REALTIME (RE)

Function

REALTIME ON informs GEORGE that the requirements of the currently loaded program are such that it must be kept permanently in store and plugged in and, optionally, specifies members of the program that are to have high, fixed priority, above GEORGE. REALTIME OFF terminates such requirements.

Format

1 REALTIME ON, PRIORITY (*list*)

2 REALTIME OFF

where *list* is a list of member numbers. The PRIORITY (*list*) parameter is optional.

Forbidden contexts

NO CORE IMAGE

Execution

FORMAT 1

A check will be made that the user has sufficient units of REALTIME stable budget. If this requirement is satisfied the program will be kept permanently in store and plugged in.

All members of the program will be given priorities above the normal range.

If the PRIORITY parameter is included, the member or members specified will be run at a priority above GEORGE itself. If a member number is given for a member that does not exist, the number will be remembered until that member is introduced, when the priority will take effect. If more than one member is specified their relative priority will be as indicated in the request slip. However the nomination of more than one priority member may be detrimental to the performance of the system and is not recommended. Programs should not rely on the relative priorities of members.

The relative priority of priority members specified by different jobs will be indeterminate.

Second REALTIME ON command

If a second REALTIME ON command with a PRIORITY parameter is given, the nominations will replace those in the previous REALTIME ON command. If a second REALTIME ON command with no PRIORITY parameter is given, any members previously nominated will lose their priority, but the program will retain its realtime status. Apart from these changes in priority, a second REALTIME ON command issued when the program is already real-time will have no effect.

FORMAT 2

All effects of any previous REALTIME ON command will be cancelled. If the program is not realtime the command will have no effect.

Notes

1 Realtime programs should never alter their size, so a CORE or SIZE command or a GIVE/4 extracode issued whilst the program is realtime will not cause a change in the amount of core store available to the program.

2 Certain peripherals, for example, flag setting peripherals, have a realtime requirement and cannot be ONLINEd unless a REALTIME ON command has been issued. Conversely, a REALTIME OFF command is not allowed until such peripherals have been released.

3 In GEORGE 4 realtime programs are run with all of their pages in core and so do no page turning. When REALTIME ON is issued a program's quota will be fixed and set equal to its size. Sparse programs are not allowed to be realtime.

Error messages

USER HAS INSUFFICIENT REALTIME BUDGET AVAILABLE

THERE ARE STILL ONLINE PERIPHERALS WITH REALTIME REQUIREMENTS
PROGRAM TOO BIG–SIZE EXCEEDS MAXQUOTA (GEORGE 4 only)
SPARSE PROGRAMS CANNOT BE REALTIME (GEORGE 4 only)
PARAMETER FORMAT ERROR

Logging messages

time,jobtime REALTIME STARTED, CLOCKED *progtime*
time,jobtime REALTIME FINISHED, CLOCKED *progtime*
CLOCKED *progtime* is only output if *progtime* is non zero.

REDON (RO)

Function

Simulates a command issued by a specified peripheral or indicates to GEORGE that a special tape or disc is to be loaded or engaged.

Format

1 Basic peripherals

REDON *peripheral description, command*

The first parameter defines the peripheral. The second parameter is the command that is to be obeyed as if it had been issued from this peripheral.

2 Magnetic tape

REDON *peripheral description, magnetic tape description*

The first parameter defines the deck.

magnetic tape description has one of two forms:

(a) For Xenotapes

(*tape serial number, name*(*generation number/reel number*))

where *tape serial number* must have an X appended to it. The other components of the description are used for checking and are optional.

(b) For non-standard tapes

name (*generation number/reel number*) (NONSTANDARD)

In this case *name* is the reference for the tape and must be included. The other details are optional.

3 Discs

REDON *peripheral description,serial number*

where *peripheral description* defines the device and *serial number* is a number of the form *#nnnnnn*, which specifies the serial number of the storage unit.

Forbidden contexts

NOT OPERATOR

Execution

1 Basic peripherals

REDON can be used to simulate commands issued from paper tape or card readers. Thus it can be used to identify unheaded paper tape or cards. For example:

REDON U27,DOCUMENT▽FRED

will cause GEORGE to identify the paper tape (for example) on device 27 as the document called FRED.

RO U29.L10.IAA1,JB REMOTE,:MANAGER

will cause GEORGE to treat the card reader IAA1 as if a card bearing the JOB command defined in the REDON had been read.

If the peripheral specified is WRONGed a command error is generated (see below).

2 Magnetic tape

The REDON command for magnetic tape is used when loading a xenotape or a non-standard tape. These tapes should not be loaded in the normal way as the latter will be rejected as non-standard and the former will cause any secure tape with the same serial number to be put into query status. The REDON command works in the following way:

The REDON command must precede the loading of the non-standard tape or xenotape in order to warn GEORGE that special action is required for the tape to be loaded on the specified deck; any tape already loaded on this deck and not in use is first of all unloaded automatically by the command. When GEORGE is ready for the tape to be loaded then the message UNIT *n* RESERVED is output on the operator's console. The operator may now load the tape. If the tape is said to be a xenotape in the command then any name and details given in the tape description are checked against the tape's header label. GEORGE also checks that the tape is not a secure tape, and the tape is finally left positioned after the header label. If the tape is said to be non-standard then GEORGE checks that the tape is not secure and leaves it positioned at start of tape. If the tape does not have a write permit ring and one is required or vice versa then the tape is unloaded (currently without an explanation).

3 Discs

The command indicates that the specified unit is non-standard and may be accessed only by a Q or S trusted program using either mode #1600 or mode #1700. If the non-standard device is disengaged, the command does not terminate until the device is engaged.

Examples

REDON U32,(276212X)	Prepares for the loading of the Xenotape on deck 32.
REDON U31,NEWTAPE (NONSTANDARD)	Prepares for the loading of the non-standard tape, to be addressed as NEWTAPE, on deck 31.
REDON U43,#7302	Prepares for the loading of the non-standard storage unit number 7302 on unit 43.

Error messages

1 Basic peripherals

COMMAND PARAMETER MISSING

NOT ALLOWED TO USE UNIT *z*

THE OPERATOR'S NUMBER PARAMETER IS MISSING

UNIT *z* IS ALREADY ALLOCATED

UNIT *z* IS WRONGED

2 Magnetic tape

UNIT *z* IS ALREADY ALLOCATED

UNIT *z* IS WRONGED

UNIT *z* IS NOT ON THIS INSTALLATION

TAPE ON UNIT *z* HAS FAILED

z DOES NOT SPECIFY NON-STANDARD OR XENOTAPE

NON-STANDARD HEADER LABEL ON UNIT *z*

TAPE ON UNIT *z* IS SECURE

ENTRANT DESCRIPTION PARAMETER MISSING

LOCAL NAME NOT SPECIFIED

RESERVATION ON UNIT *z* CANCELLED

XENOTAPE ON UNIT *z* NOT CORRECTLY IDENTIFIED

XENOTAPES MUST NOT BE NONSTANDARD

3 Discs

REDON HAS BEEN OBEYED FOR UNIT *z*

A STANDARD SU IS LOADED ON UNIT *z*

NON-STANDARD SU *z* IS NOT AVAILABLE

SERIAL NUMBER *z* IS ALREADY ONLINE

SERIAL NUMBER PARAMETER MISSING

UNIT *z* IS NOT ON THIS INSTALLATION

SERIAL NUMBER IS INVALID

REGENERATE (RG)

Function

To reinstate as available for selection jobs in the well, which are currently not marked as candidates for starting.

Format

REGENERATE *<job selection parameters>*

The job selection parameters have one of the following formats:

1 $job_1, job_2, job_3 \ldots$

2 ALL

3 ALLBUT ($job_1, job_2, job_3 \ldots$)

ALLBUT may be abbreviated to AB and each *job* may have any of these forms:

(a) job number

(b) jobname, username

(c) username, jobname

Note: ALL means all jobs in the well not marked as candidates for starting. ALLBUT means all those except the specified jobs. No check will be made that the specified jobs themselves are not candidates for starting.

If, however, a job which was abandoned is restarted while there is still a monitoring file for the job to be listed, the job will be held up until printing is finished. It is, therefore, advisable not to REGENERATE an abandoned job until its monitoring file has been listed.

Forbidden contexts

NOT OPERATOR, REMOTE

Execution

The well entries for the specified jobs are adjusted so that they become candidates for selection. The message:

JOB NUMBER *n* HAS BEEN REGENERATED

or

JOB *jobname, username* HAS BEEN REGENERATED

is output to the operator's console for each regenerated job except when ALL or ALLBUT is specified, in which case the message:

ALL REQUESTED JOBS HAVE BEEN REGENERATED

is sent.

The jobs are entered in the queue for starting according to the time of their original entry into the system. This means that regenerated jobs will probably be restarted as soon as a slot is available.

Examples

REGENERATE 127, 130

REGENERATE AB (593, :FRED, JOB1)

Error messages

JOB *jobname, username* CANNOT BE REGENERATED

JOB NUMBER *n* CANNOT BE REGENERATED

RELEASE (RL)

Function

Releases a peripheral from the current core image.

Format

RELEASE *peripheral name*

Forbidden contexts

NO CORE IMAGE

Execution

If there is a core image with the specified peripheral allocated to it, this peripheral will be released from the program. Otherwise the command has no effect.

