTTC/8 keyboard)
4	teletypewriter

When an external interruption option has been specified for a line, it remains in force until a subsequent READQ or WRITEQ is issued against that line to indicate that interruptions are not desired.

If SIR and SEEC are used, the application program must issue DIR before terminating interruption servicing. This assures that the multi-terminal task is properly initialized with regard to interruption requests.

Mixed-Mode Processing

If an application program is operating with mixed-mode processing (i.e., both interruption and polling modes), any interruption-servicing routine should also issue a CLEARQ macro instruction to the terminal it is servicing. That will clear the work flag in the TCT slot associated with the terminal and ensures that the same condition will not be processed by subsequent FINDQ macro instructions.

Inhibiting Interruptions

Prior to issuing any MTT macro instructions (within an interruption-processing routine), the application program should disable task interruptions via the SAI macro instruction; the program should enable interruptions via the RAE macro instruction upon return. This must be done because the MTT macro instructions are not recursive unless issued with a unique save-area pointer and a unique parameter list.

If the specified interruption-processing routine issues any MTT macro instruction, except WAIT, and does not inhibit interruptions, it must provide a unique save area (pointed to by register 13) and use a unique parameter list to avoid possible recursion (automatic if S-form is used). This will preserve task integrity, since an interruption may have occurred during execution of a macro instruction.

POLLING THE TCT

When the interruption mode is not used, the application program must poll the terminals' slots in the application TCT for those that have initial connections or their work flags on. Issuing a FINDQ macro instruction will detect and present any initially connected terminals, or I/O completions, by polling the TCT slots. In addition to return codes in register 15, execution of FINDQ returns, in register 1, the address of a fullword pointer which contains the address of a parameter list (DSECT CHAFNQ). This parameter list contains the relative line number assigned to the terminal being processed, the device type (e.g.,

1050/2741), and the buffer address. The relative line number must be used by the application program to address each user's terminal. The first terminal connected to an application program would be relative line number 0000, the second 0001, etc. The application program can process any input message at the indicated buffer address. When the message has been processed, the buffer can be released for further use by issuing a READQ, WRITEQ, or CLEARQ macro instruction. If one of those macro instructions is issued while an I/O operation is in progress, a busy condition will be returned to the application program; any buffer associated with the operation will be preserved.

DISCONNECTING USERS

Each connected terminal's TCT slot is permanent for the duration of an active application program, unless a FREEQ macro instruction is issued against that terminal by the application program. Application programs must define the means by which current users may elect to be disconnected. Upon receiving a defined disconnect message, the application program can issue a FREEQ to that device, logically or physically disconnecting the terminal and freeing the TCT slot.

ADDITIONAL APPLICATION PROGRAMMING CONSIDERATIONS

Data Translation/Processing

During normal I/O operations (with the READQ and WRITEQ macro instructions), the system will translate incoming and outgoing text by using the appropriate translation table: PTTC, correspondence, or teletypewriter (see Terminal User's Guide for character-translation tables). Alternatively, the application program can, on an individual basis, specify translation and provide its own translation table. The MTT macro instructions perform no editing whatsoever. Carrier returns, idle characters, tabs, and backspace characters must be supplied and processed by the application program. These standard translations will be performed:

1050	PTTC/8: EBCDIC
2741	PTTC/8: EBCDIC
2741	CORRESP: EBCDIC
teletypewriter	ASCII: EBCDIC

All lower-case alphabetic characters will be converted to upper-case.

Input-Message Editing: When "message in, complete" is noted in a terminal's TCT slot, the message area and message count includes all characters (e.g., backspaces and carrier returns), except EOB, EOT, and XOFF, for the 1050, 2741, and teletypewriter terminals respectively.

Function	Symbol	EBCDIC code
carrier return (new line)	CR(NL)	X'15'
end of block	EOB	X'26'
idle (for output only)	IL	X'17'
end-of-transmission	EOT	X'37'
transmitter off	XOFF	X'13'

Output-Message Editing: All messages transmitted via the WRITEQ macro instruction may be followed by an ending sequence:

Device	Ending sequence
1050	CR
2741	CR, IL (approximately one IL for every ten characters)
teletypewriter	none (all reads or writes automatically prefixed with carrier return and line feed)

On 1050 and 2741 terminals, the carrier return (CR) at the end of the WRITEQ message may be omitted. Then subsequent write operations will continue on the same line until the end of the printer line is reached.

For 1050 terminals, whether carrier returns have been specified in the output line or the carrier is returned automatically, writing continues "on the fly" as the carrier is returning. Idle characters (IL) should be included to allow the carrier to complete its return.

Additional information describing terminals and their character translations is in Terminal User's Guide.

Attention-Interruption Handling

The two sources from which attention interruptions may be generated are: the MTT administrator's terminal and any application-program users' terminals.

From the MTT administrator's terminal, an attention interruption allows him to interrupt the program's processing and request the program to interact with him, or request interaction between him and the users of the application program (See also "MTT Administrator's Interaction With Application Programs"). When the administrator is communicating with the application program, via the SYSIN/SYSOUT terminal, users of the application program cannot make use of it; they remain in a wait status until the administrator returns control to the application program.

From an application-program user's terminal, pressing the attention key causes that terminal's application TCT entry to be posted with an attention indication. The user can elect (via READQ and WRITEQ), for all his attention-key requests, to have an external interruption with message number 127 (see "Processing External Interruptions") queued on the multiterminal task's TSI. If the application program has defined an interruption-servicing routine, using SIR and SEEC macro instructions, the program responds to these external interruptions in the defined manner. If there is no defined interruption-servicing routine, the program can service interruptions only by polling.

The method used in polling, to detect attention interruptions, depends on whether the interruption occurs during a READQ, WRITEQ, or CLEARQ to a terminal, or while the application program is doing other work.

If the attention key is pressed during execution of a READQ or WRITEQ, the I/O operation will not be initialized; the attention indicator in the terminal's TCT entry will be cleared. The attention interruption will, however, be indicated in a return code to the application program. If the attention interruption occurs during a CLEARQ operation, that operation will be completed, the attention interruption indicator in the TCT entry will be cleared, and a return code indicating the occurrence of an attention interruption will be passed to the application program.

If the attention interruption occurs while the application program is performing work for another user, or while it is in the WAIT state, detection of the interruption should be handled differently. Then a FINDQ must be issued to poll the TCT for the interruption. When only an indication of an attention interruption is posted in the TCT entry, the interruption can be processed readily, based upon the FINDQ return code. However, if an I/O completion for a read or write also occurs and the TCT is posted with two settings, one for an attention interruption and one for the I/O completion, the FINDQ return code recognizes only the I/O completion. The attention interruption is recorded in a field (FNQATN) in the FINDQ parameter list (DSECT CHAFNQ). The application program must examine the parameter list after all I/O completion returns, if it is to process these attention interruptions.

