To: Operations

From: T. H. Van Vleck

Date: 11/09/73

Subject: Instructions for operating the initializer
and answering service

This memorandum is a complete list of operator commands and instructions for operating the initializer console.

CONTENTS

INTRODUCTION

When the Multics bootload sequence is started, by the BCS command "BOOT", a process is created called the initializer process. This process remains active as long as the system is running. It performs several functions for the system:

a) answering service operations (login, logout, etc.)

b) operator command service

c) system reloading

d) system terminal management and message routing

e) system accounting

f) user request handling (tape, new_proc, etc.)

g) system administration

The initializer process is controlled by a terminal, usually called the initializer console. The operator uses this terminal to input special commands for the purpose of controlling the system's operation.

The system cannot operate without an initializer process. If an error occurs which makes the initializer process unusable, the system will crash with the message "ATTEMPT TO TERMINATE INITIALIZER PROCESS."

The initializer process is also referred to sometimes as the system control process or the answering service process.

ADMINISTRATIVE RING COMMANDS

When the initializer terminal answers, the initializer process is in the "administrative ring" (ring 1) environment. The first thing the ring 1 environment types is a message of the form:

```
Multics SYSID - MM/DD/YY  HHMM.T est DAY
```

giving the system ID from the system tape and the current date and time. Then, the ring 1 program types

```
Command:
```

and waits for the operator to type one of the following requests:

startup	to start Multics (see page 74)
multics	to start a special session (see page 62)
reload	to do a reload
bos	to call BOS
standard	to go to the user ring
shutdown	to shut down
update	to perform a system distribution update

If "startup" or "multics" is typed, the initializer leaves the ring 1 environment and executes the command in the "user ring" (ring 4) environment.

USER RING COMMANDS

Once the initializer process begins operation in ring 4, the operator may issue initializer commands to affect the operation of the system.

If the initializer process left ring 1 by means of a "startup" or "multics" command, the first thing done in ring 4 will be the initialization of the answering service.

The initializer process is normally waiting for an initializer command from the operator. After the operator types in his command, the initializer performs it and then types "R" (for "Ready") and awaits another command. The output from a command will be typed after the "R" if the message coordinator is being used. (Sometimes, the output from a command is even routed by the message coordinator to a terminal other than the one from which the input came.)

ADMIN MODE

The initializer process is sometimes used to perform special operations which a normal process cannot perform. For instance, the initializer is the only process which can execute commands before the answering service is brought up: so the initializer must be used to repair problems which prevent any user from logging in.

In order to use the initializer process to execute an arbitrary Multics command, the operator must enter admin mode. Because the initializer process has special abilities and special limitations, admin mode is used only by qualified personnel.

The operator must supply a password in order to enter admin mode. Once in admin mode, the initializer responds to regular Multics commands instead of the commands listed in this document.

Use of admin mode is more or less like use of regular Multics, with the one important difference that there is no way to QUIT.

Admin mode and editing of the message of the day can be done from any initializer terminal; but only one terminal can be operating in this mode at a time.

To exit from admin mode, the operator types the "admin_mode_exit" (or "ame") command.

System programmers may send single commands to the initializer to be executed in admin mode, by use of "send_admin_command". If a system programmer sends such a command, a message will be typed on the initializer, the command will be executed, and the system will continue in normal operating mode.

TERMINAL USAGE

When the system is bootloaded, the answering service comes up on either the system master console (BOS typewriter) or on a terminal channel. The channel may be specified on the OPTY card in the BOS configuration deck. If the initializer terminal is a typewriter channel, the terminal is either dialed up to this port before booting, or is hard-wired permanently to the port.

The initializer process is in charge of all terminal channels known to the system. Some of these channels are connected to the answering service and used for logins and logouts; others are used by the initializer itself.

The terminal channels connected to Multics may be

a) completely unused (not in CONFIG deck)

b) unrecognized (in CONFIG deck but not in lines file)

c) used by the Message Coordinator

d) used by the Answering Service

e) used by a user process

Operator commands are available to move a terminal from one of these states to another.

Message Coordinator

The Message Coordinator programs allow the initializer to run more than one terminal channel, and lets the daemons run without terminals, sending their messages to the Initializer for disposition.

Each daemon process is considered to be a source, and the initializer process itself contains several sources, such as "as" (the answering service), "sc" (system control), and "tape" (tape mounting). Each source does its input and output over several i/o streams through the message coordinator, to a set of segments contained in >system_control_1.

The message coordinator _routes_ output messages from the various sources to one or more _virtual consoles_ for output. Each virtual console has a name, and a list of destinations, which may be terminal channels run by the initializer, or log files.

The terminals run by the initializer process are treated as if they had an independent keyboard and printer: all the keyboards may input commands to the initializer (subject to permission), but the output caused by the command will not necessarily be typed on the terminal that input it. Where output comes out depends on the routing table and the virtual console table.

SYSTEM STARTUP

A special list of commands can be set up by the system programmers to be executed when the answering service is started. These commands are kept in "system_start_up.ec", and come in three sections: those executed before the answering service is started, those execute after the answering service is ready but before phone lines are answered, and those executed after the telephone lines are answered.

Normally, the system_start_up.ec will turn on the message coordinator before running the answering service, and will automatically log in the daemons immediately after the answering service is ready. If the initializer is to operate more than one terminal channel, the additional channels will be accepted automatically at this time.

The normal mode of operation for the system will be to use the system master console (BOS typewriter) as the first initializer console, and to automatically add one or more terminal channels to the initializer during startup.

The startup sequence on the system console will look like this:

```
o) BOOT
s) MULTICS 19.1 - 02/14/72 1949.3 EST WED
s) COMMAND:
o) STARTUP
s) R
```

lines typed by the system are indicated (in this document only) by "s)" and lines typed by the operator are indicated by "o)". After the "R", the system console will not be used for most output except for the usual disk error, tape mount, programmer, and hardcore error messages.

