Technical Reference Manual

MULTSESS/HPO

Version 2.0

MVS Operating Environment

Publication number PMS0002.012

PassGo

Published by:

PassGo Technologies Ltd, Horton Manor, Ilminster, Somerset, TA19 9PY, England.

Publication number PMS0002.012

Twelth Edition (September 2001)

Changes will be made periodically to the information contained in this book. If your book does not accurately reflect the level of product you are using, this may be due to a fix being applied to the product which has still to be released as a book update.

Preface

Prerequisites for running MULTSESS/HPO

MULTSESS/HPO has been designed to run under the latest releases of MVS, MVS/XA and MVS/ESA running ACF/VTAM V3.3 and earlier, in both SNA and non-SNA environments.

Devices supported by MULTSESS/HPO

- All models of 317x, 318x, 319x and 327x (or their equivalents), utilising the maximum available screen display area
- the 3290 Information Panel with each interactive panel in 3270 mode
- LUTYPE 1 printers (SNA printers with the SCS feature)
- devices attached through NTO
- IBM 3767 and compatible devices

Application sessions using MULTSESS/HPO are supported in the terminal's native mode.

Terminal to MULTSESS/HPO sessions are supported in model 2 mode.

About the Technical Reference Manual

This Technical Reference Manual is for use by technical personnel who are going to install and customize MULTSESS/HPO.

The contents of this manual should be read before installation.

Throughout this manual the word INSTGDE denotes the MULTSESS/HPO Installation Guide.

Related publications

The MULTSESS/HPO User Installation Manual describes how to install MULTSESS/HPO in your environment.

The MULTSESS/HPO User Reference Manual contains details of commands that the user may enter at the terminal and error messages that may be issued to the terminal.

The MULTSESS/HPO User Reference Manual should be used in conjunction with this manual when performing certain installation tasks, such as defining command sequences in the user profile dataset and may be used as a standalone reference work by terminal users.

The MULTSESS/HPO Customization Reference Manual describes ways of customizing MULTSESS/HPO to suit your own requirements and your system environment.

Future Publication changes

Changes will be made periodically to MULTSESS/HPO publications to reflect new releases and facilities. When this occurs you will be supplied with update pages or a new updated manual.

Readers' comment forms

A form for readers' comments is provided at the back of this manual.

Any information supplied may be used or distributed by the authors in any manner considered appropriate without incurring any obligation whatsoever.

Table of contents

Chapter 1 - Modes of working with MULTSESS/HPO

Three major modes of working

To allow for the requirements of different users, MULTSESS/HPO provides three different major modes of working.

DYNAMIC PANEL MODE
A dynamically built menu of applications, allowing selection of application sessions by cursor position or PF key or a single digit id. A command input area is included to permit the issue of any MULTSESS command for which the user is authorized. Dynamic panel mode provides a friendly and flexible interface suitable for use by most users.

USER DEFINED PANEL MODE
An installation may customize panels to meet the needs of particular users or user groups. An unlimited number of different panels may be defined. Panels may be defined with or without a command input area. If no command input area is included, the user is restricted to the functions defined on the panel.

BASIC MODE
A command driven mode primarily for use by technical, computer-literate personnel who will be monitoring and controlling MULTSESS.

These three different modes of working with MULTSESS/HPO are described and illustrated on the following pages.

Dynamic panel mode

```
TERMINAL:-A01MS257    LOGMODE:-S2QQ88F8    TIME:-11:49:40      USERID:-CAK0004
.....................................................................
***                                                                 ***
***            HELP DESK TELEPHONE NUMBER IS NOW XTN 3456            ***
***                                                                 ***
      ..............................................................
      .  C  ID PF DESCRIPTION OF SESSION                  STATUS   .
      .                                                            .
      .  -   1  1 TSO    SYSTEM A                         AVAIL    .
      .  -   2  2 TSO    SYSTEM B                         ACTIVE   .
      .  -   3  3 CICS   ORDER ENTRY SYSTEM               ACTIVE   .
      .  -   4  4 CICS   MANUFACTURING REQUIREMENTS       AVAIL    .
      .  -   5  5 IMS    FINANCIAL CONTROL SYSTEMS        AVAIL    .
      .  -   6  6 HCF    DISTRIBUTED 8100 SYSTEMS         AVAIL    .
      .  -   7  7 NCCF   NETWORK CONTROL                  AVAIL    .
      .  -   8  8 DISSOS ELECTRONIC MAILBOX               AVAIL    .
      .  -   9  9 NCI    ELECTRONIC BULLETIN BOARD        ACTIVE   .
      .  -  10 10 CICST  TEST DATABASE SYSTEM             AVAIL    .
      ..............................................................


SCROLL PFKEYS- UP= 19 DOWN= 20       SESSCHAR= PFK24 NOKEEP AUTOFF MSKEY= #
.....................................................................
COMMAND=>
```

Dynamic panel mode is a general purpose menu-driven mode suitable for the majority of users. Its screen layout is illustrated above.

The dynamic panel consists of:

- a line showing the terminal name, the logmode in use, the time of day and userid
- a three line hot news broadcast area
- a scrollable application selection menu
- an indication of the MULTSESS control keys currently in use that may be amended by the user if required
- a command input line where the user may enter any commands for which he is authorized
- an error message line

The scrollable application menu area
The scrollable application menu area, within the dotted box on lines 6 to 19, lists the applications which the user is authorized to access, up to a maximum of 99 applications.

Each application area has:

- a single digit input area allowing the entry of abbreviated commands to activate and select a session, terminate a session and print a screen image from a session
- a numeric identifier which may be used to select the application
- the number of the PF key used to select the application. PF keys may be automatically defined for the user when the panel is built. The assigned PF key values may optionally be amended by the user.

- A description of the application as defined by the customer in the Application Definition Table (ADT)
- An application status field. If a session exists with the application, this will be the status of the sessions. If no session exists, this will be the VTAM status of the application. The following values may be reported:

  UNAVAIL - the application is not available for use, because it is inactive, has been disabled by a STOP command or has been removed from the ADT since the panel was built.

  AVAIL - the application is available for use.

  ACTIVE - the session exists with the application. Lines describing active sessions are highlighted.

  UPDATED - the session exists and data has arrived from the application since the user last accessed the session. Lines describing updated sessions are highlighted.

  ENDED - the session with the application has terminated. ENDED also implies AVAIL.

  SCRIPT - the ATP script is active on the session.

Invoking dynamic panel mode

Dynamic panel mode is invoked using the PANEL ON command. The command may be typed at the terminal, but would more usually be included in a profile member to place the user in dynamic panel mode at logon time.

When specifying a PANEL ON command, the following options are available:

PFK/NOPFK - are PF keys to be automatically assigned for selecting applications from the menu?

EXIT/NOEXIT - once in panel mode, is the user to be allowed to issue a PANEL OFF command to return to basic mode? When a user is placed directly into dynamic panel mode at logon time, specifying NOEXIT ensures that the user never accidentally goes into basic mode.

MSG/NOMSG - is the user to receive error messages from MULTSESS/HPO? Most users should receive error messages, but you may wish to use NOMSG for a number of inexperienced end-users.

UPDATE/ NOUPDATE - is the user to be allowed to amend the PF keys assigned to the listed applications?

The PANEL command is further described in detail in the MULTSESS/HPO User Reference Manual.

Scrolling the application menu

Up to 10 applications are displayed at a time. If the user is authorized to access more than 10 applications, the menu area may be scrolled as follows.

If PF keys have been assigned for scrolling, these are indicated after the words UP and DOWN on the line beginning SCROLL PFKEYS. Pressing the indicated PF key scrolls the menu.

By placing the cursor in the field after the word UP or DOWN on the line beginning SCROLL PFKEYS and pressing the <Enter> key.

Including applications on a Dynamic Panel
When a PANEL ON command is executed, the panel is built dynamically to include only those applications that the user is authorized to access, up to a maximum of 99 applications.

The applications are chosen by reference to the MULTSESS Directory of Users.

Each session statement in the user's directory entry is examined in turn and a menu entry generated for each application specified.

If the directory SESSION statement specifies a generic or masked application name, the Application Definition Table (ADT) is referenced and each defined application which matches the directory name is included on the user's menu.

The application description displayed on the menu is taken from the ADT definition of the application.

The list of applications included on the menu may be further refined with TPEXIT08. If enabled, TPEXIT08 is entered with call code 82 as each application, selected as described above, is about to be placed into the user's application menu area.

The return code from TPEXIT08 determines whether the application is to be included on the menu or discarded. Your exit routine could, for instance, reference your security database to decide whether the user is authorized to access the application. Please refer to *TPEXIT08* in the MULTSESS/HPO Customization Reference Manual for further details.

Selecting an application session
Dynamic panel mode supports four different application selection methods.

To select an application:

1. Position the cursor within the application menu area (indicated by the dotted box) anywhere on the line containing the application and press <Enter>. If a session with the application is already active, the user will be switched into the session. If a session does not currently exist, a new session will be started and the user switched into that session.

2. Type one of the following command codes against the application in the column headed **C**:
   - **S** switches into the session. If a session is not already active, a new session is created.
   - **A** creates a new session, but leaves the user in MULTSESS mode. If a session already exists, the command is ignored.
   - **T** terminates the session.
   - **E** terminates the session and any associated SIMLOGON session.
   - **P** prints the last screen image received on an active session.

   Multiple command codes may be entered in a single iteration. If multiple **S** commands are entered, sessions will be created as required, and the user will be switched into the last selected session. That is, all but the last **S** command are treated as **A** commands.

3. Type the numeric application identifier shown in the **ID** column into the command input area and press <Enter>.

4. If a PF key number is indicated against the application, press the appropriate PF key. The use of PF keys is optional and is described in the next section.

Assigning keys for application selection
PF keys may be optionally assigned to start and select application sessions:

Specify the PFK operand on the panel command which invokes dynamic panel mode. <PFK1> to <PFK24> will be assigned to the first 24 applications listed on the application menu. No PF keys are automatically assigned for scrolling the menu.

If the UPDATE operand was specified or defaulted when dynamic panel mode was invoked, the PANEL PFKSET command may be issued after invoking dynamic panel mode. This command allows <PFK1> to <PFK24>, in any order, to be selectively assigned for menu scrolling and application selection. The PANEL PFKSET command is described in the MULTSESS/HPO User Reference Manual under *PANEL ON* on page 2.43.

If the UPDATE operand was specified or defaulted on the PANEL command which invoked dynamic panel mode, the PF keys assigned for application selection and scrolling may be modified by the user by overtyping the values shown on the panel. A PF key may be deassigned by overtyping the indicated value with blanks.

Notes: If any of PF keys 13-24 have not been assigned a value, the value assigned to the equivalent key in the range 1-12 will be taken.

Use of a PFK not defined in dynamic panel mode will use the 'out-of-panel' value, assigned by a previous SET PFK command in a profile or typed at the terminal.

Passing CINIT data logon to an application
The MULTSESS SESSION command supports the passing of user specified CINIT logon data as part of a session initiation request.

When working in dynamic panel mode, most users will never type the SESSION command since there are many more user-friendly methods of starting a session.

In dynamic panel mode, CINIT logon data may be passed to an application by typing the required data in the command input area and then selecting the application by any means that allows only one session to be started at a time (cursor position or PF key). MULTSESS variables will be resolved before sending the data to the application.

Dynamic panel mode supports the automatic passing of CINIT data based on the definition of the application in the ADT. The CINIT parameter of the application definition may contain a fixed string of data to be passed on every new session with the application. The use of variables is supported for substituting such items as userid, password and terminal name.

The MULTSESS ADT option AUTO defines how a user may cause automatic CINIT logon data to be passed to an application when working in dynamic panel mode.

If AUTO=YES is specified, the CINIT logon data specified in the ADT will be automatically passed as part of a session initiation request with no special action required of the user. If the user wishes to suppress the passing of CINIT logon data, he may type an asterisk (*) in the command input area before initiating the session by any means that allows only one session to be started at a time (cursor position or PF key).

Specifying AUTO=NO means that automatic CINIT logon data will not be generated as part of a session request unless the user enters an asterisk (*) in the command input area using any means which allow only one session to be started at a time (cursor position or PF key).

