CONTROL DATA
CORPORATION

CONTROL DATA®
6000 COMPUTER SYSTEMS

INTERCOM MULTI-USER JOB CAPABILITY PROGRAMMING SYSTEMS BULLETIN
6000 VERSION 3

New features, as well as changes, deletions, and additions to information in this manual are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

REVISION RECORD

REVISION	DESCRIPTION
A	Original printing.
(2-26-71)	
B	This revision includes addition of information on a subroutine that provides user status to a muj, notes
(10-8-71)	on the creation of absolute overlay files, minor changes in the text to improve accuracy or clarify
	meaning, and corrections of typographical errors. This revision constitutes a complete reprint, since
	some of the changes affect pagination throughout the manual.
C	New information included on: TIO system error, MSYSERR routine, and expanded MUJ System Errors
(6-16-72)	Table. Pages affected: iii, v, 18, 21 thru 25, and Comment Sheet.
D	Includes muj accounting information, additions to the descriptions of the USER subroutine state parameter,
(2-13-74)	and revisions to the TIO subroutine processing sequence details. Pages affected: iii, 3 through 26, and
	Comment Sheet.

Publication No.
60327400

Additional copies of this manual may be obtained from the nearest Control Data Corporation sales office.

Address comments concerning this manual to:

CONTROL DATA CORPORATION
Software Documentation
215 MOFFETT PARK DRIVE
SUNNYVALE, CALIFORNIA 94086

or use Comment Sheet in the back of this manual

CONTENTS

INTRODUCTION

INTERCOM Version 3.0 includes a facility for a job to provide processing for many users at once. This type of job is called a multi-user job (muj). The INTERCOM facility is a set of system muj subroutines. Using these subroutines, a program can be written to handle many users simultaneously, usually in a serially reentrant manner. Normally, a program using the muj capability would be written only for heavily used applications in which users remain active for a considerable period of time.

A single-user job cannot be easily transformed to a multi-user job. A muj program must be specifically designed and coded to handle the reentrance level required by the application.

The following reference manuals provide further information on the Control Data computers, the SCOPE system, INTERCOM Version 3.0, and the compilers available for use with INTERCOM.

	Publication No.
Computer Systems Reference Manual	60100000
SCOPE 3.3 Reference Manual	60305200
ALGOL-60 2 Reference Manual	60306100
ALGOL-60 3 Reference Manual	60329000
BASIC 2 Reference Manual	60306200
COBOL 3 Reference Manual	60253000
COMPASS 2 Reference Manual	60279900
FORTRAN 2.3 Reference Manual	60174900
FORTRAN 3 Extended Reference Manual	60176600
FORTRAN Extended Reference Manual 6000(3)/7600(1)	60329100
INTERCOM 3 Reference Manual	60252800

This product is intended for use only as described in this document. CONTROL DATA cannot be responsible for the proper functioning of undescribed features or undefined parameters.

MULTI-USER JOB CAPABILITY

INTERCOM includes a facility for a job to provide processing for many users at once. This type of job is called a multi-user job (muj). The INTERCOM facility is a set of system muj subroutines. Using these subroutines, a program can be written to handle many users simultaneously, usually in a serially reentrant manner. Normally, a program using the muj capability would be written only for heavily used applications in which users remain active for a considerable period of time.

A single-user job cannot be easily transformed to a multi-user job. A muj program must be specifically designed and coded to handle the reentrance level required by the application.

MUJ ACTIVATION

Multi-user jobs are activated and associated with users when the muj name is entered as a command at a terminal. When this command is issued, the muj specified will be activated if no other user is being serviced by the muj. If it is already active, the terminal will be associated with the muj in the INTERCOM internal tables.

The terminal becomes disassociated from the muj and returns to INTERCOM command mode through a request issued by the muj to the INTERCOM system. Such a request is made because the muj job is completed for the terminal, or because the terminal user's entry is interpreted by the muj as a request to return the user to command mode.

A muj cannot terminate in the normal manner. When no users are attached, INTERCOM will terminate the muj (no reprieve is possible from this termination).

This termination involves detaching the user's local and permanent files from the muj. Normal SCOPE end-of-job processing is not performed for a muj (or any INTERCOM job); therefore, unless DISPOSE has been specified for the muj local files, all local files (such as OUTPUT) will be lost.

The INTERCOM system communicates with the muj through two tables in the muj field length. Some users may wish to define and process these tables directly. These users are referred to the INTERCOM Internal Maintenance Specification for a description of the table formats TERMIN and TERMOUT. Most users will find it more convenient to reference the tables through the set of system muj subroutines. These subroutines define and access the tables, as well as provide other processing required for proper functioning of the muj.

SYSTEM MUJ SUBROUTINES

The muj subroutines may be called from programs written in COMPASS, COBOL, or FORTRAN Extended. Because of the different data formats used by the compilers and the different ways of handling input/output, the entry point names for the muj subroutines may vary from one compiler to another; however, the basic functions of the subroutines do not change. In addition to initializing the muj when it is first loaded (INMUJ subroutine), the subroutines provide the following basic capabilities:

USER SCHEDULING AND SWITCHING

Normally, a muj is expected to process only one user task at a time. At times however, the current user reaches a point in processing where he must wait on some activity (such as I/O, a file attach request, or condition internal to the muj, such as waiting for a free buffer). The muj can perform a user switch through a call to the USER subroutine. The USER routine maintains a scheduling queue for users ready for processing and inserts users into this queue, as they become ready. In the user switch, the current user is marked as waiting for a specific activity, and a user ready for processing is returned to the muj. When USER determines that the activity for a waiting user is complete, that user will be placed in the queue and eventually switched back to the muj to continue processing.

USER AREA

The muj can maintain information associated with each user. This information is kept in a user area buffer. A user area buffer is (optionally) rolled out to mass storage by the muj subroutines when the user is not being processed by the muj and is rolled back in when processing is needed. The muj programmer may define one or more user area buffers. The USER subroutine attempts to optimize buffer usage by looking ahead whenever a buffer is free and rolling in user areas for users most likely to be processed next, rather than waiting for a user switch request.

ATTACHING USER'S FILES TO MUJ

To avoid possible problems caused by duplicate file names, a user's files are not attached to the muj automatically when the user enters the muj. User files are attached or created by a call to the USER subroutine. To handle duplicate file names, the USER subroutine inserts the FNT address of an attached (or created) file in the fifth word, bits 59-48, of the file's FET within the muj field length. When an I/O operation is performed on a user file, CIO is directed to the FNT entry by the pointer in the FET.

