MULTICS POCKET GUIDE
COMMANDS AND ACTIVE FUNCTIONS

SERIES 60 (LEVEL 68)

SOFTWARE

Honeywell
MULTICS POCKET GUIDE
COMMANDS AND ACTIVE FUNCTIONS

SERIES 60 (LEVEL 68)

SUBJECT:
Abbreviated Version of Multics Commands and Active Functions

SPECIAL INSTRUCTIONS:
This document is based on the contents of the Multics Programmers' Manual Commands and Active Functions, Order No. AG92.

SOFTWARE SUPPORTED:
Multics Software Release 3.1

DATE:
April 1976
This pocket guide presents an abbreviated version of the commands and active functions described in detail in the Multics Programmers' Manual Commands and Active Functions, Order No. AG92.

Users of this document should be familiar with some of the concepts and terminology of the Multics System. The following Multics user documentation should be consulted:

- Multics Users' Guide
- Multics Programmers' Manual:
  - Reference Guide
  - Commands and Active Functions
  - Subroutines
  - Subsystem Writers' Guide

For detailed information on Multics programming languages, refer to the following manuals:

- APL Users' Guide
- BASIC
- COBOL Reference Manual
- COBOL Users' Guide
- FORTRAN
- PL/I Language Manual

© 1976, Honeywell Information Systems Inc. File No.: 1L13
INTRODUCTION

This document is intended to serve as a quick reference and convenient memory aid for the user with some familiarity with Multics command conventions.

This guide presents an abbreviated description of the Multics commands and active functions described in detail in the Multics Programmers’ Manual Commands and Active Functions (MPM Commands), Order No. AG92. The commands are presented in alphabetical order, with each description showing the proper usage and a list of the control arguments and optional arguments that may be used. Control arguments are only listed; they are not defined in detail in this document.

The reader is expected to be familiar with the Multics command environment conventions. The following terms are defined here and are not explained at each occurrence in this document. If the reader needs more information on terminology than is given here, he should refer to “Glossary of Multics Terms” in Section I of the Multics Programmer’s Manual Reference Guide, Order No. AG91.

ACL access control list; it describes who may access an entry in the Multics storage system and in what way (see modes below).

modes access identifiers; used to define the kind of access a user has to a storage system entry. The modes are:

segments directories
r (read) s (status)
e (execute) m (modify)
w (write) a (append)

Null access can also be specified for either segments or directories: " ", n, or null.

path pathname of an entry; it can be relative or absolute.

Person_id user’s registered personal identifier; usually some form of the user’s surname; unique at site.

Project_id user’s registered project identifier; a project is an arbitrary set of users grouped together for accounting and access control purposes.

User_id access control name of the form Person_id.Project_id.tag; since the tag portion is rarely explicitly given, the term User_id is often defined as a Person_id.Project_id pair.
The format of each command description in this document is based on those found in the MPM Commands. The command name, in boldface type, is shown first, followed by the usage line. In the usage line, the following conventions apply:

1. If a command accepts more than one of a specific argument, an "s" is added to the argument name (e.g., paths, control_args).
2. To indicate one of a group of similar arguments, an "i" is added to the argument name (e.g., path_i, control_arg_i).
3. Multiple arguments that must be given in pairs are indicated by xxx_1 yyy_1 ... xxx_n yyy_n.
4. Pathnames that must be given in pairs are indicated by path_1 path_2 ... path_1n path_2n.
5. Optional arguments are enclosed in braces (e.g., {path}, {-control_args}). All other arguments are required.

To illustrate these conventions, consider the following usage line:

```
command {paths} {-control_args}
```

The lines below are just a few examples of valid invocations of this command:

```
command
command path
command path path -control_arg
command path -control_arg -control_arg
command -control_arg
command -control_arg -control_arg
```

For simplicity, when an argument takes a value other than a pathname (indicated by "path"), the value is indicated as follows:

```
XX character string
N number, decimal or octal
DT date-time character string in a form acceptable to the convert_date_to_binary subroutine described in the *Multics Programmers' Manual Subroutines*, Order No. AG93.
```
Access to the System  
(preaccess requests)  
  MAP  
  963  
  029  
  dial  
  enter  
  enterp  
  login  
  logout  

Creating and Editing Segments  
adjust_bit_count  
basic_system  
compare_ascii  
edm  
indent  
program_interrupt  
qedx  
runt  
runoff  
runoff_abs  
set_bit_count  
sort_seg  

Segment Manipulation  
adjust_bit_count  
archive  
compare  
copy  
create  
delete  
delete_force  
link  
move  
set_bit_count  
sort_seg  
truncate  
unlink  
vfile_adjust  

Directory Manipulation  
add_name  
create_dir  
delete_dir  
delete_name  
fs_chname  
link  
list  
rename  
safety_sw_off  
safety_sw_on  
status  
unlink  
vfile_status  

Access Control  
delete_acl  
delete_iacl_dir  
delete_iacl_seg  
list_acl  
list_iacl_dir  
list_iacl_seg  
set_acl  
set_iacl_dir  
set_iacl_seg  

Address Space Control  
add_search_rules  
change_default_wdir  
change_wdir  
delete_search_rules  
initiate  
list_ref_names  
nwproc  
print_default_wdir  
print_proc_auth  
print_search_rules  
print_wdir  
set_search_rules  
terminate  
terminate_refname  
terminate_segno  
terminate_single_refname  
where  

Formatted Output Facilities  
cancel_daemon_request  
dprint  
dpunch  
dump_segment  
list_daemon_requests  
print  
runt  
runtoff  
runtoff_abs  

Debugging and Performance Monitoring Facilities  
change_error_mode  
cumulative_page_trace  
dump  
display_pl11io_error  
dump_segment  
page_trace  
probe  
profile  
progress  
ready  
ready_off  
ready_on  
reprint_error  
trace  
trace_stack  

Language Translators, Compilers, Assemblers, and Interpreters  
apl  
basic  
basic_system  
bind  
cancel_cobol_program  
cobol  
display_cobol_run_unit  
format_cobol_source  
fortran  
fortran_abs  
indent  
p11  
p11_abs  
profile  
qedx  
runcobol  
runtoff  
runtoff_abs  
set_cc  
stop_cobol_run  

Input/Output System Control  
assign_resource  
cancel_daemon_request  
close_file  
console_output  
copy_cards  
display_pl11io_error  
dprint  
dpunch  
file_output  
io_call  
line_length  
list_daemon_requests  
list_resources  
print  
print_attach_table  
print_request_types  
set_cc  
set_tty  
unassign_resource  
vfile_adjust  
vfile_status
abbrev, ab

provides the user with a mechanism for abbreviating parts of (or whole) command lines in the normal command environment.

Usage: abbrev

CONTROL REQUESTS

.a <abbr> <rest of line>
  add the abbreviation <abbr> to the current profile segment.

.ab <abbr> <rest of line>
  add an abbreviation that is expanded only if found at the beginning of a line or directly following a semicolon (;) in the expanded line.

.af <abbr> <rest of line>
  add an abbreviation to the profile segment and force it to overwrite any previous abbreviation with the same name.

.abf <abbr> <rest of line>
  add an abbreviation that is expanded only at the beginning of a line and force it to replace any previous abbreviation with the same name.

.d <abbr1> ... <abbrn>
  delete the specified abbreviations from the current profile.

.f
  enter a mode (the default mode) that forgets each command line after executing it.

.l <abbr1> ... <abbrn>
  list the specified abbreviations with the things they stand for.

.la <letter1> ... <lettern>
  list all abbreviations starting with the specified letters.

.q
  quit using the abbrev processor.

.r
  enter a mode that remembers the last line expanded by abbrev.

.s <rest of line>
  show the user how <rest of line> would be expanded but do not execute it.
.u <profile>
specify to abbrev the pathname of a profile segment to use.

.P
print the name of the profile segment being used.
.<space> <rest of line>
pass <rest of line> on to the current command processor without expanding it.

BREAK CHARACTERS
Break characters (any combinations) must be used to delimit abbreviations in a command line.

<table>
<thead>
<tr>
<th>tab</th>
<th>semicolon</th>
<th>;</th>
</tr>
</thead>
<tbody>
<tr>
<td>newline</td>
<td>vertical bar</td>
<td></td>
</tr>
<tr>
<td>space</td>
<td>parentheses</td>
<td>( )</td>
</tr>
<tr>
<td>quote</td>
<td>less than</td>
<td>&lt;</td>
</tr>
<tr>
<td>dollar sign $</td>
<td>greater than</td>
<td>&gt;</td>
</tr>
<tr>
<td>apostrophe '</td>
<td>brackets</td>
<td>[ ]</td>
</tr>
<tr>
<td>grave accent \</td>
<td>braces</td>
<td>{ }</td>
</tr>
</tbody>
</table>

accept_messages, am
initializes or reinitializes the user's process for accepting messages sent by the send_message command.

Usage: am {-control_args}
    - brief, -bf
    - long
    - print
    - short

add_name, an
adds an alternate name to the existing name(s) of an entry.

Usage: an path names
    names
    additional names to be added to the entry.

add_search rules, asr
allows the user to change his search rules dynamically.