Reading and Punching Cards (IBM 1050 system only)

Application programs can define commands that, when entered at a user terminal, will direct the application program to perform READQ and WRITEQ operations on card readers and punches attached to a 1050 system (see READQ and WRITEQ macro instructions). Thus, application-program users could have data sets read in from, or punched on, a card reader/punch.

TSS/360 Commands from an Application-Program Users' Terminals

An application program can act as a middleman between its users and TSS/360 by reading in TSS/360 commands, issued at the users' terminals, and then passing them on to the TSS/360 command system via the OBEY macro instruction. In most cases, this is undesirable because of the extra overhead. However, for some TSS/360 commands (i.e., generally those that perform no terminal I/O, such as DDEF), an application program might utilize the command system on behalf of its users.

Using a Terminal's Hardware Break-Option

Use of the WRITEQ macro instruction's break option implies the existence of a hardware break-feature on the specified terminal. If the break option is specified to a terminal that does not have the hardware feature, results are unpredictable. When the break option is specified, WRITEQ will interrupt any I/O operation that is in progress to the terminal and write the specified message out at that terminal. Any interrupted I/O operation is terminated.

Use of DSECTs

Detailed descriptions of DSECTs, for the control blocks used in a multiterminal-task environment, are found in System Control Blocks or in the TSS/360 macro library, ASMMAC.

Other Coding Considerations

Application programs should be written as nonprivileged code. Programmers with an O-authority, can issue an LVPSW macro instruction to transfer to the privileged state, to take advantage of the privileged system macro instructions.

The user should be aware that nonconversational rules of the system apply to multiterminal tasks executed in the background. Thus, some instructions that can only be executed conversationally, will be ignored when issued nonconversationally.

SECTION 4: USING MTT MACRO INSTRUCTIONS

The formats and descriptions of the MTT macro instructions (CLEARQ, FINDQ, FREEQ, READQ, WAIT, and WRITEQ) use standard TSS/360 format notation and value mnemonics. The reader is assumed to be familiar with these standards. A table of value-mnemonic equivalents is in Appendix B. Assembler User's Macro Instructions includes detailed descriptions of the operand forms.

All MTT macro instructions reside in the ASMMAC macro library; a DDEF must be issued before assembling any of them. Dummy sections (DSECTs) referenced in this section are described in System Control Blocks.

CLEARQ -- CLEAR TERMINAL DEVICE STATUS (S)

The CLEARQ macro instruction clears any indications of pending work for a terminal device and releases data buffers that were assigned to that device in a previous READQ operation.

Name	Operation	Operands
[symbol]	CLEARQ	relative line number-value

relative line number

specifies the relative position of the entry (or slot) in the application's terminal control table (TCT) associated with the terminal for which any indication of pending work is to be cleared. The number must be in the range of 0-4095 and specify an active TCT slot. When register notation is used, the (binary) device number must be loaded into the two low-order bytes of the specified register before issuing CLEARQ.

The relative line number can be found in either (A) the parameter list generated by a previous FINDQ (addressability to this list can be established by using, as a base register, the pointer that is pointed to by register 1 and the DSECT CHAFNQ or (B) the communication area of any external interruption-processing routine that may have been written by the application program (see "Processing External Interruptions").

INITIALIZATION: The address of a save area must be placed in register 13.

CAUTION: This macro instruction may be used only in a multiterminal task (MTT) application program.

EXECUTION: When the CLEARQ routine receives control, register 1 points to a pointer parameter-list (which contains the relative line number) in the issuing program's PSECT. This parameter list is described by DSECT CHACLQ. CLEARQ checks the entry (or slot) in the application TCT, pointed to by the specified relative line number. The associated device number is tested for validity; if valid, the status byte of the application TCT slot is examined for work indications. If any READQ or WRITEQ is in progress, a busy signal is returned; no clearing functions are performed. Otherwise, the routine will save the contents of the TCT slot work byte for the device and reset (clear) the byte to 0's.

The slot is also checked for a buffer that is assigned to the device. If there is a buffer, a clear request will be set in the slot and an ATCS system macro instruction will be issued. This will cause TCS to release that buffer slot. Return data will be passed to the application program, based on the saved work byte indications.

Return Data: The low-order byte of register 15 is set to

Code	Significance
00	normal return
04	invalid relative line number (not connected to application program)
08	busy, previous READQ or WRITEQ not completed
0C	attention interruption received; normal clearing functions performed

PROGRAMMING NOTES: Any of these pending-work indications in an application TCT slot can be cleared using CLEARQ:

- initial connection completed
- attention received from device
- solid I/O errors on line
- message in, complete
- message out, complete

CLEARQ should be used when some conditions are signaled by an external interruption and others by a setting in the application TCT slot (mixed mode processing). Example: If initial connections are processed by an application program's interruption-servicing routine, but FINDQ is used to poll for subsequent transactions on that device, CLEARQ should be issued to clear the initial-connection indication to prevent a subsequent FINDQ from reprocessing that same indication.

Normally, buffer space assigned to a device is released automatically by subsequent executions of READQ or WRITEQ with the response option. If those macro instructions are delayed, for whatever reason, CLEARQ can be used to release buffer space.

Use of L- and E-Forms: The L-form macro instruction results in a macro expansion that consists only of a parameter list. The name field is mandatory in the L-form; in the E-form it is used to locate the parameter list. Use of L- and E-forms:

```
LFORM   CLEARQ   MF=L                relative line no. not specified

EFORM   CLEARQ   (5),MF=(E,LFORM)   relative line no. in register 5
```

EXAMPLE: In mixed-mode processing, an application program is processing initial connections from terminals via an external interruption-handling routine and subsequent transactions to those terminals by using FINDQ. When an initial connection occurs, generating an external interruption and posting the indication in the application TCT slot, control passes to EXTINT for appropriate processing. Before the interruption routine completes its processing, it issues CLEARQ to clear the initial-connection indication from the application TCT slot to prevent a subsequent FINDQ from reprocessing that indication.