However, CIO can use the FNT pointer only if the FET's for user files are at least six words long. The FNT pointer is not checked by CIO for the minimum length FET, 5-words. Three other conditions must be satisfied by FET's for attached users' files: The FET must be in core (it must not be included in a user area that may be swapped out). Only one FET must be defined for each attached user's file. The FET must not be moved. The last three conditions are required for any muj intended to run under future versions of INTERCOM in which FNT's will be swapped when a job is swapped out.

The USER routine is called also to delete or detach files from the muj. Detached files are returned to the user's list of files at control point zero. The FET must remain defined until a user's file is detached, as the FNT pointer in the FET is used also by the detach routine.

If the muj has its own files for scratch, data base, or other use, it must either define these files before any users' files are attached and follow the same rules for FET's as indicated above for attached users' files; or it must determine that no name conflicts exist between its files and attached users' files.

MUJ ACCOUNTING

Each time the USER subroutine returns a user to the muj in a ready state, the subroutine increments two servicing counts. One count is maintained in an internal table associated with the user to record the number of times the muj provided processing for that user since initialization. The second count is incremented to tally the total number of muj services for all users since the muj was initialized.

When a user's servicing count reaches a threshold value, USER performs the following accounting procedure:

Reads the total CP/PP time used by the muj since initiation.

Computes the percentage (n%) of the total muj services received by the specific user.

Calculates the charges to the user for an equal percentage (n%) of the total CP/PP time used by the muj.

Reinitializes the user's servicing count to zero to restart the accounting cycle.

If the maximum authorized time limit allotted to a user is exceeded, the state parameter returned by the USER subroutine informs the muj. The muj programmer is responsible for ensuring that the specific user is denied any further muj processing except as required for exiting from the muj without drastic consequences; for example, a multi-user text editor program could be programmed to allow the user to save his file before forcing an exit.

CREATING A MUJ USING MUJ SUBROUTINES

Assume:

The muj is present on an UPDATE program library called MUJ; the muj UPDATE deck name is AMJ.
The name or ident of the main muj routine is AMUJ.
The muj is to be written as an absolute overlay on file MUJABS.
The system muj subroutines are on an INTERCOM program library called PL8A.

Job decks as follows:

IF NO PREPROCESSORS USED AND MUJ CODED ENTIRELY IN COMPASS

```
1.   Job Card
2.   REQUEST,MUJ.                    The muj UPDATE P.L.
3.   REQUEST,PL8A,E.                 The INTERCOM P.L.(labeled)
4.   UPDATE(Q,P=MUJ)
5.   COMPASS(I=COMPILE)              Assemble AMUJ
6.   UPDATE(Q,P=PL8A)
7.   FTN(I=COMPILE,S=SCPTEXT)        Compile System muj routines.
8.   LOAD(LGO)
9.   NOGO.                           Create overlay on file MUJABS.
10.  REWIND(MUJABS)
11.  EDITLIB.
12.  7/8/9
13.  *COMPILE AMJ
14.  7/8/9
15.  *COMPILE MUJSUBS
16.  7/8/9
17.  READY(SYSTEM)
18.  DELETE(AMUJ)
19.  ADD(AMUJ,MUJABS)
20.  COMPLETE.
21.  6/7/8/9
```

IF FORTRAN PREPROCESSORS USED AND MUJ CODED IN FORTRAN (and/or COMPASS)

```
1.   Job Card
2.   REQUEST,MUJ.
3.   REQUEST,PL8A,E.
4.   UPDATE(Q,P=MUJ)
5.   FTN(I=COMPILE)                    Compile AMUJ.
6.   UPDATE(Q,P=PL8A)
7.   FTN(I=COMPILE,S=SCPTEXT)          Compile System muj routines
8.   FTN(I=COMPILE,S=SCPTEXT)          Compile preprocessors
9.   LOAD(LGO)
10.  NOGO.
11.  REWIND(MUJABS)
12.  EDITLIB.
13.  7/8/9
14.  *COMPILE AMJ
15.  7/8/9
16.  *COMPILE MUJSUBS,FTNMUJ
17.  7/8/9
18.  READY(SYSTEM)
19.  DELETE(AMUJ)
20.  ADD(AMUJ,MUJABS)
21.  COMPLETE.
22.  6/7/8/9
```

If the COBOL preprocessors are used and the muj is coded entirely in COBOL, the deck setup is similar to the deck using FORTRAN preprocessors except:

For card 5, COBOL(I=COMPILE,E=AMUJ) replaces FTN(I=COMPILE)
For card 11, substitute REWIND(COBCODE)
For card 16, substitute *COMPILE COBOMUJ,MUJSUBS
For card 20, substitute ADD(AMUJ,COBCODE)

Important Note:

The cards

```
LOAD(LGO)
NOGO.
```

shown above, have the effect of creating an absolute overlay called AMUJ which is in the proper format to be EDITLIBed. Overlay AMUJ will consist of:

a) All of the users routines from AMJ

b) The system muj routines from MUJSUBS (plus preprocessors when requested)

c) library routines referenced by (a) and (b)

However, if the user's muj consists of a main overlay and primary overlays, the overlay AMUJ created by the above sequence of control cards will consist of

a) All of the users routines from AMJ which occur before the second OVERLAY card on LGO.

b) Library routines referenced by (a)

In this latter case, AMUJ will not include the system muj routines. Only the last overlay on LGO will include the system muj routines. This is because the loader, in creating an overlay, searches LGO until it encounters another OVERLAY card, or until it reaches the end of the file (whichever comes first). If the users wishes the system muj routines to reside in the (0,0) overlay (as is most reasonable) he must reorder the relocatable binaries on LGO before creating his overlays. One suggested way to accomplish this is by using various SCOPE file utilities. Specifically (for a COMPASS muj) the user should replace cards 7 thru 9 with the following cards:

```
FTN(I=COMPILE,S=SCPTEXT,B=FTNBIN)   Put system muj routines on FTNBIN
REWIND(LGO,FTNBIN,ALLBIN)
COPYBR(LGO,ALLBIN,2)                Put first OVERLAY routine on ALLBIN
COPYBF(FTNBIN,ALLBIN)               followed by system muj routines
SKIPB(ALLBIN)                       Backspace over end-of-file
COPYBF(LGO,ALLBIN)                  Transfer remainder of LGO
LOAD(ALLBIN)
NOGO.
```

If the muj status request (MUJSTAT) is to be used by the muj, the *COMPILE card which includes MUJSUBS should be modified to include MUJSTAT as well (e.g., for FORTRAN, card 16 should be *COMPILE MUJSUBS, FTNMUJ, MUJSTAT).