If channel "tty238" is the terminal channel which will be used by the initializer for regular messages, it will be hard-wired to the system or the operator will have dialed it up before typing BOOT, as usual. The output on this console will look like this:

```
s) tty238 attached by system control.
s)  1950 as   Multics 18.6; answering service 6.12
s)  1951 as   LOGIN  Daemon io1     io1     IO.SysDaemon
s)  1951 as   LOGIN  Daemon bk      bk      Backup.SysDaemon
s)  1952 io1 IO DAEMON READY TO START
s) -->  io1
s)  1952 bk  r 1952 4.801 25+99
s) -->  bk
```

The lines beginning with "-->" indicate that the source wants input. They are called "sentinels." To input a line to the daemon, the operator uses the "reply" command.

```
o) reply io1 init prtdim prta34
s) R
o) reply bk start_dump sys_dirs xyz
s) R
s)  1953 io1 act_ctl_: IO Daemon accounting initialization.
s)  1953 io1 Is this the first or second IO daemon?
s) --> io1
s)  1953 bk  Enter primary dump-tape label:
s) --> bk
o) reply bk IC-75
s) R
o) reply io1 first
s) R
s)  1954 io1 Type "yes" if prtdim prta34 is correct:
s) --> io1
```

and so forth. The example above shows how the system intermixes output lines from all of the sources on a single console, and how the operator replies to a request for input from a source.

If more than one terminal channel is connected to the initializer, the output from the various sources (daemon processes, etc.) can be routed to divide the work between several consoles. For example, all the daemons could be handled by one terminal, and the answering service could use another. Or, if all the terminals are broken, the system can be run completely from the system console (but this setup would be bad for the system, since whenever the operator is typing in or the system is typing out on the system console, the entire Multics system is hung; and on a two-cpu configuration, the system may crash if a ring-zero message has to wait too long for the master console.)

All terminals attached to the initializer may input initializer commands. (It is possible to restrict a terminal to only certain commands.)

It is sometimes difficult to input an operator command between output messages on an initializer terminal, because the system keeps interrupting. If the operator types an empty line on an initializer terminal, the system will respond

```
    OPER:
```

and suspend output on that terminal channel. When the operator completes his command, the output will be restarted, with no messages lost. If the operator does not finish his command in one minute, the output will be restarted. (This feature does not work for the bootload console.)

Terminals may also be added to the initializer dynamically. To do this, the operator dials a terminal into Multics as if he were going to log in, but instead of typing "login", he issues a "dial" command:

```
    s) Multics 18-6: MIT, Cambridge, Mass.
    s) Load = 41.0 out of 50.0 units: users = 41
    o) dial system
```

An optional identifier may be typed after "system," to indicate which terminal has dialed up, or to serve as a password to insure that the command has been issued by an authorized operator. The dialed terminal will then get a message of the form

```
    s) TTY37 405 chn tty196 dialed to Initializer.
```

Also, on the initializer console, a message stating that the terminal has dialed up will be printed.

```
    s)   1137   as   dial_ctl_:  TTY37   405   tty196   dialed   to
    Initializer.
```

The operator should then issue a series of commands to accept the terminal channel and to route output to it.

```
o) accept tty196
s) R
o) define vc2 tty tty196
s) R
o) route dump user_i/o vc2
s) R
```

The response on the dialed terminal will be a message saying that the initializer has attached the channel:

```
s) tty196 attached by system control.
```

followed by whatever messages are routed to the terminal channel.

When the operator is finished with a dialed terminal, or if a curious user tries to dial the initializer without permission, the operator may disconnect the channel from the initializer and make it available for dialups again by typing a "drop" command:

```
o) drop tty196
s) R
```

The response on the dialed terminal will be a message like "please reissue dial command," and at this point the terminal may be re-dialed, or used for regular logins, or hung up.

USER RING INITIALIZER COMMANDS

This section contains a command write-up for each initializer command accepted by the initializer in ring 4.

For each command, the name of the command, a short explanation of its function, and an explanation of its usage is given. Examples are given for some, but not all commands. The possible error messages which the command may produce are listed. Consult MOSN-1f.2.1 for an explanation of what the message means, and what action to take when the message is printed.

Error Messages

In addition to the messages listed under each command, several messages are common to all commands.

If the operator types a command (other than "reply") with any of the illegal characters ";" or "[" in it, the system will respond with the message

```
syntax error
```

and ignore the command.

If a terminal restricted (see "accept") to issuing only certain commands, and a command which is forbidden is typed, the system will respond

```
privilege error
```

and ignore the command.

If a terminal attempts to enter admin mode, reconfigure the system, or edit the message of the day, and some other terminal is already doing one of these functions, the system will respond function busy

and ignore the command.

abs maxq

Command: abs maxq

Effect: This command sets the highest queue which is examined for absentee requests.

Usage: This command is used to change the highest queue searched for absentee requests after absentee is already up. It may be used to bypass the processing of requests in a particular queue for a particular time period. Type

```
     abs maxq M
```

to set the highest absentee queue to M.

Errors:

```
admin: Entry not found. >sc1>absentee_user_table

admin: abs error - absentee stop in progress

admin: abs error - absentee not up

admin: Expected argument missing. abs

admin: abs error - abs maxqueue has illegal value XXXX
```

---

abs maxu

Command: abs maxu

Effect: This command sets the maximum number of absentee users allowed in at any one time.

Usage: This command is used to change the maximum number of absentee users after absentee is already up. Type

```
      abs maxu N
```

to set the maximun number of simultaneous absentees to N.

Errors:

```
admin: Entry not found. >sc1>absentee_user_table

admin: abs error - absentee stop in progress

admin: abs error - absentee not up

admin: Expected argument missing. abs

admin: abs error - abs maxunits has illegal value ^XX^
```

abs stop

Command: abs stop

Effect: Stop absentees (run to completion)

Usage: This should be done about 30 minutes before "stop" is typed. If a "down" command has been issued, "abs stop" will be issued 30 minutes before the scheduled shutdown time automatically. Type

```
abs stop
```

to stop the absentee facility. The absentee programs will not let any more absentees log in, and will wait for all current ones to log out. If all absentees have not logged out in 30 minutes, absentee will automatically bump the remaining absentee users.

If all absentee users log out before 30 minutes, the message

```
admin: All absentee processes have run to completion
```

will come up. If not, then the message

```
admin: bumping all remaining absentee processes
```

will be printed.

Sometimes it is necessary to stop absentee immediately, without giving the system 30 minutes to let absentees finish. If the operator gives the command

```
abs stop now
```

then all absentee processes will be bumped as if the time limit had expired. Since this option forcibly terminates user programs, which may be difficult to restart, the "now" option

```
---------------
 abs stop
---------------
```

should be used only when specifically authorized.

Errors:

```
admin: abs error - absentee stop in progress

admin: abs error - absentee not up

admin: Expected argument missing. abs
```

Command: abs start

Effect: Start up absentee

Usage: This command starts up the absentee facility, which allows users to submit jobs for deferred execution. Type

```
     abs start N M
```

to start the absentee facility, to set the maximum number of simultaneous absentee users to N, and to set the highest queue searched for jobs to M. If N is not supplied, a default number will be assumed. Currently, this is 1. If M is not supplied, a default number will be assumed. Currently, this is 3.