Basic mode

```
q sess
MS0261  - INFO FOR USER CAK0004 ON TERM CAKT03 USING LOGMODE S2PN87F8
APPL=TSO     ,ALIAS=TSO1    ,VTERM=N/A     ,UPD=YES,SCRIPT=NONE ,ACB=MULTSESS
APPL=NCI1    ,ALIAS=NCI1    ,VTERM=POOLM1  ,UPD=NO ,SCRIPT=NONE ,ACB=MULT1
echo 10 100 1000
    ------------------------------------------------------------------
   ¦                  ¦   AVERAGE DELAY TIME IN SECONDS FOR  10 SENDS  ¦
   ¦                  ¦ INTERNAL VTAM  ¦      LINE      ¦  TOTAL       ¦
    ------------------------------------------------------------------
   ¦ RU SIZE     100  ¦    0.001       ¦    0.304       ¦    0.305     ¦
    ------------------------------------------------------------------
   ¦ RU SIZE    1000  ¦    0.001       ¦    0.104       ¦    1.105     ¦
    ------------------------------------------------------------------
q users
MS0090  - CURRENT USERS:    TOTAL = 1    ACTIVE = 1
q names
MULTOPER INTERNAL    0, CAK0004  CAKT03      2,


                                                              INPUT
```

Basic mode is a command-driven mode suitable for technical control personnel only. Basic mode is the default mode within MULTSESS. However, most users would be placed directly into either dynamic or user panel mode at logon time by including the appropriate PANEL command in the user profile.

Lines 1 through 22 of the screen comprise the information display area. Messages produced in response to MULTSESS commands are displayed here. The command responses are automatically scrolled upwards, new messages being added at the bottom.

Lines 23 and 24 contain a 120 character command input area for the entry of any MULTSESS command that the user is authorized to use. Multiple commands may be entered at once, up to a limit of 120 characters. For details about entering multiple commands, refer to *Multiple command input* on page 4.8 of the Customization Reference Manual.

A status indicator on line 24, set to either INPUT or MORE..., informs the user whether he can type input or whether MULTSESS has more data to display. MORE... will be displayed if the last command (or multiple command string) has produced more than 22 lines of output. Pressing the <Enter> key causes pending messages to be displayed.

PF keys may be assigned to provide easy access to application sessions or other frequently used functions. The PF key values may be assigned automatically at logon time by including appropriate SET commands in the user's profile member. Assigned values may be extended or amended after logon by entering the SET command(s) at the terminal. The PF key values assigned in basic mode are separate from the values assigned to application sessions in dynamic panel mode. Thus by jumping between basic and dynamic panel mode, a user with 24 PF keys has available 48 distinct PF key settings.

Basic mode PF key settings may be used while in dynamic panel mode by typing the command separation character (CMDCHAR) in the dynamic panel mode input area before pressing the required PF key.

The SESSION command generated when a dynamic panel mode user selects an application uses the characters PFK followed by the panel application number as the session alias. These sessions may be referenced outside dynamic panel mode, for instance to start sessions in the logon profile or reference a session after switching to basic mode, by using the appropriate session alias - SESSION PFK*n*.

User defined panel mode

```
################################################################################
##########                                                            ##########
##########            EXAMPLE OF A USER DEFINED PANEL                 ##########
##########                                                            ##########
################################################################################

                  PF 1    IMS      Data Base System


                  PF 2    TSO      Time Sharing Option


                  PF 3    EMAIL    Electronic Mail


                  PF 4    Disconnect


                  PF 5    Logoff

 In the event of difficulty please telephone TECHNICAL SERVICES on extn 999.

-------------------------------- MESSAGES ---------------------------------
```

An installation can design its own menu panels to provide a tailored user interface where the MULTSESS dynamic panel is considered inappropriate, as shown above.

Panel usage

Common uses of installation designed menu panels are as described below.

- To help inexperienced computer users, panels may be defined that do not contain a command input area. The user is limited to those sessions and MULTSESS functions for which PF keys have been predefined.
- To provide a PF key driven menu for the selection of application sessions and commonly used MULTSESS functions. A command input area may be defined to allow the entry of MULTSESS commands not included on the menu.

The panel's dataset

Installation defined menu panels are held in a partitioned dataset member pointed to by the PANELS DD statement in the JCL used to initiate MULTSESS. All panels are stored as part of a single member. When MULTSESS is started, all panel definitions are loaded into storage in compressed format. Thus no file I/O is involved when a user requests use of a panel. New or amended panels may be loaded into storage while MULTSESS is active by using the PLOAD command (see *PLOAD* on page 2.46 of the MULTSESS/HPO User Reference Manual). A sample panel is supplied in the control library loaded from product distribution tape.

Designing new menu panels

A panel definition consists of the following:

- A prefix record that assigns a name to the panel and contains the keyword PANEL= starting in column 1, followed by a 1-8 character name. This name is used to invoke the panel. **Note** that the keyword PANEL must be in upper-case.
- From 1 to 19, 80 character records that define the data appearing on lines 1-19 of the user terminal, normally consisting of a heading followed by descriptions of the PF key settings. Optionally, a single unprotected field, up to 120 characters long, may be defined to allow the entry of MULTSESS commands.
- If more than 19 lines appear between a PANEL= prefix statement and a subsequent PANEL= statement, the excess lines will be treated as comments.

Five line message area
A five line message area will be appended to each panel definition when the menu is displayed on the terminal.

Panel attributes
The characters defined by the PANEL= startup parameter may be used within the panel definition to assign attributes on the panel. If this parameter is omitted, the following default values apply; but note that defining a menu panel using only attributes which include the 'protected' option effectively restricts the user to those options for which PF keys have been assigned.

@	-	High Intensity, Protected. (White)
+	-	High Intensity, Unprotected. (Red)
*	-	Low Intensity, Protected. (Blue)
$	-	Low Intensity, Unprotected. (Green)
%	-	Non-Display, Unprotected.

Command input area
To include a command input area (maximum 120 characters long), define a single field preceded by an attribute character that includes the unprotected option. It is common practice to also include a (protected) descriptive prefix to inform the user that he may type commands:

```
*ENTER COMMAND ==>+                    *
```

Use of PF keys
Note that, unlike dynamic panel mode, which maintains its own set of PF key definitions, PF keys are not automatically assigned when an installation defined menu panel is invoked.

It is the responsibility of the customer to ensure that PF keys have been set to appropriate values, matching the descriptions on the menu, before displaying the panel. This is usually done by creating a profile dataset member which contains the necessary 'SET PFK' and 'PANEL' commands to correctly call up the panel.

Panel to panel switching
Panel-to-panel switching is supported using the profile name command to execute a profile member that invokes the new panel as described above. Do not forget to unset any PF keys that are not used on the new panel. This is achieved by a SET PFK IMM command with no other operands.

Exit from menus
Since installation defined menu panels are primarily intended for inexperienced end users, the 'PANEL OFF' command is not supported in this mode. This ensures that users with a command input area cannot inadvertently exit from their menu panel to the command-driven basic mode of operation.

Testing menu panels
Since MULTSESS always works with an in-storage copy of panels, the panel's member may be freely edited while MULTSESS is active. New or amended panel definitions may be brought into storage with the PLOAD command.

The fact that PANEL OFF is not honoured in user panel mode can be inconvenient when testing and amending new panels. The DISPLAY PANEL command may be used to display the in-storage copy of a newly developed panel without switching into user panel mode. The panel is displayed on the terminal as it would appear to the end user. Pressing <Enter> will return you to your previous MULTSESS display, allowing repetitive edit and display to finalize the design and saving time during panel development.

Chapter 2 - Switching between applications

Session switching

MULTSESS provides flexibility in the method by which each individual user may switch between application sessions, and between application sessions and MULTSESS.

Session switching is accomplished by use of a session switch character (the user's SESSCHAR) and, optionally, an extra MULTSESS key (the user's MSKEY), which are described in more detail on the following pages. The primary use of these keys is to return to MULTSESS from a session to allow the next session to be selected.

Defining a user to MULTSESS

When defining a user to MULTSESS, you need to decide how the user is going to switch between applications. The most common methods of setting up session switching within MULTSESS are listed below and described in more detail in this section:

- using a keyboard character as the SESSCHAR to return to MULTSESS ready to select a new session
- using a PF/PA key as the SESSCHAR to return to MULTSESS ready to select the next session
- using the SESSCHAR (keyboard character or PF/PA key) to cycle round all active sessions and return to MULTSESS at the end of the session list
- using the SESSCHAR (keyboard character or PF/PA key) to continuously cycle round all active sessions. The MSKEY is used to return to MULTSESS only when specifically required.

The session switch character

The prime consideration when switching between MULTSESS and application sessions is the session switch character, commonly referred to as the SESSCHAR.

SESSCHAR may be set to any keyboard character, any PF key or a PA key.

The global default SESSCHAR is # (X'7B') unless overridden by the SESSION-CHARACTER startup option. Each user may be given a different SESSCHAR by including a SET SESSCHAR command in the user's profile or if he/she issues the command at the terminal. In addition users in dynamic panel mode may modify their SESSCHAR by overtyping the appropriate field on the panel.

The action taken when a user enters the SESSCHAR depends upon the other options specified when the SESSCHAR is set. The more commonly used options are illustrated in the following examples. The sample commands given in the examples may be typed by the user at the terminal, but would more usually be included in the user's profile which is executed automatically at logon to MULTSESS.

Using a keyboard character

Any keyboard character may be specified as a session switch character (SESSCHAR). When this character is typed as the first character in the first unprotected field of an application screen, the user is returned to MULTSESS.

Invoked by
SET SESSCHAR *char*

where:

char may be any keyboard character.

Action
User is returned to MULTSESS.

Advantages
Does not rob the application of a PF/PA key.
User may type the SESSCHAR followed by a MULTSESS command into an application screen. The MULTSESS command will be immediately actioned. If this is a SESSION command, the user will be switched directly to the requested session.

Disadvantages
Not suitable for applications that send screen images with no unprotected fields.

Special use
If the SESSCHAR is followed by a MULTSESS command or PF key, the command or PF key is executed immediately by MULTSESS. If this is a request for another session, the user is switched to the new session. For example, if the SESSCHAR is # and TSO is application number 4 on the dynamic panel, typing #4 switches the user straight into session with TSO.

Note: An optional zap is available to enable MULTSESS to find a SESSCHAR in fields other than the first input field. This is useful for applications that send protected fields with the MDT bit set before the first input field. This zap introduces extra overhead and should only be applied where there is a definite need and all other switching methods, as described in the examples on the following pages, are unsuitable.

Using a PF/PA key

Any PF key or PA key may be designated as a session switch character (SESSCHAR). When a user hits this key while in an application session the user will be returned to MULTSESS.

Invoked by
SET SESSCHAR PFK*n* or PAK*n*

Action
The user is returned to MULTSESS.

Advantages
Suitable for working with applications which send fully protected screen images.
Single key switching.

Disadvantages
If the user wishes to send a designated switch PF/PA key to the application, two key presses are required. If the same key is depressed immediately on return to MULTSESS, the AID key value is forwarded to the application and the user is returned to the application session.

Decision
If a PF/PA key is not used by all the applications to be accessed, this is the recommended method of switching.

If all PF/PA keys are used within applications, the difference between how often a PF key will be used in the application and the number of session switches to be made will decide the SESSCHAR. For example, in ISPF <PA2> means redisplay the screen; this function is very rarely used by most users, so <PA2> may be a reasonable choice as a SESSCHAR.

The MULTSESS key

In addition to a SESSCHAR, a user may optionally specify a MULTSESS key, referred to as the MSKEY. This provides a means for users whose SESSCHAR has been set to invoke round-robin session switching to return to MULTSESS. For users not in round-robin mode the MSKEY, if set, acts as an additional SESSCHAR (refer to *Round-robin mode* below).

An MSKEY is defined with the SET MSKEY command in the user's profile, by overtyping the MSKEY field on a MULTSESS dynamic panel or by typing the command at the terminal.

Round-robin mode

Whether the session switch character is a PF/PA key or a keyboard character, round-robin session switching may be used to cycle around all active sessions before returning to MULTSESS. Round-robin mode is set by the SET SESSCHAR AUTON command which must be entered in addition to the SET SESSCHAR command that sets the key.

Invoked by
SET SESSCHAR *value*
SET SESSCHAR AUTON

In this example, the MSKEY is not set. Refer to the following example for a description of the effect of also specifying a MSKEY.

Action
When the user enters a SESSCHAR, he is switched to his next active application session. The order in which the applications are accessed is the order in which the sessions were started.

After one complete tour of the active sessions, the user is returned to MULTSESS.

Decision
This use of round-robin switching will probably be limited to technical personnel who access a number of monitoring tools and who want to return to MULTSESS to action data gathered from the sessions. Most users will use round-robin with a MSKEY set, as described in the next example.

Round-robin with an MSKEY set

This method of switching involves the use of round-robin switching with a MSKEY also set. The SESSCHAR effectively becomes a jump key for switching between application sessions. The MSKEY supplies a means of returning to MULTSESS.

Invoked by
SET SESSCHAR *value*
SET SESSCHAR AUTON
SET MSKEY *value*

Action
When the user enters his SESSCHAR, which may be a keyboard character or a PF/PA key, he is switched to his next active application session. The order in which applications are accessed is the order in which the sessions were started.