The command may be used to release both on-line and off-line peripherals. If a conceptual uniplexer or multiplexer is RELEASEd, all lines attached to it are automatically detached.

Example

RELEASE *CR0

Error messages

None

Logging messages

time FREE *peripheral name, number* TRANSFERS	This message is sent when an off-line peripheral is released by the RELEASE command.
time FREE *peripheral description, number* TRANSFERS	This message is output when an on-line peripheral is released by the RELEASE command.

RENAME (RN)

Function

Renames a file or a magnetic tape.

Format

1 RENAME *entrant description, general local name*

2 RENAME *magnetic tape description* (*MT), *worktape name*

entrant description may be a filename, or a magnetic tape description; if the latter, it must be qualified by (*MT) and must describe a standard tape. With Format 1, applicable entrant description qualifiers are NOWAIT, REPLY, FROZEN.

Forbidden contexts

NO USER

Execution

1 FILES (FORMAT 1 ONLY)

The general local name of the specified file is changed to the form specified by the second parameter, provided that the file is a terminal file belonging to the user (unless the user is suitably privileged, see *Ownership of entrants,* page 175), and that the user has an ERASE trap for it. The command may be used to change a complete name or simply some part of a name, such as the language code or generation number. If a name is specified without details, the existing language code and generation number are retained.

Since dumps of the file are recognised by name, any existing dump is lost, so the file is marked as needing to be dumped.

If the new name is already in use for another file, an error is reported.

2 SECURE TAPES (FORMATS 1 AND 2)

If the second parameter is a general local name (Format 1), the source tape specified by the first parameter has its header label altered to hold the name and details specified by the second parameter and the filestore records for this tape are updated. This is only done if the user has a write trap, the tape belongs to him (unless he is suitably privileged, see *Ownership of entrants,* page 175), and his directory does not hold a file with the local name specified in the second parameter. Information held on the tape is lost when the header label is rewritten. If no generation number is specified in the second parameter, the tape is given a generation number of zero.

If the second parameter is a worktape name (Format 2), and the first parameter specifies a worktape, then the worktape name in the directory is updated. If the first parameter specifies a worktape which does not yet have a worktape name, then the RENAME command will give it the worktape name specified in the second parameter. This converts it to a named worktape.

The RENAME command with a worktape, then, may be used either to alter the name of the worktape's header label (Format 1) or to alter the worktape name by which the tape is referred to in job descriptions (Format 2).

3 INSECURE TAPES (FORMAT 1 ONLY)

The insecure tape specified by the first parameter has its header label altered to hold the name and details specified by the second parameter. Information held on the tape is lost when the header label is rewritten.

Examples

RENAME MYFILE(3/OUT),MYFILE(3/IN)	The language of the file will be changed from OUT to IN.
RENAME :Z.CARDDATA,CD	The local name of the file will be changed from CARDDATA to CD. :Z is the name of a pseudo user whose directory belongs to the proper user.

RN FREETAPE (*MT),OTTOSTAPE	The name of the magnetic tape will be changed from FREETAPE to OTTOSTAPE.
RN SPUD(*MT),!FRED	The worktape to which the job has written the name SPUD in the header label is now known as !FRED in the directory and can be referred to by that name for the rest of the current job.

Error messages

z IS NOT A TERMINAL FILE

z ALREADY EXISTS

ALTERING DIRECTORY ENTRIES REQUIRES OWNERSHIP

MAGNETIC TAPE FAILED

z IS NOT A NAMED WORKTAPE

NAMED WORKTAPE *z* IS ALREADY ONLINE

MULTIFILES ARE NOT ALLOWED WITH THIS COMMAND

RENAMEDIR (RD)

Function

Alters the name in a tape's directory entry when the existing name in the entry is not the same as the name in the tape's header label. (This circumstance can arise when a GEORGE break has occurred and the restored version of the directory does not have the current tape name.)

Format

RENAMEDIR (*tape serial number, general local name*) (*MT)

Forbidden contexts

NO USER

Execution

If the specified tape is owned by the proper user (or the user has ownership access to the current directory, see *Ownership of entrants*, page 175), it is loaded and the name given in the parameter is checked against the header label. If they are the same this name is written to the tape's directory entry. During this process the tape is temporarily marked as being at Query status. If the directory entry is already correct an error will be generated.

Example

RENAMEDIR (365421, FREDSTAPE)(*MT)

Error messages

YOU DO NOT OWN *z*

DESCRIPTION GIVEN DOES NOT AGREE WITH HEADER LABEL

GENERAL LOCAL NAME IS ALREADY IN USE AS A FILENAME

DEVICE TYPE QUALIFIER IMPERMISSIBLE

DEVICE TYPE QUALIFIER MISSING

SERIAL NUMBER MISSING

z IS A XENOTAPE AND MAY NOT BE REFERRED TO BY THIS COMMAND

MAGNETIC TAPE DESCRIPTION FOR THIS COMMAND MUST CONTAIN A LOCAL NAME

z IS NOT KNOWN TO THE LIBRARIAN

DIRECTORY ENTRY ALREADY AGREES WITH HEADER LABEL

REPORT (RP)

Function

Specifies the selection of the monitoring file to be output at the MOP terminal during the execution of a MOP job.

Format

REPORT *action on monitoring file*

The parameters are optional. If none are given, NONE is assumed. For *action on monitoring file* parameters, see Tables 1 and 2 on page 178.3.

Forbidden contexts

NOT MOP

Execution

When a MOP terminal is engaged, the output to the MOP terminal is preset as if

REPORT FULLBUT, COMMANDS

had been issued. When a REPORT command is issued during a job, it changes the categories of MOP output for all levels of the job. The categories of MOP output to the MOP terminal will remain as set by a REPORT command until either they are changed by a subsequent REPORT command or they are reset to FULLBUT, COMMANDS by a LOGOUT or CONNECT command, or by a DISCONNECT command with no job name parameter.

Examples

REPORT FULL

REPORT POSTMORT, LISTING, DISPLAY

Error messages

z IS AN UNKNOWN MONITORING CATEGORY

ALLBUT OR FULLBUT MUST COME FIRST

RESET (RX)

Function

1 Resets the line width at the MOP terminal; that is, to number of print positions to be used

2 Informs GEORGE that a VDU MOP terminal employs a code 1 or code 2 character repertoire

Format

1 Corresponding to function 1

RESET WIDTH*n*

where *n* is the number of print positions

2 Corresponding to function 2

RESET CODE*n*

where *n* is 1 or 2, corresponding to a code 1 or code 2 character repertoire

Forbidden contexts

NOT MOP

Execution

FORMAT 1

Sets the line width of the terminal to that specified by *n*. It is possible to reset only to a width of between 40 and 511 characters. If *n* is less than 40, the width will be reset to 40; if *n* is greater than 511, the width will be reset to 511.

The command remains in effect until another RESET WIDTH*n* command is issued, or until MOP operation ceases, for example when the terminal times out in NOUSER context.

FORMAT 2

Marks a VDU MOP terminal as using character CODE*n*, where *n* is 1 or 2, so further input or output will be appropriately converted.

The command remains in effect until MOP operation ceases, including the terminal timing out in NO USER context. If the command is issued from a MOP terminal which is not a 7181/2 VDU connected via 7900 Series communications processor, it will be accepted but ignored.

Error messages

PARAMETER MISSING

PARAMETER FORMAT ERROR

RESTORE (RS)

The function of this command (to restore a core image from a saved file) can be performed just as well by the LOAD command, and it is recommended that the latter be used in preference.

RESUME (RM)

Function

Restarts an object program (the current core image) or first restores a saved file and then restarts it.

Format

1 RESUME *number,* PARAM(a_1,a_2, . . . a_n)

2 RESUME *file description, number,* PARAM(a_1,a_2, . . . a_n)

Format 1 is used if there is a current core image and format 2 if a file first has to be restored.

The *file description* may be qualified by the entrant description qualifiers NOWAIT, REPLY, FROZEN.

The *number* indicates the entry point relative to word 0 (not, as in the ENTER command, relative to word 20).

The *number* may be omitted (see *Execution,* below).

a_1,a_2, . . . a_n are optional parameters to be passed to the object program when it asks for them. These must conform to the rules that govern the formats of all parameters (see page 160).

Forbidden contexts

NO CORE IMAGE, PROGRAM, BREAKIN (Format 1)

NO USER, PROGRAM, BREAKIN (Format 2)

After the command has been obeyed the context will be CORE IMAGE.

Execution

1 If the *number* parameter is missing the object program is re-entered at the next instruction, as given by the contents of word 8. Otherwise the program is entered at the instruction indicated; that is, word 8 is set to contain the value *number,* all other bits of word 8 are cleared, and word 9 is cleared.

Format 1, therefore, apart from the way it treats word 8, is the same as the ENTER command.

Example

RESUME ,PARAM(10,12,14)	The current program will be re-entered at the next instruction. 10, 12 and 14 are object program parameters (see page 42).

2 The saved file is restored and the object program is re-entered as with format 1. Any existing core image will be overwritten.

Example

RESUME SAVEFILE,20	The saved file will be restored and entered at word 20.