Assume SIR and SEEC macro instructions have been previously issued in an application program to establish EXTINT as the routine that is to process initial connections. When EXTINT receives control, a pointer to an interrupt control block (ICB) is in general register 1. The inter-

ruption routine might perform appropriate processing such as checking the initial-connection BEGIN parameters.

```
EXTINT      .                   mask off interruptions, save registers
                                 under standard linkage conventions
            .
            .
            .
            L       2,0(1)      get address of COMAREA from ICB
            L       3,16(2)     get address if MCB + 16, the buffer
                                 address
            DROP    CURREG      drop current base register
            USING   CHATII,3    establish base register for terminal-
                                 interruption information DSECT, CHATII
            L       4,TIIWWK    locate TCT work flag, using DSECT sym-
                                 bol, to verify cause of interruption
            L       5,TIIRLN    get relative line number of initially
                                 connecting terminal
            L       6,TIIVMA    get address of input message of initial
                                 connection
            .                   examine BEGIN parameters in location
            .                    addressed by register 6
            .
            CLEARQ  (5)         before returning to main portion of
                                 application program, clear TCT slot
                                 using register notation format
            .
            .
            .                   allow for interruptions
            RETURN              return via standard linkage
*** interruption-mode processing ends here; polling-mode processing
***  follows
POLL        FINDQ               poll terminals for work
            B       ABC(15)     branch table using return code
ABC         B       NOWORK      if no work, enter wait-state; RC=00
            .
            .
            B       NITERR      if FINDQ locates an initial connection,
                                 that is error condition that should
                                 have been processed by the EXTINT
                                 routine; RC=08
            B       INCOMP      if previous READQ completed, process
                                 input, perform requested service
            .
            .
NOWORK      WAIT                wait to be redispatched by system
            B       POLL        when reactivated by system, poll
                                 application's TCT for work
NITERR      .                   perform appropriate error processing
            .
            .
INCOMP      L       2,0(1)      get address of FINDQ parameter list
            DROP    REG3
            USING   CHAFNQ,2    establish base register for FINDQ
                                 parameter list (DSECT CHAFNQ)
            L       7,FNQMSA    get address of input message using
                                 symbols from DSECT
            .                   examine input message at address in
            .                    register 7 to determine required
            .                    service
            B       POLL        when processing is complete, poll TCT
                                 TCT for more users
```

FINDQ -- FIND TERMINAL REQUIRING WORK (S)

The FINDQ macro instruction polls an MTT application program's work queue (i.e., the application terminal control table entries) for terminals that require the services of the application program, or optionally, returns the work status of a specified terminal.

Name	Operation	Operands
[symbol]	FINDQ	[relative line number-value]

relative line number
specifies the relative position of a specific terminal's associated entry (or slot) in the application's terminal control table (TCT). It must be in the range of 0-4095. When supplied by the register notation, the specified register must contain the (binary) relative line number in the two low-order bytes, before issuing FINDQ.

Default: Circular polling of the TCT; the status of the first terminal that requires the application program's services will be returned.

Note: The relative line number specified will usually be the number specified in a previous READQ, WRITEQ, CLEARQ, or FINDQ.

INITIALIZATION: The address of a save-area must be placed in register 13.

CAUTION: FINDQ may be issued only in an MTT application program. If an MTT application program has established a routine for external interruption processing to service initial terminal connections or completions of read/write operations, but has failed to issue a CLEARQ macro instruction after processing such conditions, a subsequent FINDQ will result in reprocessing the connections or completions.

EXECUTION: If the relative line number is defaulted, FINDQ polls entries in the application TCT to locate a terminal that requires servicing. If no terminals require servicing, the return is to the application program with the appropriate return code. When a terminal requires services (i.e., its application TCT entry has the work flag set on), FINDQ will clear the work indication from the slot, and pass status information and the return codes that describe the type of service required to the application program.

Return Data: FINDQ sets the low-order bytes of register 15.

Code	Significance
00	no work
04	invalid relative line number (not connected to application program)
08	initial connection of device
0C	attention received from terminal device
10	solid I/O error on terminal line
14	message out, complete (from previous WRITEQ; see also "flag byte")

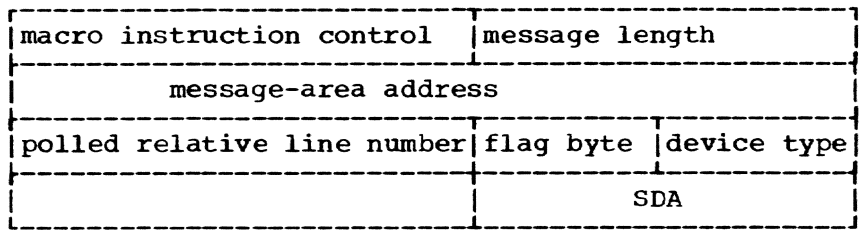
18 message in, complete (from previous READQ or WRITEQ/RESP; see also "flag byte")

1C negative response from input component

20 message in (overflowed buffer; see also "flag byte")

Also, FINDQ returns, in register 1, the address of a fullword pointer to this parameter area:

DSECT CHAFNQ



macro instruction control
contains relative line number or default code, depending on use of macro instruction

- X'FFFF' -- relative line number defaulted (polling option)
- a specific relative line number (binary), in range of 0-4095, that is to be checked

message length
meaningful only for "message in, complete" (RC=18 or 20); contains binary number of characters typed at terminal while in read-state

message area address
meaningful only for "message in, complete" (RC=18 or 20); contains pointer to buffer area containing an input message typed at terminal while in read-state; if read with translation option, input text will be in EBCDIC, otherwise in line-code form. If the attention key was pressed (see "flag byte", bit 1), while a read was on the terminal's line, the message area contains the data entered from the terminal up to the point where the attention occurred.

polled relative line number
meaningful only in polling mode; the field contains relative line number of terminal ready to work

The relative line number is assigned to a terminal during its initial connection and must be used in all subsequent transactions, to identify that terminal. The first connected terminal becomes line 0000, the second, line 0001, etc.

flag byte
these bits are defined:

bit 0: will be cleared every time FINDQ is used; if in polling mode and if resulting scan for work "wraps around" (from last terminal to first), this bit will be set to 1 when FINDQ returns

bit 1: meaningful only for "message in, complete" (return code 18), "message out, complete" (return code 14), or "message in but overflowed" (return code 20); if set to 1, an attention interruption has been received from terminal; if an

MTT application program, operating in polling mode, is to process interruptions, it must examine this bit

device type
specifies code indicating the type of terminal requesting service:

Hex code	Terminal type
01	1050
02	2741 (correspondence keyboard)
03	2741 (PTTC keyboard)
04	teletypewriter

SDA
system's "symbolic device address" of terminal

PROGRAMMING NOTES: If an application program has not defined an interruption-processing routine to process external interruptions generated at initial connection (and optionally upon completion of reads and writes), a FINDQ macro instruction must be issued to detect those interruptions and completions.