If a "down" command has scheduled an automatic shutdown, issuing an "abs start" will automatically schedule an "abs stop" 30 minutes before the scheduled shutdown time.

Errors:

```
admin: Entry not found. >sc1>absentee_user_table

admin: abs error - absentee stop in progress

admin: abs error - absentee already up

admin: Expected argument missing. abs

admin: abs error - abs maxunits has illegal value XXX

admin: abs error - abs maxqueue has illegal value XXX

absentee_user_manager_: ERROR_MESSAGE. Cannot create absentee
event channel

absentee_user_manager_: Error in initializing absentee. Don't
bring up absentee facility.

absentee_utility_: ERROR_MESSAGE. >sc1>whotab
```

---

abs start

---

absentee_utility_: ERROR_MESSAGE. Creating new >sc1>absentee_N.ms

absentee_utility_: ERROR_MESSAGE. Unable to create >sc1>absentee_N.ms

absentee_utility_: ERROR_MESSAGE. Unable to set extended access on >sc1>absentee_N.ms

absentee_utility_: ERROR_MESSAGE. Unable to initiate new >sc1>absentee_N.ms abs bump

Command: abs bump

Effect: Bump absentee user

Usage: This command is used to cause an absentee user to be bumped. If the absentee job has been declared restartable, the job will be left in the queue and retried again later. This command can be used when shutting down absentee quickly, because of some error, usually at the request of a system programmer. If a user calls and asks to have his absentee job bumped, make sure he does not mean for you to use the "abs cancel" command instead. Type

```
     abs bump NAME PROJ
```

to bump the absentee user with name NAME and project PROJ. Either or both of NAME and PROJ may be an asterisk (*), meaning everybody.

It is also possible to bump the absentee user on a particular "slot number", just as it is possible to bump regular users by channel number. Type

```
     abs bump absNN
```

to bump the absentee user on slot NN. These slot numbers are typed by "who".

Example: To bump all absentee users, type

```
     abs bump * *
```

Errors:

admin: Expected argument missing. abs

abs bump

```
admin: abs error - no abs bump signalled for NAME PROJ

admin: abs error - abs bump has illegal arguments
```

Command: abs cancel

Effect: Bump absentee user, do not restart

Usage: This command is used to cause an absentee user to be bumped. It differs from "abs bump" in that restartable absentee requests will be removed from the queue, and not restarted. Use this command to get rid of an absentee job that is in trouble (for instance, one that seems to be re-issuing itself in a loop, or one that calls for a tape which does not exist) or when a user requests that his job be bumped. Type

```
abs cancel NAME PROJ
```

to cancel the absentee user with name NAME and project PROJ. Either or both of NAME and PROJ may be an asterisk (*), meaning everybody.

It is also possible to cancel the absentee user on a particular "slot number". Type

```
abs cancel absNN
```

to cancel the absentee user on slot NN. These slot numbers are typed by "who".

Example: To cancel all absentee users, type

```
abs cancel * *
```

Errors:

```
admin: Expected argument missing. abs

admin: abs error - no abs cancel signalled for NAME PROJ
```

----

aos cancel

```
admin: aos error - aos cancel has illegal arguments
```

accept

Command: accept

Effect: accept a device channel and connect it to initializer

Usage: accept TTYXXX -RESTRICT-

This command is used to pick up a terminal channel and add it to the initializer's device complement. If RESTRICT is not specified, or if it is "full", the device will be able to issue all operator commands. RESTRICT may also be

none	no commands allowed
reply	only "reply" is allowed
query	only "who" and "hmu" are allowed

If the channel appears in the answer_table, then it must either have state 0 (not in lines file) or be dialed to the initializer.

Response: TTYXXX attached by system control.

Errors:

```
admin: expected argument missing. accept

admin: accept error - tty not dialed to initializer: ttyXXX

admin: accept error - unknown privilege code

admin: Ioname already attached and active. accept

admin: ERROR_MESSAGE. accept
```

---

addcpu

Command: addcpu

Effect: Add cpu to Multics configuration

Usage: This command is issued by the operator to add a CPU. Type

```
addcpu CPUID PORT
```

to add the cpu with id CPUID and processor port PORT. The command will give very full instructions and ask you whether all switches are set before starting the added cpu. Consult MOSN 6.6 for more details.

Making a mistake in adding a cpu can crash the system.

Example: To add CPU b on port 5, type

```
addcpu b 5
```

Errors:

see MOSN 6.6

addmem

Command: addmem

Effect: Add a memory to the Multics configuration

Usage: This command is issued by the operator to add a memory. Only memories which are defined by BOS cards at bootload time can be added. Type

```
     addmem MEM -CTL-
```

to add the memory with id MEM. Specify the second argument, CTL, only if you must specify the control processor for this memory. The command will give very full instructions and ask you whether all switches are set before starting the added memory. Consult MOSN 6.5 for more details.

Errors:

see MOSN 6.6

addpage

Command: addpage

Effect: Add paging device records

Usage: This command dynamically reconfigures the system to add part or all of a paging device.

See MOSN 6.6 for details.

Errors:

see MOSN 6.6

Command: addtape

Effect: Add tape drive

Usage: This command tells the system that it may use a specified tape drive.

See MOSN 6.6 for details.

Errors:

See mosn 6.6

---

admin

Command: admin

Effect: Enter "admin" mode

Usage: Admin mode allows the operator to use the initializer console to execute normal Multics commands. Because the initializer has extraordinary power and special limitations, admin mode should only be used at the request of a system programmer or system administrator. To enforce this, a password is required in order to enter admin mode. Type

```
     admin
```

to enter admin mode. The system will respond

```
     Password
```

and turn off the printer. Type the password. (A password will not be requested if the system administrator has set the admin mode password to "*".) The system should then respond with a ready message. Execute any desired commands, and then type the command

```
     ame

     (or "admin_mode_exit")
```

to return to initializer command level.

Errors:

```
system_control_: admin password incorrect
```

Command: attach

Effect: Attach channel to answering service

Usage: Sometimes it is necessary to attach an additional terminal channel to the answering service, so that users may use the channel for logins. This may be because the channel was not in the "lines" file when the system was being started up, or because the channel was removed either manually or due to an error. Type

```
     attach ttyXXX ttyYYY ...
```

to cause the channels listed to be attached to the answering service. The arguments may be of the form ttyXXX, where XXX is a terminal channel number, or netXXX, where XXX is a network channel number.