If the size of the program is in excess of the object program quota (an installation parameter), then the following message will be output:

> EXECUTION OF YOUR PROGRAM MAY BE DELAYED AS ITS CORE SIZE EXCEEDS THE PROGRAM QUOTA

Error messages

FORMAT 1

THIS COMMAND IS NOT ALLOWED IN NO CORE IMAGE CONTEXT (if no core image exists.)

z IS A GENERAL PURPOSE LOADER

z HAS A CHECKSUM ERROR

z IS NOT IN BINARY FORMAT

INAPPROPRIATE/INVALID BLOCK TYPE

z HAS NO ENTRY BLOCK

z HAS A BLOCK COUNT ERROR

Logging messages

GEORGE 3

time milltime CORE GIVEN *number,* CLOCKED *progtime*

where *number* is the size of the program.

time milltime DELETED, CLOCKED *progtime* — This message is sent if there was a core image when the RESUME command (format 2) was issued.

CLOCKED *progtime* is output only if *progtime* is non-zero.

GEORGE 4

time milltime SIZE GIVEN *number,* CLOCKED *progtime*

or

time milltime SIZE GIVEN *number* SPARSE, CLOCKED *progtime*

depending on whether the program is dense or sparse, *number* is the size of the program.

time milltime DELETED, CLOCKED *progtime*
MAXQ = *number*K, PT = *number,* USED *number*K — This message is sent if there was a core image when the RESUME command (format 2) was issued.

CLOCKED *progtime* is output only if *progtime* is non-zero.

GEORGE 3 AND GEORGE 4

All these messages occur only if format 2 has been used

time milltime REALTIME FINISHED — This message is sent if there was a core image when the RESUME command was issued and a REALTIME ON command has been obeyed, but not a subsequent REALTIME OFF command.

time FREE *peripheral name, number* TRANSFERS — This message is sent when an off-line peripheral is released by the RESUME command.

time FREE *number,number* TRANSFERS — This message is output when an on-line peripheral is released by the RESUME command.

RETRIEVE (RV)

Function

Enables a file to be retrieved from a dump tape autonomously.

A description of CANCEL RETRIEVE follows this specification.

Format

RETRIEVE *file description*$_1$,*file description*$_2$, . . . ,*file description*$_n$

Each parameter must specify the name of a file.

Forbidden contexts

None.

Execution

If the file is not already on-line, it will be brought on-line by a dump tape activity running independently of the user's job. No trap checking or checking on the files being user-frozen is performed.

Error messages will be produced for command errors, such as an incorrectly formatted name, that can be detected immediately. If the file is already on-line, the following message will be sent in the COMMENT category:

z IS ALREADY ONLINE

z is of the form *:username.general local name*

If several files are already on-line, the message is as follows:

FILES ALREADY ONLINE;- $z_1,z_2,z_3, \ldots$

If the user is at a MOP terminal, the final status of the file (that is, whether or not it has been successfully retrieved) will be indicated to him by one of the following messages:

z IS NOW ONLINE

z IS TEMPORARILY UNAVAILABLE

z IS PERMANENTLY LOST

z DOES NOT EXIST

If the files are not on-line the user will not be able to determine the final status of the file except by attempting to open it (by ASSIGN or LOAD commands, etc.) or by issuing another RETRIEVE or a CANCEL RETRIEVE command. Appropriate messages concerning the status of the file will then be produced.

When a job is set waiting for an implicit retrieve (for instance, when a LOAD command is issued for a file which is not on-line) the following message, indicating that retrieval has been started, is sent to the monitoring file:

z IS BEING RETRIEVED

If the user is at a MOP terminal and does not wish to wait, he can break in and continue to issue commands, such as QUIT; the file will still be retrieved.

The RETRIEVE command may be used in macros to initiate retrieval of files that will be needed during execution of the macro but may not be on-line when it is entered.

Use of the RETRIEVE command will cause the files not to be thrown off-line for a fixed period (normally 150 minutes) from when the command was issued. This period does not include time when the system is not running.

Examples

```
RETRIEVE    :LIB.PROGRAM▽XE20,
:LIB.SUBGROUPS–CM
```

Two activities will be set up to retrieve the files PROGRAM▽XE20 and SUBGROUPS–CM

CANCELLATION OF RETRIEVALS

Function

Cancels a previously issued RETRIEVE command, or an implicit retrieve (see *Execution* below).

Format

CANCEL RETRIEVE *file description*

Forbidden contexts

None, but the command must be issued in the same context as the command that started the retrieval. In addition, if the context is USER, the command must be issued from the same job as the command that started the retrieval; however, in NO USER context retrieval may be cancelled from any source.

Execution

If the retrieval was initiated by a RETRIEVE command it can be terminated by a CANCEL command unless the file is now on-line.

If the retrieval was implicit (for instance, due to a LOAD command) break-in causes it to become autonomous and again a CANCEL command can be used to terminate the retrieval.

Error messages

If no retrieval initiated from this job for this file is outstanding when a CANCEL command is issued, one of the following messages is output:

z IS ALREADY ONLINE

z IS NOT ONLINE

RETURN (RT)

Function

Frees a magnetic tape from the user and returns it, or deletes a named worktape's entry from the job's temporary directory (see *Acquiring worktapes,* page 73) and returns the tape to the worktape store.

Format

RETURN *magnetic tape description*

The magnetic tape description must include the qualifier *MT.

Forbidden contexts

NO USER

Execution

1 If the tape is not a named worktape, the directory entry for the named magnetic tape will be deleted from its directory, provided that the tape belongs to the proper user (or the user has ownership access to the current directory, see *Ownership of entrants,* page 175). The entry for the tape in the index file is updated to show that the tape has been returned to the pool and the user's budget of SPACEMT is updated.

If the tape is loaded, POOL▽TAPE is written to the header label.

Assuming that the operation is successful, the message

MT *number* IS NOW A POOL TAPE

is output on the operator's console; the number is the tape serial number of the tape that has been returned to the pool.

If a tape is RETURNed when it is not loaded, a request to load the tape will not be generated and the tape will be put into query status as it will not have POOL▽TAPE written to its header label. When the tape is next loaded, POOL▽TAPE will be written to its header label and it will be taken out of query status.

2 If the tape is a named worktape, the entry in the job's temporary directory (see page 73) is deleted and the tape becomes available for use as a worktape by other jobs. A named worktape may only be RETURNed by the job that is using it.

Examples

RETURN MAGDATA (500) (*MT)

RETURN (7000, :MP.A (2)) (*MT)

The user :MP in the magnetic tape name must be either the proper user or a pseudo user whose directory belongs to the proper user.

RETURN !WORKTAPE (*MT)

Error messages

DEVICE TYPE QUALIFIER IMPERMISSIBLE

DEVICE TYPE QUALIFIER MISSING

z IS A XENOTAPE AND MAY NOT BE REFERRED TO BY THIS COMMAND

RIGHT (RI)

Function

Makes a peripheral available for use by GEORGE.

Format

RIGHT *peripheral description 1, peripheral description 2, . . .*

where the parameters specify the local or remote peripherals to be made available for use.

Forbidden contexts

NOT OPERATOR

Execution

If this command is issued from a remote console, it may be used only to RIGHT a peripheral in its own cluster. If it is issued from the central console, it may be used to RIGHT any central or remote peripheral.

Provided that the command can be issued from the issuing console and that the peripheral exists, the message

UNIT *peripheral description* RIGHTED

is output on the operator's console from which the command was issued. GEORGE will mark the peripheral RIGHT, irrespective of whether it was previously marked WRONG (see the WRONG command, page 400.33).

Examples

RI 24,U25,IAA2,I62

RI U29.L10.IAA1,IAA6,U60,I73,I72

Error message

z NOT RECOGNISED

RUNJOB (RJ)

Function

Initiates a background job defined by a filed job description.

Format

1 RUNJOB *username, jobname, filename,* PARAM (a_1,a_2, . . . a_n), JOBDATA (b_1,b_2, . . . b_n), NOWELL, *restart parameters*

2 RUNJOB *jobname, filename,* PARAM (a_1,a_2, . . . a_n), JOBDATA (b_1,b_2, . . . b_n), NOWELL, *restart parameters*

Format 1 is issued in NO USER context and the first three parameters are mandatory although the positions of *user name* and *job name* are interchangeable.

Format 2 is issued in USER context and the first two parameters are mandatory.

In both formats the *job name* must not coincide with the name of any other job which the user currently has in the system.

The *filename* is the name of the job description file. If the filename is not a local name then only this file is opened. There is the usual trap checking when the file is opened.

If only a local name is specified then the file specified must be directly below either the current directory, a directory superior to the current directory, or :SYSTEM.MACROS.

Note that if the command is issued in the USER context, although the search for the job description file is made relative to the current directory, the files set up by the command to execute this job are set up below the proper user's *proper* directory, Thus, any budgeting and accounting applies to the proper user and not to the user associated with the current directory. If the command is issued in the NO USER context, the current directory is the proper user's proper directory.

The parameters a_1,a_2, . . . a_n if present, are the job parameters which must conform to the rules that govern the formats of all parameters (see *General descriptions of parameter formats,* page 160). They will be substituted in the job description where parameter substitution is required (denoted by %A, %B etc.). They are passed to the system Journal and so they may also be used to indicate an account code.