FINDQ polls initially at the beginning of the application TCT and, subsequently, immediately after the TCT entry on which the previous execution of FINDQ stopped.

Every time FINDQ returns with a "work" indication, the work status of that entry is cleared; FINDQ will never return an indication a second time when that indication has been previously processed by a FINDQ. It will, however, present work-status information a second time, when operating in the interruption mode, if CLEARQ has not been used.

Use of L- and E-Forms: The L-form macro instruction results in a macro expansion consisting of only a parameter list. The name field is mandatory in the L-form; in the E-form it is used to locate the parameter list. Use of L- and E-forms:

```
LFORM  FINDQ   MF=L

EF1    FINDQ   (4),MF=(E,LFORM)

EF2    FINDQ   MF=(E,LFORM)
```

The instruction labeled LFORM creates a parameter list in the issuing program's PSECT. EF1 inquires about the status of the terminal whose relative line number is in register 4. EF2 requests polling mode for any terminal requiring service.

EXAMPLE: An application program polls its applications's TCT for any terminals requiring service. Action is then based on an appropriate return code from FINDQ. When no work is found, the application program enters the wait-state until redispatched by the system.

```
POLL        FINDQ
            B       ABC(15)    branch table using return codes
ABC         B       NOWORK     no work ready; enter wait-state
            B       INVNO      invalid relative line number specified;
                                branch to error-message routine
            B       NITCON     initially connected terminal requires
                                servicing
            B       ATTNINT    attention interruption from specified
                                terminal; any outstnading READQ or
                                WRITEQ was not completed and must be
                                reissued
            B       IOERR      solid I/O error on terminal line
            B       OUTCOMP    previous output completed from a WRITEQ;
```

```
                                    continue servicing that terminal (e.g.,
                                     issue subsequent READQ or WRITEQ)
          B         INCOMP          previous READQ or WRITEQ with response
                                     completed; process input line
          B         NEGINP          branch to process negative input value
          B         OVFLO           input message to long; issue error
          .                          message
          .

NOWORK    WAIT                      wait to be redispatched upon work
                                     completion
          B         POLL            when reactivated, poll application's
                                     TCT for work
INVNO     .
          .
```

FREEQ -- DROP A TERMINAL DEVICE (S)

The FREEQ macro instruction causes a terminal associated with the specified relative line number (or all terminals) to be physically or logically disconnected from an MTT application program. It will also, optionally, write a message to that terminal before disconnecting.

Name	Operation	Operands
[symbol]	FREEQ	{relative line number-value} {ALL} [,message pointer-addr] [,disconnect-{PD}] [,disconnect-{LD}]

relative line number
specifies relative position of an entry (or slot), in application TCT, associated with terminal that is to be logically or physically disconnected; if ALL is specified, all terminals will be disconnected; unless physical disconnection is specified, terminals will be logically disconnected. During logical disconnection, a user must,within 2 minutes, connect to TSS (i.e., LOGON) or to an application program (i.e., BEGIN). If no action is taken in that time, physical disconnection occurs. If register notation is used, the (binary) relative line number must be loaded into the two low-order bytes of the specified register before execution of FREEQ.

The relative line number can be found in either (A) the parameter list generated by a previous FINDQ (addressability to this list can be established by using, as a base register, the pointer that is pointed to by register 1 and the DSECT CHAFNQ) or (B) the communication area of any external interruption-processing routine that may have been written by the application program (see "Processing External Interruptions").

message pointer
specifies address of pointer to a message area, which must be in this format:

length	message text	CR

1 byte

The length byte should indicate the length of the message plus one (for the length byte itself).

If register notation is used, the address of the pointer must be loaded into the specified register before executing FREEQ.

disconnect
specifies logical or physical disconnection; LD or PD should be indicated.

PD specifies that the device, associated with the application TCT entry indicated by the relative line number, will be physically disconnected from the system; the line to the device will be disabled.

LD specifies that the device will be logically disconnected (i.e., all related tables cleared) and a read CCW placed on the terminal's line. A timer interruption is set for 2 minutes; if no subsequent interruption is received from the device before the time expires, the device will be physically disconnected.

Default: LD

INITIALIZATION: The address of a save-area must be placed in register 13.

CAUTION: This macro instruction may be specified only in a multiterminal task.

EXECUTION: When the FREEQ routine is entered, register 1 will point to a parameter list pointer (DSECT CHAFRQ) that contains the macro instruction parameters. The entry (or slot) in the application TCT pointed to by the specified relative line number is examined. The device associated with that entry is checked for validity; if valid, any I/O operation to that device will be terminated, and the terminal will be physically or logically disconnected from the task. If the message option was specified, a message would be written to the appropriate terminal before its line is disconnected. If the ALL option was specified, any message specified will be written to all connected terminals; all connected terminal lines will be physically or logically disconnected and all application TCT pages and buffer pages (except one of each of these pages) will be released.

Return Data: Register 15 will be set.

Code	Significance
00	normal return
04	invalid relative line number specified (not connected to application program)
08	should not normally be received by users; indicates disconnect byte must be 00 or FF
0C	zero length was specified
10	illegal use of the L-form of the macro instruction.

PROGRAMMING NOTES: When logical disconnection is requested, the user has the option of becoming a TSS user or connecting to an MTT application program, without having to redial the system. The message "TERMINAL LOGICALLY DISCONNECTED, RECONNECT OR HANG UP" will be displayed at his termminal; he will then have two minutes to respond with a LOGON or BEGIN command.

When the message option is specified, the message text must be in the EBCDIC form. The message will be routed to printer 1 (see "WRITEQ") for 1050 terminals. Control characters will be inserted by the system where necessary.

The message length specified by FREEQ should not exceed 72 characters when being sent to a teletytewriter.

Assignment of relative line numbers to a device is dynamic, following entry of a BEGIN command at a terminal; this assignment continues until FREEQ is issued. If a terminal is logically or physically disconnected and later reconnected to the same application program, a new relative line number probably will be assigned.