Changes, as described in the INTERCOM V3 Installation Handbook, are needed in tables in INTERCOM PP routines 1SJ and 1QP to recognize a program as a muj.

The descriptions of the multi-user subroutines follow. The subroutine name and the parameters are required for calling the subroutine from COMPASS. For calls from COBOL or FORTRAN programs, if the entry point name or the parameters differ, they are described under the sections Pre-processors and Calling Sequences.

Arrays, or central memory storage blocks may be set up as tables in COBOL; normally the data type is not relevant, although all information should begin and end on word boundaries.

INMUJ

Performs initialization for multi-user jobs. The file TERMIOF is connected; this file is used for terminal I/O. The addresses of the TERMIN/TERMOUT tables used by the muj are initialized in the muj table in central memory resident. The muj passes, as parameters to the INMUJ routine, several tables used by the USER, TIO and MUJSTAT subroutines; the muj must allocate space for these tables, but the tables are used only by the subroutines. These tables, along with others internal to the multi-user subroutines, are initialized by the INMUJ subroutine.

CALLING SEQUENCES

FORTRAN Extended:

```
    CALL INMUJ (mujtbl, nfile, nfet, nuser, narea, larea,uarea)
```

COMPASS:

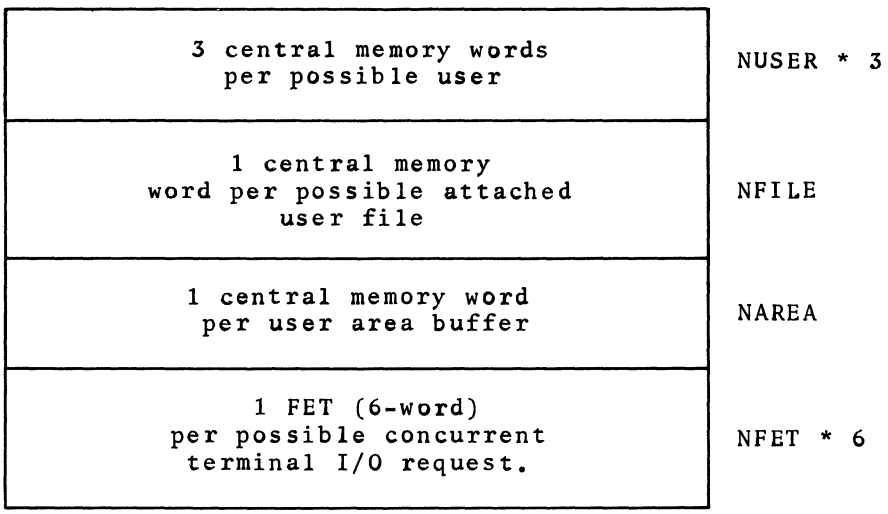
```
              SA1   INPAR
    +         RJ    INMUJ
              .
              .

    INPAR     VFD   60/mujtbl,60/nfile,60/nfet
              VFD   60/nuser,60/narea,60/larea,60/uarea
              DATA  0
```

COBOL:

```
    ENTER     INMUJCO   mujtbl, nfile, nfet, nuser, narea,
                        larea, uarea
```

PARAMETERS

mujtbl - A block of central memory words used by the muj subroutines for internal table space. This block is an array in FORTRAN, a BSS or other storage reservation instruction in COMPASS, or a table in COBOL. The length of the block is determined by the value of the other parameters in the INMUJ call. The algorithm for determining the length of the block is nfile + 6*nfet + 3*nuser + narea. Format of mujtbl is shown below:



nfile - The maximum number of user files that can be attached simultaneously to the muj. These files are attached by a call to the USER subroutine. Each attached file requires a 1-word entry in the mujtbl to identify the user who owns the file and the file's FET address. The number of attached files may be related to the number of terminals; for example, one file per terminal. nfile is a binary number in COMPASS, an integer variable in FORTRAN, and a computational-1 item in COBOL.

nfet - The number of 6-word file environment tables provided in the mujtbl for terminal I/O. Ideally, one FET should be provided for each terminal; however, this may not be possible because of memory restrictions. If no FET is available when a terminal I/O request is issued through TIO, the request will be rejected; and it will be necessary to re-issue the request later. nfet is a binary number in COMPASS, an integer variable in FORTRAN, and a computational-1 item in COBOL.

nuser - The number of users the muj may process simultaneously. When the number of users in the muj is equal to nuser, other users will be denied access; and a message will instruct them to try later. The mujtbl requires three words for each user for status information, accounting, etc. nuser is a binary number in COMPASS, an integer variable in FORTRAN, and a computational-1 item in COBOL.

narea - The number of user area buffers provided by the muj for rolling user areas in and out of central memory. The mujtbl contains one word per user area for scheduling user area buffers. If the muj is not employing user areas, narea may be zero, or the last three parameters may be unspecified. If specified, narea is a binary number in COMPASS, an integer variable in FORTRAN, or a computational-1 item in COBOL.

larea - The length of each user area buffer, in central memory words. If user areas are not employed by the muj, larea may be zero or not specified. It is a binary number in COMPASS, an integer variable in FORTRAN, or a computational-1 item in COBOL.

uarea - The name of the block of central memory provided for user area buffers. The length of the block should be narea * larea. If user areas are not employed by the muj, uarea may be zero or not specified. It may be a BSS or other storage allocation instruction in COMPASS, an array in FORTRAN, or a table in COBOL.

PRE-PROCESSORS

FORTRAN: none

COBOL: INMUJCO: Converts computational-1 items to binary integer numbers.

SUGGESTIONS FOR USE

The INMUJ routine must be called only once by the muj before any other multi-user subroutines are called.

The time for user switches and the number of rejected file action requests or terminal I/O requests decreases in relation to the following factors: Amount of storage allocated for terminal FETs and file name entries in the mujtbl; Number of user area buffers allocated.

USER

Provides user scheduling and switching and performs file action requests, such as attaching and detaching user files. The muj calls the USER subroutine to place, in its scheduling queues, a user task that is temporarily halted and to request another user ready for processing. The USER routine keeps track of user status and schedules users for muj processing when they are ready.