The JOBDATA parameter is optional and may be abbreviated to JD. It indicates to the high level scheduler the resources needed by the job. The b_1,b_2, . . . b_n specify the scheduling requirements in the same form as the scheduling information in the JOBDATA command. The NOWELL parameter specifies that the job should be started immediately, even if circumstances would otherwise cause it to go into the well. The restart parameters are any of RESTARTABLE, GRESTARTABLE, and GRNRERUN. NOWELL, RESTARTABLE, GRESTARTABLE and GRNRERUN may be abbreviated to their first four characters.

Forbidden contexts

USER (Format 1)

NO USER (Format 2)

In both cases as a result of the command, the context of the new job becomes USER and OFF-LINE.

Execution

FORMAT 1

A check is made that the named user is in the Dictionary and is permitted to start background jobs. HIGH and NORMAL security users, when starting a job in NO USER context on-line, must type in their password: HIGH security users are not permitted to start background jobs from a reader in NO USER context (see *SECURITY command,* page 397).

The *job name* is then checked to see that is is acceptable (see *Format,* above), and the *filename* is checked to see that the file is entered in either the current directory, :SYSTEM.MACROS or a directory superior to the current directory (they are searched in that order). Provided that an entry is found, the user traps are checked to ensure that the user is allowed to access the file.

If any of the above requirements is not satisfied, the command is abandoned. Otherwise if the job cannot be started immediately, it enters the well. A user with NOWELL privilege can avoid this by including the NOWELL parameter. If a job is to be started immediately by virtue of this parameter, the message

username, jobname BYPASSING THE WELL

is sent to the central operator's console.

The scheduling information is stored in the job's :SYSTEM.JOBLIST entry, and when the system is ready to start the job, the monitoring file is opened with the name given in the first parameter, and the specified filename is issued as a macro with the $a_1, a_2, \ldots a_n$, if any, as parameters. Once the job has started, commands are read and obeyed from the job description file until either an ENDJOB command is encountered or the end of the file is reached at which point the job is terminated. If no ENDJOB command is encountered, the job is suspended for a length of time determined by the WAITTIME installation parameter (see *INSTPARA command,* page 294).

The presence of the restart parameters has no direct effect on the job, but indications of their presence appear in certain output from the WHATSTATE command. Some installations may use these parameters to convey information to operators as to whether a job is suitable for restart in various circumstances.

FORMAT 2

Action is the same as that described for format 1 except that no check is made on the user.

Examples

RUNJOB OLDJOBNO1,:FRED,JOBFILE	OLDJOBNO1 is the name; :FRED is the username; JOBFILE is the name of the (existing) job description file.
RUNJOB J3, JDF3,PARAM (*CR0,*LP0,11,01)	J3 is the job name, JDF3 is the name of the (existing) job description file. The other parameters are job parameters.

Note: Background jobs with full security may be started from a MOP console by means of format 1. The password will be called for and checked.

Error messages

FILE NAME PARAMETER MISSING

JOB NAME PARAMETER MISSING

JOB NAME *z* NOT UNIQUE

THE LIMIT ON THE NUMBER OF JOBS HAS BEEN REACHED

USER NAME PARAMETER MISSING

z HAS OVERDRAWN HIS MONEY BUDGET

z IS NOT A CORRECTLY FORMED NAME (job name or user name)

z IS NOT A REAL USER

USER NAME/PASSWORD INVALID

THE SECURITY OF USER *z* DOES NOT ALLOW THIS COMMAND TO BE OBEYED IN THIS CONTEXT

YOU CANNOT SPECIFY A WORKFILE NAME

JOBDATA RESOURCE(S) INCORRECT

Logging messages

The logging messages for RUNJOB are exactly the same as for JOB, See page 300.

SAVE (SV)

Function

Saves the state of the current core image in a file.

Format

SAVE *file description,device type*

where *device type* is either *CP or *DA and is optional; if it is omitted *DA is assumed. *File description* may be the name of a new file to be created or of an existing file to which the user has WRITE access. It may be followed by the qualifiers TRAPGO and TRAPSTOP to set the required traps for the file; other applicable qualifiers are NOWAIT, REPLY, FROZEN, OWNERACC, APPEND. If *file description* refers to an existing file the *device type* specified (or implied) must correspond to the type of the existing file; that is, if it is a serial file the *device type* must be *CP, and if it is a disc file *device type* must be *DA or omitted. Otherwise an error will result.

Note: *ED or *FD will be accepted as equivalent to *DA in the current mark.

Forbidden contexts

NO CORE IMAGE

Execution

A binary dump of the program in a format acceptable to the LOAD, LOADENTER, RESTORE and RESUME commands is sent to the file specified. The file name can be obeyed as a macro only if the *device type* is *CP.

GEORGE 4 disc type files will contain details of pages with special permission or shareability.

If *device type* is *CP the file produced will be of GRAPHIC type. If *device type* is *DA or is omitted the file will be a disc file in standard binary program format with 128-word buckets.

If the program has any peripherals a message is output to the monitoring file system for each one, in the form:

peripheral name NOT SAVED

This means that if the saved file is restored, these peripherals will be lost.

Notes

1 The SAVE command does not alter the state of a program in any way.

2 When an overlay program is loaded, the program file is (traps permitting) ASSIGNed to the program so that the program can access its overlays. This file is released by SAVE as are all other peripherals connected to the program. This also applies to an ONLINEd magnetic tape in the case of a program overlaying from magnetic tape.

If a program is loaded from a GRAPHIC file, the 'retain load peripheral' bit may be set so that the program can read data following the binary program in the file. As above, traps permitting, the file is ASSIGNed to the program by LOAD and released by SAVE. If this file is ASSIGNed again when the saved file is restored, it is not repositioned at the start of the data.

3 It should be noted that the following details in the saved file are taken from the current state of the program, and they will be attributed to the program when it is reloaded.

The request slip in the saved file contains the size of the core image at the time the SAVE command was issued. A supplementary request slip will be formed if one existed when the program was first loaded or if the program was not in compact mode when the SAVE command was issued. In either case the current branch and address modes will be recorded in the supplementary request slip.

GEORGE 4 disc type files will record the permissions and shareability of each page which does not have full permission or is shareable.

4 In GEORGE 4 the user is warned that if the program has used GIVE/17 to 19 extracodes and contains more than 60 changes of permission across its virtual store, then SAVE to a disc type file will produce a format that the library manipulation programs (such as #XPEU) cannot handle.

5 If the size of the core image to be saved (that is, the non-zero area in GEORGE 3 or the number of accessed pages in GEORGE 4) is larger than can be saved in a filestore file, the SAVE will not take place and an error message (see below) will be output.

6 In GEORGE 4, if an attempt is made to save a core image with special permission areas to a GRAPHIC file, the user is warned by the following message:

SPECIAL PERMISSION AREAS ARE NOT SAVED IN GRAPHIC FILES

and the SAVE continues.

7 If the SAVE command is issued for a core image that has not been completely loaded (that is, a LOAD command has been obeyed but the core image has not been yet formed in core), the SAVE command will force completion of the loading process.

Examples

Command	*Resulting messages*	*Explanation*
SAVE SFILE(3)	*CR0 NOT SAVED *LP0 NOT SAVED	The current core image has two peripherals. These will not be saved, but messages will be sent to the monitoring file system (COMMENT category) indicating that they have been lost. The saved file that will be created or overwritten will be of type *DA and will be entered in the proper user's directory.
CE ! SAVE !		The current core image will be written to the workfile at the top of the stack.

Error messages

FILE NAME PARAMETER MISSING

PROGRAM SIZE TOO LARGE FOR SAVING

z IS NOT A CORRECT PERIPHERAL NAME

SCHEDULE (SC)

Function

Requests an action or actions for a particular job either from the built-in scheduling routines or from the high level scheduler subject program. It may also be used to convey a message to the HLS without reference to a particular job.

Format

SCHEDULE *<job identification>*, *<action(s)>*, MESSAGE (*text*), *GENRES parameters*

MESSAGE may be abbreviated to ME.

The first parameter or group of parameters identify the job for which the specified actions are required and are in one of the following formats:

1 *Job number*

2 *Jobname, username*

3 *Username, jobname*

4 Null (that is, neither of the first two parameters starts with a colon and the first parameter is not numeric)

An *action* parameter must be one of the following:

1 START (ST)

2 EXPRESS (EP)

3 CANCEL (CC)

Action parameters are only allowed if a job identification is present, when more that one may be given.

The MESSAGE parameter is used to convey the specified text (which may be up to twenty-four characters long) to the HLS and as such is only relevant if the HLS is in use. Both *action* and MESSAGE parameters are optional. The combinations of *action* and MESSAGE parameters which are allowed without an HLS and with the standard HLS are specified below.

The *GENRES parameters* are used by operators to convey information to a non-candidate job regarding what happened to it following a dump from which a general restore is being performed. The possible parameters, with abbreviations, are:

GRQUERY (GQ)

GRBREAK (GB)

GRRERUN (GN)

GRRUNDUMP (GP)

These parameters can only be used with a non-candidate job for which the GENRES condition is true. The only action parameter allowed is CANCEL (CC), to reinstate a changed condition, and MESSAGE must not be present.