Usage: asr path1  {-control_arg path2} ..
    . path1 { -control_arg path2 } path1
    pathname of a directory to the current search rules (certain keywords may also be used).

adjust_bit_count, abc
sets the bit count of segments that for some reason do not have the bit count set properly.

Usage: adjust_bit_count paths {-control_args}
    - character, -ch
    - long, -lg

answer
provides a preset answer to a question asked by another command.

Usage: answer ans {-control_args} command_line
    ans
desired answer to any question.
    command_line
    any Multics command line.
    control_args
    - brief, -bf
    - times N

apl
invokes the Multics APL interpreter.

Usage: apl

archive, ac
combines an arbitrary number of separate segments into one single segment.

Usage: archive key path components
    components
    components of the archive segment.
    key
    listed below by function.
    Table of Contents Operations:
    t print the entire table of contents if no components are named by the path arguments.
Assign Operations:
- append named components to the archive segment.
- append and delete.
- append and deleteforce.
- copy and append.
- copy, append, and delete.
- copy, append, and deleteforce.

Replace Operations:
- replace components in, or add components to the archive segment.
- replace and delete.
- replace and deleteforce.
- copy and replace.
- copy, replace, and delete.
- copy, replace, and deleteforce.

Update Operations:
- update.
- update and delete.
- update and deleteforce.
- copy and update.
- copy, update, and delete.
- copy, update, and deleteforce.

Delete Operations:
- delete from the archive those components named by the path arguments.
- copy and delete.

Extract Operations:
- extract from the archive those components named by the path arguments, placing them in segments in the storage system.
- extract and deleteforce.

Assign_resource, ar
- calls the resource control package (RCP) to assign a resource to the caller's process.

Usage: assign_resource type { -control_args }
   type     punch
disk       reader
console    special
printer
control_args
- comment XX, -com XX
- density N, -den N
- device XX, -dv XX
- line_length N, -ll N
- long, -lg
- model N
- system, -sys
- track N, -tk N
- train N, -tn N
- volume XX, -vol XX

Basic
- invokes the BASIC compiler.
Usage: basic path { -control_arg }
   type     compile
- compile
- ti.ne N

Basic_system, bs
- standard BASIC source editor and run dispatcher.
Usage: basic_system { path }

Requests
- delete all
  or
- delete first { last }
  deletes the specified lines.

Exec command_line
- passes the command_line argument to the Multics command processor.

Get { path }
- clears the internal buffers so that the user can work on a different program.

Line_number
- deletes that source line if such a line number exists.

Line_number source_line
- adds or replaces a BASIC source line (source_line) in proper sequence.

List
- prints the entire current internal segment.
quit
exits from basic_system and returns to command level.

rseq {first} {increment}
resequences the line numbers so that they differ by
a fixed increment.

run
calls the BASIC compiler to run the current
internal source segment.

save {path}
stores the current internal source segment in the
segment whose pathname is specified by path.

time N
establishes a time limit of N CPU seconds on the
execution of the program.

bind, bd
produces a single bound object segment from one or
more unbound object segments.

Usage: bind paths {—control_arg}
—list, —ls
—map
—update paths, —ud paths

calc
provides the user with a calculator.

Usage: calc

REQUESTS
< expression > type value of
expression.
< variable > = < expression > assign value of ex-
pression to variable.
list
q

EXPRESSIONS
order of evaluation
1. expression within parentheses
2. function references

3. prefix +, prefix —
4. * *
5. +, /
6. + , —

FUNCTIONS
sin, cos, tan, atan, abs, ln, log

cancel_abs_request, car
allows a user to delete a request for an absentee
computation.

Usage: cancel_abs_request path {—control_args}
—all, —a
—brief, —bf
—queue N, —q N

cancel_cobol_program, ccp
cancels one or more programs in the current COBOL
run unit.

Usage: cancel_cobol_program names {—control_arg}
control_arg
—retain_data, —retd
names
name specified in the PROG-ID statement.

cancel_daemon_request,cdr
cancels a dprint or dpunch request.

Usage: cancel_daemon_request path {—control_args}
—all, —a
—brief, —bf
—queue N, —q N
—request_type XX, —rqt XX

change_default_wdir, cdwd
sets a specified directory as the user's default working
directory for the duration of the current process or
until the next change_default_wdir command is
issued.

Usage: change_default_wdir {path}
change_error_mode, cern
controls the amount of information printed by the default handler for system conditions.
Usage: change_error_mode \{-control_args\}
   -brief, -bf
   -long, -lg

change_wdir
changes the user's working directory to the directory specified as an argument.
Usage: change_wdir \{path\}

check_info_segs, cis
prints a list of new or modified segments.
Usage: check_info_segs \{-control_args\}
   -brief, -bf
   -call command_line
   -date DT, -dt DT
   -long, -lg
   -no_update, -nud
   -pathname star_name_path, -pn star_name_path

close_file, cf
 closes specified FORTRAN and PL/I files.
Usage: close_file \{-control_arg\} filenames
   control_arg
   -all
   filenames
   names of the open files.

cobol
invokes the COBOL compiler.
Usage: cobol path \{-control_args\}
   -brief, -bf
   -check, -ck
   -debug, -db
   -format, -fmt
   -list, -ls
   -map
   -severityN, -svN
   -source, -sc
   -symbols, -sb
   -table, -tb
   -time, -tm
   -tn

compare
compares two segments and lists their differences.
Usage: compare path1 \{offset1\} path2 \{offset2\}
   \{-control_args\}
   path1, path2
   pathnames of segments to be compared.
   offset1, offset2
   octal offsets within the segments to be compared.
   control_args
   -length N, -ln N
   -mask N

compare_ascii, cpa
compares two ASCII segments and prints the changes made to the segment specified by path1 to yield the segment path2.
Usage: compare_ascii path1 path2 \{minchars\}
   \{minlines\}
   path1, path2
   pathnames of segments to be compared.
   minchars
   decimal number specifying the minimum number of characters that must be identical before the segments are again assumed to be "in sync" after a difference in the two segments.
   minlines
   decimal number specifying the minimum number of lines that must be identical.

cconsole_output, co
directs the user_output to the terminal. (See file_output.)
Usage: console_output

copy, cp
creates copies of specified segments and/or multisection files in the specified directories with the specified names.
Usage: copy path1{\{path2\}}... path1n{\{path2n\}}
   \{-control_args\}
path1
pathname of segment to be copied.

path2
pathname of a copy to be created from path1.

copy_cards
copies specified card image segments from system pool storage into a user's directory.

Usage: copy_cards deck_name {path}

deck_name
name entered on deck_id card.

create, cr
creates a storage system segment in a specified directory (or in the working directory).

Usage: create paths

create_dir, cd
creates a specified storage system directory branch in a specified directory (or in the working directory).

Usage: create_dir paths {--control_args}

--access_class XX, --acc XX
--quota N

cumulative_page_trace, cpt
accumulates page trace data so that the total set of pages used during the invocation of a command or subsystem can be determined.

Usage: cumulative_page_trace command_line

{--control_args}

command_line
character string to be interpreted as a command line.

copy_cards
--count, --ct
--reset, --rs

debug, db
interactive debugging aid to be used in the Multics environment.

Usage: debug

DATA REQUESTS
Format of Data Request
<generalized address> <operator> <operands>

Generalized Address
[/segment name/] [offset] [segment ID]
[relative offset]

/seg name/ offset segment ID
pathname number &t text
ref name symbol &s stack
seg number symbol &l linkage
&n seg name &a source line
seg$entry &p parameter
&i internal static

rel offset number register

Operators
print operands
assign input list
< set break function
> transfer list
:= call

Output Modes
o octal
h half-carriage octal
d decimal
a ASCII
i instruction
p pointer
s source statement
l code for line number
n no output
c floating point
f floating point
b bit string
g graphic
CONTROL REQUESTS

.decode
reconstructs an original segment from an enciphered segment according to a key that need not be stored in the system. (See encode.)

Usage: decode path1 { path2 }

path1
pathname of enciphered segment.

path2
pathname of deciphered segment to be produced.

.defer_messages, dm
suspends printing of messages sent by the send_message command on the user's terminal.

Usage: defer_messages

.delete, dl
deletes the specified segments and/or multisection files.

Usage: delete paths

.delete_acl, da
removes entries from the ACLs of segments, multisection files, and directories.

Usage: delete_acl { path } { User_ids } {--control_args}
- all, -a
- brief, -bf
- directory, -dr
- segment, -sm

.delete_dir, dd
deletes the specified directories (and any segments, links, and multisegment files they contain).

Usage: delete_dir paths

.delete_force, df
deletes the specified segments or multisegment files, regardless of whether or not the safety switch is on.

Usage: delete_force paths

.decode
reconstructs an original segment from an enciphered segment according to a key that need not be stored in the system. (See encode.)

Usage: decode path1 { path2 }

path1
pathname of enciphered segment.

path2
pathname of deciphered segment to be produced.

.defer_messages, dm
suspends printing of messages sent by the send_message command on the user's terminal.

Usage: defer_messages

.delete, dl
deletes the specified segments and/or multisection files.

Usage: delete paths

.delete_acl, da
removes entries from the ACLs of segments, multisection files, and directories.