Since the MSKEY is set to provide a means by which the user returns to MULTSESS on request, round-robin switching cycles continuously around all active sessions. For users with only two active sessions this technique provides a flip-flop or jump switch.

Advantages
Suitable for most true end-users whose use of MULTSESS is mainly confined to session switching and who require only occasional access to the many other functions of MULTSESS, e.g. sending a screen image to the help desk or printing an application screen image. By setting the MSKEY to a valid 3270 character which does not appear on the keyboard in use (e.g. <PAK3>) and setting up all the user's sessions in the profile at logon time, MULTSESS may be made 'invisible' to the user.

Note: When the user's last session is terminated and the user is in round-robin mode with an MSKEY set, the user is automatically logged off from MULTSESS.

Using the ATTN key

The ATTN key may be used as a hot interrupt key for SNA terminals. Pressing the ATTN key has the same effect as SESSCHAR, unless of course the ATTN key has been allocated to the MSKEY. If the user is not in round-robin mode, pressing ATTN returns the user to MULTSESS. If round-robin is in effect, pressing ATTN switches the user to the next active session. If the MSKEY is set to ATTN, the user will be returned to MULTSESS mode.

The ATTN key will be effective even if the keyboard is locked. For example, it is possible to start a long running CLIST in TSO, switch to another session using ATTN while the CLIST is processing and return to view the results of the CLIST at a later time.

On terminals which support ATTN, this method of switching is available in addition to the normal SESSCHAR switch. The user may of course choose to assign the ATTN key as the SESSCHAR or MSKEY.

Users not in round-robin mode may send ATTN to the active application by pressing the ATTN key twice, passing the attention interrupt to the application and leaving the user in session. Alternatively the user may return to MULTSESS mode and enter an ATTN command to send an attention interrupt to any of his application sessions.

TPEXIT01 (call code 13) will be driven when a user presses ATTN. This exit may stack the command sequence ATTN;SESS to pass the attention interrupt to the application and leave the user in session transparently without disturbing the terminal display.

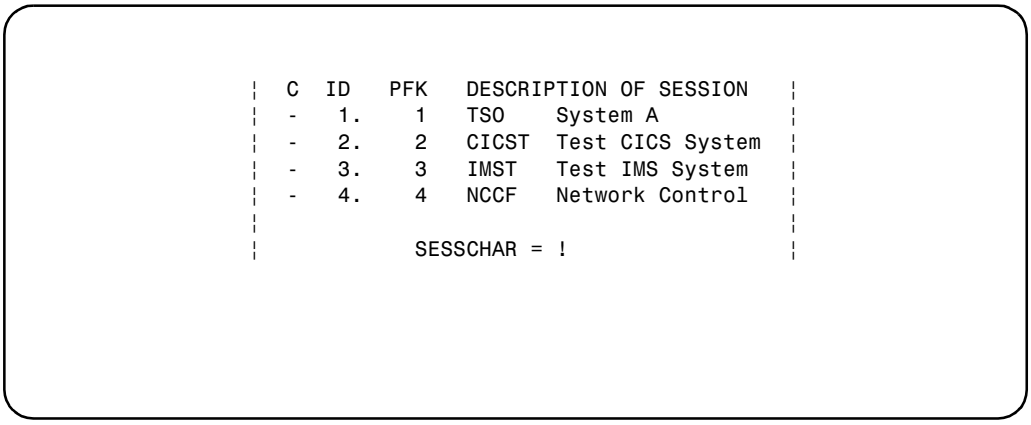
Pressing ATTN (or a PA key) causes the terminal to do a short send to the controlling application, in this case MULTSESS. This means that any data typed but not entered is not sent to the CPU. However, because MULTSESS uses the READBUF command to take a snapshot of the screen at session switch time, data typed but not entered will be captured and will still be available when the user returns to the session.

Jumping directly between applications

If the SESSCHAR is a character, any data entered into the input field after the switch character is passed to MULTSESS for processing, is treated as if it had been entered into the MULTSESS command input area. The aid value (<Enter>, <PFK*n*>, etc.) used to enter the switch character is also captured by MULTSESS and is processed according to the current MULTSESS PF key settings.

This enables the user to jump directly between application sessions without displaying an intervening MULTSESS panel.

The following extract from a dynamic panel application menu illustrates the various options available:



Assuming the user is in the TSO session, the following sequences could be used to jump directly to any of the other sessions:

Type: **!** and press <PFK2>

switches direct to CICST, since <PFK2> is assigned to CICST on the MULTSESS menu.

Type: **!3** and press <Enter>

switches direct to IMST, since IMST is ID number **3** on the MULTSESS menu.

Type: **!SESS NCCF** and press <Enter>

switches direct to NCCF, since ! is followed by a valid MULTSESS command.

The KEEP and NOKEEP options

The two commands SET SESSCHAR KEEP and SET SESSCHAR NOKEEP provide a facility for either keeping the SESSCHAR / MSKEY on a screen image, or removing it from a screen image, when the screen is reconnected.

Summary of SESSCHAR, MSKEY and ATTN processing

The previous pages have discussed the more commonly used combinations of SESSCHAR options and MSKEY usage. The table below summarizes all the available options.

SETTINGS			ACTIONS								
SESSCHAR	SET SESSCHAR AUTON or AUTOFF	MSKEY	SESSCHAR entered	SESSCHAR and AID key pressed	SESSCHAR plus data entered	MSKEY entered	MSKEY and AID key pressed	MSKEY plus data entered	ATTN pressed	SESSCHAR plus data with NOKEEP	MSKEY plus data with NOKEEP
CHAR	AUTOFF	NOT SET	R	P	P	-	-	-	R	D	-
CHAR	AUTON	NOT SET	NR	I	I	-	-	-	NR	D	-
PFK	AUTOFF	NOT SET	R	-	I	-	-	-	R	I	-
PFK	AUTON	NOT SET	NR	-	I	-	-	-	NR	I	-
CHAR	AUTOFF	CHAR	R	I	I	R	P	P	R	D	D
CHAR	AUTON	CHAR	NF	I	I	R	P	P	NF	D	D
PFK	AUTOFF	CHAR	R	-	I	R	P	P	R	I	D
PFK	AUTON	CHAR	NF	-	I	R	P	P	NF	I	D
CHAR	AUTOFF	PFK	R	I	I	R	-	I	R	D	I
CHAR	AUTON	PFK	NF	I	I	R	-	I	NF	D	I
PFK	AUTOFF	PFK	R	-	I	R	-	I	R	I	I
PFK	AUTON	PFK	NF	-	I	R	-	I	NF	I	I
CHAR	AUTOFF	ATTN	R	I	I	R	-	-	R	D	-
CHAR	AUTON	ATTN	NF	I	I	R	-	-	R	D	-
PFK	AUTOFF	ATTN	R	-	I	R	-	-	R	I	-
PFK	AUTON	ATTN	NF	-	I	R	-	-	R	I	-

CHAR = keyboard character
PFK = PF/PA key
R = return to MULTSESS
NR = switch to next active session, return to MULTSESS at end of round-robin tour
NF = switch to next active session, switch to first active session at end of round-robin tour (continuous jump key processing)
P = processed
I = ignored
D = deleted from screen image
- = not applicable

Chapter 3 - Profiles

What are Profiles?

Profiles are automatic commands. They provide a means of quickly and easily executing a predefined set of MULTSESS commands.

Profiles exist as members of a partitioned dataset pointed to by the PROFILE DD statement in the JCL used to start MULTSESS.

Executing profiles

A profile may be executed:

- automatically when the user first logs on to MULTSESS:
  - achieved by specifying the required profile dataset member name on a PROFILE statement in the user's MULTSESS directory entry
  - using TPEXIT07 (call code 71) to return a profile name in the EXITPROF field of the standard parameter list
- on demand by use of the PROFILE command, either by typing the command or using a PF key which has been set to execute the required PROFILE command.

Common uses of profiles

Common uses of profiles are:

- setting up the environment for a user when he logs on to MULTSESS, for instance:
  - setting up the PF key values to be used when a user is not in dynamic panel mode
  - assigning the session switch character and MSKEY for the user
  - switching the user into dynamic panel mode or into an installation designed menu
- automatically starting sessions and switch the user directly into an application
- invoking or switching between different installation designed menu panels. The profile member would contain commands to redefine the PF keys to match the new menu and display the panel
- executing a predefined set of commands at intervals, e.g. a series of ECHO commands to send various different sized RUs to monitor network response, for technical personnel only.

Automatically executing sessions at logon time

A profile may be executed automatically when a user logs on by coding the PROFILE statement in the user's directory entry.

Any profile may contain SESSION commands to start application sessions when the profile is executed.

The treatment of multiple session commands within a profile member is handled like multiple commands typed on a single terminal input line.

A SESSION command that is followed by another command will cause a new session to be activated if one does not already exist. The user will not be switched into the session and execution of other commands in the profile continues.

If the SESSION command is the last command in the profile, a new session will be established if one does not already exist and the user will be switched into session with the application.

Examples of profiles

Example 1
Define the user's session switch character as '/' instead of the MULTSESS default, and place the user into dynamic panel mode.

```
*
* Override the Session Switch Character
*
SET SESSCHAR /
*
* Put user into dynamic panel mode
* with automatic PF key assignments
*
PANEL ON PFK
```

After logging on, the first screen the user sees will be the MULTSESS dynamic panel with PF keys set for session selection.

Example 2
Set up the user's PF key values.

Define the switch character as <PFK12>.

Display menu panel MENU1.

Start sessions with IMS and TSO.

Because the session request for TSO is the last command in the profile, the user is switched into TSO.

```
*
* Set users PF keys
*
SET PFK1 IMM SESSION TSO 1ST    ( USER1/PASS1
SET PFK2 IMM SESSION TSO 2ND    ( USER2/PASS2
SET PFK3 IMM SESSION IMS IMS1   TERM001
SET PFK4 IMM SESSION IMS IMS2   TERM002
SET PFK5 IMM SESSION CICS CICS1 TERM001
SET PFK6 IMM SESSION CICS CICS2 TERM002
SET PFK7 IMM SESSION ROSCOE
*
*  Override the Session Switch Character
*
SET SESSION PFK12
*
*  Switch to user menu panel MENU1
*
PANEL MENU1
*
*  Start application sessions
*
SESSION IMS IMS1
SESSION TSO 1ST ( USER1/PASS1
```

After logging on, the first screen the user sees will be the TSO 'Ready' screen. To return to MULTSESS, press <PFK12>.

Note: Different VTERM names are used for the two IMS sessions. These could be defined to the IMS Security Maintenance Utility to allow access to different sets of transactions.

The customer-defined panel MENU1 would describe these transaction groups in terms familiar to the user, e.g. Payroll, MRP.

Example 3
Set up any options unique to this user.

Execute profile member PROFPAN1 to invoke panel MENU1

Set <PFK12> is allow switching between MENU1 and MENU2.

User's main profile pointed to by his directory entry:

```
*
*  Set user unique options
*
SET SESSCHAR /
SET SESSCHAR AUTON
SET MSKEY   !
*
*  Invoke panel MENU1
*
PROFILE    PROFPAN1
```

Profile dataset member PROFPAN1:

```
*
*  Set PF keys for panel MENU1
*
SET PFK1 IMM SESSION TSO 1ST * (&SYSUSER./&SYSPASS
SET PFK2 IMM SESSION TSO 2ND  *
SET PFK3 IMM SESSION IMS IMS1 POOLIMS*
SET PFK9 IMM DISC
*
*  Set PFK12 to switch to panel MENU2
*
SET PFK12 IMM PROFILE PROFPAN2
*
*  Display panel MENU1 to the user
*
PANEL MENU1
```

Profile dataset member PROFPAN2:

```
*
*  Set PF keys for panel MENU2
*
SET PFK1  IMM Q TIME
SET PFK2  IMM Q NAMES
SET PFK3  IMM Q USERS
SET PFK11 IMM DISC
*
*  Set PFK12 to switch to panel MENU1
*
SET PFK12 IMM PROFILE PROFPAN1
*
*  Display panel MENU2 to the user
*
PANEL MENU2
```

Masking of profile names

The logon profile name specified in the MULTSESS directory may be generated on the basis of either the users userid or physical terminal luname, or a combination of both, using a character masking technique.

With a suitable mask, it is possible to create a general purpose generic directory entry, or a general purpose logon profile which executes other profiles based on userid or luname, to display a menu tailored to the needs (or security authorization) of the user or the terminal.

MASK CHARACTER	ACTION
+	This character position in the profile name will be replaced with the corresponding character from the terminal luname.
?	This character position in the profile name will be replaced with the corresponding character from the userid.
&	The remaining character positions of the profile will be filled with as many characters from the luname as are required to complete the 8 digit name, starting from the beginning of the physical terminal luname.
%	As for & above, except that the fill characters are taken from the start of the userid.