Use of L- and E-Forms: The L-form macro instruction results in a macro expansion consisting of only a parameter list. The name field is mandatory in the L-form; in the E-form, it is used to locate the parameter list. While the relative-line ALL operand is mandatory, it may be specified in either the L- or E-form. Illustrations of using the L- and E-forms:

```
LF1    FREEQ    MF=L              option left to E-form

EF1    FREEQ    (6),MF=(E,LF1)    relative line number specified by
                                  register 6

EF2    FREEQ    ALL,MF=(E,LF1)
```

EXAMPLE: An MTT application program processes an input line from one of its connected terminals and finds that the terminal has finished using the application program. A FREEQ is then issued to the terminal associated with the relative line number in the FINDQ parameter list. The message "YOUR APPLICATION LOGOFF IS COMPLETED" is written to the terminal and it is logically disconnected. The user at the terminal can then proceed to log on as a normal TSS/360 user, or issue the BEGIN command for another (or the same) application program.

```
***          Locate terminal ready to work
POLL         FINDQ
             B       ABC(15)    branch to appropriate routine based on
                                return code
ABC          .
             .
             .
             B       INCOMP     previous WRITEQ with response com-
                                pleted,RC=18
             .
             .
*** Has terminal completed use of application program?
INCOMP       SR      6,6
             SR      5,5
             L       2,0(1)     get pointer to FINDQ parameter list
             DROP    CURREG     drop current base register
             USING   CHAFNQ,2   establish base register for DSECT CHAFNQ
             L       5,FNQMSA   get input message address
             L       7,1(5)     get first word of input message
             C       7,APPOFF   compare first word of input with logoff
                                 code defined in application program
             BE      DISCON
             .
             .
DISCON       LH      7,8(2)     get relative line number
             FREEQ   (7),QUITMSG
             .
             .
```

```
APOFF      DC      C'****'   logoff indicator defined in application
                               program
QUITMSG    DC      AL4(QUITAP) pointer to "quit" message
QUITAP     DC      AL1(L'MSG)
MSG        DC      C'YOUR APPLICATION LOGOFF IS COMPLETED'
```

READQ -- INITIATE READ OPERATION TO TERMINAL (S)

The READQ macro instruction causes the line to a specific terminal device to be placed in the read status; control is immediately returned to the MTT application program.

Name	Operation	Operands
[symbol]	READQ	relative line number-value [,TRNSL={N|Y}]
		[,INTRPT={Y|N}] [,COMPSEL=value]

relative line number
specifies relative line number of entry in the application TCT associated with a terminal to be read; when supplied as register notation, the specified register must contain the (binary) relative line number in the two low-order bytes.

The relative line number can be found in either (A) the parameter list generated by a previous FINDQ (addressability to this list can be established by using, as a base register, the pointer that is pointed to by register 1 and the DSECT CHAFNQ) or (B) the communication area of any external interruption-processing routine that may have been written by the application program (see "INTRPT" option below).

TRNSL
specifies character translation option that indicates if a terminal's character set (or line code) should be translated into internal EBCDIC code. If specified as Y, the input line code will be translated to EBCDIC using the standard translation table for the specific device:

Device	Line code	Internal translation
1050	PTTC/8 (folded)	EBCDIC
2741	PTTC/8 (folded)	EBCDIC
2741	Correspondence (folded)	EBCDIC
teletypewriter	ASCII	EBCDIC

If specified as N, no translation; the input from the terminal will be recorded internally in line form.

Default: TRNSL=Y

INTRPT
specifies external interruptions (with external message number 127) are to be generated and queued on the application's TSI on completion of this read operation (and after any subsequent I/O operations, unless specified otherwise). Y indicates external interruptions should be generated for processing by the application; N indicates no external interruption and that a subsequent FINDQ must be issued to locate completed read operations.

Default: initially N; when this option has been changed, however, the default thereafter will be either Y or N, depending on the most recently issued READQ or WRITEQ in which the INTRPT option has been explicitly specified.

COMPSEL (for IBM 1050 system only)
specifies decimal value indicating type of 1050 unit; if register notation is used, low-order byte of specified register must contain desired value in binary form.

Value	Unit type
5	terminal keyboard
6	reader 1
7	reader 2
0	any input component

Default: COMPSEL=5

INITIALIZATION: Address of a save-area to be placed in register 13.

CAUTION: This macro instruction may be used only in multiterminal task (MTT) application program.

EXECUTION: When READQ receives control, register 1 points to the address of a parameter list (containing the READQ input parameters) in the issuing program's PSECT. The DSECT, CHARDQ, describes this list.

The READQ routine examines this parameter list and checks the entry (or slot) in the application TCT pointed to by the specified relative line number. The associated device number is tested for validity; if valid, the status byte of the application TCT slot is examined for work indications. If an indication of a pending attention interruption or previous I/O operation still in progress is found, return codes are set and then control returns to the application program. Otherwise, the read request and associated options are posted in the TCT slot and control is passed to the terminal communication subprocessor (TCS) via the ATCS system macro instruction. When READQ resumes control, it rechecks for any attention interruptions that occurred between the first check and the issuance of start I/O (SIO) by TCS. If an interruption has been received, the I/O will not be initiated; the appropriate return code will be passed to the application program. If no interruption was entered, an attempt will be made to start the I/O for a channel program. If successful, a READ CCW is on line to that device. Control is then returned to the application program before processing of the read CCW with return code 00 in register 15.

When the read CCW for a terminal is processed, the terminal will be placed in the read status. The terminal user can then enter an input line that will be placed (translated or untranslated) into the input buffer by the channel operation. When the read operation has been completed, the application TCT slot associated with the device will be posted as "message in, complete." An external interruption can optionally be queued on the task's TSI.

Return Data: Control is returned to the issuing application program with the low-order byte of register 15 set to

Code	Meaning
00	normal return
04	invalid relative line number (not connected to application)

08 busy; previous READQ or WRITEQ not completed

0C attention interruption received from terminal

10 solid error occurred during initiation of starting I/O

The read operation will be initiated by this macro instruction only when the return code = 00. For RC=0C, the attention interruption indication will be cleared.

PROGRAMMING NOTES: The implied maximum length of any input message is the buffer length specified by the MTT command that enables the application program's task. Characters beyond this length will be truncated. The message length and its location are returned to the issuing program when the message transmission is complete; they can be found in a parameter list associated either with an external interruption that has external message number 127 (DSECT CHATII) or with the FINDQ macro instruction (DSECT CHAFNQ).

When the translation tables do not apply, an application program may apply its own translation function. In such cases the TRNSL=N option should be specified; the application translation function should be applied against the input recorded in the buffer.

A WRITEQ with a response option is more efficient than issuing separate WRITEQ and READQ macro instructions. The application program can issue a prompting message and place the terminal in a read state; input can then be entered.

If INTRPT=Y is specified, the application program must use the SEEC and SIR macro instructions to set up a routine to service the external interruptions indicating completion of read operations. An external message number of 127 should be specified with SEEC. See also "Processing External Interruptions" under "Application Program Processing."