When the USER subroutine passes a user to the muj for processing, conditions which rendered the user ready are indicated in the state parameter. The muj may influence user scheduling by requesting that a specific user be returned for processing. If a user requested through the newu parameter is ready for processing, the user is returned to the muj, along with a state indication. A user not ready for processing is not returned for processing; but his status is indicated in the state parameter.

The actn parameter specifies the action to be performed for the user being returned to the scheduling queues (by a call to USER) and the conditions under which the muj will again accept this user for processing. The action may be a request to perform a file management function, such as attaching or detaching a user file from the muj; or it may be a request to detach a user from the muj. The return condition may be based on the status of the terminal - for instance, when the terminal is performing an I/O operation. Another condition for which the muj may request a user switch is a wait on completion of some operation not necessarily associated directly with the user. This operation's completion is indicated by the setting of a complete bit, bit 0, in a central memory word; the complete bit is zero if the operation is not complete, and one when it is completed. This return condition allows user processing to wait for completion of I/O operations. Consequently, the complete bit is expected to appear as the complete bit in the code-and-status field of a file environment table. In FORTRAN and COBOL programs, the location of the complete bit is specified in the subroutine call by a logical unit number (FORTRAN) or a file name (COBOL).

If the muj is employing user areas, the USER routine will switch user areas as well as users. The user area associated with the old user is specified by the muj with an indication of whether or not it should be kept in central memory (as it would be, for instance, if it contained an FET or a buffer for an I/O request currently in process). A pointer to the user area associated with the new user will be passed to the muj on return from USER.

CALLING SEQUENCES

FORTRAN:

```
CALL USERFO (oldu, actn, state, newu, iarea, lwait, flun, fname)
```

COMPASS:

```
        SA1   USPAR
        RJ    USER
        .
        .
        .
USPAR   VFD   60/oldu,60/actn,60/state,
              60/newu,60/iarea,60/complt,60/fetadd
        DATA  0
```

COBOL:

```
ENTER USERCO   oldu, actn, status, ibreak, iperms, newu, iarea, fwait, fname
```

PARAMETERS

oldu - The user ID (the ID given to the muj by USER when this user was passed to the muj for processing) of the old user to be placed in the scheduling queue, or detached from the muj. If no user is being returned, oldu=0. This parameter is a binary number in COMPASS, integer variable in FORTRAN, and a computational, synchronized left variable in COBOL; it is an input parameter. The ID is actually two display code alphanumeric characters, left-justified with trailing binary zeros.

actn - A number representing the action to be performed on this user switch. Most actions refer to the old user, oldu and define what must be done before this user is returned to the muj. This number is a binary number in COMPASS, an integer variable in FORTRAN, and a computational-1 item in COBOL; it is an input parameter.

The value for actn can be zero or actn may assume the following positive (+) or negative (-) values:

actn < 0 — The user area associated with the old user (pointed to by iarea) will not be rolled out; it will be left in central memory. This action may increase the time required for user switches, as the number of available user area buffers is decreased.

actn > 0 — The information in the user area associated with the old user will be rolled out to the user area swap device (ECS or rotating mass storage); and the buffer in which it resides may be used for other user areas. If no user areas are employed actn may be either positive or negative.

actn=0 — No old user is being returned; the only action requested is to switch in a new user for processing. When the muj first calls USER, actn = 0 because the muj has no user to return.

actn=±1 oldu should be returned for processing when input from his terminal is ready. The muj must not issue a read request (via TIO) for the terminal until input is ready.

actn=±2 oldu may be returned for processing any time. If any other terminals are ready for processing, however, they will be returned before this user.

actn=±3 oldu should be returned for processing when the complete bit (bit 0) whose address is in the complt word is set to 1. Complt may be anywhere in the muj field length; it is referenced from FORTRAN or COBOL programs respectively by a logical unit number or a file name.

actn=±5 Request to detach oldu from the muj. The INTERCOM system will be notified that this user is no longer attached and his user area will be released by the USER subroutine. All files currently attached to the muj for this user will be detached by INTERCOM.

actn=±6 Wait for RUN complete. (For use by INTERCOM EDITOR only)

actn=±7 Wait for the buffer in use by a terminal output operation to be free. The output FET will also be completed and available for another terminal I/O request.

actn=±9 Wait until all output has reached the terminal. Normally, this action is taken when the USER subroutine informs the muj that the specific user's output is backlogged (see Suggestions for Use under the TIO subroutine, and flag bit 57 of the USER state parameter).

actn=±11 Create a new local user file associated with the user oldu. The FET specified by fetadd will be used to specify the file name. oldu will be returned for processing when the various table entries for the file have been made.

actn=±12 Attach the user file named in the FET specified by fetadd to the muj. This file will be attached by extracting it from the list of files associated with oldu and making entries in SCOPE tables to associated the file to the muj. In the internal tables of the USER subroutine, however, the file will remain associated with oldu; so that it may be returned to the proper user when it is detached from the muj. oldu will be returned for processing when the file is attached.

actn=±13 Delete the file named in the FET specified by fetadd. An unload operation will be requested. oldu will be returned for processing when the file has been deleted. The muj must have previously attached or created the file before issuing the delete request.

actn=±14 Detach the file named in the FET specified by fetadd from the muj, placing or replacing it in the list of local user files from which it was extracted. Users' files should be detached as soon as possible after they are attached to avoid overflowing the file name table in the USER subroutine. oldu will be returned for processing when the file is detached.

actn=±16 If the file in the FET specified by fetadd exists, attach it to the muj; otherwise, create a new local user file by that name.

state - A number representing the state of the user, newu, being sent to the muj for processing. This parameter is a binary number in COMPASS, an integer variable in FORTRAN; it is an output parameter. Bits 0-17 contain the state value. If this value is in the range 10-19, bits 18-22 contain additional information (below). In COBOL, three computational-1 output parameters are needed to contain status information: status contains the state bits 0-17; iperms contains the five permission bits described as 18-22 of state below; ibreak contains the break bit described below as bit 59 of state.