Errors:

```
admin: ttyXXX NOT attached: restricted channel name

admin: ttyXXX NOT attached: channel already attached

admin: ttyXXX NOT attached: Answer Table full.

admin: ttyXXX NOT attached: TTY DIM error. Channel removed.

admin: bad arg "XXX"

admin: attach not done
```

---

bos

<u>Command</u>: bos

<u>Effect</u>: Call BOS

<u>Usage</u>: This command causes BOS to be entered. All Multics operation is suspended.

When the system is in trouble, it is sometimes necessary to enter BOS to use the DUMP, PATCH, or BLAST commands. Type

```
      bos
```

to enter BOS. Typing

```
      GO
```

on the BOS console will cause Multics to be restarted.

<u>Errors</u>:

none

Command: bump

Effect: bump user

Usage: This command causes a user to get an immediate automatic logout, or an automatic logout in a specified time. It does this by causing the initializer to signal itself to bump the user, so if the initializer is so sick that it can't handle logins, "bump" probably won't work either. Type

```
bump NAME PROJ MM MESSAGE TO USER
```

to cause the user with name NAME and project PROJ to be bumped after MM minutes. The string MESSAGE will be typed on the user's console. If MM is omitted, the user will be bumped immediately. If MESSAGE is omitted, the user will not be told why he is being bumped. Either or both of NAME and PROJ may be asterisk (*), meaning everybody. Users with the "nobump" privilege, like the system administrators, cannot be bumped by this form of the command. A message will be sent to such users telling them that an attempt to bump them has been made.

Type the command

```
bump ttyXXX MM MESSAGE TO USER
```

to cause the user on terminal channel ttyXXX to be bumped. This is the only way to bump a user with the "nobump" privilege.

Sometimes, the "bump" command will appear to work -- the message "Name.Proj bumped" will come out -- but the user will not be logged out. This may happen if the user's answer table entry is in an inconsistent state. If this happens, try the "remove" command on the user's TTY channel.

See the "abs bump" and "abs cancel" commands for instructions on bumping absentee users. See the "logout" command for instructions on logging out daemon processes.

---

bump

Examples:

```
     bump * * 3 Problems with reconfiguration. Up in 10 min.

     bump Jones Multics

     bump tty113
```

Errors:

```
admin: Expected argument missing. bump

admin: bump has illegal arguments

admin: no bump signalled to NAME PROJ

admin: NAME PROJ has "nobump"
```

Command: cripple

Effect: Prepare for reload

Usage: This command is used before a reload which will reload the system library directories or the initializer's directory ">system_control_1".

To do such a reload, the operator brings up a special session and logs in the daemon process to be used for reloading, Dumper.Sysdaemon.

The message coordinator should not be used during such a reload, since its programs and data bases are likely to be replaced by the reload too. (See the write-up of the "nostart" command.)

After the reloader is logged in, type

```
cripple
```

to shut off the answering service so that no unexpected faults will occur if one of the answering service data bases or programs is deleted by the reloader.

When the reload is done, do not attempt to shut down, as this is likely to cause a fault. instead, re-enter BOS manually. (A "reset" command will reset the effect of "cripple", but is not recommended.)

define

Command: define

Effect: associate virtual console with channel

Usage: define VCONS TYPE DEST

This command creates a new virtual console if VCONS does not already exist. The destination DEST is then added to the destination list for VCONS. A virtual console may have up to 8 destinations. If TYPE is "tty" then DEST must be a channel ID which has been accepted previously. If TYPE is "log" then DEST is the name of a log file to which messages will be added as they are sent to VCONS. (These logs can be printed with "print_log".) If TYPE is "sink" then DEST can be any name: output sent to a sink vanishes.

Errors:

admin: expected argument missing. define

admin: Ioname not found. define

admin: Typename not found. define

admin: There is no room to make requested allocations. define

Command: delcpu

Effect: Delete cpu

Usage: This command removes a CPU from the Multics configuration. See MOSN 6.6 for more instructions. Type

```
     delcpu CPUID
```

to delete the cpu with id CPUID. Instructions for changing the control processor switch will be given.

Errors:

see MOSN 6.6

----

delmem

Command: delmem

Effect: Delete memory

Usage: This command removes a memory from the Multics configuration. See MOSN 6.6 for more instructions. Type

```
     delmem MEMID
```

to delete the memory with id MEMID.

Errors:

see MOSN 6.6

__Command__: delpage

__Effect__: Delete paging device records

__Usage__: This command dynamically reconfigures the system to drop part or all of the paging device.

See MOSN 6.6 for details.

__Errors__:

see MOSN 6.6

----

deltape

Command: deltape

Effect: Delete tape drive

Usage: This command tells the system that a specified tape drive is no longer available for assignment.

See MOSN 6.6 for details.

Errors:

see MOSN 6.6.

Command: deroute

Effect: remove virtual console from stream

Usage: deroute SOURCE STREAM OLD_VCONS

This command removes a virtual console from the output list for a given SOURCE and STREAM. If the stream is left with no virtual consoles, output will be sent to the default virtual console, which is usually defined to the system master console.

Errors:

admin: Expected argument missing. deroute

admin: Ioname not found. deroute

---

detach

Command: detach

Effect: Detach channel & bump user

Usage: This command causes a channel to be removed from the answering service. It will no longer answer the telephone. If any user is logged in on the channel, he will be bumped. This is done by signalling, like "bump". Type

```
detach NAME PROJ MESSAGE TO USER
```

to detach the channel which the user with name NAME and project PROJ is o. If MESSAGE is given, the message will be typed on the user's console. If it is not given, the user will not be told what is happening. Either or both of NAME and PROJ may be asterisk (*), meaning everybody.

Type the command

```
detach ttyXXX MESSAGE TO USER
```

to detach the channel whose id is ttyXXX.

If the message "ttyXXX detached" comes up, then the detach has been signalled. If the user on the channel does not log out shortly, then the answer table entry for the channel may be in a state which does not allow detaches. Try the "remove" command.

Errors:

```
admin: Expected argument missing. detach

admin: detach has illegal arguments

admin: no detach signalled to NAME PROJ
```

Command: develop

Effect: Reload development-system tapes

Usage: This command is typed on the development system initializer after the system library dump tapes have been reloaded, in order to set the development machine up for a test session. Type

```
     develop
```

to perform this setup.