Usage: delete_acl { path } { User_ids } {--control_args}
- all, -a
- brief, -bf
- directory, -dr
- segment, -sm

.delete_dir, dd
deletes the specified directories (and any segments, links, and multisegment files they contain).

Usage: delete_dir paths

.delete_force, df
deletes the specified segments or multisegment files, regardless of whether or not the safety switch is on.

Usage: delete_force paths

.decode
reconstructs an original segment from an enciphered segment according to a key that need not be stored in the system. (See encode.)

Usage: decode path1 { path2 }

path1
pathname of enciphered segment.

path2
pathname of deciphered segment to be produced.

.defer_messages, dm
suspends printing of messages sent by the send_message command on the user's terminal.

Usage: defer_messages

.delete, dl
deletes the specified segments and/or multisection files.

Usage: delete paths

.delete_acl, da
removes entries from the ACLs of segments, multisection files, and directories.

Usage: delete_acl { path } { User_ids } {--control_args}
- all, -a
- brief, -bf
- directory, -dr
- segment, -sm

.delete_dir, dd
deletes the specified directories (and any segments, links, and multisegment files they contain).

Usage: delete_dir paths

.delete_force, df
deletes the specified segments or multisegment files, regardless of whether or not the safety switch is on.

Usage: delete_force paths
delete_iacl_dir, did
  deletes entries from a directory initial ACL in a specified directory.
Usage: delete_iacl_dir {path} {User_ids}
  {–control_args}
    –brief, –bf
    –ring N, –rg N

delete_iacl_seg, dis
  deletes entries from a segment initial ACL in a specified directory.
Usage: delete_iacl_seg {path} {User_ids}
  {–control_args}
    –brief, –bf
    –ring N, –rg N

delete_name, dn
  deletes specified names from entries that have multiple names.
Usage: delete_name paths

delete_search_rules, dsr
  allows the user to delete current search rules.
Usage: delete_search_rules paths

dial, d
  connects an additional terminal to an existing process.
Usage: dial dial_id Person_id.Project_id
  dial_id
  keyword that uniquely specifies a logged-in process that is accepting dial connections.
  Person_id.Project_id
  the Person_id and Project_id of the process the user wishes to connect to.

display_cobol_run_unit, dcr
  displays the current state of a COBOL run unit.
Usage: display_cobol_run_unit {–control_args}
  –all, –a
  –files
  –long, –lg

display_pl1io_error, dpe
  describes the most recent file on which a PL/I I/O error was raised and displays diagnostic information associated with that type of error.
Usage: display_pl1io_error

do
  expands a command line according to the arguments supplied following the command string.
Usage: do “command_string” {–control_args}
  command_string
  a command line in quotes.
  control_args
  a character string argument to replace a parameter designated by &1 in command_string.
  modes
    –absentee
    –brief, –bf
    –go
    –interactive
    –long, –lg
    –nogo

dprint, dp
  queues specified segments and/or multisegment files for printing on the line printer.
Usage: dprint {–control_args} {paths}
  –access_label, –albl
  –bottom_label XX,
  –bottom_label XX, –blbl XX
  –brief, –bf
  –copy N, –cp N
  –delete, –dl
  –delete_type XX
  –destination XX, –ds XX
  –header XX, –he XX
  –indent N, –in N
  –label XX, –lbl XX
  –line_length N, –ll N
  –no_endpage, –nep
  –no_label, –nlbl
  –pl N
  –queue N, –q N
  –request_type XX,
  –req N
  –single, –sg
  –top_label XX,
  –tbl N
  –truncate, –tc
dpunch, dpn

queues specified segments and/or multisegment files for punching by the card punch.

Usage: dpunch {-control_args} {paths}
   -brief, -bf
   -copy N, -cp N
   -delete, -dl
   -destination XX, -ds XX
   -header XX, -he XX
   -mcc

queues specified segments and/or multisegment files for punching by the card punch.

dump_segment, ds

prints, in octal format, selected portions of a segment.

Usage: dump_segment path {first} {num}
   {-control_arg}

edm

invokes a simple Multics context editor.

Usage: edm {path}

REQUESTS

. enter input mode; exit when a line with only "." is typed.
   -N back up N lines,
   ' enter "comment" mode; exit when a line with only "." is typed.
   = print current line number.
   b go to bottom of file, enter input mode.
   c N/s1/s2/ change all occurrences of string "s1" to "s2" for N lines.

d N deletes N lines.
updelete delete all lines above current line.
E line execute "line" as a Multics command line.
f string find a line beginning with "string".
i line insert "line" after current line.
merge path insert segment "path" after current line.
move M N beginning with line M, remove N lines and insert them after the current line.
k enter brief mode (no response after f, n, l, c, and s requests).
l string locate a line containing "string".
N move down N lines.
p N print N lines.
q exit from edm.
qf exit directly from edm with no question.
r line replace current line with "line".
s N/s1/s2/ same as "c".
t go to top of file.
v enter verbose mode (opposite of k request).
w path write edited copy of file into "path".
upwrite path write all lines above current line into "path".

encode

enciphers a segment's contents according to a key that need not be stored in the system. (See decode.)

Usage: encode path1 {path2}
   path1 pathname of segment to be enciphered.
   path2 pathname of the enciphered segment to be produced.

enter, e
enterp, ep

used by anonymous users to gain access to Multics.

Usage: enter {anonymous_name} Project_id
   {-control_args}
anonymous_name

treated like person identifier.

Project_id

identification of the user’s project.

control_args

- brief, -bf
- force
- home_dir path, -hd path
- no_preempt, -np
- no_print_off, -npf
- no_start-up, -ns
- print_off, -pf
- process_overseer path, -po path

enter_abs_request, ear

requests that an absentee process be created.

Usage:
enter_abs_request path { -control_args }
- arguments XX, -ag XX
- brief, -bf
- limit N, li N
- output_file path, -of path
- queue N,
- restart, -rt
- time DT, -tm DT

exec_com, ec

executes a series of command lines contained in a segment.

Usage:
exec_com path { optional_args }

optional_args

character strings substituted for &i in the exec_com segment.

Each &i (where i is an integer) in the exec_com segment is replaced by the corresponding argument to the exec_com command; &ec name is replaced by the entryname portion of the exec_com pathname without the ec suffix; &0 is replaced by the path argument to the exec_com command.

CONTROL STATEMENTS

&label location

identifies location.
&goto location

transfers control to &label specified.

&attach

attaches user_input to exec_com segment.
&detach

detaches user_input.
&input_line on

writes input lines on user_output.
&input_line off

does not write out input lines.
&command_line on

writes command lines on user_output prior to execution.
&command_line off

does not write out command lines.
&ready_on

invokes ready message after execution of each command line.
&ready_off

turns off ready message; default.
&print char_string

prints char_string on user_output.
&qquit

returns exec_com to caller.
&if [ACTIVE_FUNCTION -argi ... -argn -]

executes & then clause if ACTIVE_FUNCTION returns "true"; executes &else clause if ACTIVE_FUNCTION returns "false"; otherwise error. Each argi can also be an active function.
&then THEN_CLAUSE

can include a command line, input line, null statement, and most control statements.
&else ELSE_CLAUSE

can include a command line, input line, null statement, and most control statements.

file_output, fo

directs the user_output to a segment. (See console_output.)

Usage:
file_output { path }

format_cobol_source, fcs

converts pseudo free-form COBOL source programs to the standard fixed-format COBOL source programs.

Usage:
format_cobol_source path1 path2
path1
pathname of input segment containing pseudo free-form COBOL source code.

path2
pathname of output segment containing converted fixed-format COBOL source.

**fortran, ft**
invokes the FORTRAN compiler.

**Usage:**
```plaintext
fortran path { --control_args }
-brief, -bf  -optimize, -ot
-brief_table, -profile, -pf
-bftb     -severityN, -svN
-card     -source, -sc
-check, -ck -subscriptrange, -subrg
-convert   -symbols, -sb
-debug, -db -table, -tb
-list, -ls  -time, -tm
-map
```

**fortran_abs, fa**
submits an absentee request to perform FORTRAN compilations.

**Usage:**
```plaintext
fortran_abs paths { --ft_args } { --dp_args }
{ --abs_control_args }

ft_args
control arguments accepted by the fortran command.

dp_args
control arguments (except -delete) accepted by the dprint command.

abs_control_args
-queue N, -q N
-hold
-output_file path, -of path
```

**fs_chname**
manipulates strangely named segments because none of the special command system symbols (e.g., *, >) are interpreted. When oldname and newname are not null strings, fs_chname is equivalent to using the rename command; null string for oldname is equivalent to using the add_name command; null string for a newname is equivalent to using the delete_name command.

**Usage:**
```plaintext
fs_chname dir_name entryname oldname
newname

dir_name
directory name portion of the segment.
entryname
entryname portion of the segment.
oldname
old entryname to be deleted.
newname
new entryname to be added.
```

**gcros, gc**
invokes the GCOS environment simulator to run a single GCOS job, immediately, in the user’s process.

**Usage:**
```plaintext
gcos job_deck_path { --control_args }

job_deck_path
pathname of segment containing a GCOS job deck.

control_args
listed below by function.
```

**Input Specifications:**
```plaintext
-ascii, -aci
-no_canonicalize, -nocan, -no_gcos, -gc
-truncate, -tc
```

**Output Specifications:**
```plaintext
-dprint, -dp
-hold, -hd
-dprint_options "options", -list, -ls
-dpo "options", -lower_case, -lc
-dpunch, -dpn
-raw
-dpunch_options "options", -dpno "options"
```

**Creation of Files:**
```plaintext
-brief, -bf
-continue, -ctu
-debug, -db
-job_id id, -id id
-long, -lg
```

(no content continues on the page)
**gecos_card_utility, gcu**

Copies GCOS card image files, altering their format, content, and medium, as specified by the user.