Forbidden contexts

NOT OPERATOR, REMOTE

Execution

1 HLS OFF. The message parameter is not allowed. In the case of START action, the specified job, which must be a candidate for starting, is taken from the well and started. In the case of EXPRESS action, the job will automatically be started if it is in the well and then expressed when it requests to be made fully started

2 HLS ON. The effect of the action parameter is as for HLS OFF if the HLS is not controlling the starting or fully starting of jobs of this type. Otherwise the HLS is notified of the required action by setting up an HLSTEMPQ block with the appropriate bits set in word 8. This block will contain the text of the MESSAGE

parameter if one is present. The standard HLS will then take action to start, express or cancel express the job.

If the standard HLS does not recognize an action, it will display the message:

INVALID ACTION IN SCHEDULE COMMAND

If it does not recofnize a MESSAGE or if a MESSAGE is given together with an action, it will display the message:

UNRECOGNISED SC/EP MESSAGE: %A

where %A is the first four characters of the message. However in the latter case the action will be performed.

3 GENRES parameter(s) present. This variant of SCHEDULE is independent of the HLS, and is used by operators to convey information to a job that is being restarted from its well record, which has been reconstituted from a dump from which a general restore is being performed. This information can be conveyed to a job whilst it is non-candidate, that is, not able to start. Later, when it has been REGENERATEd and has started; it can test, by the IF command, which conditions have been set TRUE for it, and regulate its action accordingly.

In the absence of any operator action to reset conditions, that corresponding to GRQUERY is TRUE and the others FALSE. The operators can set any of these others to TRUE, thereby automatically falsifying the QUERY condition. If a mistake is made, a further SCHEDULE command with a CANCEL parameter can be used to rectify matters.

Examples

SC FRED, :BERT, START

SCHEDULE, ME (SWITCH)

SC 54321, GRBREAK

Error messages

z DOES NOT EXIST

NO SUCH JOBNUMBER AS *z*

z IS NOT A CORRECTLY FORMED NAME

z IS NOT A VALID ACTION

JOBNAME PARAMETER MISSING

PARAMETER MISSING

z IS ALREADY STARTED

JOB CANNOT BE STARTED

SCREENEDIT (SD)

Function

Provides screen-editing facilities for a VDU MOP terminal. These facilities enable the user to copy or skip through a file displaying on the screen any records in the file. The displayed records may then be edited using the editing facilities provided by the control section of the VDU keyboard, and then sent to a new file.

Format

SCREENEDIT *oldfile, newfile*

oldfile is the file description of the old file to be edited. This parameter is mandatory and must be the first parameter. Relevant qualifiers are NOWAIT, REPLY, FROZEN.

newfile is the file description of the file to which the edited information is to be written. If this parameter is null the screen editor will create a file with the same file description as the file described in the first parameter, except that the file generation number will be incremented by one. Relevant qualifiers are LIMIT, NOWAIT, REPLY, FROZEN, APPEND, TRAPGO, TRAPSTOP, OWNERACC.

Forbidden contexts

NOT VDU MOP, NO USER

This command is not allowed from terminals running under the control of the VIDIMOP subject program.

Execution

Each parameter is checked and the *oldfile* and *newfile* are opened. If an error is detected during this time the *oldfile,* if opened, is closed, and the *newfile,* if opened, is returned to its original state and a standard error message is generated.

When the files have been opened successfully the VDU screen is cleared and an invitation to type a screen editing instruction is given. Each request for an editing instruction is prefixed by a numerical string that gives the value of a pointer to the next record to be accessed in the *oldfile.* The pointer is initially at the first record of the file; that is, record 0.

Shift files are treated as graphic in the course of editing, the output being in GRAPHIC (card) format; any input from VDUs is converted to its graphic equivalent (in particular, lower case letters are converted to upper case). Any records in the *oldfile* more than 80 characters long will be truncated to 80 characters in the *newfile* and on the screen when displayed. Each line on the screen corresponds to one record in the old or new file.

Error message

THIS COMMAND IS NOT PERMITTED FROM THIS DEVICE

SECURITY (SE)

Function

Assigns a security level to the jobs of the proper user.

Format

1 SECURITY HIGH

2 SECURITY NORMAL

3 SECURITY LOW

Forbidden contexts

NO USER

Execution

A user who is at HIGH or NORMAL security is restricted in the type of job (MOP or BACKGROUND) that he is allowed to initiate in NO USER context, although in some cases he is merely requested to give his password. The requirements of the user who issues any of the commands that initiate jobs (JOB, RUNJOB, LOGIN and CONNECT) or create filestore files (INPUT) in NO USER context are tabulated below.

Context	*Command*	*SECURITY HIGH*	*SECURITY NORMAL*	*SECURITY LOW*
MOP,NO USER	INPUT JOB RUNJOB LOGIN CONNECT	Password required	Password required	No password required
BACKGROUND NO USER	INPUT JOB RUNJOB	Not allowed	No password required	No password required

Thus, users in an environment where they are not concerned with security, may set themselves at LOW security and avoid having to type a password each time they LOGIN.

Note that users are initially set at NORMAL security.

Error messages

PARAMETER FORMAT ERROR *z*

'HIGH' MAY BE SPECIFIED ONLY WHEN YOU ARE LOGGED IN

THE SECURITY OF USER *z* DOES NOT ALLOW THIS COMMAND TO BE OBEYED IN THIS CONTEXT

SETPARAM (SP)

Function

Creates, or resets to a new value, one of the variable parameters available to the current command processor level.

Format

SETPARAM *parameter indicator, new value*

The parameter indicator may be represented in one of four ways:

1 As an alphabetic character between A and X inclusive, indicating the number of the parameter. 'A' represents the first parameter, 'B' the second, and so on to the twenty fourth.

2 As an enclosed string, optionally preceded by a decimal number *n* in the range 1 to 24. The number indicates that the parameter being referred to is the *n*th parameter beginning with the specified string. If *n* is missing, a value of 1 is assumed. For the format of an enclosed string, see page 162. In comparisons between the parameter and the string, occurrences of space characters in both the parameter and the string are ignored. If the string is null (for example ()), no comparison is made and it is assumed that the *n*th parameter is the one being referred to.

3 As an alphabetic character between A and X inclusive preceded by the character @. The alphabetic character has the same significance as in 1. However, the parameter referred to by this alphabetic character must itself have one of the formats for a parameter indicator, and will refer the command to the parameter at the level above the current level. Thus @A refers to the current 1st parameter and if this is set to the value B then the 2nd parameter at the level above the current level will be set to the *new value.*

4 As an enclosed string, optionally preceded by a decimal number *n*, itself preceded by the character @. The enclosed string has the same significance as in 2, and the @ character generates the same operation as in 3 except that when the character string is found, the *remainder* of the parameter must be one of the formats for a parameter indicator and indicates a parameter at the level above.

It should be noted that with formats 3 and 4 a parameter several command processor levels above the current level may be accessed but all parameters accessed on the way must exist.

The new value may be represented in one of the following ways:

1 *enclosed string*

Leading and trailing spaces are ignored in setting the new value, but internal spaces are always significant.

2 MESSAGE ($number_1$,$number_2$)

For the format of a number, see page 163.

The value of the parameter is set to all or part of the current program event message (see *Program events*, page 22.2). The part of the message is indicated by the two numbers that follow MESSAGE in parentheses. These give the character positions in the message text of the first and last characters required. For example, where the last message text was ABCDEFG

MESSAGE (2, 8 - 4)

would indicate the new value to be BCD.

If only $number_1$ is given, or if $number_1 = number_2$, it is assumed that the new value is the single character so defined. If MESSAGE alone is given, it is assumed that the whole message text represents the new value. Leading and trailing spaces of the message text are ignored, but internal spaces are significant. If the whole message text is null, the specified parameter becomes null.

3 DISPLAY ($number_1$, $number_2$)

The value of the parameter is set to all or part of the current display (see page 22.2). The part of the display text is indicated in the same way as in case 2.

4 REPLY ($number_1$, $number_2$)

The value of the parameter is set to all or part of the last message sent to the monitoring system, excluding messages in the COMMANDS and ONLINE categories, using a procedure analogous to case 2.

5 *enclosed string* (*number*$_1$, *number*$_2$)

The value of the parameter is set to all or part of the parameter specified, using a procedure analogous to case 2.

6 VALUE *enclosed string*

The value of the parameter is set to the *evaluated* enclosed string, which should be a number. The number is always converted to decimal.

7 TEXT (*number list*)

The value of the parameter is set to the evaluated number list where the latter has the definition given on page 164. Each number, being single length, is regarded as four characters with leading zero characters removed. Thus, #41 will be treated as the character A, #4141 as the two characters AA, #004100 as the two characters A0 and 0 as the character 0. The characters from every number in the list are concatenated and leading and trailing spaces are removed.

8 INSTPARA (*installation parameter*)

The value of the parameter is set to the current value of the specified installation parameter converted to decimal, excepting MINTRACE, MOPTRACE and JOBTRACE, when the setting will have the format *action on monitoring file*, and CONTEXT, when the setting will consist of the appropriate combinations of A, B, C separated by commas.

9 DATE

The value of the parameter is set to the current date in the format DDMMMYY where DD is the day, MMM the month (in alpha characters) and YY is the year.

10 TIME

The value of the parameter is set to the current time in the format HH.MM.SS where HH is the hours, MM is the minutes and SS the seconds.