The values which state may assume fall into four general areas:

1. If one or more of the state parameter's flag bits (59, 58, and 57) are set, user newu is affected as follows:

   a. If bit 59 of state is set, a break (%A, ESC A, or CTRL-Z A) has been entered at the newu terminal, or a temporary terminal disconnect has occurred. The remainder of state may be tested to determine other conditions under which newu is returned for processing. (Normally, state=1 indicates that input from the terminal is ready for user newu; however, if state=1 and bit 59 is set, input from the terminal may or may not be ready. Programming the muj to call subroutine USER again with actn=±1 returns the user to the muj only when input is ready.)

   b. If bit 58 of state is set, user newu has exceeded his maximum time limit as specified in the INTERCOM password file. When the user runs out of time, all further processing should be denied him except as required for exiting from the muj without drastic consequences.

   c. If bit 57 of state is set, user newu has an output backlog which has not yet reached his terminal. Subsequent attempts to send output to that terminal may be delayed unless the backlog is cleared before the next write is issued (see actn=±9 of the USER subroutine and Suggestions for Use under the TIO subroutine).

2. If 1 ≤ state bits 0-17 ≤ 9, newu is either a user who has just been attached to the muj or who was returned to the scheduling queues earlier with an actn parameter of 1-9 and is now ready for further processing.

3. If 10 ≤ state bits 0-17 ≤ 19, the user newu was returned to the scheduling queues earlier with an actn parameter of 11-16 (request for file action) and is now ready for further processing. For attach file requests, bit 22 of state will be set to one if the file is a permanent file. Bits 21-18 are set to indicate the permissions granted for the file: bit 21=1 for control; bit 20=1 for modify; bit 19=1 for extend; and bit 18=1 for read permission.

4. If 20 ≤ state bits 0-17 ≤ 30, the user newu was specifically requested for processing by the muj (see description of newu below), but he is not ready according to the conditions under which he was placed in the scheduling queues. State indicates his current status. This user must be requested again but may not be returned to the USER subroutine until given to the muj by USER in a ready status.

The state bits 0-17 may assume the following values:

state=0 newu is ready; this user had been given to the muj for processing once before and has not been returned to USER since. Therefore, this user is still regarded as ready. This case will arise only when the muj is doing its own user scheduling and has specifically requested newu for processing. It will not arise if the current user is always returned whenever a new user is requested because the muj is processing only one user at a time. A muj system error will be indicated if it requests a user it is currently processing.

state=1 Input from the terminal is ready for newu.

state=2 newu is ready and is being returned under no special conditions, but rather it is this user's turn for processing.

state=3 The complete bit whose address was in the complt word (indicated when newu was returned to USER from the muj as oldu, with action indicator of ±3) has been set to one.

state=4 newu is not attached to the muj, but was passed to USER as the oldu to be returned for scheduling. If a specific newu was requested, this case is the only one in which the status of that specific newu will not be returned.

state=5 The terminal user has just issued a command to become attached to the muj.

state=6 RUN is complete. (For use by INTERCOM EDITOR only)

state=7 The buffer used in the muj for output to this user's terminal is free.

state=8 The user was logged out automatically after a disconnect. This state occurs if the user does not log back into INTERCOM before a prescribed period of time (installation option). The muj should perform any clean-up for this user, and it must detach the user from the muj.

state=9 All backlogged output for this user has reached the terminal (see Suggestions for Use under subroutine TIO).

state=10 The file has been attached to the muj for the user newu.

state=11 The file could not be created because newu already has a file by that name.

state=12 The file could not be found in list of files for user newu (or, for detach and delete, in the muj local files). It could not be attached (or detached or deleted).

state=13 The file could not be attached because it was connected to the terminal and not disconnected before newu became attached to the muj.

state=14 The file belonging to newu was deleted.

state=15 The file was created as a new user file for newu.

state=16 The file was detached from the muj and returned to the newu list of files.

state=17 The file could not be created or attached because the USER subroutine tables were full. (The maximum number of attached user files, specified in the call to INMUJ, has been reached.)

state=18 The file could not be added to the list of newu files because his INTERCOM file limit has been reached.

state=20 newu is not attached to the muj. This state can occur only if newu was requested specifically by the muj.

state=21 newu is waiting for input from the terminal.

state=22 newu is ready for processing but no user area buffer is free to be assigned to newu. The muj must process at least one user before requesting this user again.

state=23 newu is waiting for the complete bit to be set in the word whose address is in the complt word.

state=24 newu is waiting for backlogged output to reach the terminal.

state=25 newu is waiting for the terminal output buffer to be free.

state=26 The user area assigned to newu is in the process of being rolled in or out to disk.

state=27 newu is waiting for a file attach or create request.

state=28 newu is waiting for a file detach or delete request.

state=29 newu is waiting for completion of a RUN operation (for use of INTERCOM EDITOR only).

state=30 newu is in the process of being detached from the muj (following a USER call with actn=±5).

newu - The user ID of the new user being returned to the muj for processing. This parameter is a binary number in COMPASS, an integer variable in FORTRAN, and a computational, synchronized left variable in COBOL; it is both an input and an output parameter. If newu is zero on entry to the USER subroutine, the subroutine will process the scheduling queues to find the next user ready for processing and submit it as newu. If newu is not zero on entry, it will be interpreted as an input parameter representing the user ID of a user specifically requested by the muj for processing. This user will be returned for processing only if he is ready. Ready or not, however, his status will be indicated in the state parameter. (See states 20 through 29).

iarea - A pointer to the relative location in the array of user area buffers employed to hold the user area for the old/new user. This pointer is a binary number in COMPASS, an integer variable in FORTRAN, and computational-1 item in COBOL. It is used as both an input and an output parameter. On input (entry to USER) it points to the user area used by oldu, on output (exit from USER) it points to the user area used by newu. If newu is not ready for processing, iarea is meaningless. If the muj is not employing user areas, iarea should be zero or unspecified. iarea is a pointer in the following sense: if uarea is the name of an array or table allocated for user area buffers, the first word of the user area to which iarea points could be referenced in FORTRAN or COBOL programs as uarea (iarea); in COMPASS, as uarea+iarea-1. If the user area is associated with a user who just entered the muj, the area will be zeroed by the USER subroutine.

complt - a word containing the address of a central memory word (anywhere in the muj field length) in which bit 0 will be set by some external program (such as the SCOPE I/O routines) to indicate completion of an operation. For FORTRAN and COBOL programs, this parameter is passed as a reference to a file (file name in COBOL, logical unit number in FORTRAN). This parameter is meaningful only when the actn parameter is ±3. It must be specified in the calling sequence, however, for all file action requests (actn ±11 to ±16).

fetadd - A word containing the first word address of a file environment table (FET) for I/O operations on a user file. This parameter is meaningful and need be specified in the calling sequence only for USER calls in which the actn parameter is ±11 to ±16. The first word of the FET must contain the name of the file in display code characters, left justified and zero filled. For attach and create requests, the FNT address for the file will be stored by USER in bits 59-48 of the fifth word of the FET. For detach and delete requests, USER will expect to find the FNT address in this position. In FORTRAN calls to USER, a file name (fname) and a logical unit number (flun) must be specified to allow the FORTRAN pre-processor to find and prepare the FET for the file. In COBOL, a file name (fname) is specified in the USER call. Refer to the discussion of attaching user files for a list of rules regarding the FET at fetadd.