**Usage:**

gcu input_specification output_specification

input_specification, output_specification

pathnames (or tape numbers) and control arguments. Control arguments are listed below by function.

**Input and Output Specifications:**

- input, -in
- output, -out

**File Formats:**

- ascii, -aci
- comdk, -cdk
- gcos_ascii, -gca
- gcos, -gc

**File Contents:**

- imcv XX
- library XX, -lib XX

**Tape Files:**

- attached, -att
- detach, -det
- label XX, -lbl XX
- tape N

**Partial Copying:**

- count N, -ct N
- first N, -ft N
- last N, -lt N

**Output File Duplication:**

- append, -app

**Input and Output Lists:**

- all
- file_input path, -fi path
- file path

**Terminal Output:**

- brief, -bf
- debug, -db
- long, -lg

---

**gecos_sysprint, gsp**

Converts a print file (either SYSOUT or simulated printer) produced by the GCOS environment simulator, from BCD to ASCII.

**Usage:**

gcos_sysprint input_path \{ output_path \}

\{--control_args\}

**Output Path**

pathname into which the ASCII output lines are written.

**control_args**

- lower_case, -lc
- temp_dir path, -td path

---

**gecos_syspunch, gspn**

Converts a GCOS standard system format file, containing BCD and binary card images, to a format suitable for punching using the Multics dpunch command with the -raw argument.

**Usage:**

gcos_syspunch path

---

**get_com_line, gcl**

Prints on the user’s terminal the maximum length allowed for an expanded command line.

**Usage:**

get_com_line

---

**get_quota, gq**

Returns information about the secondary storage quota and pages used for a specified directory.

**Usage:**

get_quota paths \{--control_arg\}

- long, -lg

---

**help**

Assists users in obtaining online information about such things as commands, subsystems, system status, or changes.

**Usage:**

help \{ name \} \{--control_args\}
name
the name of an info segment which refers to a
command or general topic.

control_args
- header, - he
- pathname path, - pn path
- search XX, - sh XX
- section XX, - sc XX
- title

REQUESTS
Each info segment is divided into paragraphs delimited
by double blank lines. After each paragraph, help asks
"More help?" The user may reply:

yes print next block
no print no more from this segment
rest rest of this segment
skip skip next block and proceed
title print remaining section titles, no questions
quit exit from help
search \{XX\}, sh \{XX\} search forward for string XX
section \{XX\}, sc \{XX\} find section named XX

how_many_users, hmu
prints how many users are currently logged in.
Usage: how_many_users {args} {-control_args}
control_args
- absentee, - as
- brief, - bf
- long, - lg
args
Person_id
.Project_id
Person_id.Project_id

immediate_messages, im
restores the immediate printing of messages sent by
the send_message command.
Usage: immediate_messages

indent, ind
improves the readability of a PL/I source segment by
indenting it according to a set of standard conventions.
Usage: indent oldpath \{newpath\} {-control_args}
oldpath pathname of input PL/I source segment.
newpath pathname of output PL/I source segment.
control_args
- brief, - bf
- comment N, - cm N
- indent N, - in N
- lmargin N, - lm N

initiate, in
enables users to initiate segments directly, i.e., not
using the normal search rules.
Usage: initiate path \{ref_names\} {-control_arg}
ref_names reference names for the segment.
control_arg
- long, - lg

io_call, io
performs an operation on a designated I/O switch.
Usage: io attach switchname modulename \{args\}
Usage: io detach switchname
Usage: io open switchname mode
Usage: io close switchname
Usage: io get_line switchname \{n\} {-control_args}
Usage: io get_chars switchname \{n\} {-control_args}
Usage: io put_chars switchname \{string\} {-control_args}
Usage: io rewrite_record switchname \{string\} {-control_args}
Usage: io seek_key switchname
Usage: io position switchname type
Usage: io delete_record switchname
Usage: io read_record switchname \{n\} {-control_args}
Usage: io write_record switchname \{string\} {-control_args}
Usage: io rewrite_record switchname \{string\} {-control_args}
Usage: io delete_record switchname
Usage: io position switchname type
Usage: io seek_key switchname key
Usage: io read_key switchname

Usage: io read_length switchname

Usage: io control switchname order

Usage: io modes switchname {string} {--brief}

Usage: io find_iocb switchname

Usage: io look_iocb switchname

Usage: io move_attach switchname switchname2

Usage: io destroy_iocb switchname

Usage: io print_iocb switchname

switchname

name of the I/O switch.

module name

name of I/O module used in attachment.

args

any arguments accepted by the I/O module used in attachment.

mode

stream_input, si
stream_output, so
stream_input_output, sio
sequential_input, sqi
sequential_output, sqo
sequential_input_output, sqio
sequential_update, squ

n

decimal number.

string

any character string.

type

bof, -1 set to beginning of file.
eof, 1 set to end of file.
f N set forward N records or lines.
r N set back N records.
other

key

string of ASCII characters with 0 ≤ length ≤ 256.

order

one of the orders accepted by the I/O module used in the attachment of the I/O switch.

control_args

--brief, -bf
--lines
--nl
--nnl
--segment path {offset}, --sm path {offset}
--segment path {length}, --sm path {length}
--segment path {offset} {length},
--sm path {offset} {length}

line_length, ll

sets the maximum length of a line output to the device that a process is connected to through the user_output I/O switch.

Usage: line_length max_length

max_length maximum length of line.

link, lk

creates a storage system link with a specified name in a specified directory pointing to a specified segment or directory.

Usage: link path1 path2 ... path1n {path2n}

path1i

pathname of the segment to which path2i is to point.

path2i

pathname of the link to be created.

list, ls

prints information about entries contained in a single directory.

Usage: list {entrynames} {--control_args}

entrynames names of entries to be listed.

control_args listed below by function.

Directory:

--pathname path, --pn path
list_acl, la
lists the ACLs of segments, multisegment files, and directories.
Usage:
```
list_acl {path} {User_ids} {-control_args}
-brief, -bf
-directory, -dr
-ring_brackets, -rb
-segment, -sm
```

list_daemon_requests, ldr
prints information about dprint and dpunch requests.
Usage:
```
list_daemon_requests {-control_args}
-all, -a
-long, -lg
-queue N, -q N
-request_type XX, -rqt XX
-total, -tt
```

list_iacl_dir, lid
lists some or all of the entries on a directory initial ACL in a specified directory.
Usage:
```
list_iacl_dir {path} {User_ids} {-control_args}
-brief, -bf
-ring N, -rg N
```

list_iacl_seg, lis
lists entries on a segment initial ACL in a specified directory.
Usage:
```
list_iacl_seg {path} {User_ids} {-control_args}
-brief, -bf
-ring N, -rg N
```

list_ref_names, lrn
lists the absolute pathname and reference names associated with a segment.
Usage:
```
list_ref_names paths {-control_args}
-all, -a
-brief, -bf
-from N -to N
```

list_abs_requests, lar
prints information about absentee requests.
Usage:
```
list_abs_requests {-control_args}
-all, -a
-long, -lg
-queue N, -q N
-total, -tt
```

Entry Type:
- all, -a
- branch, -br
- directory, -dr
- file, -f
- link, -lk
- multisegment_file, -msf
- segment, -sm

Columns:
- count, -ct
- date_time_contents_modified, -dtcm
- date_time_entry_modified, -dtem
- date_time_used, -dtu
- length, -ln
- link_path, -lp
- mode, -md
- name, -nm
- record, -rec

Totals/Header Lines:
- no_header, -nhe
- total, -tt

Multiple-Name Entries:
- match
- primary, -pri

Entry Order:
- reverse, -rv
- sort XX, -srt XX

Entry Exclusion:
- exclude entryname, -ex entryname
- first N, -ft N
- from DT, -fm DT
- to DT

Output Format:
- brief, -bf
- short, -sh
list_resources, lr
lists resources that are assigned or attached to the calling process by the resource control package (RCP).

Usage: list_resources {--control_args}
  --assignments, -asm
  --attachments, -atm
  --device XX, -dv XX
  --long, -lg
  --type XX, -tp XX

login, l
used to gain access to the system.

Usage: login Person_id {Project_id} {--control_args}
Person_id
user's personal identifier.
Project_id
identification of the user's project.

control_args
  --authorization XX, -auth XX
  --brief, -of
  --change_de-fault_auth, -cda
  --change_de-fault_project, -cdp
  --change_password, -cpw
  --force
  --generate_password, -gpw
  --home_dir path, -hd path
  --modes XX

logout
terminates a user session.

Usage: logout {--control_args}
  --brief, -bf
  --hold

mail, ml
sends a message to another user or prints messages in a mailbox.