11 JOB

The value of the parameter is set to the jobname. The command must be used in USER context.

12 USER

The value of the parameter is set to the username without the initial colon. The command must be issued in USER context.

13 ENVIRONMENT (n_1, n_2)

The value of the parameter is set to the reply to the GIVE/5 extracode, converted to decimal. n_1 and n_2 are the lowest and highest numbered bits required: they must lie between 1 and 47 inclusive but the difference between them must not be greater than 22 (that is, no more than 23 bits may be specified at one time). For example, (18, 28) specifies 11 bits and is therefore permitted but (18, 47) specifies 30 bits and is not allowed. If only one bit is required, only n_1 need be specified.

14 SIZE

The value of the parameter is set to the current size, in words, of the core image. The command must be issued in CORE IMAGE context.

15 CORE

This is identical in effect to SIZE.

16 DIRENT (*file description*) (*number*$_1$, *number*$_2$)

The *file description* may not be a workfile name. The value of the parameter is set to all or part of a fixed length character string in the following format:

:*directory name. local name* (*fgn/lang*)

where the fields have the following lengths:

directory name: 12 characters

local name: 12 characters

fgn (file generation number): 4 characters

lang (language code): 4 characters

Each field is of the specified fixed length, left justified and space filled. If the file has no language code, four spaces are inserted.

The specified or implied directory is searched for the specified file name. If no file generation number is specified, the highest generation number in the directory is given.

The part of the resulting string accessed is indicated in the same way as when *new value* is MESSAGE.

A command error is reported if:

(a) This form of the SETPARAM command is issued in NO USER context

(b) The format of *file description* is invalid

(c) The proper user has neither READ access to the specified or implied directory nor any open trap to the specified file

(d) The file description specified is a workfile name

It should be noted that only the first four characters of any operator need be given. So the example in 2 may be written MESS (2, 8 - 4).

Forbidden contexts

A SETPARAM command must not be issued at command processor level zero or at the top break-in level.

Execution

In order to understand the action of SETPARAM, it is necessary to realise that starting a job or calling a macro sets up a new command processor level (see *Command processor levels*, page 4) and that at each command processor level so formed, a block of 24 variable length parameter locations is made available for use by the job or macro. Initially, only the locations corresponding to the parameters (if any) of the JOB, RUNJOB or macro command are set.

SETPARAM is used to manipulate the contexts of any of the 24 locations; the action of SETPARAM can be thought of in terms of altering or creating parameters at the current level. The block of locations for a given level is deleted on termination of the macro or job which set up the level.

It is possible to alter or create parameters at levels above the current macro. This is done using a '*call by name*' technique as described earlier (that is, using the @ character); it is thus possible to communicate between macros.

Examples

A job is currently obeying commands of a macro expansion. The macro command was issued with two parameters 'OLD 1' and 'OLD 2'. The text of the current program message is 'WHATSNEW?'.

1 Any of the following commands will reset 'OLD 2' to 'NEW':

```
SETPARAM   B, (NEW)
SETPARAM   2 "OLD", (NEW)
SETPARAM   2( ),MESSAGE(3*2,8)
SETPARAM   "OLD2" ,MESSAGE(6,16/2)
```

2 The following command will set the fourth current parameter location to 'NEW':

```
SETPARAM   D, "NEW"
```

3 The following version of the command will set the fourth current parameter location to 17

```
SETPARAM    D, VALUE(#10+4+5)
```

4 Either of the following commands, if issued while the sixth parameter at the current level is 'A', will set the first parameter at the command processor level above to 'FAIL':

```
SETPARAM @F,(FAIL)
SETPARAM @6(),(FAIL)
```

Error messages

EXPRESSION *z* INVALID

NO DISPLAY

NO MESSAGE (that is program event message)

NO REPLY

NO CORE IMAGE

PARAMETER SPECIFIED BY *z* NON EXISTENT (in 'call by name' this may refer to a parameter several levels above the parameter specified by the user)

PARAMETER FORMAT ERROR*z* (in 'call by name' this may mean that a parameter several levels above the parameter given by the user has not the format 1, 2, 3 or 4)

NOT ENOUGH COMMAND LEVELS FOR CALL BY NAME

CALL BY NAME FORMAT ERROR

THE COMMAND MAY NOT BE ISSUED DIRECTLY FROM A CONSOLE OR READER

SIZE (SZ)

Function

Alters the size of an object program. The description of the command given here applies only to GEORGE 4. In GEORGE 3 the command has the same effect as the CORE command (see page 242).

Format

SIZE *number*

number represents the required size of the program, in words.

Forbidden contexts

NO CORE IMAGE

Execution

The *number* is rounded up to the next highest multiple of 1024 in GEORGE 4 (or 64 in GEORGE 3). For restrictions on the size of a program see page 31.

A SIZE command may be used to alter the size of both dense and sparse programs but cannot be used to change a dense program to a sparse one or vice versa.

The size of a sparse program cannot be reduced below the area of store the program has used in the current run. If a SIZE command is issued specifying a value less than the area of store already used, the program size will only be reduced to the number of pages used. The actual size given will be indicated by the logging message.

Example

SIZE 8000 The size will be rounded up to 8192 in GEORGE 4.

Error messages

PARAMETER MISSING

PROGRAM SIZE REQUESTED IS INVALID

If a REALTIME ON command has been obeyed but not a REALTIME OFF command:

REALTIME PROGRAMS CANNOT ALTER THEIR SIZE

Logging messages

time milltime SIZE GIVEN *number*, CLOCKED *progtime*

or, if the program is sparse:

time milltime SIZE GIVEN *number* SPARSE, CLOCKED *progtime*

CLOCKED *progtime* is output only if *progtime* is non-zero.

SPEAK (SK)

Function

Causes a message to be sent to a specified operator's console or consoles.

Format

1 SPEAK *,text*

2 SPEAK *console property name,text*

3 SPEAK *,text*,B

Forbidden contexts

NOT OPERATOR

Format 1 may not be issued from the central operator's console.

Format 2 may be issued from any operator's console.

Format 3 may only be issued from the central operator's console.

Execution

FORMAT 1

The specified text is output on the central operator's console.

FORMAT 2

The specified text is output on the specified operator's console (central or remote).

FORMAT 3

The specified text is output on all remote operators' consoles.

The text, as sent, is also output on the console which issues the command. If a message contains commas it must, of course, be enclosed by parentheses or quotes. A console may not SPEAK to itself.

Examples

SPEAK	CENTRAL,I'M SIGNING OFF NOW	either of these will cause the message *console property name*:- I'M SIGNING OFF NOW to be output on both the central operator's console and the originating console.
SPEAK	,I'M SIGNING OFF NOW	
SK	,"MOPPING OFF, 10 MINS",B	will cause the message CENTRAL:- MOPPING OFF, 10 MINS to be sent to all remote operator's consoles.
SK	READING, WE'VE JUST RECEIVED YOUR TAPE PUNCH OUTPUT	will cause an exchange of messages between two remote operators' consoles.
SK	BRACKNELL, (OK, WE'LL SEND SOMEONE OVER TO COLLECT IT)	

Error messages

YOU ARE THE NOMINATED CONSOLE FOR PROPERTY *z*

YOU MAY NOT BROADCAST FROM A REMOTE CONSOLE

z IS NOT A CONSOLE PROPERTY

SPOOL (SO)

Function

Informs GEORGE whether spooling is to take place on an RJE terminal when the specified devices are addressed.

Format

SPOOL *mode parameter, peripheral description*$_1$, *peripheral description*$_2$ . . .

The *mode parameter* is one of the following:

ATTENDED (AT) if spooling is being switched on.

OFF (OF) if spooling is being switched off.

The *peripheral description* describes peripherals which have been declared as spooling devices. These will have been specified in the HARDWARE command if the connection is by way of a character buffering multiplexer, and in the DCP specification tape if the connection is by way of a message buffering multiplexer.

Forbidden contexts

NOT OPERATOR

Execution

If the specified devices have been declared as able to be spooled, operation in the mode specified will begin immediately if no input or output is currently taking place and the multiplexer to which they are connected is MOPped ON. Otherwise, the specified mode will take effect when the current input or output operation is finished, or when the multiplexer is MOPped ON. Paper tape punch spooling is not permitted.

If this command is issued from a remote operator's console, it may refer to devices in that console's cluster only. If it is issued from the central console, there is no such restriction.

Examples

SPOOL AT, I9, I10	Switches spooling on for message buffering identifiers 9 and 10.
SO OFF, IAA1	Switches spooling off for peripheral number 1 on terminal AA on a character buffering system.

Error messages

SPOOLING IS NOT PERMITTED ON UNIT *z*

STARTTIME

Function

Notifies the high level scheduler of a time and date before which the job must not be fully started.

Format

STARTTIME *time, date*

If the earliest start date is omitted, it is assumed to be the day on which the command was issued.

Forbidden contexts

NO USER

Execution

The STARTTIME command is a system macro which makes use of the JOBDATA command. It stores the information given in its parameters in the file :SYSTEM.JOBLIST for use by the high level scheduler.

Examples

STARTTIME 6.30	The job is not to be fully started before 6.30 a.m. on the day on which the command is issued.
STARTTIME 13.30,15/11/68	The job is not to be fully started before 1.30 p.m. on the 15th November 1968.