Any buffer space assigned by a READQ or WRITEQ to the device associated with the application TCT slot is released automatically by subsequent executions of READQ, or a WRITEQ with the response option. However, if the input data is processed, but further READQ or WRITEQ processing is delayed for some reason, the application program can release buffer space by issuing CLEARQ.

Use of L- and E-Forms: The L-form results in a macro expansion consisting of only a parameter list. The name field is mandatory in the L-form; the E-form is used to locate the parameter list. The only operand required is the relative line number, which may be specified as either L- or E-forms.

```
LFORM  READQ  MF=L                relative line number not specified

EFORM  READQ  (5),MF=(E,LFORM)    relative line number in register 5
```

EXAMPLE: FINDQ indicates that a previous WRITEQ has just been completed to a specific terminal, which should now be ready to furnish additional input. Thus, READQ is issued and, if the read is successfully initiated, the application program continues to poll its application's TCT for additional work. When the initiated read CCW reaches the specified terminal, the terminal will be placed in a read-state; the user can enter input. When transmission of this input is completed, an external interruption will be generated but ignored due to the INTRPT option, and the application TCT slot-status byte will be set to indicate the completion. Polling via FINDQ will eventually reflect the completion of the read operation and the application program can then process the data in the input buffer.

```
*** poll terminals for work
POLL        FINDQ
            B       ABC(15)     branch to appropriate routine based on
                                 return code
ABC         .
            .
            .
            B       OUTCOMP     previous WRITEQ completed; RC=14
            B       INCOMP      previous READQ or WRITEQ with response
                                 completed, RC=18
            .
            .
OUTCOMP     L       2,0(1)      get parameter list address
            LH      6,8(2)      load relative line number from FINDQ
                                 parameter list into low-order byte of
                                 register 6
            READQ   (6),Y,N     read device associated with relative

            B       BRTBLE(15)line number in register 6
BRTBLE      B       POLL        continue servicing application's TCT
            .                    word queue
            .
INCOMP      B       PROCESS     service input to application program
```

WAIT -- WAIT FOR TERMINAL STIMULI (O)

The WAIT macro instruction temporarily suspends execution of an MTT application program pending receipt of an external terminal interruption representing work for the task.

Name	Operation	Operands
symbol	WAIT	[environment - A]

environment
specifies task in which WAIT was issued as a multiterminal task; it should be specified as code A or defaulted.

Default: A

CAUTION: Execution of the WAIT SVC 204 is restricted to tasks having authority O or P.

EXECUTION: The WAIT routine will check register 0; if it is 0 (indicating an MTT task), will locate application's multiterminal status control block (MTSCB) pointed to by the issuing task's TSI. It then extracts the virtual storage address of the first application TCT page and scans the pages for any work posted since the last application-program scan. If work has been posted, the task is then redispatched for processing. If no work is found, the task's TSI will be placed in a wait-state. Any incoming work from an application user causes the system to reactivate the application task.

After execution of the system's posting routines or an application program's interruption-servicing routine, in response to a work request, control is returned to the instruction after WAIT.

PROGRAMMING NOTES: If an MTT application program has defined an external interruption servicing routine, when subsequent work requests (i.e., following entering the wait-state) generate interruptions that routine can process the work. Upon return from that interruption routine, processing is continued at the instruction following WAIT.

Any interruption the task allows (or enables), causes the appropriate interruption servicing routine to be redispatched.

EXAMPLE: An MTT application program has polled its users for input and has not found any ready to work. Then it issues WAIT to place the application's TSI in the wait-state until a user enters a work request. Any subsequent requests will be posted in the application program's terminal control-table slot associated with that user's terminal; TCS will redispatch the application program at the next sequential instruction following WAIT (i.e., at SEARCH). FINDQ will then poll the application TCT a second time and find the posted work indication. Control is passed to INCOMP where the work indication can be processed by examining the FINDQ parameter list pointed to by register 1.

```
POLL      FINDQ                 poll user's terminals for work
          B       ABC(15)       branch table using return codes
ABC       B       NOWORK        if no work is ready, enter wait-state
          .                      RC=00
          .
          .
          B       INCOMP        previous READQ completed, process input
          .                      .
          .                      .
NOWORK    WAIT                  wait to be redispatched by TCS
SEARCH    B       POLL          when reactivated, poll application TCT
                                 for work
INCOMP    L       2,0(1)        get parameter list address
          DROP    CURRER        process FINDQ parameter list
          USING   CHAFNQ,2
```

WRITEQ -- WRITE A MESSAGE TO TERMINAL (S)

The WRITEQ macro instruction causes a message to be sent to a specified terminal and, optionally, a user's response to be read from that terminal.

Name	Operation	Operands
[symbol]	WRITEQ	relative line number-value ,area-addr,length-value [,INTRPT={Y|N}] [,BREAK={Y|N}] [,COMPOUT=value] [,TRNSOUT={N|Y}] [,RESP={Y|N}] [,COMPIN=value] [,TRNSIN={N|Y}]

relative line number
specifies relative line number of entry (or slot) in application's TCT associated with terminal to which message is being sent; when register notation is used, specified register must contain (binary) relative number in two low-order bytes.

The relative line number can be found in either (A) the parameter list generated by a previous FINDQ (addressability to this list

can be established by using, as a base register, the pointer that is pointed to by register 1 and the DSECT CHAFNQ) or (B) the communication area of any external interruption-processing routine that may have been written by the application program (see INTRPT option below).

area
specifies address of first byte of output message; byte preceding location should contain length of output message.

length
specifies number of bytes to be written to terminal; when register notation is used, specified register must contain (binary) message length in two low-order bytes; value must not exceed 4080 bytes.

INTRPT
specifies if external interruptions (with external message number 127) are to be generated and queued on application's TSI by system on completion of write operation or, if response option is used, on completion of read operation. Specification of this operand will also affect any subsequent I/O operations, unless those operations specify different options. Y indicates external interruptions are to be generated; in this case the application program can process interruptions. N indicates no external interruptions. When N is specified, a subsequent FINDQ must be issued to locate any completed WRITEQ or a WRITEQ with response.

Default: Initially N; when this option has been changed, however, the default thereafter will be either Y or N, depending on the most recently issued READQ or WRITEQ in which the INTRPT option has been explicitly specified.

BREAK (only for terminals having hardware break-feature)
allows message to be written to terminal even if previous reads or writes are in process; current I/O operation will be interrupted and terminated. Y indicates this message should interrupt any current I/O operation. N indicates that if terminal is in read- or write-state when WRITEQ is executed, no breaks are allowed; busy signal is returned. If the break option is specified for a write issued to a terminal that does not have the hardware break facility, results will be unpredictable.