Usage: (sending)
  ml path Person_id Project_id ...
  { Person_id } { Project_id }

  (printing)
  ml { path } {--control_arg}
    --brief, -bf

SENDING
If path is *, mail responds with "Input:" and accepts lines from the terminal until a line consisting of a period (.) is typed.

PRINTING
If no path argument is given, the contents of the default mailbox are printed.

CREATING A MAILBOX
A default mailbox is created automatically the first time a user types "mail"; the default mailbox is:
>user_dir_dir>Project_id>Person_id>Person_id.mbx

MAP
tells system user has terminal that generates only uppercase characters; system then maps each typed character to lowercase unless it is preceded by a backslash (\).

Usage: MAP

memo
maintains an interactive notebook and reminder list.

Usage: memo {--control_arg} {optional_args}
  { memo_text }

  control_args
    --brief, -bf
    --delete, -dl
    --list, -ls
    --off
    --on
    --pathname path, -pn path
    --print, -pr
optional_args
  memo_number
  -alarm, -al
  -call
  -date DT, -dt DT
  -invisible, -iv
  -match XX
  -repeat intvl, -rp intvl
  -time DT, -tm DT
memo_text
text of memo being set.

move
moves a designated segment or multisegment file (and its ACL and all names on the designated file) to a new position in the storage system hierarchy.

Usage:
move path1 \{path2\} .. path{n} \{-control_arg\}
  \{\-control_arg\}
path1j
  pathname of segment to be moved.
path2j
  pathname to which path1j is to be moved.
control_arg
  -brief, -bf

move_quota, mq
moves storage quota between two directories, one immediately inferior to (contained in) the other.

Usage:
mov
  e_quota path1 quota change1 .. pathn quota changep
  \{-control_arg\}
path1j
  pathname of directory.
quota_changej
  number of records to be moved between the containing directory quota and the path1j quota.

new_proc
destroy the user's current process and creates a new one, using the control arguments given initially with the login command, and the optional control argument to the new_proc command itself.

Usage:
new_proc \{-control_arg\}
  \{-control_arg\}
  -authorization XX, -auth XX

page_trace, pgt
prints a recent history of page faults and other system events within the calling process.

Usage:
page_trace \{\count\} \{-control_arg\}
  \count
  prints the last count of system events recorded for the calling process.
  \{\control_arg\}
control_arg
  -long, -lg

pll
invokes the PL/I compiler.

Usage:
pll path \{-control_args\}
  \{-control_args\}
  -brief, -bf
  -brief_table, -bftb
  -check, -ck
  -debug, -db
  -list, -ls
  -map
  -optimize, -ot

pll_abs, pa
submits an absentee request to perform PL/I compilations.

Usage:
pll_abs paths \{-pll_args\} \{-dp_args\}
  \{-abs_control_args\}
  \{-pll_args\}
pll_abs
  control arguments accepted by the pll command.
dp_args
  control arguments (except -delete) accepted by the dprint command.
abs_control_args
  -queue N, -q N
  -hold
  -output_file path, -of path

print, pr
prints a specified ASCII segment on the user's terminal.

Usage:
print path \{begin\} \{end\}
print_attach_table, pat
prints a list of attached I/O switches, their attach
descriptions, and opening mode.
Usage:  print_attach_table { switch_names }
switch_names
names of I/O switches.

print_auth_names, pan
prints the names of the sensitivity levels and access
categories defined for the installation.
Usage:  print_auth_names { -control_args }
-all, -a
-brief, -bf
-category, -cat
-level

print_default_wdir, pdwd
prints out the pathname of the current default work-
ing directory.
Usage:  print_default_wdir

print_messages, pm
prints any interprocess messages that were received
(and saved in the user's mailbox) while the user was
not accepting messages.
Usage:  print_messages

print_motd, pmotd
prints out changes to the message of the day since the
last time the command was called.
Usage:  print_motd

print_proc_auth, ppa
prints the access authorization of the current process
and current system privileges (if any).
Usage:  print_proc_auth { -control_args }
    -all, -a
    -long, -lg

print_request_types, prt
prints a list of all request types handled by the I/O
daemon.
Usage:  print_request_types { -control_args }
    -access_name XX, -an XX
    -brief, -bf
    -gen_type XX, -gt XX

print_search_rules, psr
prints the search rules currently in use.
Usage:  print_search_rules

print_wdir, pwd
prints the pathname of the current working
directory.
Usage:  print_wdir

probe, pb
provides symbolic, interactive debugging facilities for
programs compiled with PL/I, FORTRAN, or COBOL.
The program to be debugged must be compiled with
the -table control argument.
Usage:  probe { procedure_name }
procedure_name
the symbolic name of the form,
reference_name$offset_name, of an entry to a
procedure or subroutine.

REQUESTS
after a       Set a break after a statement.
before b       Set a break before a statement.
call c         Call an external procedure.
continue c      Return from probe.
execute e       Execute a Multics command.
profile

prints information about the execution of each statement in PL/I or FORTRAN programs. The -profile control argument must have been used when the program was compiled.

Usage:

```
profile paths {--control_args}
```

control_args

-brief, -bf
-cput N
-off
-on
-output_switch XX, -os XX
-realt N

command_line
character string created by concatenating all the arguments to progress (excluding the first if it is a control argument) with blanks between them. The string is executed as a command line.

profile

prints information about the execution of each statement in PL/I or FORTRAN programs. The -profile control argument must have been used when the program was compiled.

Usage:

```
profile paths {--control_args}
```

control_args

-brief, -bf
-cput N
-off
-on
-output_switch XX, -os XX
-realt N

command_line
character string created by concatenating all the arguments to progress (excluding the first if it is a control argument) with blanks between them. The string is executed as a command line.

progress, pg

executes a specified command line and prints information about how its execution is progressing in terms of CPU time, real time, and page faults.

Usage:

```
progress {--control_arg} { command_line }
```

control_arg

-brief, -bf
-cput N
-off
-on
-output_switch XX, -os XX
-realt N

command_line
character string created by concatenating all the arguments to progress (excluding the first if it is a control argument) with blanks between them. The string is executed as a command line.

qedx, qx

c context editor used to create and edit ASCII segments in Multics.

Usage:

```
qedx {path} {optional_args}
```

path
pathname of an ASCII segment from which the editor takes its initial instructions.

optional_args
appended, each as a separate line, to the buffer named args.

REQUESTS

Listed below in four categories giving: format, default in parentheses, and brief description. For value of ADR, see "Addressing" below; regexp, see "Regular Expression."

INPUT REQUESTS

These requests enter input mode and must be terminated with \f.

ADRa (.a)
append lines after specified line.

ADR1,ADR2c (...c)
change existing line(s); delete and replace.

ADR1 (.i)
insert lines before specified line.
BASIC EDIT REQUESTS
ADR1,ADR2d (...d) delete line(s).
ADR1,ADR2p (...p) print line(s).
ADR= (=) print line number.
q exit from qedx editor.
ADRr path ($r path) append contents of path after specified line.
ADR1,ADR2s/regexp/string (...) substitute every regexp in the line(s) with string. If string contains &, & is replaced by regexp. First character after s is delimiter; it can be any character not in either regexp or string.
ADR1,ADR2w {path;}(1,$w) write lines into segment names path; if path omitted, default pathname used.
/reexp/

EXTENDED EDIT REQUESTS
e <command line> execute command line without leaving editor.
ADR1,ADR2gX/regexp/(/,$gX/regexp/) perform operation on lines that contain regexp; X must be d for delete, p for print, or = for print line numbers.
ADR1,ADR2vX/regexp/(/,$vX/regexp/) perform operation on lines that do not contain regexp; X must be d for delete, p for print, or = for print line numbers.
/x give the status of all buffers in use.
ADRn (.n) set the value of "." to line addressed.
ADR" ("") ignore rest of line; used for comments.

ADDRESSING
Most editing requests are preceded by an address specifying the line or lines in the buffer on which the request is to operate. Lines in the buffer can be addressed by absolute line number; relative line number, i.e., relative to the "current" line; and context. Current line is denoted by period (.), last line of buffer, by dollar sign ($).

REGULAR EXPRESSION
The following characters have specialized meanings when used in a regular expression. The user can reinvoke the last used regular expression by giving 3.
* signifies any number (or none) of the preceding character.
\ when used as the first character of a regular expression, signifies the character preceding the first character on a line.
\ when used as the last character of a regular expression, signifies the character following the last character on a line.
$ matches any character on a line.