Error messages

'z' NOT RECOGNISED

PARAMETER MISSING

PARAMETER NULL

STOPLIST (SL)

Function

Deletes all or some entries in the file :SYSTEM.OUTPUT of files waiting to be listed following a LISTFILE command. Note that STOPLIST is not provided to stop a listing which is in progress; TERMINATE may be used to do this.

Format

1 STOPLIST *output peripheral type,* PROPERTY *console property name*

2 STOPLIST *output peripheral type*

3 STOPLIST *user name, job name, output peripheral type, file name*

4 STOPLIST *user name, job name*

Forbidden contexts

NOT OPERATOR, REMOTE (Format 1)

NOT OPERATOR (Formats 2, 3 and 4)

Execution

1 FORMAT 1

All entries corresponding to files waiting to be listed on the specified peripheral type on the cluster specified by the console property name are deleted from the file :SYSTEM.OUTPUT. This format allows the operator to delete all the required entries in the case where the specified peripheral was available when a LISTFILE command was analysed but has become unavailable before the listing could be produced.

If the console property name is not specified, all LISTFILE entries for a peripheral of the specified type are deleted if the listing was intended for a central device.

2 FORMAT 2

All entries corresponding to files waiting to be listed on the specified peripheral type on the cluster the console of which was used to issue the STOPLIST command are deleted from the file :SYSTEM.OUTPUT.

Note that the result is undefined if the specified peripheral is still available and a file is in the process of being listed. Once a peripheral has been allocated by GEORGE for a listing, that listing is considered to be in progress and this command cannot be used to stop the listing.

3 FORMAT 3

The entry or entries in :SYSTEM.OUTPUT, corresponding to the specified file waiting to be listed following a LISTFILE command, are deleted. This format allows the operator to delete the required entry when a user asks that the listing of a file be abandoned. It is assumed that the user requested the listing of the file by a LISTFILE command.

4 All entries in :SYSTEM.OUTPUT that correspond to files waiting to be listed following a LISTFILE command issued by the specified job are deleted.

If the required file is actually being listed, or if the LISTFILE command is being implemented the following message is output on the operator's console:

REQUIRED FILE BEING LISTFILED ON UNIT *n*

where *n* is the operator's number of the device.

The file name specified in the STOPLIST command must be exactly the same as that given in the user's LISTFILE command.

If there are no files of the type specified the following message is output on the operator's console:

NO SUCH FILES

Note that if format 1 or 2 is used, and a peripheral has already been allocated for the listing this message may occur even if the listing does not appear to have started, for example if the peripheral concerned has not yet been engaged.

Error messages

z IS NOT A TERMINAL FILE

INVALID NUMBER OF PARAMETERS

NO PARAMETER

PROPERTY NAME *z* UNKNOWN

z IS NOT A CONSOLE PROPERTY

TAPERIGHT (TH)

Function

Tapes specified by the parameters are freed from the effect of a TAPEWRONG command done in the current context (OPERATOR or USER).

Format

TAPERIGHT *tape serial number$_1$,tape serial number$_2$, . . . ,tape serial number$_n$*

The tape serial numbers must be qualified by *MT. The other qualifiers permitted are OWNERACC and FROZEN.

Forbidden contexts

1 NOT OPERATOR, REMOTE

2 NO USER

Execution

1 OPERATOR context

The 'tape wronged by operator' marker is unset in the tape's entry in :SYSTEM.SERIAL and also, if it is a dump tape, in the tape's entry in :SYSTEM.INCINDEX

2 USER context

If the tape is owned by the proper user (or the user has ownership access to the current directory, see *Ownership of entrants,* page 175) then the 'tape wronged by user' marker is unset in the tape's entry in :SYSTEM.SERIAL and also, if it is a dump tape, in the tape's entry in :SYSTEM.INCINDEX.

All the tapes specified in the parameters will be dealt with apart from those about which error messages are output. Since the tapes are not dealt with in the order they are specified in the parameters, if break-in occurs no assumptions should be made about which tapes have or have not been TAPERIGHTed. When both 'tape wronged by user' and 'tape wronged by operator' markers are unset the tape is available for normal use.

Example

TAPERIGHT (360245) (*MT),(360246)(*MT)

Error messages

DEVICE TYPE QUALIFIER IMPERMISSIBLE

DEVICE TYPE QUALIFIER MISSING

MAGNETIC TAPE DESCRIPTION FOR THIS COMMAND MUST CONSIST OF A SERIAL NUMBER ONLY

z IS A XENOTAPE AND MAY NOT BE REFERRED TO BY THIS COMMAND

SERIAL NUMBER MISSING

SERIAL NUMBER ALREADY GIVEN IN THIS COMMAND

z IS NOT KNOWN TO THE LIBRARIAN

THIS COMMAND IS ONLY ALLOWED IN OPERATOR OR USER CONTEXT

YOU DO NOT OWN *z*

TAPEWRONG (TW)

Function

Tapes specified in the parameters will be recorded as not available for use until a TAPERIGHT command is given in the same context.

Format

TAPEWRONG *tape serial number*$_1$,*tape serial number*$_2$, . . . ,*tape serial number*$_n$

Tape serial numbers must be qualified by *MT. The other qualifiers permitted are OWNERACC and FROZEN.

Forbidden contexts

1 NOT OPERATOR, REMOTE

2 NO USER

Execution

1 OPERATOR context

A 'tape wronged by operator' marker is set in the tape's entry in :SYSTEM.SERIAL and also, if it is a dump tape, in the tape's entry in :SYSTEM.INCINDEX.

2 USER context

If the tape is owned by the proper user (or the user has ownership access to the current directory, see *Ownership of entrants*, page) then the 'tape wronged by user' marker is set in the tape's entry in :SYSTEM.SERIAL and also, if it is a dump tape, in the tape's entry in :SYSTEM.INCINDEX.

In either context once the current user has finished with the tape nobody can access it until all 'tape wronged' markers for the tape have been removed by TAPERIGHT commands.

All the tapes described by the parameters will be dealt with except those about which error messages are output. Since the tapes are not dealt with in the order in which they are specified in the parameters, if break-in occurs no assumptions should be made about which tapes have been or have not been TAPEWRONGed.

If the dump tape is the last tape available containing a particular set of increments, the following message will be output to the operator's console:

MT *n* WAS THE LAST TAPE CONTAINING INCREMENTS *a, b, c,* AND FILES IT HELD WILL NOW BE LOST

The effect of TAPEWRONG can be cancelled by the TAPERIGHT command.

Example

TAPEWRONG (360245) (*MT),(360247)(*MT)

Error messages

DEVICE TYPE QUALIFIER IMPERMISSIBLE

DEVICE TYPE QUALIFIER MISSING

MAGNETIC TAPE DESCRIPTION FOR THIS COMMAND MUST CONSIST OF A SERIAL NUMBER ONLY

z IS A XENOTAPE AND MAY NOT BE REFERRED TO BY THIS COMMAND

SERIAL NUMBER MISSING

SERIAL NUMBER ALREADY GIVEN IN THIS COMMAND

z IS NOT KNOWN TO THE LIBRARIAN

THIS COMMAND IS ONLY ALLOWED IN OPERATOR OR USER CONTEXTS

YOU DO NOT OWN *z*

TERMINATE (TE)

Function

Terminates the activity of a local or remote basic peripheral.

Format

TERMINATE *peripheral description*$_1$, *peripheral description*$_2$ . . . *peripheral description*$_n$

where each parameter specifies a peripheral whose activity is to be terminated.

Forbidden contexts

NOT OPERATOR

Execution

If the command is issued from a remote console, it may be used only for a basic peripheral in its own cluster. If it is issued from the central console it may be used for any local or remote basic peripheral.

Provided that the command can be issued from the issuing console, a check is made that the peripheral is of a basic type (CR, LP, TR, TP, CP). If these conditions are satisfied, the system request is cancelled: otherwise an error is signalled.

If the request is cancelled, subsequent action depends upon the nature of the peripheral activity. If the peripheral is being used by an object program the command has the same effect as the command

CANTDO ENGAGE UNIT *z*

where *z* is the peripheral description (see the CANTDO command, page 226) and the object program will be put into a peripheral-failed condition.

If the peripheral is being used by the system the system operation that is in progress is terminated at the point reached. For example, if an INPUT or JOB command is being implemented, the TERMINATE command will close the file that is being created. If a LISTFILE command is being obeyed, the TERMINATE command will remove the appropriate entry from the output queue. The message

UNIT *z* TERMINATED

will be output on the operator's console from which the command is issued.

Examples

TERMINATE 4

TERMINATE 4,U8,U29.L10.IAA2,I72

Error messages

PERIPHERAL *z* NOT ALLOWED IN THIS COMMAND

UNIT *z* NOT IN REQUIRED STATE

TIME (TI)

Function

Sets the program timer to the specified value.

Format

TIME *milltime*

If neither MINS nor SECS is given in the parameter, seconds are assumed.

Forbidden contexts

NO CORE IMAGE

Execution

The program timer counts down on mill time used by subject or object programs. RESUME, ENTER and LOAD cause the timer, if it is not positive, to be set to the value of the PROGTIME installation parameter.