Examples:

```
Directory or command:        PROFILE  P+PROF++
User logs on at luname:               T1234567
PROFILE name generated:               P1PROF67

Directory or command:        PROFILE  P&
User logs on at luname:               T1234567
PROFILE name generated:               PT123456

Directory or command:        PROFILE  ???PROF?
Userid:                               TSGUSER1
PROFILE name generated:               TSGPROF1

Directory or command:        PROFILE  ???P&
Userid:                               TSGUSER1
User logs on at luname:               ROMELU01
PROFILE name generated:               TSGPROME
```

General purpose directory entry example

```
*
*  This directory entry could be used for all
*  users. Different menu panel and PF key
*  settings are used by calling a profile based
*  on the first 3 digits of the userid.
*
*  Although SESSION * is used, users can only
*  access those applications included on the
*  menu, assuming the menu has no unprotected
*  command input area.
*
USER  *   RACF   G   3
PROFILE   ???MENU
VTERM     POOLGEN
SESSION   *
```

General purpose profile member example

```
*
*  Set up options common to all users.
*
SET SESSCHAR /
SET SESSCHAR AUTON
SET MESKEY   !
*
*  Call another profile to invoke a menu panel
*  depending on the terminal in use.
*
PROFILE ++++PROF
```

Chapter 4 - Virtual Terminals

The need for virtual terminals

The use of virtual terminals is motivated by the need for a session manager such as MULTSESS. A VTAM logical unit that represents a physical terminal has an implicit session limit of one, i.e. a terminal can only be in session with one application at a time. Many users require concurrent access to more than one application, hence the justification for a **session management productivity tool**.

MULTSESS works by intercepting all data flow from the terminals that it is managing and distributing it to the various applications. Thus the physical terminals are only ever in session with one application, MULTSESS.

MULTSESS may be managing many hundreds of application sessions on behalf of the terminal users. The applications with which MULTSESS is communicating think they are talking to 3270 terminals, which are all called MULTSESS. Applications such as TSO, ROSCOE and HCF function normally with this arrangement since they are not sensitive to the fact that they appear to be holding multiple conversations with the same terminal. The use of VTERMs is not required for such applications. MULTSESS will use its own ACB to maintain all the sessions with them.

The use of virtual terminals is a convenient way convincing such applications that they are talking to several different 3270 terminals when they are in fact running multiple sessions with MULTSESS.

IMS and CICS applications

Certain applications, notably IMS and CICS, rely on the fact that a 3270 logical unit can only ever hold one connection at a time. They use the terminal name as a unique identifier for session control, security and auditing.

What is a virtual terminal?

Any VTAM application program requires a gateway through which to pass requests for service to VTAM, an ACB. This assigns a name to the application that VTAM will recognize when servicing requests, in a similar fashion to the terminal name on a physical terminal definition. Virtual terminals are merely ACB definitions in SYS1.VTAMLST that provide extra gateways, each with a different name, and thereby allow MULTSESS to initiate session requests via VTAM to the applications that MULTSESS wishes to communicate with.

When VTAM initially contacts the application to start the session, the name of the VTAM resource requesting the session is passed to the application. Without MULTSESS this is the terminal luname. With MULTSESS, the name passed is the name of the virtual terminal ACB requesting the session. Thus, even though MULTSESS may be running many sessions to CICS for example, CICS views each session as originating from a different source.

Applications needing virtual terminals

As a rule-of-thumb, applications that need real terminals to be defined to them, usually in some sort of table, do not support parallel sessions to the same logical unit and therefore require the use of virtual terminals when working with MULTSESS. In addition, some other applications also require virtual terminals - there follows a list, by no means exhaustive, of the more commonly used applications

CICS	IMS
NCCF	NCI
ENVIRON1	PHOENIX
VM/VCNA	UCC7

Defining and sharing of virtual terminals

Virtual terminals are defined to VTAM by adding APPL statements to SYS1.VTAMLST. They should also be defined in any applications that require internal terminal definitions, e.g. they should be added to the CICS TCT.

Expensive hardware solutions that provide multiple sessions (e.g. PC/3270) require a unique logical unit definition and hence consume VTAM storage, for each potential session whether the application requires it or not and even if the user is not actively using the logical unit. Wherever possible and consistent with both integrity and auditing, MULTSESS should share virtual terminal ACBs to minimize storage consumption and maintenance.

If all the applications with which a user will communicate support parallel sessions, no virtual terminal is required. MULTSESS will use its own ACB for all sessions. If a user requires a maximum of one session with an unlimited number of different applications, any of which require the use of a virtual terminal, a single ACB may be shared by all the user's sessions.

If a user requires more than one concurrent session with an application not supporting parallel sessions, the maximum number of VTERMs required to support all the user's sessions will be equal to the maximum number of sessions with any such application. Note that the extra ACB(s) need not be allocated to the user until he actually starts the extra session(s).

Two techniques are available for virtual terminal allocation: assigning specific VTERMs to specific users and a VTERM pooling technique. These are described in the following sections.

Assigning specific VTERMs

Authorization
A user is authorized to use a particular virtual terminal by the presence of one or more VTERM statements in the user's directory entry.

Assigning to a user
A specific virtual terminal is assigned to a user by coding, on the user's VTERM statements, the required ACB name.

Generating a VTERM ACB name
A masking technique is available to generate the VTERM ACB name at logon time based on the userid, the name of the real terminal in use, or a combination of both. This is described in the section *Generating specific VTERM names* on page 4.6.

Indicating a session vterm
The VTERM to be used for a particular session may be indicated on a SESSION statement in the directory entry, on a SESSION command assigned to a PF key or as part of a SESSION command typed by the user. In all cases use of the requested VTERM will only be allowed if matched by a VTERM statement in the directory entry for the user.

Vterm not specified
If a VTERM is not specified on a SESSION statement or command and the application requires the use of a virtual terminal, the VTERM ACB is taken from the first VTERM statement for the user requesting the session.

If no vterm statements have been defined for a user, sessions with applications requiring the use of virtual terminals will not be allowed.

Note: The VTERM and SESSION statements are described in detail in the User Installation Manual, *Chapter 4 - Defining users to MULTSESS/HPO*.

The advantages of specific VTERMS

1. Some applications manage security by terminal name. By assigning specific VTERMs based on real terminal name, security routines will require very little change, if any.
2. Standard reporting tools supplied with many applications report by terminal name. By allocating a specific VTERM to a user, or generating the name based on userid using the masking technique, such reports relate to specific users and their value is consequently enhanced.
3. Users are guaranteed access since their VTERM is reserved for their use. If the pooling technique is used and all VTERMs in a pool are in use, a user may be unable to start certain sessions.
4. Problem determination and administration is made easier because the virtual terminal used for any session can be reliably predicted.

The disadvantages of specific VTERMS

1. If many more users are defined than will concurrently use MULTSESS, more virtual terminal ACBs need to be defined than will be used at any given time.
2. Maintenance is increased since adding a new user also involves defining a new virtual terminal.

Notes: The two possible VTERM allocation techniques, specific ACBs and pooling, are not mutually exclusive. For example:

- use pooling for the majority of users, but specific VTERMs for politically sensitive users to guarantee them access even if the pools are fully used.
- use pooling for widely used systems such as CICS and IMS, but use specific VTERMs for NCCF.

More than one user may be assigned use of the same specific virtual terminal. The first user to request a session using the VTERM will obtain use of the ACB. The other users must wait until the VTERM is freed. This is a useful technique for allowing a group of users access to an application, but ensuring only one is logged on at a time.

Examples

The following are examples of directory entries illustrating the use of specific virtual terminals. All examples assume that CICS, IMS and NCCF are defined in the ADT with VTERM=YES. TSO is defined with VTERM=NO.

1. Sessions with all applications requiring use of a virtual terminal will share ACB VTERM001. Because the user is only authorized to use one VTERM ACB, he can only have one session at a time with each of CICS, IMS and NCCF.

```
USER     .........
VTERM    VTERM001
SESSION  *
```

2. A single session with each of CICS and IMS will always share the virtual terminal ACB named VDB001. Sessions with NCCF will always use NCCFV001.

```
USER     ........
VTERM    VDB001   VT1
VTERM    NCCFV001 VT2
SESSION  IMS  VT1
SESSION  CICS VT1
SESSION  NCCF VT2
```

3. This user is allowed two concurrent sessions with CICS. Because CICS does not support parallel sessions, a separate VTERM is required for each CICS session. The first VTERM will be shared for any other session.

```
USER     ........
VTERM    VACB001   VSHARE
VTERM    VACB002   VCICS2
SESSION  CICS   VCICS2
SESSION  *
```

4. This user has access to a range of virtual terminals and may choose any one for use with any session. If you are starting a session to an application that requires a VTERM and a VTERM name is not specified on a session command, then VTERM001 will be taken as the default (from the first VTERM statement).

```
USER     ........
VTERM    (VTERM001  VT1)  (VTERM002  VT2)
VTERM    (VTERM003  VT3)  (VTERM004  VT4)
VTERM    (VTERM005  VT5)  (VTERM006  VT6)
SESSION  *
```

5. The same VTERM ACB is allocated to a group of users. The first user requesting an IMS session will obtain use of the VTERM. Other users must wait until the VTERM becomes free. This technique provides a means of making an application available to a group of users while limiting access to one user at a time.

```
USER     DEV*
VTERM    VIMS001
SESSION  TSO
SESSION  IMSPROD  VIMS001
```

Generating specific VTERM names

Specific VTERM names

The virtual terminal ACB name to be used for a particular session may be generated based on the real terminal name at which the user is logged on, or the userid in use, or a combination of userid and luname.

This may be achieved by use of the following mask characters in the ACBNAME specified on the VTERM statements in the user's directory entry.

MASK CHARACTER	ACTION
+	This character position in the ACB name will be replaced with the corresponding character position from the real terminal logical unit name (luname).
?	This character position in the ACB name will be replaced with the corresponding character position from the userid.
&	The remaining character positions of the ACB name will be filled with as many characters from the terminal name as are required to complete the 8 digit name, starting from the beginning of the real terminal name. This technique enables ACB names to be generated by prefixing the real terminal logical unit name.
%	As for &, except the fill characters are taken from the start of the userid.

Examples

```
Directory specifies:          VTERM V+TERM++ symname
User logs on at luname:       T1234567
Vterm ACBNAME generated:      V1TERM67

Directory specifies:          VTERM V& symname
User logs on at luname:       T1234567
Vterm ACBNAME generated:      VT123456

Directory specifies:          VTERM ???TERM? symname
Userid:                       IMSUSER1
Vterm ACBNAME generated:      IMSTERM1

Directory specifies:          VTERM ???V++++ symname
Userid:                       IMSUSER1
User logs on at luname:       ROMELU01
Vterm ACBNAME generated:      IMSVLU01
```

Generating the virtual terminal name by means of the masking technique is particularly useful in conjunction with generic entries in the MULTSESS directory.

Large groups of users
The same directory entry may be used to define a large group of users but assign a unique virtual terminal to each user at logon time.

User group directory definition
Example with a separate virtual terminal assigned to each user based on the real terminal in use.

```
USER     TECH*     RACF   G   3
VTERM    V+++++++  VT1
SESSION  CICS  VT1
SESSION  IMS   VT1
SESSION  TSO
```

Virtual terminal pooling

VTERM allocation delay
The virtual terminal pooling technique provides a means of delaying allocation of a vterm to a user until it is actually needed. A vterm ACB is dynamically allocated from a pool at session initiation time.

For installations that have many more users defined to MULTSESS than will ever be logged on at the same time, virtual terminal pooling provides a means of limiting the number of virtual terminals that need to be defined. In addition, pooling provides a flexible and powerful technique for controlling allocation of virtual terminals to users. Reducing the overall number of virtual terminals eases maintenance and reduces storage requirements in VTAM.

Invoking pooling
Virtual terminal pooling is invoked by including a VPOOL DD statement that points to a partitioned dataset member which contains one or more pool definitions.

Defining pooling
The statements used to define virtual terminal pools are described in the MULTSESS/HPO User Installation Manual.

An **unlimited** number of virtual terminal pools may be defined containing **any number** of virtual terminal ACBs. If desired, an ACB may be defined in more than one pool.

Each VTERM pool is assigned a 5 to 8 character name which must begin with the letters 'POOL'.

Authorization
A user is authorized to access the virtual terminal pool(s) by one or more VTERM statements in the user's directory entry:

VTERM POOLaaaa

Generic names
Generic pool names are supported, authorizing the allocation of a single ACB from any of a group of pools:

VTERM POOLDB*

continued....

Logmodes and pools
When defining virtual terminal ACBs to VTAM, the DLOGMOD parameter is not normally specified. This provides flexibility in the allocation of VTERMs and may reduce the total number of ACBs that need to be defined. MULTSESS uses a 'late bind' technique to assign bind parameters at session initiation. This means, for example, that an ACB may be used once for an SNA model 3 session and later for a non-SNA model 4 session.