Errors

reloader errors

---

down

Command: down

Effect: Schedule auto shutdown

Usage: This command schdules an automatic shutdown for some later time. When the scheduled time comes, all users will be bumped with three minutes' warning. Type

```
     down TIME BACK REASON
```

to schedule an automatic shutdown at time TIME. TIME must be a military time: it refers to the next possible such time, so that you cannot schedule a shutdown more than 24 hours in advance. BACK is another military time, which will be announced to users as the time when the system will come back up. REASON is the reason for the shutdown.

When the system is started up, a check is made to see if a "down" command had previously been issued. If one is found, and more than 30 minutes remain before it takes effect, the "down" command will be re-issued automatically.

If absentee is up when the "down" command is issued, an automatic "abs stop" will be set up for 30 minutes before TIME. If absentee is not up when the "down" command is issued, but it is brought up later, the automatic "abs stop" will still be set up for 30 minutes before shutdown. If absentee is already being shut down when the "down" command is issued, the shutdown of absentee will continue.

Example: To schedule a shutdown at 5:45 AM, and to tell the users that the system will be back at 9, type

```
     down 0545 0900 Regularly scheduled shutdown.
```

To cancel a scheduled shutdown, type

```
     down 0
```

Errors:

```
admin: Expected argument missing. down

admin: down error - there is no scheduled shutdown

admin: down error - XXX illegal: shutdown time must  be  4  digit
time or 0

admin: down error - XXX illegal: invalid clock time

admin: resetting previous shutdown at MM/DD/YY HHMM
```

---

drop

Command: drop

Effect: remove a device channel from system control

Usage: drop TTYXXX

This command causes a device channel to be removed from the message coordinator. Any pending output for the channel is lost. If the channel was dialed to the initializer, it is disconnected.

```
Response: please reissue dial command
          (only if channel was dialed)
```

Errors:

```
admin: drop finds ttyXXX not dialed to initializer

admin: ERROR_MESSAGE. drop

admin: Expected argument missing

admin: Ioname not found. drop

admin: ERROR_MESSAGE. drop
```

exec

__Command__: exec, x

__Effect__: Execute special commands

__Usage__: This function should only be used when a system programmer or administrator has given you instructions. The "exec" command permits the installation to make temporary commands for the initializer, by editing a file called "admin.ec". The first argument to exec is the name of the function in "admin.ec" which is selected. Type

```
     exec FUNC ARGS ...
```

to execute function FUNC.

The functions in the version of "admin.ec" which is distributed with the system are:

io	Start IO Daemon
inc	Start incremental dump
cat	Start catchup dump
month	monthly reset of answer table
trace_on	trace answering service
trace_off	stop trace
copy_dump	copy "fast" dump
salv	print online salvager output
network	start up network
abs_dump	dump absentee
set_fdump	set FDUMP number
sfdn, set_fdump_number	..
login_print_dump,lpd	login Print_Dump.SysDaemon
print_dump,pd	process FDUMP
delete_dump,dd	delete FDUMP
print_queues,pq	list total absentee requests
create_ddd	re-create >daemon_dir_dir
net_stop	stop network
net_restart	restart network
net_remove	remove network channel

----

exec

Errors:

depend on the command.

Command: force

Effect: Force answering service to reset itself

Usage: This command is like the "reset" command but does additional resetting. If administrators are unable to "install" system tables, this function can sometimes clear the jam. Type

```
force
```

to force a reset.

Errors:

none

---

go

Command: go

Effect: Second half of answering service initialization

Usage: This command causes all channels attached to the answering service to be initialized, so that they will answer the phone. It is the second half of "startup". Typing "multics" and then "go" is just like "startup" except that the login word is set to a random number, so that only users who know the special word can log in. Type

```
go
```

to cause all lines in the "lines" file to be initialized.

Parts two and three of the system start_up.ec will be executed in response to a "go" command. Part two will be done before user lines are answered, part three will be done afterwards.

A check is made to see if a down command had previously been issued. If one is found, and more than 30 minutes remain before it takes effect, the "down" command will be re-issued automatically.

Errors:

as_init_: type multics first

as_init_: cannot type go twice

as_init_: No lines file found. No consoles will answer unless attached.

as_init_: ttyXXX is restricted name

as_init_: ttyXXX already known

as_init_: No room for ttyXXX

---

go

---

as_init_: tty-dim error on ttyXXX

load_ctl_: configurantion not in tables. X cpu, Y mem, shift Z

---

hmu

Command: hmu

Effect: Print how many users logged in

Usage: This command prints how many users are logged in. Since it also prints the system ID, it can be used to check the success of a "sysid" command. Type

```
     hmu
```

to get the following message:

```
     Multics SYSID; INSTALLATION
     Load = XX.X out of YY.Y units; users = ZZ
     Absentee users = W; Max absentee users = K
```

Errors:

none

Command: log

Effect: Leave message for system programmers

Usage: This command can be used by the operator to enter a line in the system log. Type

```
     log ANY MESSAGE AT ALL
```

to cause the rest of the line to be entered into the log.

---

login

Command: login

Effect: operator login of daemon

Usage: login PERSON PROJECT SOURCE

This command causes the login of a daemon process at operator request. The PERSON.PROJECT must be a registered user with the "daemon" attribute.

Usually, the "login" command is used to cause a daemon process to be logged in without a terminal of its own; such a process sends all its output via the message routing DIM to the message coordinator for output on one of the message coordinator's output destinations, and all input to such a daemon process must be done via the initializer command "reply". (See page 69)

The "login" command can also be used to log in a daemon user which is to have its own terminal channel. If SOURCE is of the form "ttyXXX", and the channel ttyXXX is not in use by the answering service, then the daemon process will be logged in and given channel ttyXXX.

Examples:

```
login IO SysDaemon io

login Retriever SysDaemon tty101
```

Errors:

admin: Expected argument missing. login

admin: login error - tty not dialed to initializer: ttyXXX

daemon_user_manager_: ERROR_MESSAGE. Creating event channel for PERSON PROJECT daemon_user_manager_: ERROR_MESSAGE. declaring event call channel for PERSON PROJECT

daemon_user_manager_: PERS1.PROJ1 already logged in on SOURCE

daemon_user_manager_: Channel error on ttyXXX for PERSON.PROJECT

daemon_user_manager_: cannot login PERSON.PROJECT - ttyXXX is hung up

daemon_user_manager_: ERROR_MESSAGE. Creating process for PERSON PROJECT

daemon_user_manager_: tty failure logging in PERSON PROJECT ttyXXX

daemon_user_manager_: no login PERSON PROJECT REASON

logout

Command: logout

Effect: operator logout of daemon

Usage: logout PERSON PROJECT SOURCE

This command causes the logout of a daemon process at operator request. If PERSON, PROJECT, or SOURCE is "*", all users which match are logged out. SOURCE, or SOURCE and PROJECT, may be omitted, and are then assumed to be "*".