The program timer's running out is a program event which causes the program to be stopped and a TIME UP message to be sent to the monitoring file system (OBJECT category).

This command overrides the default settings of the program timer set by other commands until

1 Another TIME command is issued.

2 The program is deleted.

3 The program timer runs out.

See *The program timer,* page 19.

Examples

TIME 5

TIME 5SECS

Both these commands will reset the program timer to five seconds.

Error messages

PARAMETER FORMAT ERROR :*z*

TIME PARAMETER MISSING

TRACE (TA)

Function

Limits output to the job's monitoring file to a specified set of categories unless a FULLTRACE command has been issued, in which case no TRACE command in the job will have any effect. The minimum selection of categories allowable is determined by the installation parameter MINTRACE.

Format

TRACE *action on monitoring file*

Forbidden Contexts

NO USER

Execution

When a job is started, the categories of output to its monitoring file are determined by installation parameter (JOBTRACE for background jobs, MOPTRACE for MOP jobs).

The first TRACE command in a job changes the set of categories to that specified in its parameter list. A TRACE command supersedes any previous TRACE commands issued at the same command processor level. Any subsequent TRACE command issued at a lower level changes the category set to include those categories specified in its parameter list which have not been excluded by a TRACE command at a higher level. When control returns to the higher level (at the end of a macro) the category set reverts to that which obtained when the macro was issued.

Any TRACE commands issued at the top break-in command processor level will control tracing at that and lower levels. (Commands issued in BREAK-IN context are at lower levels than the command broken in on.) Immediately after break-in the tracing level will be the same as that at the next higher level; that is, the level being obeyed when break-in occurred.

The minimum tracing level is set by the Installation Parameter MINTRACE and the categories specified by this parameter will always be included in the categories set by a TRACE command, even if they were not specified in its parameter list. Further details are given in *The TRACE system* on page 18.

If a FULLTRACE command is issued, any further TRACE commands in the job will have no effect.

Examples

If MINTRACE is LOGGING:

1	TRACE	LISTING,COMERR	Categories output to the job's monitoring file will be LISTING, COMERR and LOGGING.
2	TRACE	FULLBUT,COMMANDS	All categories will be output except COMMANDS, ENGINEER, JOURNAL, CENTRAL and CLUSTER.
3	TRACE	ALLBUT,COMMANDS,ENGINEER, JOURNAL,OPERATOR	

Error messages

z IS AN UNKNOWN MONITORING CATEGORY

Note

The user should make quite sure that he will not require information which he suppresses (using TRACE) as it will not be available at ENDJOB or LOGOUT time.

TRAPCHECK (TC)

Function

Enables a user to check in what modes he, another user, or a group of users is allowed to open a named entrant.

Format

TRAPCHECK *entrant description, user name,* GROUP

The second parameter may be omitted, in which case the proper user is assumed to be referring to himself.

The GROUP parameter is also optional (see Table 4 on page 178.5).

Except for the first parameter, null parameters are ignored. Duplicated trapmodes are commented on.

Forbidden contexts

NO USER

Execution

A user may only enquire about his own access to an entrant or about another user's access to one of his own entrants. The enquiry includes a check on any modes of access permitted to the named or assumed user as a result of his membership of a group (see TRAPGO).

If the GROUP parameter is given, a check is made only on the group traps of the given user and his superiors; any trap that applies to him individually is ignored.

A reply is sent to the monitoring file system (COMMENT category) specifying the modes in which the user is allowed access to the entrant. This reply has the following format:

PERMITTED ACCESS MODES IN *entrant description* BY *user name: access mode, access mode,. . . .*

The access modes and their meaning are given in Table 4 page 178.5).

Where a check on the access of another user to one of the proper user's entrants indicates that no access is permitted, the following reply is sent:

THIS USER IS NOT ALLOWED ANY ACCESS TO THIS ENTRANT

Where a check by a user on his own access to another user's entrant indicates that no access is permitted, the following reply is sent:

YOU ARE NOT ALLOWED ANY ACCESS TO THIS ENTRANT

Examples

TRAPCHECK MYFILE, :JONES, GROUP

Reply PERMITTED ACCESS MODES IN MYFILE BY :JONES: GROUP, READ

TRAPCHECK (1010, MFDATA(/200), GROUP

Reply PERMITTED ACCESS MODES IN (1010,MFDATA(/200)) BY :PROPERUSER: GROUP, READ

TRAPCHECK OTHERFILE

Reply YOU ARE NOT ALLOWED ANY ACCESS TO THIS ENTRANT

TRAPCHECK MYFILE, :FRED

Reply THIS USER IS NOT ALLOWED ANY ACCESS TO THIS ENTRANT

Notes:

1 See *File security,* page 53, for a full description of the trap system

2 An entrant's traps may be checked even if it, or its directory, is user-frozen

Error messages

ENTRANT DESCRIPTION PARAMETER MISSING

YOU DO NOT HAVE THE AUTHORITY TO ASCERTAIN THE ACCESS OF THIS USER IN THIS ENTRANT

UNRECOGNIZED TRAPMODE PARAMETER

THIS USER IS A PSEUDO USER AND MAY NOT BE GIVEN TRAPS

TRACING AND REPORTING ARE SUCH THAT YOU WILL GET NO OUTPUT FROM THIS COMMAND

TRAPGO (TG)

Function

Permits a specified user or users to have access to an entrant that belongs to the proper user.

Format

TRAPGO *entrant description,username,access mode,access mode . . . . .*

The entrant name must be the name of an entrant that belongs to the proper user. The user name may be that of any user known to the system. If it is omitted the proper user is assumed.

Any number of access modes may be given. The access modes and their meanings are specified in Table 4 on page 178.5.

Any unrecognized parameter is ignored.

Except for the first parameter, null parameters are ignored. Duplicated trapmodes are commented on.

Forbidden contexts

NO USER

Execution

A user trap is set in the relevant entry of the proper user's directory. This contains the name of the user who is being allowed access to this entrant and indicators which specify the modes in which access is permitted.

If GROUP is included as a mode parameter, access will be granted to the named user and to all users inferior to him. A separate trap is set in the entry for the named user's group. If GROUP is specified with the username :MANAGER, then access is granted to all users in the filestore, whether or not they are inferior to :MANAGER.

See *File security,* Chapter 3, for a full description of the trap system.

Examples

TRAPGO MYFILE, :JAMES, GROUP, ALL	:JAMES and all the users inferior to him will be granted access to MYFILE as a GROUP in every mode.
TRAPGO TAPEDATA(/230),WRITE	The proper user will be granted WRITE access to the named magnetic tape.
TRAPGO :Z.F120,:FRED, READ, EXECUTE	:FRED will be granted READ and EXECUTE access to :Z.F120. :Z is a pseudo user whose directory belongs to the proper user.

Error messages

ENTRANT DESCRIPTION PARAMETER MISSING

TRAPMODE PARAMETERS MISSING (if no access modes are given, or they all have the wrong format)

YOU ARE NOT OWNER OF THIS ENTRANT

z IS A XENOTAPE AND MAY NOT BE REFERRED TO BY THIS COMMAND

UNRECOGNIZED TRAPMODE PARAMETER

THIS USER IS A PSEUDO USER AND MAY NOT BE GIVEN TRAPS

TRAPSTOP (TS)

Function

Unsets a group trap or an individual user trap governing specified modes of access to an entrant belonging to the proper user.

Format

TRAPSTOP *entrant description,username,access mode,access mode . . .*

The parameters are as in the TRAPGO command. As with TRAPGO, the user name may be omitted, in which case the proper user is assumed.

Except for the first parameter, null parameters are ignored. Duplicated trapmodes are commented on.

Forbidden contexts

NO USER

Execution

GEORGE searches the directory entry of the named entrant for the appropriate user trap and removes those indicators specified in the mode parameters (other than GROUP). If, as a result of this, no further access is permitted to the named user, the trap is deleted.

If GROUP is given as one of the mode parameters, the appropriate group trap is treated in the same way.

Notes:

1 See *File security*, Chapter 3, for a full description of the trap system.

2 When a user has been denied access to an entrant as an individual he may still be able to access the entrant as a member of a group (and vice versa).

Examples

TRAPSTOP MYFILE, GROUP, WRITE	The proper user's GROUP (if there is one) will be denied WRITE access to MYFILE.
TRAPSTOP XF1(3), :JOHN, APPEND	:JOHN will be denied APPEND access to the named magnetic tape.
TRAPSTOP :PUSER.A1(/AB), :BILL,READ	:BILL will be denied READ access to :PUSER.A1(/AB). :PUSER is a pseudo user whose directory belongs to the proper user.

Error messages

ENTRANT DESCRIPTION PARAMETER MISSING

TRAPMODE PARAMETER MISSING (if no access modes are given, or they all have the wrong format)

YOU ARE NOT OWNER OF THIS ENTRANT

THIS USER IS A PSEUDO USER AND MAY NOT BE GIVEN TRAPS

A DUPLICATED TRAPSTYLE HAS BEEN GIVEN

THE USER REFERRED TO HAS NO SUCH TRAP(S) TO THIS FILE

UNRECOGNIZED TRAPSTYLE PARAMETER

z IS A XENOTAPE AND MAY NOT BE REFERRED TO BY THIS COMMAND

z IS NOT AN ACCESS MODE