PRE-PROCESSORS

FORTRAN USERFO: Two additional parameters, flun and fname, must be specified instead of fetadd. The fname parameter is a file name in display code, left justified with zero fill. flun is an integer variable representing the logical unit number used to reference the file fname in I/O requests from the muj. If no file action is requested in the call to USER, flun and fname need not be specified in the calling sequence. Another difference in the FORTRAN call is that the location of the complt word is specified by the parameter lwait, an integer representing the logical unit number of the file for which a wait on I/O complete is to take place. The FORTRAN pre-processor uses this logical unit number to locate the FET and the complete bit in the code-and-status field of the FET for this file.

COBOL USERCO: In the COBOL call, the location of the complt word is specified as a file name, fwait. The pre-processor uses this file name to find the FET and the complete bit in the code-and-status field of the FET. The fname parameter contains the file name; it is specified only for file action requests. The pre-processor also converts actn, and iarea from computational-1 items to binary integer numbers; an associated post-processor reconverts these parameters on exit from USER.

SUGGESTIONS FOR USE

If employed, the user area buffers may be the biggest obstacles in user scheduling. Even if a user is ready for processing, a user area buffer must be available before the user can be passed to the muj for processing. Consequently, it is recommended that an adequate number of user area buffers be allocated, and that information for a specific user not be retained in central memory in the user area buffers during a user switch unless absolutely necessary. The use of FETs or I/O buffers in the user areas should be avoided if possible.

The muj should always check the state of a new user returned to it for processing to determine if a break has been entered or a temporary disconnect has occurred. The muj must determine the meaning of the break and the processing required for it. Entry of the break at a terminal will terminate output currently being sent to the terminal, and it will unset certain wait conditions in effect for the user; for instance, if the user was waiting for input when the break was entered, the muj must again request that the user be switched out, waiting for input, even though bits 0-17 of state indicate input is ready.

When the USER subroutine returns newu for processing, it performs only a normal subroutine return. In many cases, the muj will select a branch to various parts of the program depending on the user and his state. An easy way to accomplish this (if user areas are employed) is to maintain a re-entry address in the user area. If a re-entry address is kept, care must be taken that associated muj routines are re-entrant. This requirement may mean that other information, including return addresses for nested subroutines, must be saved in the user area and that the passing of parameters in re-entrant subroutines must be avoided.

TIO

Performs terminal input/output for a specific user. The file name used for terminal I/O is TERMIOF. The types of I/O requests are described under the iocode parameter. If an I/O operation is currently in process for the specified user, this I/O request will be rejected. If the muj is operating with the other system muj subroutines, all terminal I/O must be requested through the TIO subroutine.

CALLING SEQUENCES

FORTRAN:

```
        CALL            TIO  (user, iocode, buffer, leng, err)
```

COMPASS:

```
        SA1       TIPAR
        RJ        TIO
        .
        .
        .
TIPAR   VFD       60/user,60/iocode,60/buffer,60/leng,60/err
        DATA            0
```

COBOL:

```
        ENTER     TIOCO user, iocode, buffer, leng, err
```

PARAMETERS

user - The user ID of the user requiring the I/O operation. This parameter is a binary number in COMPASS, an integer variable in FORTRAN, and a computational, synchronized left variable in COBOL. user is an input parameter usually obtained from the user ID returned on a user switch. It contains two left-justified alphanumeric display code characters with trailing binary zero.

iocode - A number representing the type of I/O requested. It is a binary number in COMPASS, an integer variable in FORTRAN, and a computational-1 item in COBOL. It is an input parameter. The values are as follows:

iocode=1 Read request; must be made only after the USER routine indicates that input from the terminal is ready for a user. Control will not be returned to the muj until the data requested by the READ has been transferred.

iocode=2 Write request to send output to a terminal. Control will be returned immediately after the WRITE is issued. The FET or buffer may not yet be free at this time.

iocode=3 Read only one line from the terminal input. Differs from iocode=1 in the one line limitation. Any other input for this user will remain in the INTERCOM buffers and may be read later.

iocode=4 Perform a Readskp from the terminal. If the first line will not fit in the circular buffer, data will be read until the buffer is full; the remainder will be discarded. No error code will be returned in the FET. Otherwise this code acts as normal read (iocode=1)

iocode=5 Perform a Readskp from the terminal but read only one line. If the line will not fit in the circular buffer, data will be read until the buffer is full; the remainder will be discarded. No error code will be returned in the FET. Otherwise this code acts as a normal read of one line only (iocode=3).

buffer - An array or table which holds the data to be sent to the terminal for a write request or receives data from the terminal for a read request. This array must be in the muj field length. For write requests, the buffer must contain all information to be sent to the terminal in the correct format with carriage control characters, SCOPE end-of-line indicators, and in display code.

The INTERCOM Reference Manual lists carriage control characters. The end-of-line indicator must contain binary zero as the 12 bits in positions 0-11.

leng - For write requests, the number of words of information to be sent to the terminal; for read requests, the number of words in the buffer array or table. It is a binary number in COMPASS, an integer variable in FORTRAN, and a computational-1 item in COBOL. It is an input parameter for all requests, and an output parameter (the number of words read) for read operations. If the buffer cannot accommodate all data to be read, the data will be truncated. On return from a read request, leng indicates the actual number of words read. The buffer length is measured in central memory words. If the number of characters of data is one less than an even multiple of ten or an exact multiple of ten, leng includes the word needed by SCOPE to contain the end-of-line byte.

err - Output parameter which indicates errors during I/O processing. This parameter is a binary number in COMPASS, an integer variable in FORTRAN, and a computational-1 item in COBOL. err may be set as follows:

err=0	No errors occurred.
err=1	An I/O operation is in process for the specified user; the requested operation cannot be performed. This condition should not arise if user switching is done properly.
err=2	An error was made in the TIO call (usually an illegal value for iocode).
err=3	No terminal FET was available for the I/O request; it must be re-issued later.
err=4	A read operation was requested, but the user was not switched in with input ready.
err=5	The user ID is not that of a user attached to the muj.
err=6	The buffer is shorter than the input which was truncated accordingly.
err=7	Information in buffer is improperly formatted; no end-of-line zero byte in last word.