ACB characteristics
The characteristics of the ACB will be assigned by MULTSESS at session initiation based on the logmode specified on the SESSION statement, or defaulted if an overriding logmode is not specified.

Optional logmode name
Each virtual terminal pool you define may optionally have a logmode name associated with it. Before allowing allocation of an ACB from such a pool, MULTSESS will compare the pool logmode with the logmode used to request the session and only allocate an ACB from that pool if the logmodes match.

Differing logmode names
Since an ACB may occur in more than one pool, and the pools may have differing logmode names, the same ACB may be used for sessions on behalf of any terminal type.

Predefining terminal characteristics
For applications that require characteristics to be predefined in an internal table, the ACB should only be included in pools with a logmode entry that corresponds to the application's table definition of the ACB.

Starting the session
Logmode names specified on VPOOL definitions should be the name used to start the application session. This will probably not be the logmode that the user logged on with, but will be a name assigned from the MULTSESS logmode table, unless an overriding name has been specified on the SESSION command. Please refer to *Chapter 5 - MULTSESS/HPO use of logmodes* for further details.

Notes
The use of a logmode name on pool definitions is optional and its purpose is to allow selective allocation based on differing terminal types. If your terminal population is entirely or mainly of one type, logmodes names may not be required to distinguish different groups of ACBs.

If you are doubtful about the correct logmode name to specify on a VPOOL definition for a particular group of terminals, connect the terminal to MULTSESS with the logo panel displayed. The MULTSESS console file will contain a message:

TPVLOGON - SESSION CREATED WITH NODE *luname* LOGMODE *name*

where:

luname	is the terminal name
name	is the logmode that MULTSESS has allocated from its own modetab. Unless overridden on a SESSION command, this is the logmode that MULTSESS will use when initiating sessions on behalf of the terminal and is the name to specify on the VPOOL definition.

Allocation of pooled VTERMs
An ACB usually need only be allocated from a pool when a user first starts a session. Once allocated, a pooled vterm may be shared for all a user's sessions. Sharing is described below.

Search rules
The search rules for virtual terminal pools are as follows:

1. Pools are searched top down in the order they occur in the VPOOL dataset.
2. The pool name is matched against the absolute or generic pool name specified by the user.
3. If a pool has an associated logmode name, this is matched against the logmode used to request the session. Otherwise no matching on logmode is done.
4. The matched pool will be searched for an unallocated ACB. If all ACBs in the pool are in use, the search continues for another pool that matches the search criteria.
5. If the selected ACB cannot be opened, e.g. is inactive to VTAM, the search continues for another free ACB in this or another matching pool.

Sharing pooled VTERMs
Once a user has been allocated a virtual terminal from a pool, MULTSESS will attempt to share the ACB for all the user's other sessions.

Rules
The rules governing sharing of vterms allocated from a pool for new sessions are:

1. If the new session does not require use of a virtual terminal (VTERM=NO being specified in the ADT definition of the application), the MULTSESS ACB will be used.
2. If the pool name specified on the session request does not match (uniquely or generically) the pool from which the ACB was originally allocated, the ACB cannot be shared.
3. If the ACB was allocated from a pool with an associated logmode name, the logmode of the new session is compared with the logmode of the original pool. If the names do not match, the ACB cannot be shared. If the ACB is already in use with the application for which the new session has been requested, i.e. the user is requesting a second session with the application, the ADT definition of the application is examined.

   If VTERM=UNQSYM or UNQNODE was specified, indicating that the application does not support parallel sessions with the same VTERM, the ACB cannot be shared.

ACB sharing
The ACB will be shared for unique sessions with multiple applications provided the poolnames and (optionally) logmode names match. The same ACB may also be shared for multiple sessions with applications that support parallel sessions (VTERM=YES specified in the ADT). If the original ACB cannot be shared, MULTSESS will attempt to allocate another ACB from the specified pool(s).

Notes

Each VTERM entry in the user's directory entry authorizes the use of one VTERM ACB. If a user has only one VTERM statement, he can only use one VTERM at a time. If a user requires concurrent use of two VTERMs from different pools, he needs two VTERM statements.

Similarly, to run two concurrent sessions with an application that does not support parallel sessions (VTERM=UNQ being in the ADT) he needs two VTERM statements. Specifying only one VTERM statement is a convenient way of limiting a user to one session with applications such as IMS and CICS.

Examples

1. Simple environment.

   All terminals the same model.

   One common pool for all users.

   VTERM pool definitions.

```
VPOOL POOL1
ACBNAMES  ACB1 ACB2 ACB3 ......
ACBNAMES  ACB120 ACB121 ACB122 ......
```

   User's directory entry.

```
USER    *
VTERM   POOL*
SESSION *   POOL*   *
```

2. Mixed environment.

   Differing terminal types.

   Groups of terminals of different models defined in CICS TCT.

   VTERM pool definitions.

```
*
*  ACBs defined to CICS as 24 x 80.
*
VPOOL    POOLC2   S2PN87C7
ACBNAMES ACB21  ACB22 ACB23.....
*
*  ACBs defined to CICS as 32 x 80.
*
VPOOL    POOLC2   S3AN87C7
ACBNAMES ACB31  ACB32 ACB33.....
*
*  ACBs defined to CICS as 43 x 80.
*
VPOOL    POOLC2   S4AN87C7
ACBNAMES ACB41  ACB42 ACB43.....
```

User's directory entry.

```
USER  *
VTERM POOLC2
SESSION CICS *
```

All pools are called POOLC2.

When user requests a CICS session, MULTSESS will allocate an ACB from the POOLC2 that has been defined with a logmode matching the user's real terminal.

3. Mixed environment.

   Differing terminal types.

   All IMS TERMINAL definitions are model 2.

   VTERM pool definitions.

```
*
*   General use pool
*
VPOOL    POOLA1
ACBNAMES ACB1  ACB2  ACB3....
ACBNAMES ACB8  ACB9  ACB10...
*
*   Pool for IMS model 2 usage.
*
VPOOL    POOLIMS   S4C32782
ACBNAMES ACBA  ACBB  ACBC....
ACBNAMES ACBX  ACBY  ACBZ....
*
*  Repeat of POOLIMS, but without logmode
*  restrictions
*  Only non-IMS sessions will be allowed to
*  use this pool
*
VPOOL    POOLA2
ACBNAMES ACBA  ACBB  ACBC....
ACBNAMES ACBX  ACBY  ACBZ....
```

   User's directory entry.

```
USER    *
VTERM   POOL*
SESSION IMS POOLIMS S4C32782
SESSION *   POOLA*  *
```

   Sessions with IMS may only use VTERMs from POOLIMS. Since IMS only has model 2s defined to it, POOLIMS will only allocate an ACB for sessions using a model 2 logmode.

The session statement for IMS specifies a model 2 logmode, thus forcing model 2 mode for any terminal type. Sessions with applications other than IMS may use any pool whose name begins POOLA.

Because POOLA1 occurs first in the VPOOL definitions, sessions other than IMS will try to allocate a VTERM from this pool first.

If an ACB cannot be allocated from POOLA1, overflow into POOLA2 is allowed. POOLA2 contains the same ACB names as POOLIMS. Thus stealing of IMS ACBs is allowed when all other ACBs are exhausted. To prevent overflow and thus always reserve ACBs for use with IMS, remove the definition of POOLA2.

Note: In order to allow overflow into the next pool, a single VTERM statement that permits access to both pools is used.

Chapter 5 - MULTSESS/HPO use of logmodes

Controlling the session

For every terminal in your network there exists a logmode table entry that describes to VTAM and the applications the terminal characteristics and certain control parameters that partners, terminal and application will use to control the session.

Logmode tables and entries

All logmode tables are held in SYS1.VTAMLIB. The logmode table used by the terminal is pointed to as part of the terminal definition in SYS1.VTAMLST, by the MODETAB= operand on a LINE, GROUP, PU or LU statement.

The logmode entry, often called simply the logmode, is defined either by the DLOGMOD= parameter in SYS1.VTAMLST, or is inserted by VTAM as part of USS table processing, or may be specified explicitly by the user as part of his logon command.

MULTSESS is supplied with its own logmode table, membername MULTSESS in the supplied load library. The source for this table is supplied in the sample library on the distribution tape.

This load module is typically copied into SYS1.VTAMLIB as part of MULTSESS installation, and MULTSESS and its VTERMs use this table by virtue of the MODETAB=MULTSESS operand on the ACB APPL statements in SYS1.VTAMLST which define MULTSESS and the VTERMs to VTAM.

The MULTSESS logmode table is supplied for your convenience and contains definitions suitable for most terminal types. However, you may choose to use your own existing logmode table instead. Please refer to *Using your own logmode table* on page 5.7 for more details.

MULTSESS use of table

MULTSESS makes use of the table for two quite separate purposes at different points in its processing. These are described in the following sections:

Terminal logon requests

When a terminal connects to MULTSESS, the logon exit is passed a set of bind parameters by VTAM. These parameters are derived from the terminal logmode.

MULTSESS will search its own table, held as member MULTSESS in the steplib load library, looking for an entry where all the individual bind parameters match, with the exception of the PACING, VPACING and COS parameters.

Note: At this stage MULTSESS is not interested in the logmode name used on the logon request. With older releases of VTAM, MULTSESS does not have knowledge of the original name used, since it is not made available by VTAM.

If a matching entry is found, the logon is allowed and the name of the matching entry is saved for later use in session establishment.

Note: This is the name of the MULTSESS table entry, not the logmode used on the logon request.
It is essential therefore that the MULTSESS logmode table contains entries applicable to all the logmodes in use by the terminals in your installation, if those terminals are to be able to logon to MULTSESS.

If a matching entry cannot be found, the logon is denied and the terminal is released back to VTAM.

Supplied table

The logmode table supplied with MULTSESS contains various entries for the more commonly used combinations of protocol type, terminal type and rusizes, and should prove adequately wide-ranging for most installations.

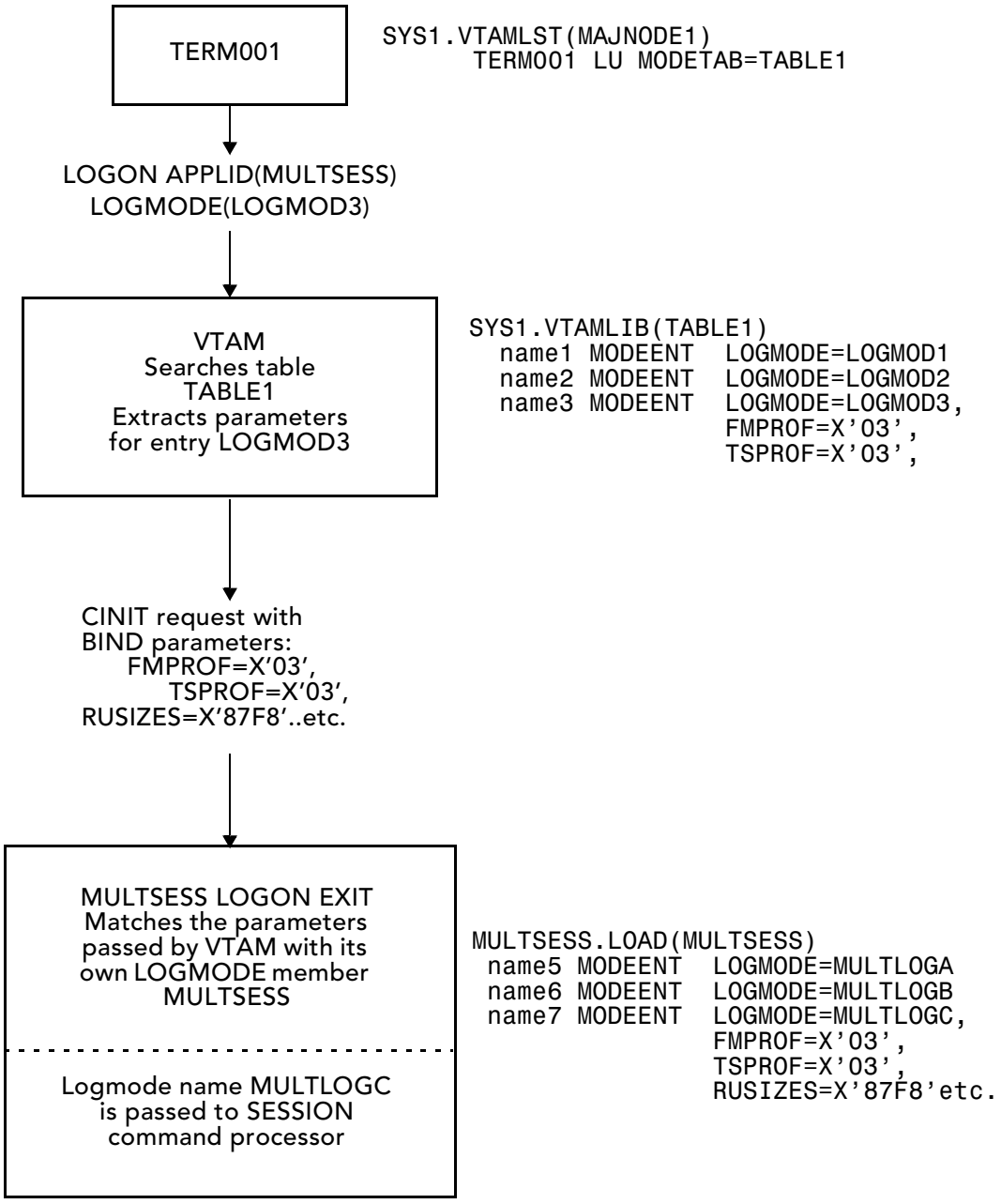
However, should you encounter difficulties connecting a particular terminal, it may be that MULTSESS is refusing connection because of logmode validation failure. In this case the MULTSESS console file will contain an error message followed by a listing of the logmode which should be added to the MULTSESS table.