Example: To log out all daemon processes, type

```
     logout * * *
```

Errors:

```
admin: Expected argument missing. logout

daemon_user_manager_: Entry not found. PERSON PROJECT
```

maxunits

Command: maxunits

Effect: Set the maximum number of load units

Usage: This command alters the maximum number of load units allowed on. If it is set to below the current number of units, no users are bumped, but only those users with "guaranteed login" privilege can log in. Type

```
     maxu NNN
```

to set the maximum number of load units to NNN/10.

To cause the system to set the "standard" value of maxunits from the configuration table set by the system administrator in "installation_parms", type

```
     maxu auto
```

The "maxu auto" command is normally issued after every reconfiguration.

Example: To set the maximum load to 41.3 load units, type

```
     maxu 413
```

Errors:

```
admin: Expected arguemnt missing. maxunits

admin: maxumits error - XXX illegal value for maxunits
```

---

mc_list

Command: mc_list

Effect: list message coordinator tables

Usage: This command lists the message coordinator tables, in the form of a list of "define" and "route" commands which could be issued by the operator or the system_start_up.ec to create the current message routing tables.

Errors:
none

__Command__: mc_on, mc

__Effect__: start message coordinator

__Usage__: mc_on

This command causes system control to start the message coordinator.

__Errors__:

system_control_: Ioname not attached. cannot attach mc_i/o

system_control_: Ioname not attached. error from mc_init

message

<u>Command</u>: message

<u>Effect</u>: Edit "message_of_the_day"

<u>Usage</u>: This command invokes the Multics "edm" editor to edit the file "message_of_the_day", which most (but not all) users print out automatically when they log in. Type

```
     message
```

to edit the message. The system will reply

```
     Edit.
```

in red, and accept edit requests. Some of the legal requests are:

i line	to insert text
d	to delete a line
n	to go to the next line
p	to print the current line
-	to back up a line
w	to write the message out
q	to exit from edm, to initializer command level

There are many other requests, and options for these requests. See the Multics Programmer's Manual write-up of the edm command for a long explanation of this command. Note that the "E" request is not permitted. An error message will be typed if the E request is used.

Example: To insert a whole new message, the following sequence of commands and edit requests might be used:

```
mess
Edit.
d99
EOF
iSpecial shutdown at 04:45 for PM.
c/PM/Preventive Maintenance/
Special shutdown at 04:45 for Preventive Maintenance.
w
q
```

It is a good idea to keep the message brief, but not cryptic, and to use no lines longer than 80 characters, for the convenience of users at terminals with narrow carriages.

Errors:

see "edm" writeup in MPM for edm errors

system_control_: no external command processing. use "admin" mode

---

multics

Command: multics

Effect: Bring up special session.

Usage: This command initializes the answering service, but does not answer the telephones. It is the first half of a "startup". A junk login word will be made up, which the users must use to log in on this special session. Type

```
      multics
```

to initialize the answering service and make a random login word. Then, type "go" to cause the lines to answer. To revert to a normal session, type "word login".

Part one of the system_start_up.ec will be executed before the answering service is initialized.

Errors:

```
as_init_: multics already typed

as_init_: Entry not found. >sc1>sat - cannot bring up system

as_init_: Entry not found. >sc1>pnt - cannot bring up system

as_init_: ERROR_MESSAGE. >sc1>SEGNAME - cannot bring up system

as_init_: SAT version inconsistent with declarations used by this
program.

as_init_: installation_parms was missing. Creating new one

dialup_: Entry not found. login_help

lg_ctl_: creating hash table for pnt

lg_ctl_: unable to create hash table for pnt: REASON

lg_ctl_: login word is "XXX"
```

```
load_ctl_: Entry not found. master_group_table

act_ctl_: ERROR_MESSAGE. Creating update channel.

act_ctl_: ERROR_MESSAGE. Cannot unlock >sc1>pdt>PROJ.pdt

act_ctl_: Cannot locate Initializer.SysDaemon pdt entry.

as_meter_: ERROR_MESSAGE. stat_seg

as_meter_: ERROR_MESSAGE. SEGID

up_sysctl_: ERROR_MESSAGE. proj_admin_seg

up_sysctl_: made new proj_admin_seg, check ACL

up_sysctl_: ERROR_MESSAGE. cannot find >sc1>update

daemon_user_manager_: init entered with null DUT ptr. Can't init

device_acct_: ERROR_MESSAGE. device_table

cpg_: ERROR_MESSAGE. creating pit_temp_

tape_opr_: ERROR_MESSAGE. cannot initialize tapes.
```

nostart

Command: nostart, ns

Effect: Inhibit system start_up.ec execution

Usage: This command prevents the subsequent execution of the system_start_up.ec segment until the next system bootload. See page 9 for an explanation of the use of this segment.

This command is only useful if executed in ring 4, before a "startup", "multics", or "go" command.

This command should not be used by the operator unless he has been specifically instructed to do so.

Errors:

none

Command: quit

Effect: send quit to a source process

Usage: quit SOURCE

This command sets a flag in the segment "mc_message" indicating that a quit has been sent. If the source process has called

```
ios_$order (STREAM, "quit_enable", null, status);
```

on one or more of its streams attached through mrd_, the message coordinator will send a QUIT signal to the process by a special supervisor interface.

Errors:

admin: Process lacks permission to alter device status. quit

admin: Expected argument missing. quit

admin: Ioname not found. quit

admin: Ioname not active. quit

redefine

Command: redefine

Effect: interchange one destination with another

Usage: redefine VCONS OLD_DEST NEW_TYPE NEW_DEST

This command removes one destination from a virtual console and adds another. NEW_TYPE and NEW_DEST are as above. If OLD_DEST is a device channel which currently has output queued for it, no more output will be queued but all the queued output will be printed.

Errors:

admin: Expected arguemnt missing. redefine

admin: Ioname not found. redefine

admin: Typename not found. redefine

admin: There is no room to make requested allocations. redefine reload

Command: reload

Effect: Call reloader

Usage: This command initiates a ring-4 reload. This is rarely used, since segments from rings 1, 2, and 3 will not be restored correctly. A warning message will be typed. Type

```
     reload
```

to call the reloader.

If the system libraries or >system_control_1 are to be reloaded, the "cripple" command must also be used.