ESCAPE SEQUENCES
\f exit from input mode and terminate the input request.
\c suppress the meaning of the escape sequence or special character following it.
\b(X) redirect editor stream to read subsequent input from buffer X.
\r temporarily redirects the input stream to read a single line from the user's terminal.

ready, rdy

Usage: ready
ready_off, rdf
   turns off the ready message.
Usage:      ready_off

ready_on, rdn
   prints a ready message after each command line has
   been processed.
Usage:      ready_on

release, rl
   releases the stack history that was automatically pre­
   served after a quit signal or unclaimed signal.
Usage:      release {¬control_arg}
   -all, -a

rename, rn
   replaces an entry name by a specified new name, with­
   out affecting any other names the entry might have.
Usage:      rename path1 name1 ... pathn name n
   pathi
      old name that is to be replaced.
   namei
      new name that replaces the entryname portion
      of pathi.

reprint_error, re
   prints information, from the system condition handler,
   about a condition that has already been handled and
   for which stack history is preserved.
Usage:      reprint_error {¬control_args}
   -all, -a
   -brief, -bf
   -depth N, -dh N
   -long, -lg

resource_usage, ru
   prints a report of resource consumption for current
   billing period.
Usage:      resource_usage {¬control_arg}
   -brief, -bf
   -long, -lg
   -total, -tt

run_cobol, rc
   initiates execution of a COBOL run unit in a specified
   "main program."
Usage:      run_cobol name {¬control_args}
   name
      reference name or pathname of the "main
      program" to be initiated.
   control_args
      -cobol_switch N, -cs N
      -no_stop_run, -nsr

runoff, rf
   types out text segments in manuscript form.
Usage:      runoff paths {¬control_args}
   control_args
      -(ball N, -bl N  — page N, -pg N
      -character, -ch — parameter arg
      -device N, -dv N — pm arg
      -from N, -fm N — pass N
      -hyphenate, -hph — segment, -sm
      -indent N, -in N — stop, -sp
      -no-pagination, — to N
      -npgn — wait, -wt
      -number, -nb

CONTROL WORDS
Conventions to specify arguments of control requests.
# integer constant
c character
cd character pair
exp expression (either numeric or string)
n integer expression
± ± indicates update by n; if sign not present, set
to n
f segment name
t title of the form 'part1'part2'part3'

CONTROL REQUESTS
If the request has a default, it is in parentheses fol­
lowing the definition.
.ad right justify text (on)
.ar arabic page numbers (arabic)
.bp begin new page
.br  break, begin new line
.cc c  change special character from % to c (%)
.ce n  center next n lines (1)
.ch cd  . note "c" in chars segment as "d"
.ds  double space (off)
.ef # t  defines even footer line #
.eq n  next n lines are equations (1)
.ex text  call command processor with "text"
.ft t  format of footnote processor with "text" (underscore)
.fi  fill output lines (on)
.fo # t  equivalent to : .ef # t, .of # t
.fr c  controls footnote numbering: "t" reset each page, "f" continuous, "u" suppress numbering
.ft  delimits footnotes
.gb xxx  "go back" to label xxx
.gf xxx  "go forward" to label xxx
.he # t  equivalent to: .eh # t, .oh # t
.if f exp  segment f.runoff inserted at point of request; value of "exp" assigned to "Parameter"
.in ±n  indent left margin n spaces (0)
.ia xxx  define label xxx
.ii ±n  next n lines treated as text (1)
.im ±n  line length is n (65)
.ma ±n  equivalent to: .mai ±n, .m4 ±n (4)
.mp ±n  print only every nth page (1)
.ms ±n  multiple space of n lines (1)
.ml ±n  margin above headers set to n (4)
.m2 ±n  margin between headers and footers set to n (2)
.m3 ±n  margin between text and footers set to n (2)
.m4 ±n  margin below footers set to n (4)
.na  do not right justify (off)
.ne n  need n lines; begin new page if not enough remain (1)
.nf  do not fill output lines; print them exactly as entered (off)
.of # t  defines odd footer line #
.oh # t  defines odd header line #
.op  next page number is odd
.pa ±n  begin page n
.pi n  skip n lines if n remain; otherwise skip n on next page before any text (1)
.pl ±n  page length is n (66)
.rd  read one line of text from the user_input I/O switch and place it in place of .rd line
.ro  roman numeral page numbers (arabic)
.rt  "return" from this input segment
.sk n  skip n lines (1)
.sp n  space n lines (1)
.sr sym exp  assign value of "exp" to variable named "sym"
.ss  single space (on)
.tr cd  . translate nonblank character c into d on output
.ts n  process next input line only if n is not zero (1)
.ty xxx  write "xxx" onto error_output I/O switch
.un n  indent next text line n spaces less (left margin)
.ur text  substitute values of variables in "text", and scan line again
.wt  read one line of text from user_input I/O switch and discard it
.*  comment line; ignored
~  comment line; ignored, but included in chars output segment

BUILT-IN SYMBOLS
runoff has over 50 internal variables, which are available to the user. In addition, the user can set his own variables with the .sr control request. See the runoff command in the MPM Commands for the list of built-in symbols.

EXPRESSIONS
Expressions can be either arithmetic or string and consist of numbers and operators in appropriate combinations. The operators and order of precedence are:

-    (bit-wise negation), -(unary)
* / \  (remainder)
* , + , -  (binary)
= , < , > , *  (all are comparison operators that
yield –1 for true or 0 for false)

\( \land \) (bit-wise AND)

\( \lor \) (bit-wise OR), \( = \) (bit-wise equivalence)

runoff_abs, rfa

submits an absentee request to process text segments
using the runoff command.

Usage:

\[ \text{rfa} \text{ paths \{-rf\_args\} \{-ear\_args\} \{-dp\_args\} \{-abs\_control\_args\}} \]

rf\_args

c control arguments accepted by the runoff command.

ear\_args

c control arguments accepted by the enter\_abs\_request command (except –brief).

dp\_args

c control arguments (except –brief and –truncate) accepted by the dprint command.

abs\_control\_args

–copy N, –cp N

–hold

–queue N, –q N

safety\_sw\_off, ssf

turns off the safety switch of a segment, directory, or
multisegment file, thus permitting the segment, directory, or multisegment file to be deleted.

Usage:

\[ \text{safety\_sw\_off \{paths\}} \]

safety\_sw\_on, ssn

turns on the safety switch of a segment, directory, or
multisegment file, thus preventing deletion of that segment, directory, or multisegment file.

Usage:

\[ \text{safety\_sw\_on \{paths\}} \]

send\_message, sm

sends messages (one or more, always sent one line at a
time) to a given user on a given project.

Usage:

\[ \text{send\_message Person\_id Project\_id \{message\}} \]

send\_message types “Input:” and accepts
lines that it sends, one at a time, with each newline character. In this case, input is terminated
by a line consisting solely of a period.

set\_acl, sa

manipulates the ACLs of segments, multisegment files,
and directories.

Usage:

\[ \text{sa path mode1 User\_id1 \ldots modep User\_idp \{-control\_args\}} \]

–directory, –dr

–segment, –sm

set\_bit\_count, sbc

sets a specified bit count on a specified entry.

Usage:

\[ \text{set\_bit\_count path1 count1 \ldots pathp countp}} \]

\[ \text{counti} \]

is the bit count, in decimal, desired for pathi.

set\_cc

sets the carriage control transformation for a specified
FORTRAN formatted file either on or off.

Usage:

\[ \text{set\_cc fileN \{-control\_arg\}} \]

fileN

name of the FORTRAN file in the range of
file01 to file99.

control\_arg

–off

–on

set\_com\_line, scl

changes the maximum size of expanded command
lines.

Usage:

\[ \text{set\_com\_line \{size\}} \]

size

is the new maximum expanded command line
size.
set_iac1_dir, sid  
adds entries to a directory initial ACL in a specified  
directory or modifies the access mode in an existing  
directory initial ACL entry.

Usage:  sid path model1 User_id1 ... modep {User_idp}  
{-control_arg}  
-ring N, -rg N

set_iac1_seg, sis  
adds entries to a segment initial ACL in a specified  
directory or modifies the access mode in an existing  
segment initial ACL entry.

Usage:  sis path model1 User_id1 ... modep {User_idp}  
{-control_arg}  
-ring N, -rg N

set_search_rules, ssr  
allows the user to set his dynamic linking search rules  
to suit his individual needs with only minor restric­  
tions. Two types of search rules are permitted: abso­  
lute pathnames of directories to be searched and  
keywords.

Usage:  set_search_rules path

set_tty, stty  
modifies the terminal type and modes associated with  
terminal I/O.

Usage:  set_tty {-control_args}  
-io_switch XX, -is XX  
-modes XX

sort_seg, ss  
orders the contents of a segment according to the  
ASCII collating sequence.

Usage:  sort_seg path {-control_args}  
-all, -a  
-ascending, -asc  
-block N, -bk N  
-delimiter XX, -dm XX  
-descending, -dsc  
-field S1 L1 S2 L2 ... Sn Ln,  
-fl S1 L1 S2 L2 ... Sn Ln  
-replace, -rp  
-segment path, -sm path  
-unique, -uq

start, sr  
resumes execution of the user's process from the point  
of interruption after a signal has suspended execution.

Usage:  start {-control_arg}  
-no_restore, -nr

status, st  
prints status information about storage system entries.

Usage:  status paths {-control_args}  

Segments, Multisegment Files, and Directories:

-all, -a  
-author, -at  
-date, -dt  
-device, -dv  
-length, -ln  
-mode, -md  
-name, -nm  
-type, -tp
stop_cobol_run, scr
causes the termination of the current COBOL run unit.

Usage:
\[ \text{stop_cobol_run} \{ \text{-control_arg} \} \]
\[ \text{-retain_data, -retd} \]

terminate, tm
terminate_segno, tms
terminate_refname, tmr
terminate_single_refname, tmsr
terminates reference names for a segment, unsnaps links to the segment, and makes the segment unknown if it has no reference names left.