For further details refer to *Updating logmode entries* on page 5.8.

RUSIZE parameters

RUSIZEs are irrelevant to non-SNA terminals and therefore the RUSIZE parameters are not checked for non-SNA terminals.

The validation process is illustrated below:



Establishing sessions with other applications

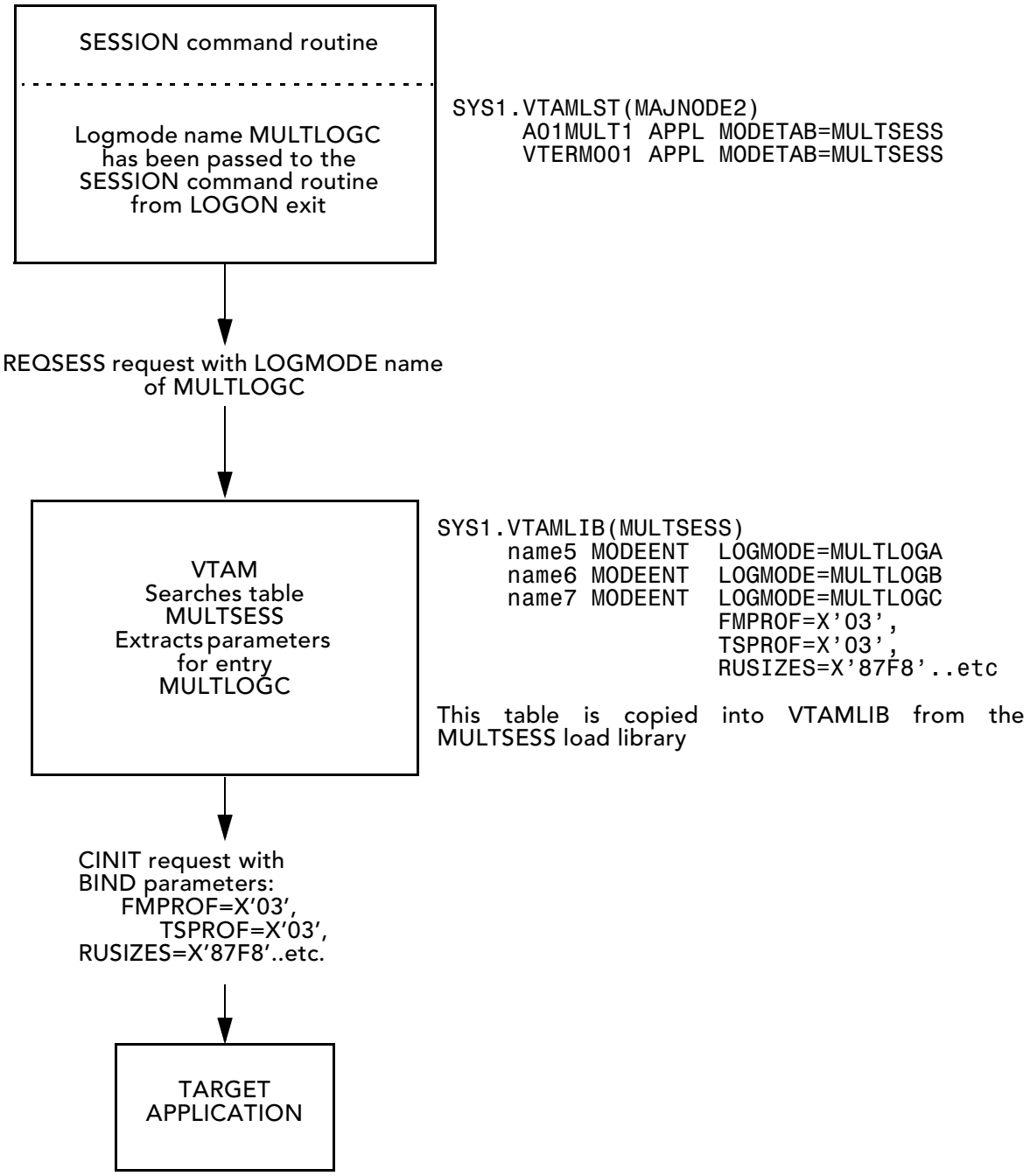
When a user requests MULTSESS to establish a session with another application, MULTSESS will pass to VTAM, as part of the session request, the name of the logmode to be used for the session. The target application will use this as part of session establishment, just as it would a logmode specified on a logon command entered at a physical terminal.

The name which MULTSESS passes is the name it obtained from its own table in the logon validation process described previously. VTAM will use this name to extract the bind parameters from the MULTSESS table in SYS1.VTAMLIB.

It is possible to specify a logmode name on a SESSION command in which case this logmode will be used in place of that saved during logon validation. However the logmode so named must be present either in the MULTSESS table in SYS1.VTAMLIB or the IBM supplied default table ISTINCLM.

If the virtual terminal pooling technique is to be used, a VTERM ACB will be allocated from a pool at session initiation time. If pools are specified with an associated logmode name, the logmode names used must be the names as used, on the LOGMODE= parameter, in the MULTSESS logmode table.

The use of logmodes at session establishment is illustrated below:



Specifying a logmode on the SESSION command

A logmode name may be specified on a SESSION command or a SESSION statement in the directory. The named logmode will be used during session establishment in place of that saved during the logon validation process described previously. However, the logmode specified must be present in the MULTSESS logmode table in SYS1.VTAMLIB or in the IBM supplied default table ISTINCLM.

The overriding logmode specified must be compatible with that used when logging on to MULTSESS. It should only be used to refine the description of the terminal, not to completely redefine it. For example, you cannot logon on as a Model 3 and then expect to run sessions in Model 4 mode.

Note: Remember that VTAM and hence MULTSESS has no knowledge of the physical characteristics of the terminal other than those defined in the logmode used at logon time. If you sit down at a Model 5 terminal and logon to MULTSESS using a logmode which specifies a primary screen size of 24 x 80 with no alternate (PSERVIC=.....1850.....7E), as far as MULTSESS is concerned you are sitting at a Model 2 screen. You cannot then request a session specifying a Model 5 logmode (PSERVIC.=..18501B847F) and expect to work in Model 5 mode.

Logmode considerations for users of HCF and similar applications

The facility of specifying a logmode on the SESSION command is of particular interest to users of HCF (and similar applications that do not support alternate screen sizes), but who may wish to use the alternate screen facility with other applications. To illustrate this, consider the following case.

The user sits down at a Model 4 terminal and logs on to MULTSESS with a logmode specifying a 24 x 80 primary screen size and a 43 x 80 alternate size (PSERVIC=....18502B507F). He starts a session with TSO without specifying an overriding logmode on his SESSION command. Since TSO will always work in alternate mode if an alternate screen size is defined in the logmode, the user's TSO session will operate in 43 x 80 mode.

If the user now wishes to connect to HCF, the SESSION command should specify a logmode that specifies a primary screen size of either 24 x 80 or 43 x 80, but has no alternate screen size defined:

PSERVIC=....1850....7E

or

PSERVIC=....2B50....7E

Note: The logmode name specified on the SESSION command must exist in the MULTSESS logmode table or the IBM default table ISTINCLM.

Logmode considerations for users of NCCF

Older versions of NCCF require that all terminals connecting to it use a logmode called DSILGMOD, regardless of terminal type.

If all the terminals used to access NCCF are the same type and model, amend entry DSILGMOD in the MULTSESS table to match the terminals in use.

If terminals of various types and models are to be used with NCCF, multiple logmode definitions, all called DSILGMOD, are required to describe each terminal type in use. Since each entry in a logmode table must have a unique name, each different version of DSILGMOD must be assembled into a different table.

To ensure that the correct version is used when initiating a session with NCCF, define a special set of VTERMs to be used with NCCF. Define the VTERMs to VTAM with the MODETAB operand pointing to the table that contains the appropriate version of DSILGMOD. When starting a session with NCCF, select the appropriate VTERM for the type of terminal in use.

Using your own logmode table

The MULTSESS logmode table (load module MULTSESS in the load library) is supplied for your convenience and contains sample logmode entries suitable for most terminal types. However, you may chose to use your own logmode table instead.

This is usually only practical if ALL the terminals using MULTSESS share a SINGLE mode table member in SYS1.VTAMLST. Using your own table has the advantage that logmode names are consistent for the terminal-MULTSESS and MULTSESS-application sessions.

If you wish to use your own logmode table, do the following:

1. Copy or reassemble your own logmode table into the MULTSESS load library, replacing the existing load module called MULTSESS.
2. Amend the definition for MULTSESS and its virtual terminals in SYS1.VTAMLST.

   Change the MODETAB=operand on all definitions to read:

   MODETAB=*yourtab*

   where *yourtab* is the member of SYS1.VTAMLIB containing the logmode table you wish MULTSESS to use.

Updating logmode entries

The source of the MULTSESS LOGMODE table is distributed on file 3 of the distribution tape and should be updated to contain definitions of all the terminals that are to use MULTSESS.

Normally, there is no need to update the table as most terminal types are already defined. However if it needs to be updated it must be assembled to both the MULTSESS load library and SYS1.VTAMLIB for any changes to take effect and for it to be invoked correctly. The member ASMSAMP in the MULTSESS sample library on the distribution tape may be used for this purpose.

It is important to note that any MODETAB updated this way will not be effective until all the VTAM major nodes containing references to these definitions have been varied inactive and active again using the appropriate VTAM commands, for example:

'V NET,INACT,ID=MAJNODE'

'V NET,ACT,ID=MAJNODE'

If at any time MULTSESS detects a terminal attempting to connect to it that is not defined in the MULTSESS MODETAB an error message will be issued to the CONSOLE DD file together with the parameters required. You should add an entry with these parameters to the MULTSESS source member, recompile and reactivate the relevant major node definitions.

Restrictions

For a terminal to work correctly with MULTSESS it is important that its PRIMARY screen width is defined as 80 characters. The height may be any IBM supported height, but MULTSESS itself will only use the first 24 lines. The alternate screen size may be any IBM supported size and will be supported for sessions invoked through MULTSESS.

Example
LOGMODE entry for a Model 2 SNA terminal:

```
                                                col 72
S2DN87C7 MODEENT LOGMODE=S2DN87C7,                 +
          FMPROF=X'03',TSPROF=X'03',               +
          PRIPROT=X'B1',SECPROT=X'90',             +
          COMPROT=X'3080',RUSIZES=X'87C7',         +
          PSERVIC=X'020000000000000000000000' +
```

MULTSESS use of RUSIZES

When an SNA terminal logs on to MULTSESS, the maximum outbound and inbound RUSIZES are specified in the LOGMODE entry used by the terminal to log on to MULTSESS. This LOGMODE entry can be found by reference to the MODETAB operand specified or defaulted on the LU definition statement in SYS1.VTAMLST. When a session is requested by the terminal, MULTSESS initiates the session specifying a logmode name determined by the procedure outlined in the previous section.

This logmode entry can be found by reference to the MODETAB operand on the VTERM APPL definition statement in the VTAMLST library (or MULTSESS APPL definition statement if VTERMs are not being used). This set of parameters is merely a suggestion and the application can use different values if it wishes. TSO will use the requested parameters, but IMS and CICS will use the values generated into them. (For example: CICS will use the DFHTCT macro RUSIZE parameter as maximum inbound RUSIZE and the BUFFER parameter as maximum outbound RUSIZE. IMS will use the COMM macro RECANY parameter as the maximum inbound RUSIZE and the terminal macro OUTBUF parameter as the maximum outbound RUSIZE).