Errors:

reloader error messages

---

remove

Command: remove

Effect: Remove channel from system

Usage: This command is the opposite of "attach". When a channel is removed, the answering service stops paying any attention to it. If a user is on the channel, he is immediately dumped, with no message. This command should not be used unless the user complains of channel trouble, and cannot be bumped. Once a channel is removed, the telephone associated with the channel will not answer and the channel cannot be used for logins again until an "attach" command reattaches it to the answering service. Type

```
      remove ttyXXX ttyYYY ...
```

to remove the channels ttyXXX, ttyYYY, etc.

Errors:

```
admin: remove not done

admin: remove has illegal args
```

Command: reply, r

Effect: send input line to a source

Usage: reply SOURCE REST OF LINE

This command sends an input line to the given source. The input line is placed in the segment "SOURCE_message" and a wakeup sent to the source. When the source calls to read via mrd_, it will extract the message from the segment.

Errors:

system_control_: Ioname not found. not sent

system_control_: Ioname not active. not sent

system_control_: ERROR_MESSAGE. not sent

---

reroute

Command: reroute

Effect: Reroute output from a source to new virtual console

Usage: reroute SOURCE STREAM OLD_VCONS NEW_VCONS

This command changes the routing table so that the output from SOURCE on stream STREAM will be sent to the virtual console NEW_VCONS instead of OLD_VCONS. This command is equivalent to the sequence

```
deroute SOURCE STREAM OLD_VCONS
route SOURCE STREAM NEW_VCONS
```

Errors:

admin: Expected argument missing. reroute

admin: Ioname not found. reroute

admin: There is no room to make requested allocations. reroute

Command: reset

Effect: Reset answering service

Usage: This command is used to cause the initializer to reset itself. If an unexpected fault happens, it is possible that the initializer will become confused. Type

```
     reset
```

to reset the initializer process and attempt to restart all terminals.

Errors:

none

---

route

Command: route

Effect: direct output from a source to virtual consoles

Usage: route SOURCE STREAM VCONS

This command sends the output from the source SOURCE written on the stream STREAM to the virtual console VCONS. If no entry for SOURCE, or for STREAM under SOURCE, exists in the MRT, one will be created. There may be up to 16 sources. Each source may have up to 8 streams, and each stream may have up to 8 virtual consoles. VCONS must have been previously defined. It is added to the virtual console list for STREAM.

Errors:

admin: Expected arguemnt missing. route

admin: Ioname not found. route

admin: There is no room to make requested allocations. route

shutdown

Command: shutdown

Effect: Shut down Multics supervisor and call BOS

Usage: This command is used after all users are logged out to make a normal exit from Multics. Type

```
     shutdown
```

to shut the system down. If any users are still logged in, the system will ask if you really want to shut down. Usually, you should not. Answering "yes" causes an intentional crash. Answering "no" returns you to initializer command level so that you can bump the users.

This command should only be issued after a "stop" command, unless this is the end of a special session in which "startup" or "multics" was never typed.

Errors:

```
system_control_: XX users still on. Do you want to shut down?
```

---

startup

Command: startup

Effect: Start Multics operation for normal session.

Usage: This command is usually the first thing the operator types when bringing up Multics for a normal user session. Type

```
     startup
```

to initialize the answering service and cause all channels to answer.

All parts of the system_start_up.ec will be executed in response to this command. Part one will be executed before answering service initialization. Part two will then be executed, and then all terminal channels will be instructed to answer. After the terminal channels have been started, part three will be executed.

Errors:

see the list of errors for "multics" (page 62) and "go" (page 50)

Command: stop

Effect: Warn users, then "bump * *" after 3 minutes

Usage: This command is used to begin the shutdown process. Type

```
stop
```

to cause all users to get a message of the form

```
************
From Operator: Multics will shut down in 3 minutes
************
```

The "stop" command also executes a "word shutdown" command, so that no more users may log in. After three minutes, the system will automatically execute a "bump * *" to bump all users. Users with the "nobump" privilege will not be bumped.

The next step in the shutdown procedure is to let the daemons and absentee finish up and log out. Then, type the "shutdown" command.

Issuing the "stop" command again will not cause the bump to be signalled again or the message to be re-sent. If all users are logged out, the "stop" command will say

```
admin: all users are out. You may shut down.
```

If you type "stop" and then change your mind, do the following:

```
unbump * * "shutdown cancelled"
word login
```

```
---------------
 stop
---------------
```

Errors:

```
admin: no arguments are allowed for "stop"
```

storage

Command: storage

Effect: Print out current disk/drum usage

Usage: This command causes the current level of use of all storage devices to be typed out. Type

```
     storage
```

to see the current level of use of the secondary storage devices.

This command calls the "device_meters" command to do the actual printing. The information of use to an operator is on the rows labeled "Left" (the actual number of records left on the device) and the two rows giving the counts of "N Errors" (non-fatal errors) and "F Errors" (fatal errors).

Errors:

none

substty

Command: substty

Effect: swap one terminal channel for another

Usage: substty TTYXXX TTYZZZ

This command causes TTYZZZ to be attached and TTYXXX to be dropped. All output queued for TTYXXX will be placed in the queue for TTYZZZ.

Response: TTYZZZ attached by system control.
same message as for "drop" on TTYXXX

Example: Suppose that the regular typewriter is tty100, and that its printing mechanism jams. To switch all output to tty102, dial tty102 to the initializer (if it is controlled by the answering service) and then type

```
substty tty100 tty102
```

Errors:

admin: Expected arguemnt missing. substty

admin: Ioname not found. substty

admin: substty error - cannot substty for same channel

admin: substty error - tty not dialed to initializer: ttyXXX

admin: drop finds ttyXXX not dialed to initializer sysid

Command: sysid

Effect: Change current system ID

Usage: This command is used to change the system ID typed out by the "who" command and when users dial up. Type

```
     sysid NAME
```

to change the system id to NAME.

Errors:

admin: Expected arguemnt missing. sysid

---

terminate

Command: terminate

Effect: Cause user to get new process

Usage: This command causes the initializer to signal itself that a user should have his old process destroyed and a new process created for him. Type

```
     terminate NAME PROJ MESSAGE TO USER
```

to terminate the process for the user with name NAME and project PROJ. The string MESSAGE TO USER will be typed at the user's console if present. Either or both of NAME and PROJ may be asterisk, meaning everybody.

Type the command

```
     terminate ttyXXX MESSAGE TO USER
```

to cause the user whose channel is ttyXXX to get a new process.