Default: N

COMPOUT (for IBM 1050 system only)
specifies decimal value indicating type of 1050 unit to be used as output; if register notation is used, low-order byte of specified register must contain the desired value.

Value	Unit type
1	printer 1 (keyboard)
2	printer 2
3	punch 1
4	punch 2
9	any or all output components

Default: COMPOUT=1

TRNSOUT
specifies whether output string text will be translated from EBCDIC to line code using standard translation table for the device. Y indicates standard translation; N no translation. If N is specified, text string will be sent over the line as it exists in the message area.

Device	Unit type
1050	PTTC/8 unfolded
2741	PTTC/8 unfolded
2741	CORRESPONDENCE unfolded
teletypewriter	ASCII

Default: TRNSOUT=N

RESP

specifies whether response is expected immediately after writing output message. Y indicates response is expected after WRITEQ and both write and read operations are to be initiated; N no response expected.

Default: N

COMPIN (only for WRITEQ to 1050 system with RESP=Y)
specifies decimal value indicating type of 1050 unit from which response is expected; if register notation is used, low-order byte of specified register must contain binary form of value.

Value	Unit type
5	keyboard
6	reader 1
7	reader 2
0	any input component

Default: COMPIN=5

TRNSIN (only for WRITEQ with RESP=Y to 1050 system)
indicates if input text string will be translated from line code to EBCDIC (folded), using appropriate translation table defined under TRNSOUT. Y indicates input line code will be translated; N, no translation and input message will be in buffer in line code form.

Default: TRNSIN=Y

INITALIZATION: Address of save-area must be in register 13.

CAUTION: This macro instruction may be used only in an MTT application program; if break option is sent to device that does not have the hardware break facility, results will be unpredictable.

EXECUTION: When the WRITEQ routine receives control, register 1 points to a pointer to a parameter list (containing the WRITEQ parameter specifications) generated in the issuing program's PSECT. The DSECT, CHAWRQ, describes this list. The WRITEQ routine checks the entry (or slot) in the application TCT pointed to by the specified relative line number. The associated device number is tested for validity; if valid, the status byte of the application TCT slot is examined for work indications. If there is an indication that an attention key has been hit or a previous I/O operation is still in progress, return codes are set and then control returns to the application program. If the break option had been specified, and a previous read-I/O operation was found, the previous operation is canceled and the WRITEQ message length is checked.

The WRITEQ request and associated options are posted in the application TCT slot and control is passed to the terminal communication subprocessor to initialize the I/O operation via the ATCS macro instruction. WRITEQ then checks for any attention interruptions that may have occurred between the first check and the issuance of start-I/O TCS. If an interruption was received, the I/O is not initiated and the appropriate return code is passed to the application program. If no interruption was received, an attempt is made to start I/O for a channel program. If successfully initiated, a WRITE CCW and, if RESP=Y, a READ CCW, have been placed on line to that device. Control is then returned to the issuing program before those CCW operations are completed with a return code of 00 in register 15.

On subsequent execution of the channel program, the output message will be transmitted and, if RESP=Y, the terminal will immediately be put in the read state, and an input line will be read into the buffer. Completion of the write/read operation will be signaled by posting the application TCT slot and, optionally, by system generation of an external interruption of the task's TSI.

Return Data: Control is returned to the issuing application program, with a code set in the low-order byte of register 15.

Code	Significance
00	normal return
04	invalid relative line number (not connected to application)
08	busy, previous READQ or WRITEQ not completed (this return will not occur when using break option)
0C	attention interruption received from terminal
10	solid error during start-I/O
14	message length exceeds 4080 bytes

The write operation will be initiated by this macro instruction only when the return code = 00. For RC=0C, the attention interruption indication will be cleared.

PROGRAMMING NOTES: The implied maximum length of any response message is the buffer length specified by the MTT command that enabled the application task. Any characters transmitted beyond this length are truncated.

The actual message length and its location are returned to the issuing program when the message transmission is complete; they can be found in a parameter list associated with either an external interruption or the FINDQ macro instruction.

When the translation tables do not apply, an application program may apply its own translation function. In such cases, the TRNSL=N option should be specified and the application translation function should be applied against the input recorded in the buffer. If the RESP option is specified, and the INTRPT option was specified, no interruption will occur until the subsequent input is completed; the next return from FINDQ for this terminal will indicate "message in, complete."

When a WRITEQ macro instruction is executed, a copy of the output message will be made by the system for transmission. Therefore, it is not necessary for the issuing program to retain the original message after successful execution of the macro instruction.

A WRITEQ with the response option is a more efficient procedure than issuing separate WRITEQ and READQ macro instructions; a prompting message can be issued; the terminal will be in the read-state and input can be entered.

If INTRPT=Y is specified, the application program must have used the SEEC and SIR macro instructions to set up a routine to service the external interruptions that indicate completions of read operations. External message number 127 should be specified with SEEC. (See also "Processing External Interruptions" under "Application Program Processing.")

Any buffer space assigned by a READQ or WRITEQ to the device associated with the application TCT slot is released automatically by subsequent execution of READQ, or WRITEQ (with or without the response option). However, if the input data is processed, but further READQ or WRITEQ processing is delayed for some reason, the application program can release buffer space by issuing CLEARQ. READQ and WRITEQ do not clear work indications; only FINDQ and CLEARQ will clear them.

The channel program will transmit the message, as it is in the message area, using the specified translation. It is not necessary, on 1050 terminals, to put addressing, polling, and EOB characters into the message. The user program must, however, provide the CR if necessary. Furthermore, if the message is to be printed as multiple lines, the issuing program must insert a CR character to end each line, followed by sufficient idle characters to allow the carriage-return action to be completed, before continuing with the text string in the next line.

Use of L- and E-Forms: The L-form macro instruction results in a macro expansion consisting of only a parameter list. The name field is mandatory in the L-form; it is used by the E-form to locate the parameter list. Examples:

```
LFORM   WRITEQ   ,MSGOUT,RESP=Y,MF=L     (relative line-number length
                                          not specified)

EFORM   WRITEQ   (3),,(4),MF=(E,LFORM)   (area not specified)
```

The L-form established a parameter list, defining the message-output area (MSGOUT) and specifies the RESPONSE option. The E-form assumes the message has been placed at MSGOUT. Register 3 contains relative line relative line number and register 4 contains the message length. Omitted I-form operands assume the value established by the L-form (e.g., the E-form macro above does not default to RESP=N).