CICS users should be aware that the default RUSIZEs are different from CICS V1.5 and CICS V1.6.

When this set is returned to MULTSESS, a check is made against the set used on the terminal to the MULTSESS session. The following fields must be compatible for the two sessions to work correctly. The message issued if the check fails is indicated below followed by 'E' (the session is not allowed), or 'W' (warning only - the session is allowed).

APPL LUTYPE	NE	TERM LUTYPE	- MS0119 E
APPL OUTBOUND RUSIZE	GT	TERM INBOUND RUSIZE	- MS0115 W
APPL INBOUND RUSIZE	LT	TERM OUTBOUND RUSIZE	- MS0016 W
APPL QUERY CAPABILITY	NE	TERM QUERY CAPABILITY	- MS0122 W

Even if the session is allowed by MULTSESS when a warning message has been given, it may still hang or produce unpredictable results if any of the problem areas are encountered.

The warning message will not be seen by the user until a return to MULTSESS mode is done and will never be seen if the user is in panel mode with no message area. However, a message is always written to the CONSOLE DD file.

MULTSESS highlights RUSIZE differences to assist the customer in finding incompatibilities that have been introduced into the network by independent tuning of different components. For instance, the CICS systems programmer with knowledge of the CICS transactions may code values in the TCT to deliver all output in a single send.

On the other hand, the network systems programmer may deliberately define the terminals with smaller RUSIZEs to optimize the interleaving of all network traffic.

For integrity reasons MULTSESS does not reblock data from the application when sending it on to the terminal, and vice-versa. If it were to reblock three RUs from the application into four RUs before sending it on to the terminal and the terminal were to reject the third, MULTSESS would not know which block from the application to associate the negative response with. For database updates this could be serious if it resulted in a corrupted database.

continued ....

Knowledge of the internal workings of the relevant application is required which MULTSESS does not and should not require. For this reason, the RU will be sent unaltered and if it exceeds the maximum allowed, a terminal or application dependent error will occur. This is normally a negative response followed by a hanging condition. RUSIZE problems do not occur on NON-SNA terminals as all data requests are sent as one element and no chaining is performed.

MULTSESS use of screen sizes

When a terminal logs on to MULTSESS, the logmode entry provides the primary and alternate screen sizes to be used during the session. As outlined previously, MULTSESS will only use the primary size which must be defined as 80 columns.

When connecting to an application, the screen sizes specified in the returned bind are checked against the terminal. For success, the APPL primary screen must match either the terminal primary screen or the terminal alternate screen. The APPL alternate screen, if specified, must also match. Where screen switching is required, for example, APPL primary to terminal alternate, this will automatically be done by MULTSESS.

If no match can be found, the session will be denied and the message MS0120 or MS0121 will be issued as follows:

```
APPL DEF SCREEN SIZE NE TERM DEF/ALT SCREEN  -  MS0120
APPL ALT SCREEN SIZE NE TERM DEF/ALT SCREEN  -  MS0121
```

On non-SNA terminals, the screen sizes need not be specified. Where this is so, MULTSESS will assume that the application is in control and no switching will be performed. Errors may occur if the application assumes an incorrect screen size.

Chapter 6 - Copying data between sessions

The COPY command

The COPY command allows up to five blocks of data to be copied in a single invocation. The source blocks for a copy operation may overlap or be positioned to overlap on the target screen. The COPY command provides automatic protection against copying into protected fields.

The data to be copied may be defined as a **box**, where only data lying within the box is copied, or as a **string**, where all data is copied from the screen buffer start position to the end position, wrapping around lines as necessary.

Copying data

MULTSESS allows data to be copied from a session screen image to another session (or the same session) based on a logical box enclosing the data to be copied, or as a continuous string selected from the screen buffer.

The start and end point of boxes and strings can be specified using the **(option** field of the COPY command (refer to *COPY* on page 2.13 of the User Reference Manual). If start and end positions are not specified, they can be entered interactively by positioning the cursor and pressing an appropriate PF key.

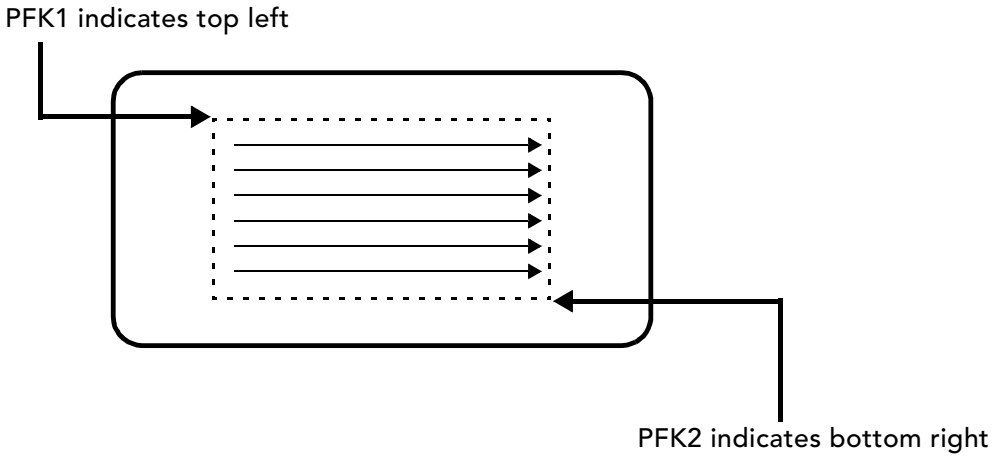
PFkeys 1 thru 10 are available for the interactive selection of blocks of data to be copied. The odd numbered PF key of each odd/even pair denotes the start point, the even numbered key of each pair indicates the end point. Thus <PFK1> and <PFK2> through <PFK9> and <PFK10> may be used to select up to 5 blocks of data to be copied in a single copy operation.

Box data

To copy a box of data, position the cursor at top left and press the odd numbered PF key. Position the cursor at bottom right and press the even numbered PF key.

Data within this logical box will be available for copy to the target session screen image.

Box copy is invoked by specifying (or defaulting) the BOX keyword on the COPY command.

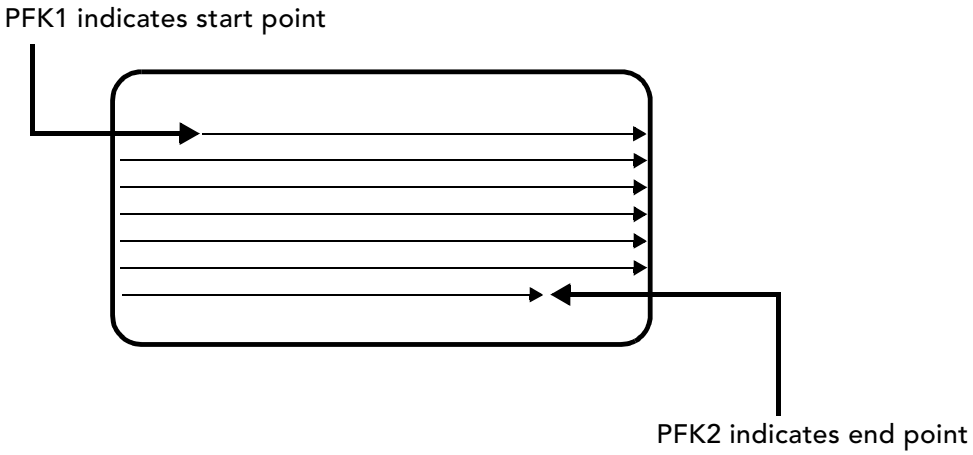


String data

To copy a string of data, position the cursor at start point and press the odd numbered PF key. Position the cursor at end point and press the even numbered PF key.

The data in the screen buffer between the start and end points will be available for copying to the target screen image.

'String' copy is invoked by specifying the DATA keyword on the COPY command.



Copying boxes

After entry of the COPY command, the screen image from APPL1 is displayed.

The box enclosing the data to be copied is defined using even/odd pairs of PF keys. PFkeys 1 through 10 are supported, allowing up to 5 boxes to be defined for a single copy operation.

An odd numbered PF key defines the top left corner of the box. The corresponding even numbered PF key defines the bottom right corner.

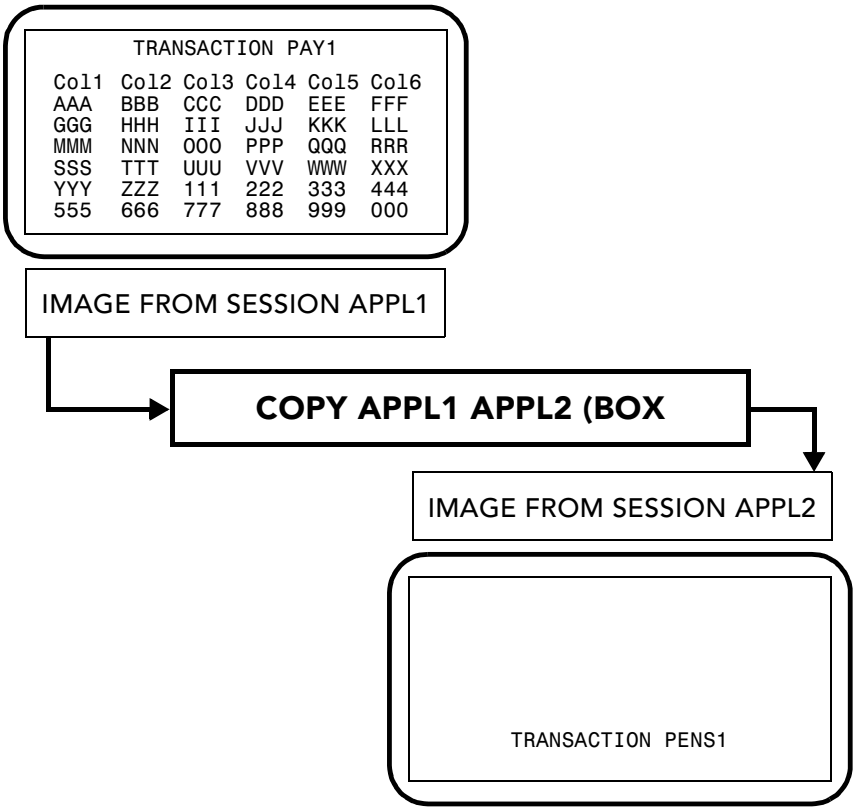
Boxes may overlap on either the source or target screen.

To illustrate the box copy facility, refer to the screen diagrams on the next page. We will copy the entire contents of columns headed AAA and BBB as one box and the top three lines of the three columns headed BBB, CCC and DDD as another box.

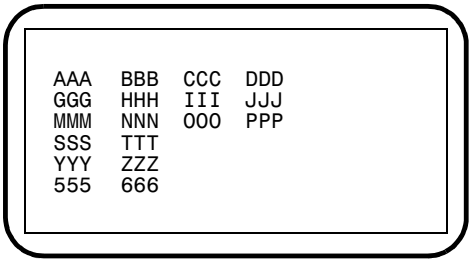
Position the cursor on the first A at the top of column 1 and press <PFK1>. This sets the top left corner of the copy box.

Position the cursor on the last 6 at the bottom of column 2 and press <PFK2>. This sets the bottom right of the box.

Next select the second box by positioning the cursor on the first B of column 2 and pressing <PFK3> and putting the cursor on the last P of the PPP on line three of column 4. Optionally, press <Clear> to check the selected data before copying to the target screen.



In our example, pressing <Clear> would display:



Press <Clear> again to return to the APPL1 screen to make further selections, or to adjust the selections already made, by reusing the same PF keys to select new coordinates.

Press <PAK1> to abandon a copy operation and return to the MULTSESS menu.

Press <Enter> to confirm the selections. The screen image from the target session, APPL2 in our example, is displayed to allow you to position the copied data.

Each box of selected data is put into the target application screen by moving the cursor to the screen location where the top left of the copied box is to be positioned and pressing the odd numbered PF key that was used to originally select the box.