This command is only to be used if the user calls up and requests that his process be terminated.

Example: To cause all users on the project "Multics" to get a process termination, type

```
     terminate * Multics
```

Errors:

admin: Expected argument missing. terminate

admin: terminate has illegal arguments

terminate

```
admin: no terminate signalled to NAME PROJ
```

tape

Command: tape

Effect: Respond to tape requests

Usage: When a user requests the mounting of a tape, a message will come out from the "tape" source like

```
   1233 tape 99 (123071) drive 2 Doe.Multics -- MOUNT ring,auth
```

The operator will mount the tape and then issue the "tape" command to reply to request 99 as follows:

```
      tape 99 ok vcd
```

The general form of the "tape" command is

```
      tape MN FUNC args
```

where MN is the request index, and FUNC may be one of

ok XXX	tape is mounted, authentication is XXX
notape	tape cannot be found
redun	tape already mounted
nodrive	no drive available
shutdown	too close to shutdown
repeat	repeat mount message

The operator may reply to tape requests in any order.

To find out what tapes are mounted, and what tape requests are still pending, type

```
      tape list
```

Errors:

```
admin: Expected argument missing. tape

tape_opr_: "XX" is not a legal tape command

tape_opr_: invalid reply index MN

tape_opr_: no pending request MN

tape_opr_: authentication required MN

tape_opr_: authentication for MN (VOLSER) is not XYZ -- retry

tape_opr_: error mounting MN (VOLSER) on drive D

tape_opr_: user process for MN (VOLSER)  drive  D  terminated  --
dismount
```

----

unbump

Command: unbump

Effect: cancel a pending (timed) bump

Usage: This command cancels a request to bump a user. It cannot be used unless the user has been given some "grace time" in which to clean up and log out. Type

```
unbump Name Proj MESSAGE
```

to cancel a bump for "Name.Proj".

To cancel a bump on a given channel, type

```
unbump ttyXXX
```

Errors:

```
admin: Expected argument missing. unbump

admin: unbump has illegal arguments

admin: no unbump signalled to NAME PROJ
```

Command: undefine

Effect: remove destination from virtual console

Usage: undefine VCONS OLD_DEST

This command removes a destination from a virtual console. If VCONS is left with no destinations and output is routad to it, the output will be typed on the bootload console.

Errors:

admin: Expected arguemnt missing. undefine

admin: Ioname not found. undefine

warn

Command: warn, w

Effect: Send message to user

Usage: This command is used to blast a message onto a user's typewriter, right in the middle of whatever other output he is doing. Type

```
     warn NAME PROJ MESSAGE TO USER
```

to send the message MESSAGE TO USER to the user with name NAME and project PROJ. Either or both of NAME and PROJ may be asterisk, meaning everybody.

If a user to be warned has specified the "-no_warning" ("-nw") argument at login, no warning message will be sent to him, and the operator will be told that the user could not be warned.

This command should not be used for casual communication, because it may splatter the message right in the middle of something the user is typing out. Some users have complained that their output was ruined when one of these messages was sent to them. Use the command only when the system is coming down shortly, or is likely to crash, or when you have an urgent message for a particular user whom you cannot reach by telephone.

Example: To send a message to "Jones.Multics", type

```
     warn Jones Multics Can't find your tape. Call x7739
```

If the user "Jones.Multics" is logged in, he will receive a message of the form

```
     ************
     From Operator: Can't find your tape. Call x7739
     ************
```

Errors:

```
admin: NAME.PROJ has "no_warning"

admin: Expected arguemnt missing. warn

admin: warn error - NAME PROJ not found
```

word

Command: word

Effect: Change login word and dialup buffer

Usage: This command can be used to change either the login word or the dialup buffer typed out when a user dials up, or both. Type

```
     word LOGINWD DIALUP MESSAGE
```

to change the login word to LOGINWD and the dialup buffer to DIALUP MESSAGE. DIALUP MESSAGE may be omitted. If it is, then if LOGINWD is "login" the message buffer is reset; if LOGINWD is "shutdown" the message buffer is set to "Multics is shutting down"; if LOGINWD is anything else, the message buffer is set to "Special session in progress".

If the login word is set to "shutdown", no users will be allowed to log in. If a user dials up, he will be told that the system is shutting down, and his terminal will be hung up immediately.

Example: To leave the login word as normal, but to type a message to all users at login, type

```
     word login Only one CPU until 1300
```

To set up a special session, type

```
     word secret Test session until 0945
```

Errors:

```
admin: Expected arguemnt missing. word

admin: word error - length of login_word must not exceed 8
```

---

word

---

characters

admin: word error - XXX is a reserved word

---

who

__Command__: who

__Effect__: print list of logged-in users

__Usage__: This command causes the list of users logged in to be typed. Type

```
who
```

to get a list of all users, showing name, project, channel ID,

A heading is printed, which gives the system and installation ID, the current and maximum load units, the number of users, the current and maximum number of absentee users, the number of daemon users, the time the system came up, the time of shutdown set by "down", and the time and number of the last crash or shutdown. terminal ID, weight, time of login, and flags.

Anonymous users are flagged with a "*" before the user name. Users who don't list on a regular "who" issued by normal users are flagged with an "N". Users with the "nobump" privilege are flagged with a "+". (These users can only be bumped by channel ID). Users who may be bumped by others in their project, but whose "grace" has not run out are flagged with ">". Users who have been bumped, but still have some of their three minutes' warning time left before automatic logout are flagged with "X".

Absentee users are listed next. Each absentee is flagged with "A", his name is followed by the name of his absin file in parentheses, and his queue and slot number are indicated instead of idcode and channel number.

Daemon users are listed last. Each daemon is flagged with "D" and his source identifier shown.

Example: Here is a sample of the output.

```
Multics 20.4; MIT, Cambridge, Mass.
Load = 5.0 out of 54.0 units; users = 5
Absentee users = 1; Max absentee users = 3
Daemon users = 1
System up since 08/29/71  1752.5
Scheduled shutdown at 08/29/71 2300.0
Last crash (ERF# 342) was at 08/29/71  1712.5

    Login at        TTY  Load   Chan   PN  User ID

08/29/71  1754.2  412   1.0   tty218 +   IO.SysDaemon
          2240.1  226   1.0   tty100 >N  Repair.SysAdmin
          2245.0  none   .5   tty111    *Smith.Class
          2247.1  Q 3   1.0   abs1   A   Roe.Multics (x1)
          1755.8  bk    1.5   bk     D   Backup.SysDaemon
```

Errors:

none