PRE-PROCESSORS

FORTRAN none

COBOL TIOCO: This pre-processor converts computational-1 items to binary integer numbers. An associated post-processor converts err from an integer variable to a computational-1 item.

SUGGESTIONS FOR USE

The muj must check that the data to be sent to the terminal is formatted properly in display code with a carriage control character and an end-of-line indicator (a 12-bit zero byte in bits 0-11 of a central memory word). FORTRAN programs may use ENCODE to format output message; COBOL may use MOVE or some type of editing operation. However, the programmer must assure presence of the binary zero end-of-line indicator in the last word to be sent to the terminal.

The FET's used for terminal input and output are in the array passed to the multi-user subroutines by the INMUJ call (the mujtbl array). Since they are not in the user area, it need not be retained in central memory during terminal I/O unless the I/O buffer is in the user area.

If an I/O request was rejected because no terminal FET is available, a user switch should be requested, with the current user to be returned under no special conditions (actn=±2). The I/O request may then be re-issued.

The normal sequence of processing is as follows:

A read request is issued through subroutine TIO only after the USER subroutine has been called and has informed the muj that input is ready for a specific user at his terminal; calling USER with actn=±1 guarantees this. A write request, issued by calling subroutine TIO with iocode=2, initiates data transfer from the specified buffer to INTERCOM's output buffers in central memory resident; from there, it is sent to the user's terminal. The muj considers output complete when all data is removed from the specified buffer in the muj's field length and the buffer is free for use in another operation.

After each write request, the muj should call subroutine USER with actn=±7 to wait for the output buffer to be freed; otherwise the muj must be prepared to receive an error code of 1 on the next call to TIO for this user and issue a call to USER with actn=±7 at that time.

If a muj generates output for a specific user faster than his terminal can display it, the output accumulates in INTERCOM's output buffers as backlog. When the backlog becomes excessive, INTERCOM eventually refuses to accept additional output for that user until the backlog is diminished. This situation is handled for the muj by calling subroutine USER with actn=±7. The subroutine waits until the backlog is cleared and ensures that any refused output is sent at that time; however, the buffer holding the refused output is not available until it has been cleared. To prevent the user's next request from being rejected and minimize the time required to clear the buffer, the muj can do the following:

Call user with actn=±7 after each write request.

Examine bit 57 when the user is returned to the muj with state=7; this state indicates that the muj buffer used for his output is free. If bit 57 is set, the user's output is backlogged and his next write request probably will be refused. Call USER with actn=±9 when bit 57 is set, to wait until the backlogged output has reached the user's terminal; this call guarantees acceptance of the next write request and ensures earlier availability of the buffer holding the output data.

MUJSTAT

Provides status information to the calling program. All parameters are optional except the first (usercnt).

CALLING SEQUENCES

FORTRAN:

```
CALL MUJSTAT(usercnt, rdycnt, uawtcnt, userx, statusx, idarray)
```

COMPASS:

```
            SA1   MUPAR
            RJ    MUJSTAT
            .
            .
            .
MUPAR       VFD   60/usercnt,60/rdycnt,60/uawtcnt,60/userx,60/statusx,60/idarray
            DATA  0
```

COBOL:

A calling sequence for COBOL is not supplied. A COBOL muj may call MUJSTAT from a COMPASS subprogram using the COMPASS calling sequence.

PARAMETERS

usercnt - A count of the number of users currently attached to the muj and those waiting to become attached. It is a binary number in COMPASS, an integer variable in FORTRAN.

rdycnt - A count of the number of users currently ready for processing by the muj. This count includes the user currently being processed by the muj. It is a binary number in COMPASS, an integer variable in FORTRAN.

uawtcnt - A count of the number of users, otherwise ready for processing, who are waiting for buffers for their user areas. It is binary number in COMPASS, an integer variable in FORTRAN.

userx - The user ID of a specific user. If this parameter is specified, MUJSTAT returns the status of this user in statusx (see below). It is a binary number in COMPASS, an integer variable in FORTRAN. userx is an input parameter which contains two left-justified alphanumeric display code characters with trailing binary zero.

statusx - A number representing the state of the user userx. This parameter must be specified if userx is specified. It has the same format as the state parameter for subroutine USER.

idarray - An array allocated by the calling program. It must contain at least nuser+1 words, where nuser is the parameter passed to subroutine INMUJ. MUJSTAT places one word in idarray for each user currently attached to the muj. This word is a number of the same format as statusx (see above) and state (see subroutine USER), with the addition that bits 36-47 of the word contain the two-character ID of the user. Users waiting to become attached are not included in this array; consequently, there may be fewer than usercnt entries. To solve this dilemma, a word of all zero follows the last entry.

PRE-PROCESSORS

FORTRAN: none

COBOL: not applicable

SUGGESTIONS FOR USE

MUJSTAT is intended to provide a muj with information to make the following decisions:

1. Should a user switch be performed?

   If usercnt=1, there is no need to switch users.

2. When switching users, should the old user's user area by rolled out to disk?

   If there are fewer users than user area buffers (usercnt is less than the parameter narea passed to INMUJ) there is no need to roll out the user area.

3. Is this a good time to do background work unrelated to a specific user:

   If rdycnt=0, perhaps yes.

The information returned by MUJSTAT may also be used to give the muj user an idea of who his competition is, to give the muj coder an idea how well his muj is performing, or to do anything else a creative mind cares to do with it.

INPUT/OUTPUT REQUEST TO TERMINAL

Since many users may be associated with the muj at one time, terminal I/O must specify the user involved. The user must be specified in the sixth word of the FET for the file connected to terminal keyboards. The TIO subroutine will perform this action, set up the FET and issue an I/O request for a terminal. If the muj subroutines are used, rather than the direct table interface with the INTERCOM system, TIO must be called for all terminal I/O. In addition to setting up the special FET's, the TIO subroutine checks its FET's to insure that no other I/O request is active for the requesting user. TIO also handles other special processing required of a muj for terminal I/O.