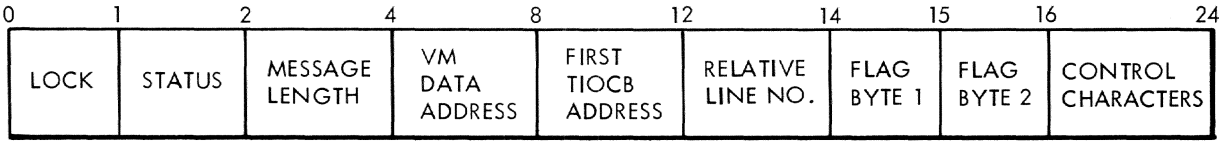
EXAMPLE: FINDQ indicates that a terminal has just logged on to use the application program. WRITEQ, with a response option, is then issued to prompt that terminal for input that will be read into a buffer. If the write, with response option, is successfully initiated, the application program continues to poll the application TCT for additional work.

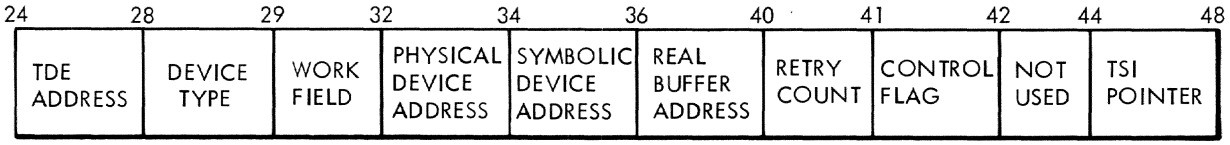
When the initiated channel program is executed, "ENTER APPROPRIATE INPUT" will be displayed at the terminal, terminal will be placed in the read status, and an input line will subsequently be read into the buffer.

```
*** locate terminal ready to work
***
POLL        FINDQ
            B       ABC(15)    branch based on return code
            .
            .
ABC         .
            .
```

```
                B       NITCON    terminal initially connected to
                                   application program; RC=08
                .
                .
NITCON          L       2,0(1)    get address of FINDQ parameter list
                LH      6,8(2)    load relative line number of initially
                                   connected terminal, from FINDQ
                                   parameter list, into low-order
                                   byte of register 6
                WRITEQ  (6), PRMPT,LNGTH,N,,,TRNSOUT=Y,RESP=Y,,Y
                B       TBL(15)
TBL             B       POLL      successful I/O initiation
                .
                .
LNGTH           DC      AL2(L'PRMPT)     length of output message
PRMPT           DC      C'ENTER APPROPRIATE INPUT'
```

An entry (or slot) in an application's terminal control table (TCT) is assigned to every user connected to an application program. The layout of a typical TCT entry is shown below. Application programs indirectly scan these entries (via the MTT macro instructions: READQ, WRITEQ, FINDQ, etc.) to determine if terminals have requested services. A DSECT (CHATCT) is available for symbolically addressing fields in the TCT entries; it is described in System Control Blocks.





TCT Entry (48 bytes)

Status-byte settings (set by TCS routine) are:

Hex code	Meaning
X'80'	prepare CCW indicator bit
X'40'	read required bit
X'20'	buffer overflow
X'10'	PCI bit
X'08'	two-page data area
X'04'	supervisor page for write data
X'02'	complete I/O bit
X'01'	halt I/O flag

FLAG BYTE 1 -- (set by macro instruction SVCs)

Hex code	Meaning
X'80'	read operation
X'40'	write operation
X'20'	write/response operation
X'10'	clear operation
X'08'	free operation

FLAG BYTE 2 -- (set by macro instruction SVCs)

Hex code	Meaning
X'80'	interruption required on task
X'40'	translate on in messages
X'20'	translate on out messages
X'10'	break to be issued
	control characters

DEVICE TYPE

Hex code	Meaning
X'00'	slot available mask
X'01'	1050 PTTC/8
X'02'	2741 CORRESPONDENCE
X'03'	2741 PTTC/8
X'04'	teletypewriter ASCII

WORK BYTE

Hex code	Meaning
X'180'	message in
X'40'	message out
X'20'	attention
X'10'	initial connection
X'08'	unrecoverable error
X'04'	negative polling response
X'02'	buffer overflow flag

Value mnemonics help the user remember the forms a particular operand may assume. The value mnemonics used in TSS/360 publications are:

relexp
addr
addrx
addx
integer
absexp
value
text
code
symbol
characters
name
specsym
alphnum

In macro instruction descriptions, each positional operand is specified by a meaningful name hyphenated with a value mnemonic, as illustrated:

Name	Operation	Operand
[symbol]	EXAMPLE	name-value mnemonic

Each keyword operand is specified by the keyword, an equal sign, and a value mnemonic, as illustrated:

Name	Operation	Operand
[symbol]	EXAMPLE	KEYWI=value mnemonic

One or more operand forms may be substituted for each value mnemonic. For example, the value mnemonic, relexp, denotes that a relocatable expression may be written as the operand form; the value mnemonic, addx, specifies that an explicit address or an implied address may be written.

The 10 operand forms are:

relocatable expression
register notation
explicit address
implied address
symbol
decimal integer
absolute expression
code
text
characters
data set name
special symbol
alphameric characters

Operand Forms	Value Mnemonics													
	relexp	absexp	addr	addrx	addx	integer	value	text	code	symbol	characters	name	alphnum	specsym
Relocatable Expression	X		X											
Register Notation			X	X			X							
Explicit Address				X	X									
Implied Address (Indexed)				X	X									
Symbol										X				
Decimal Integer						X								
Absolute Expression		X					X							
Code									X					
Text								X						
Characters											X			
Data Set Name												X		
Alphameric Characters													X	
Special Symbol														X

Note: An X indicates that the operand form may be written.

Multiterminal Task Programming and Operation Printed in U.S.A. GC28-2034-1

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 Technical Newsletter

File Number	S360-50
Base Publication No.	GC28-2034-1
This Newsletter No.	GN28-3184
Date	9/30/71
Previous Newsletters	None

IBM SYSTEM/360 TIME SHARING SYSTEM
MULTITERMINAL TASK PROGRAMMING AND OPERATION

This Technical Newsletter, a part of Version 8, Modification 1, of the IBM System/360 Time Sharing System, provides replacement pages for the subject publication. Pages to be inserted and/or removed are:

Contents

1-2

5-6

A change to the text is indicated by a vertical line to the left of the change.

Summary of Amendments

This Technical Newsletter changes the user authority code required for the MTT command.

IBM Corporation, Programming Publications, Dept. 636, Neighborhood Road, Kingston, N.Y. 12401

PRINTED IN U.S.A.