Usage:
\[ \text{terminate paths} \]
\[ \text{tms seg_nos} \]
\[ \text{tmr ref_names} \]
\[ \text{tmsr ref_names} \]
seg_nos
segment numbers (in octal).
ref_names
reference names.

trace
a debugging tool that monitors all calls to a specified set of external procedures.

Usage:
\[ \text{trace} \{ \text{-control_args} \} \text{ names} \]
control_args
\[ \text{-after N} \]
\[ \text{-argument N, -ag N} \]
\[ \text{-before N, -brief, -bf} \]
\[ \text{-depth N, -dh N} \]
\[ \text{-every N, -ev N} \]
\[ \text{-execute XX, -execute XX} \]

unassign_resource, ur
unassigns a resource that has been assigned to the caller's process.

Usage:
\[ \text{unassign_resource} \text{ resource} \{ \text{-control_args} \} \]
resource
  specifies the name of the resource to be unassigned.

c control_args
  -comment XX, -com XX
  -admin, -am

unlink, ul
  deletes the specified link entry.
Usage:  unlink paths

vfile_adjust, vfa
  adjusts a storage system file left in an inconsistent state by an interrupted opening.
Usage:  vfile_adjust path {-control_arg}
  -set_bc
  -set_nl
  -use_bc {N}
  -use_nl

vfile_status, vfs
  prints the apparent type (unstructured, sequential, blocked, or indexed) and length of storage system files.
Usage:  vfile_status path

walk_subtree, ws
  executes a command line in a given directory (called the starting node) and in directories inferior to the starting node.
Usage:  walk_subtree path command_line {-control_args}
  command_line
  command line to be executed (multiple-word command line should be typed as a quoted string).
  control_args
  -bottom_up, -bu
  -brief, -bf
  -first N, -ft N
  -last N, -lt N

where, wh
  searches for a given reference name using the standard search rules and initiates the segment if found.
Usage:  where ref_names

who
  lists User_ids and other information about current users of the system.
Usage:  who {args} {-control_args}
  args
    Person_id
    .Project_id
    Person_id. Project_id
  control_args
    -absentee, -as
    -brief, -bf
    -long, -lg
    -name, -nm
    -project, -pj

963
  tells system user has terminal similar to EBCDIC IBM Model 2741 that must be recognized before he can log in.
Usage:  963

029
  tells system user has terminal similar to Correspondence code IBM Model 2741 that must be recognized before he can log in.
Usage:  029
ACTIVE FUNCTIONS

The format of each active function in this document is based on those found in the MPM Commands. The active function name is shown in boldface type followed by a brief description of the value this active function returns. In the usage line, after the name and description, the following conventions apply:

1. For simplicity, four common types of arguments accepted by active functions have been abbreviated as follows:

   str    any character string.
   t f    character string that has the value “true” or “false”.
   de c   character string that represents a decimal number.
   dt    character string that represents a date and time (see also item 5 below).

2. If an active function accepts more than one of a specific argument, “_args” is added to the argument name (e.g., tf_args).

3. Arguments that must be given in pairs are indicated with an “A” and “B” (e.g., strA strB).

4. Optional arguments are enclosed in braces (e.g., {strB}). All other arguments are required.

5. Each dt argument must be in a form acceptable to the convert_date_to_binary subroutine described in the Multics Programmers' Manual Subroutines, Order No. AG93. If an optional dt argument is not given, information about the current date and time is returned.

6. The term star_name means any pathname that conforms to the star convention.

The active functions listed below are grouped according to operation. An abbreviated description for each active function is given in the following pages, arranged in alphabetical rather than operational order.

Arithmetic
   ceil
   divide
   floor
   max
   min
   minus
   mod
   plus
   quotient
   times
   trunc

Character String
   format_line
   index
   index_set
   length
   search
   string
   substr
   verify

Date and Time
   date
   date_time
   day
   day_name
   hour
   long_date
   minute
   month
   month_name
   time
   year

Logical
   and

equal
   exists
   greater
   less
   nequal
   ngreater
   nless
   not
   or

Segment Name
   directories, dirs
   directory
   entry
   files
   get_pathname, gpn
   home_dir
   links
   nondirectories, nondirs
   nonlinks, branches
   nonsegments, nonsegs
   path
   pd
   segments, segs
   strip
   strip_entry, spe
   suffix
   unique
   wd

User Parameter
   have_mail
   system
   user

Question Asking
   query
   response
and
true if all the tf_args = true; otherwise false.
Usage: [ and tf_args ]

ceil
smallest integer > dec.
Usage: [ ceil dec ]

date
date abbreviation in the form “mm/dd/yy”.
Usage: [ date { dt } ]
date_time
date abbreviation, a time from 0000.0 to 2359.9, a
time zone abbreviation, and a day of the week
abbreviation.
Usage: [ date_time { dt } ]

day
one- or two-digit number of a day of the month, from
1 to 31.
Usage: [ day { dt } ]

day_name
name of a day of the week.
Usage: [ day_name { dt } ]
directories, dirs
names (separated by blanks) of all directories matching
star_name.
Usage: [ directories star_name ]
directory
directory portion of the absolute pathname of path.
Usage: [ directory path ]
divide
integer part of the value of decA / decB.
Usage: [ divide decA decB ]

date
date abbreviation in the form “mm/dd/yy”.
Usage: [ date { dt } ]
date_time
date abbreviation, a time from 0000.0 to 2359.9, a
time zone abbreviation, and a day of the week
abbreviation.
Usage: [ date_time { dt } ]
day
one- or two-digit number of a day of the month, from
1 to 31.
Usage: [ day { dt } ]
day_name
name of a day of the week.
Usage: [ day_name { dt } ]
directories, dirs
names (separated by blanks) of all directories matching
star_name.
Usage: [ directories star_name ]
directory
directory portion of the absolute pathname of path.
Usage: [ directory path ]
divide
integer part of the value of decA / decB.
Usage: [ divide decA decB ]

entry
entryname portion of the absolute pathname of path.
Usage: [ entry path ]
equal
ture if strA = strB; otherwise false.
Usage: [ equal strA strB ]
exists
checks for the existence of various types of items
depending on the value of key.
Usage: [ exists key str ]
key
argument
true if it has been passed an argument str;
otherwise false.
branch
true if a branch with pathname str exists;
otherwise false.
directory
true if a directory with pathname str exists;
otherwise false.
entry
true if an entry with pathname str exists;
otherwise false.
file
true if a segment or multisegment file str
exists; otherwise false.
link
true if a link with pathname str exists; other­
wise false.
msf
true if a multisegment file with pathname
str exists; otherwise false.
non_null_link
true if a link with pathname str exists and
points to an existing segment, directory, or
multisegment file; otherwise false.
segment
true if a nondirectory segment with pathname
str exists; otherwise false.
files
names (separated by blanks) of all segments, directories, links, and multisegment files matching a given star_name.

Usage:  [ files star_name ]

floor
largest integer \leq \text{dec}.

Usage:  [ floor dec ]

format_line, fl
formatted character string that is constructed from a control string and other optional arguments.

Usage:  [ format_line control_string { args } ]
control_string
is an ioa_ control string that is used to format the return value of the active function.
args
substituted in the formatted return value, according to the ioa_ control string.

get_pathname, gpn
absolute pathname of the segment that is designated by the reference name or segment number specified.

Usage:  [ get_pathname {-name} arg ]
-name
indicates that arg (which looks like an octal segment number) is to be interpreted as a segment name.
arg
reference name or segment number.

greater
true if strA > strB; otherwise false.

Usage:  [ greater strA strB ]

have_mail
true if there is mail in the user’s current default mailbox or in a specified mailbox; otherwise false.

Usage:  [ have_mail { path } ]

home_dir
pathname of the user’s home directory (usually of the form \text{>user\_dir\_dir\_Project\_id\_Person\_id}).

Usage:  [ home_dir ]

hour
one- to two-digit number of an hour of the day, from 0 to 23.

Usage:  [ hour { dt } ]

index
character position in strA where strB begins. If strB does not occur in strA, 0 is returned.

Usage:  [ index strA strB ]

index_set
sequence of numbers from 1 through \text{g} (where \text{g} is a decimal integer), separated by spaces.

Usage:  [ index_set \text{g} ]

length
character representation of the number of characters in str.

Usage:  [ length str ]

less
true if strA < strB; otherwise false.

Usage:  [ less strA strB ]

links
names (separated by blanks) of all links matching a given star_name.

Usage:  [ links star_name ]

long_date
month name, a day number, and a year in the form “month, day, year”.

Usage:  [ long_date { dt } ]

max
numerical maximum of dec_args.