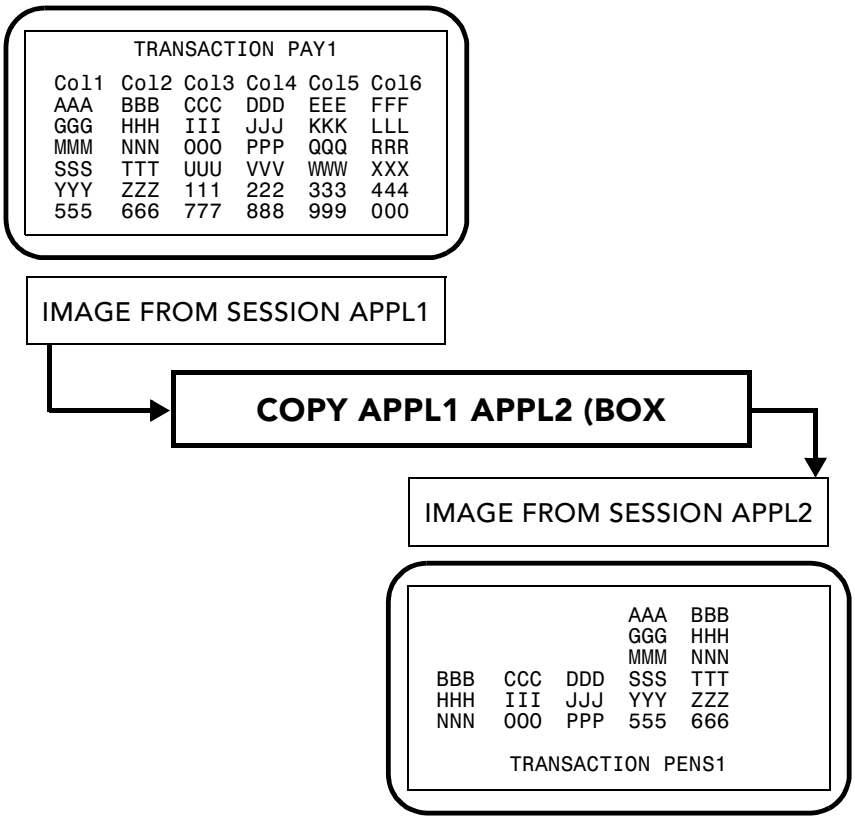
During positioning, the <Clear> key flip-flops the display between the 'before' and 'after' image of the target application screen image.

After positioning, using the **odd** numbered PF key, the box may be 'undone' by pressing the associated **even** numbered PF key. The data associated with the PF key pair is still available and may be repositioned using the odd numbered PF key.

A copy operation may be abandoned at any point using <PAK1>.

Boxes may be overlapped during positioning. Data originally selected using a **higher numbered PF key** is laid **on top** of data selected with a **lower numbered PF key** e.g. a box selected with <PFK7/8> will overlay a box selected with <PFK5/6>, regardless of the order in which the selections were made or the order in which the boxes were put into the target screen image.

To complete the copying of the two boxes selected in the example on the previous page, position the cursor to indicate the location of our first box and press <PFK1>, then position the cursor ready for the second box and press <PFK3>:



After positioning the copied data, press <Enter> to complete the COPY operation.

When placing data into the target screen image, MULTSESS will not place data in protected fields. If the target area contains protected fields, the data that would have overlaid any protected areas will be dropped and data insertion will continue from the next unprotected screen location.

The next screen displayed on completion of a COPY command depends on the return parameter specified on the COPY command. You may return to the MULTSESS menu (the default), your last accessed session, the 'from' session or the 'to' session specified on the COPY command.

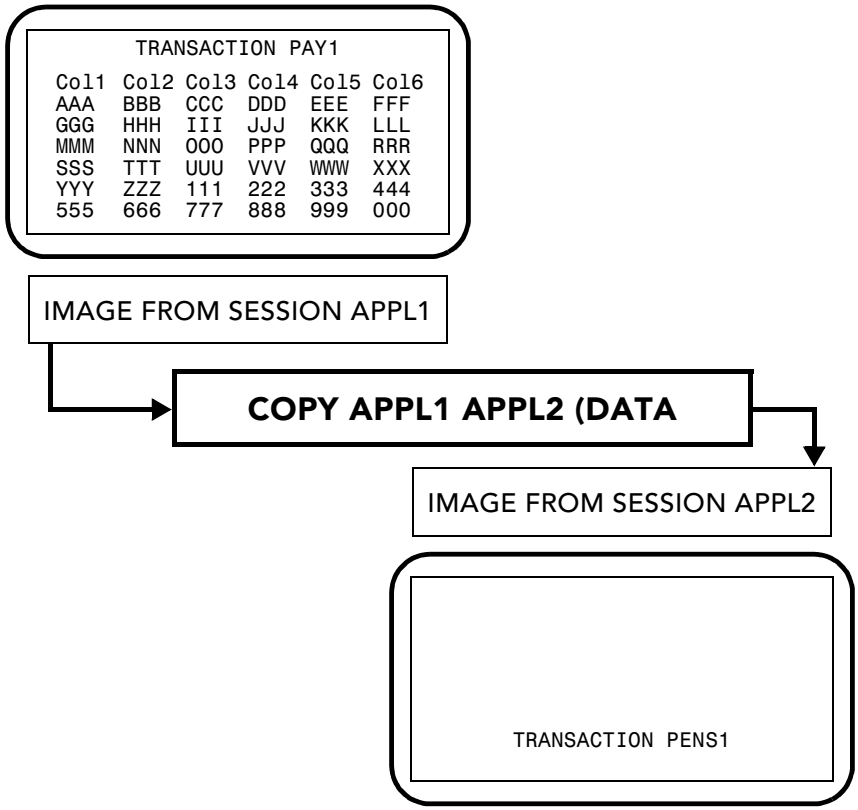
Copying strings of data

Copying a continuous string of data is invoked by specifying the DATA operand on the COPY command.

The COPY operation is carried out in exactly the same way as previously described for the BOX operand, except that the PF keys select a continuous string of data from the screen buffer, starting at the cursor location indicated by an odd numbered PF key and ending at the cursor location selected using the associated even numbered PF key.

Data selection wraps around line endings as necessary, until the end position is reached.

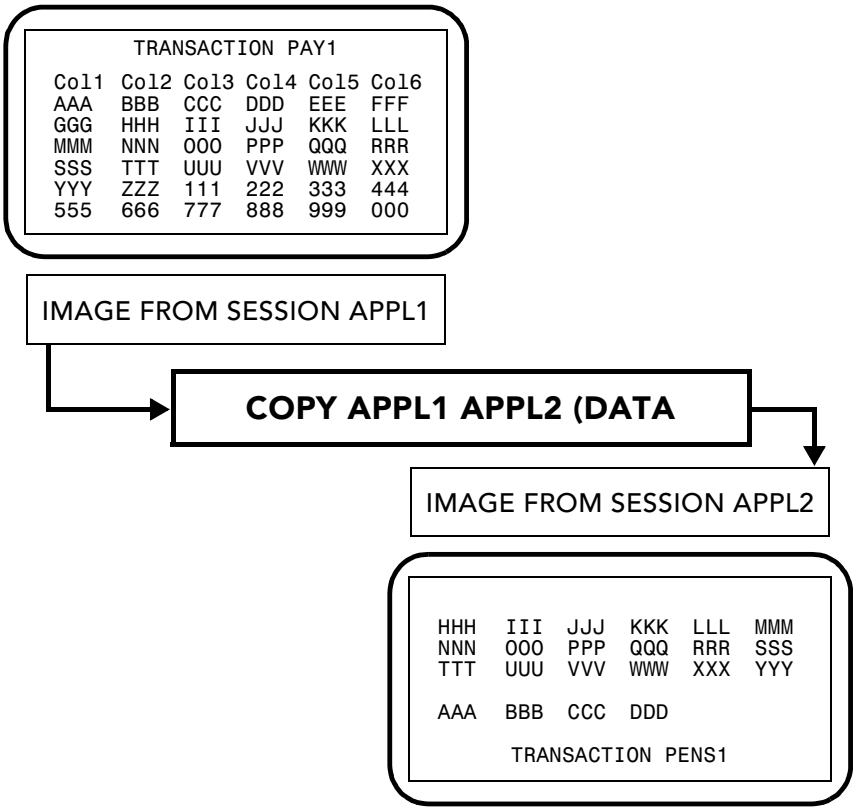
As an example, we will copy two strings, as follows:



The first begins with AAA and ends with DDD, a string selected from line 1 of our data.

The second starts with HHH from line 2 and ends with YYY on line 5.

The result of the copy operation is shown on the next page.


TRANSACTION PAY1
Col1 Col2 Col3 Col4 Col5 Col6
AAA BBB CCC DDD EEE FFF
GGG HHH III JJJ KKK LLL
MMM NNN OOO PPP QQQ RRR
SSS TTT UUU VVV WWW XXX
YYY ZZZ 111 222 333 444
555 666 777 888 999 000
IMAGE FROM SESSION APPL1
COPY APPL1 APPL2 (DATA
IMAGE FROM SESSION APPL2
HHH III JJJ KKK LLL MMM
NNN OOO PPP QQQ RRR SSS
TTT UUU VVV WWW XXX YYY
AAA BBB CCC DDD
TRANSACTION PENS1

Summary of aid key usage

DURING DATA SELECTION

<PFK1> thru <PFK10>

Used in conjunction with the current cursor position.

ODD numbered key indicates the start position of data to be copied.

EVEN numbered key indicates the end position of data to be copied.

Used in matched pairs, <PFK1> and <PFK2>, <PFK3> and <PFK4> etc, to select up to 5 data blocks.

Selected data blocks may overlap.

<Clear> KEY

Press once to display data selected for copy only.

Press again to return to source screen image.

<PAK1>

Abandon COPY operation and return to MULTSESS menu.

<Enter> key

Indicates data selection complete. The target screen image is displayed.

Other keys

Ignored.

DURING DATA PLACEMENT

<PFK1> thru <PFK10>

Used in conjunction with the current cursor position.

Odd numbered key places the data block originally selected with the same PF key into the target screen image, starting at the current cursor location.

Even numbered key removes data previously inserted by the associated odd numbered PF key.

If blocks are placed such that they overlap, the data associated with a higher numbered PF key pair will overlay data from a lower numbered PF key pair.

<Clear> KEY

Press once to display the original (before) target screen image.

Press again to return to updated target screen image.

<PAK1>

Abandon copy operation and return to MULTSESS menu.

<Enter >key

Indicates data copy complete. The next screen displayed depends on the option specified or defaulted on the COPY command.

Other keys

Ignored.

This page intentionally left blank

Chapter 7 - Help information

Comprehensive help panels are supplied for all MULTSESS commands, subcommands and messages.

Help panels for commands and subcommands are held in a partitioned dataset pointed to by the HELP DD statement. Help panels for messages are held in a partitioned dataset pointed to by the MSGLIB DD statement. Both datasets are fixed length with a record length of 80 and must not contain sequence numbers.

Help for commands

Help panels for commands are stored as members prefixed with the characters TP followed by as many characters from the command text as are needed to complete the 8 character member name. For example, when a user types:

HELP LOGOFF

the panel stored in member TPLOGOFF of the HELP library is displayed.

MULTSESS only displays help information relative to the command authority of the user issuing the HELP command. For instance, if a G class user enters HELP IMAGE, he will be given information about all options of the IMAGE command except IMAGE GET, which is reserved for C class users and above.

Help text for commands is delimited by statements containing the string:

. / END-OF-X

where X denotes the command class for information preceding the . / END-OF-X statement.

Review member TPIMAGE for an example of the . / END-OF-X statement. In member TPIMAGE, all information down to the line . / END-OF-D will be displayed to all users; whereas text between this line and the line . / END-OF-A will only be displayed to users with a command class of A, B or C.

Help for subcommands

Help text for subcommands is stored within the member relating to the major command and is displayed by entering the major command and subcommand name. For example:

HELP IMAGE VIEW

will display help information for the VIEW option stored in member TPIMAGE, starting from the line containing . / VIEW. Refer to member TPIMAGE in the help library for an example of how to subdivide a help member into subcommand sections. Help text for the major command must always be first. Text for subcommands follows in any order, each section beginning with a *./ subcommand* statement.

Help for messages

Help panels for MULTSESS messages are stored in members which have names corresponding to the messages being described. For example, when a user types:

```
HELP MS0044
```

the help text stored in member MS0044 of the MSGLIB library is displayed.

Changing messages and associated help text

If you want to amend a message, you must edit the message in the TPMSGS module and reassemble the module.

If you want to amend help text for a message you must edit the appropriate member in the MSGLIB library.

Using help members to display information

You can add new help members to enable users to display any new information relevant to your installation, for instance a table of application availability times or news about new applications that is too long to be fully described in a broadcast message. Refer to members TPUSER1 and TPUSER2 in the supplied help library for examples.

A new member can have any name which starts TP and does not clash with a MULTSESS command name. For example you could use TPAPPLS so that the user can display the information by typing HELP APPLS or pressing a function key set to issue this command.

Critical information may be displayed to all users at logon time by including the HELP command in a logon profile.

Index

V

Reader's Comment Form

If you find any discrepancy in the information contained in this publication, please complete this form and mail it to the address below.

The authors may use, or distribute, any of the information you supply in any way they consider appropriate without incurring any obligation whatsoever.

Publication number PMS0002.012 - Twelth Edition (September 2001)

Please write your comments below and on the following page and return this form to the

Documentation Manager
PassGo Technologies Ltd
Horton Manor
Ilminster, Somerset
TA19 9PY
England

Reader's Comment Form