CHANGING SIZE OF TERMIN AND TERMOUT

The muj subroutines use the TERMOUT table to post requests to the INTERCOM system. Replies are placed in the TERMIN table. Since the volume of requests depends on the nature of a muj as well as the number of users, it is difficult to state exactly the appropriate size of TERMIN and TERMOUT. As a guide, the INTERCOM EDITOR, which is a MUJ, uses 5 TERMIN words and 10 TERMOUT words when configured to process up to 30 simultaneous users.

If a muj is configured to handle substantially more than 30 users, performance may be improved by increasing the size of TERMIN and TERMOUT. They can be increased by changing the DIMENSION statement of MUJCOM.6 and the EQU statements of CMUJCOM.4 and CMUJCOM.5. These changes should immediately precede the COMPILE MUJSUBS card shown in the muj installation decks.

DEBUGGING A MUJ

The reentrant nature of a muj renders debugging difficult. Because a muj services many users simultaneously, an abort affects all users; therefore, it is especially important to make sure a muj is essentially error-free before it is put into the INTERCOM system.

If however, some internal check in the code of a muj determines that a system error has occurred, the muj must choose a course of action:

1. Ignore the situation
2. Issue messages and/or dumps indicating the error has occurred, and then:
   a. Abort the command of the user currently being processed.
   b. Abort the entire muj.

The system muj subroutines have a number of internal checks for system errors which select either option 2a or 2b above, depending upon the nature of the error. They use the MSYSERR routine to handle messages and dumps. This routine is also available to the muj proper.

MSYSERR

Generates system error messages and dumps for the calling program. The first two parameters are required; the third is optional.

CALLING SEQUENCE

FORTRAN:

```
CALL MSYSERR (number, iflags, auxsub)
```

COMPASS:

```
        SA1   MSPAR
        RJ    MSYSERR
        .
        .
        .
MSPAR   VFD   60/number, 60/iflags, 60/auxsub
        DATA  0
```

COBOL:

A calling sequence for COBOL is not supplied. A COBOL muj may call MSYSERR from a COMPASS subprogram using the COMPASS calling sequence.

PARAMETERS

number — System error number to appear in messages and dumps:

- 0-49 — Reserved for the system muj subroutines (see chart MUJ System Errors)
- 50-99 — Available for the MUJ proper

iflags — Bit-structured parameter telling MSYSERR what diagnostic information to generate:

Bits

- 0-2 — 0 = take no dumps
1 = dump field length
2 = dump FL and 70K of low core
3 = dump FL, low core, and 1K of high core
- 3-5 — Reserved (should be 0)
- 6 — 0 = return after generating diagnostic information
1 = abort the muj (all dumps taken)
- 7 — 0 = send no message to users
1 = send system error message to all muj users
- 8 — 0 = issue no dayfile message
1 = issue system error message to dayfile
- 9 — Reserved for use by subroutine REKOVER (should be 0)
- 10-59 — Unused

auxsub — Name of a subroutine which MSYSERR should call immediately before disposing the file OUTPUT to the central site printer. This parameter is optional.

MUJ SYSTEM ERRORS

Number	Issued By	Error
1	USER	User area lost internally
2	SERVICE	User area lost internally
3	SERVICE	Bit KWCOM should not be set for this value of MMACT (FATAL)
4	SWAPOK	Error (from CIO) on last user area swap
5	SWAPOK	Illegal CIO function code on last user area swap (FATAL)
6	SWAPOK	User area lost on swap-out (FATAL)
7	not used	
8	not used	
9	USER	Muj returning user area not assigned to it. (User error) (FATAL)
10	USER	Invalid ACTN code sent by muj (User error) (FATAL)
11	USER	Invalid info from 1QP
12	USER	Internal logic error
13	USER	A non-ready user was marked as ready
14	LUNSRCH	Logical unit number was specified in call to USERFO, but corresponding file was not declared on muj PROGRAM card (User error) (FATAL)
15	not used	
16	USER	Muj is returning user not assigned to it. (User error) (FATAL)
17	USER	User's files cannot be returned when user leaves muj

*User errors are noted, all others are system errors

TRACE PRINTOUT

In addition to the abort dump, the muj subroutines may be compiled optionally to generate a trace printout which helps in diagnosing muj aborts. This trace printout is generated and included in the abort dump if a *YANK MDEBUG card is included in the UPDATE job which takes the muj subroutines off the INTERCOM program library.

The trace printout is generated by FORTRAN PRINT statements. Consequently, the muj programmer must provide, at address x, a word containing the left justified display code characters OUTPUT in bits 59-18 and the address of an FET for the file named OUTPUT in bits 17-0. Address x must satisfy the condition RA+2≤x≤RA+53(octal). None of the words preceding address x may be binary zero. The SCOPE Reference Manual contains information concerning the File Environment Table.

If the muj main program is coded in FORTRAN, the above process is accomplished by including the file name OUTPUT on the PROGRAM card. If the muj main program is coded in a language other than FORTRAN, the process can be accomplished by a COMPASS subprogram.

In order to handle an abort, INMUJ calls the SCOPE RECOVR routine. If the muj that is being written also calls RECOVR, this call should be included after the muj calls INMUJ. The last call to RECOVR will be executed first to effectively stack reprieve requests.

COMMENT SHEET

CONTROL DATA CORPORATION

TITLE: INTERCOM Multi-User Job Capability PSB, 6000 Version 3

PUBLICATION NO. 60327400 REVISION D

This form is not intended to be used as an order blank. Control Data Corporation solicits your comments about this manual with a view to improving its usefulness in later editions.

Applications for which you use this manual.

Do you find it adequate for your purpose?

What improvements to this manual do you recommend to better serve your purpose?

Note specific errors discovered (please include page number reference).

CUT ON THIS LINE

General comments:

FROM NAME: _____ POSITION: _____

COMPANY NAME: _____

ADDRESS: _____

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Documentation Department
215 Moffett Park Drive
Sunnyvale, California 94086

CUT ON THIS LINE

FOLD

FOLD

STAPLE

STAPLE

► ►CUT OUT FOR USE AS LOOSE-LEAF BINDER TITLE TAB

CONTROL DATA
CORPORATION

CORPORATE HEADQUARTERS, 8100 34th AVE. SO., MINNEAPOLIS, MINN, 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

Pub. No. 60327400

Litho in U.S.A.