Usage:  [ max dec_args ]
\textbf{min}
\hspace{1em} numerical minimum of \texttt{dec\_args}.
\textit{Usage:} [ min \texttt{dec\_args} ]

\textbf{minus}
\hspace{1em} result of \texttt{decA} – \texttt{decB}.
\textit{Usage:} [ minus \texttt{decA \texttt{decB}} ]

\textbf{minute}
\hspace{1em} one- or two-digit number of a minute of the hour, from 0 to 59.
\textit{Usage:} [ minute \{ \texttt{dt} \} ]

\textbf{mod}
\hspace{1em} \texttt{decA} modulo \texttt{decB}.
\textit{Usage:} [ mod \texttt{decA \texttt{decB}} ]

\textbf{month}
\hspace{1em} one- or two-digit number of a month of the year, from 1 to 12.
\textit{Usage:} [ month \{ \texttt{dt} \} ]

\textbf{month\_name}
\hspace{1em} name of a month of the year.
\textit{Usage:} [ month\_name \{ \texttt{dt} \} ]

\textbf{nequal}
\hspace{1em} true if \texttt{decA} = \texttt{decB}; otherwise false.
\textit{Usage:} [ nequal \texttt{decA \texttt{decB}} ]

\textbf{ngreater}
\hspace{1em} true if \texttt{decA} > \texttt{decB}; otherwise false.
\textit{Usage:} [ ngreater \texttt{decA \texttt{decB}} ]

\textbf{nless}
\hspace{1em} true if \texttt{decA} < \texttt{decB}; otherwise false.
\textit{Usage:} [ nless \texttt{decA \texttt{decB}} ]

\textbf{nondirectories, nondirs}
\hspace{1em} names (separated by blanks) of all segments, links, and multisegment files matching a given \texttt{star\_name}.
\textit{Usage:} [ nondirectories \texttt{star\_name} ]

\textbf{nonlinks, branches}
\hspace{1em} names (separated by blanks) of all segments, directories, and multisegment files matching a given \texttt{star\_name}.
\textit{Usage:} [ nonlinks \texttt{star\_name} ]

\textbf{nonsegments, nonsegs}
\hspace{1em} names (separated by blanks) of all directories, links, and multisegment files matching a given \texttt{star\_name}.
\textit{Usage:} [ nonsegments \texttt{star\_name} ]

\textbf{not}
\hspace{1em} false if \texttt{str} = true; true if \texttt{str} = false; otherwise an error diagnostic.
\textit{Usage:} [ not \texttt{str} ]

\textbf{or}
\hspace{1em} true if any \texttt{tf\_arg} = true; otherwise false.
\textit{Usage:} [ or \texttt{tf\_args} ]

\textbf{path}
\hspace{1em} absolute pathname of \texttt{path\_arg} (which is a pathname).
\textit{Usage:} [ path \texttt{path\_arg} ]

\textbf{pd}
\hspace{1em} pathname of the process directory of the process in which it is invoked.
\textit{Usage:} [ pd ]

\textbf{plus}
\hspace{1em} sum of \texttt{dec\_args}.
\textit{Usage:} [ plus \texttt{dec\_args} ]
query
true if the user's answer to the question was "yes";
false if the user's answer was "no"; otherwise an error
diagnostic. If the question is more than one word (arg
contains blanks), it must be enclosed in quotes.
Usage: [ query arg ]

quotient
result of decA / decB.
Usage: quotient [ decA decB ]

response
answer typed by the user in response to the question
specified by arg. If arg contains blanks, it must be en-
closed in quotes.
Usage: [ response arg ]
arg
the question to be asked.

search
first character position in strA that meets the follow-
ing test: does any character in strB occur in strA? If
no character of strB occurs in strA, 0 is returned.
Usage: [ search strA strB ]

segments, segs
names (separated by blanks) of all segments matching
a given star_name.
Usage: [ segments star_name ]

string
single character string. If no str_args are present, a null
character string is returned. If one or more str_args are
present, then any quotes in these are doubled when
str_args are placed in the quoted return string.
Usage: [ string { str_args } ]

strip
absolute pathname of the specified entry with the
last component removed. If str is specified, the last
component is removed only if it matches str.
Usage: [ strip path { str } ]

strip_entry, spe
entryname portion of the absolute pathname returned
by the strip active function. If str is specified, the last
component is removed only if it matches str.
Usage: [ strip_entry path { str } ]

substr
portion of str starting with decA and continuing for
decB characters (the default for decB is 1).
Usage: [ substr str decA { decB } ]

suffix
last component of the entryname portion of the speci-
fied segment. If that entryname has only one com-
ponent, it returns the null string.
Usage: [ suffix path ]

system
various installation-dependent system parameters
Usage: [ system key ]
key
company
per-system parameter company name.
date_up
date that the system was brought up, in the
form "mm/dd/yy".
department
derm-system parameter computer center de-
partment name.
down_until_date
date that the system will next be brought up,
if specified by operator, in the form
"mm/dd/yy".
down_until_time
time that the system will next be brought up,
if specified by operator, in the form
"hhmm.tm".
ds_company
per-system parameter company name, with
the characters of the name double spaced.
ds_department
- per-system parameter computer center department name, with the characters of the name double spaced.

installation_id
- per-system parameter installation identification.

last_down_date
- date that service was last interrupted, whether by shutdown or crash.

last_down_reason
- reason for the last system service interruption, if known. The reason may be:
  - shutdown normal system shutdown
  - crash system crash (no number assigned)

last_down_time
- time that service was last interrupted.

max_units
- current maximum number of load units, in the form “nnn.n”.

max_users
- current maximum number of users.

n_units
- current number of logged-in load units including daemon and absentee, in the form “nnn.n”.

n_users
- current number of logged-in users including daemon and absentee.

next_down_date
- date that system will next be shut down, if specified by operator.

next_down_time
- time that system will next be shut down, if specified by operator.

next_shift
- next shift number.

reason_down
- reason for next shutdown, if specified by operator.

shift
- current shift number.

shift_change_date
- date on which current shift number will change to next_shift.

shift_change_time
- time at which current shift number will change to next_shift.

sysid
- version number of the hardcore system tape currently running.

time_up
- time that system was brought up, in the form “hh:mm.t”.

time
- four-digit time of day in the form “hh:mm” where 00 <= hh <= 23 and 00 <= mm <= 59.

Usage: [ time { dt } ]

times
- result of decA * decB.

Usage: [ times decA decB ]

trunc
- largest integer whose absolute value is <= absolute value of dec.

Usage: [ trunc dec ]

unique
- unique character string as generated by the unique_chars_subroutine.

Usage: [ unique ]
user

various user parameters.

Usage: [ user key ]

key

absentee
true if the user is an absentee user; otherwise false.

absin
absolute pathname of absentee user’s absentee input segment including the absin suffix; otherwise a null string.

absout
absolute pathname of absentee user’s absentee output segment; otherwise a null string.

anonymous
true if the user is an anonymous user; otherwise false.

auth
short string for the authorization of the user’s process or system_low.

auth_long
long string (in quotes) for the authorization of the user’s process or “system_low”.

brief_bit
true if the user specified the –brief control argument in his login line; otherwise false.

cpu_secs
user’s CPU usage (in seconds) since login, in the form “sss.t” with leading zeros suppressed.

log_time
user connect time (in minutes) since login, the form “mmm.t”.

login_date
date at login time, in the form “mm/dd/yy”.

login_time
time of login, in the form “hhmm.t”.

max_auth
short string for the max authorization of the user’s process or system_low.

max_auth_long
long string (in quotes) for the max authorization of the user’s process or “system_low”.

name
user’s User_id at login time.

outer_modulo
initial outer modulo for the terminal channel.

preemption_time
time at which the primary user becomes eligible for group preemption, in the form “hhmm.t”.

process_id
user’s process identification in octal.

project
user’s Project_id.

protected
true is user currently a primary user and protected from preemption; otherwise false.

secondary
true if the user is currently subject to preemption; otherwise false.

term_id
user’s terminal ID code. It is “none” if the user’s terminal does not have the answerback feature.

term_type
user’s terminal type. It can have one of the following values:

“Absentee” “CORR2741” “TN300”
“Network” “TTY33” “ARD”
“1050” “TTY37” “ASCII”
“2741” “TTY38”

verify
first character position in strA that fails the following test: does any character in strB occur in strA? If every character of strB occurs in strA, 0 is returned.

Usage: [ verify strA strB ]
wd
    pathname of the working directory.

Usage:   [ wd ]

year
    two-digit number of a year of the century.

Usage:   [ year { dt } ]

MULTICS ASCII CHARACTER SET

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>NUL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010</td>
<td>BS</td>
<td>HT</td>
<td>NL</td>
<td>VT</td>
<td>NP</td>
<td>CR</td>
<td>SO</td>
</tr>
<tr>
<td>020</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>030</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>040</td>
<td>Space</td>
<td>!</td>
<td>&quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
</tr>
<tr>
<td>050</td>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
</tr>
<tr>
<td>060</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>070</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
<td>&lt;</td>
<td>=</td>
<td>&gt;</td>
</tr>
<tr>
<td>100</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>110</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
</tr>
<tr>
<td>120</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
</tr>
<tr>
<td>130</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>[</td>
<td>\</td>
<td>]</td>
<td>^</td>
</tr>
<tr>
<td>140</td>
<td>`</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
</tr>
<tr>
<td>150</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td>n</td>
</tr>
<tr>
<td>160</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
</tr>
</tbody>
</table>
| 170 | x | y | z | { | | | | }

Unused Characters

These characters are reserved for future use:

SOH 001 ACK 006 DC4 024 EM 031
STX 002 DLE 020 NAK 025 SUB 032
ETX 003 DC1 021 SYN 026 ESC 033
EOT 004 DC2 022 ETB 027 FS 034
ENQ 005 DC3 023 CAN 030 GS 035
     RS 036
     US 037