

SCF Commands for the Kernel Subsystem

This section describes the following SCF commands that support the Kernel subsystem.

Command	Page
ABORT	6-4
ADD	6-6
ALTER	6-15
CONTROL	6-28
DELETE	6-29
INFO	6-31
NAMES	6-41
START	6-48
STATUS	6-52
STOP	6-62
VERSION	6-66

Other commands that are generally supported by SCF, such as the ASSUME and ENV commands, are not documented in this manual. The *SCF Reference Manual for G-Series RVUs* provides general information about SCF commands. You should be familiar with general SCF usage before using this manual.

Topic	Page
Supported Commands and Object Types	6-2
Sensitive and Nonsensitive Commands	6-3
Wild-Card Support	6-3
Descriptions of SCF commands for the Kernel subsystem	6-4

Supported Commands and Object Types

[Table 6-1](#) lists the SCF commands supported by the Kernel subsystem and the object types to which they apply. The object types are described in [Section 5, SCF Object Types and Object Names](#).

Table 6-1. SCF Commands and Object Types for the Kernel Subsystem

Command	Object Types			
	null	PROCESS	SERVERNET	SUBSYS
ABORT	--	X	--	--
ADD	--	X	--	--
ALTER	--	X	--	X
CONTROL	--	--	--	X
DELETE	--	X	--	--
INFO	--	X	--	X
NAMES	X	X	X	X
START	--	X	X	--
STATUS	--	X	X	X
STOP	--	--	X	--
VERSION	X	X	X	X

X = The command currently supports this object type.

-- = The command does not support this object type.

Sensitive and Nonsensitive Commands

SCF commands are either sensitive or nonsensitive.

Sensitive SCF commands can cause communications to cease if the commands are improperly used. Only a super-group user (255,*n*), the owner of the subsystem, or a member of the group of the owner of the subsystem can issue a sensitive command.

Nonsensitive SCF commands request information or status but do not affect operation. SCP does not perform any security checking on these commands.

[Table 6-2](#) lists the sensitive and nonsensitive SCF commands for the Kernel subsystem.

Table 6-2. Sensitive and Nonsensitive SCF Commands

Sensitive Commands	Nonsensitive Commands
ABORT	INFO
ADD	NAMES
ALTER	STATUS
CONTROL	VERSION
DELETE	
START	
STOP	

If you do not specify an object type or object name in the command, SCF uses the assumed object type and object name (set by the user with a previous ASSUME command) to expand any missing or partially qualified portions of these arguments. Based on the device type of the object named in the command, SCF selects the appropriate subsystem to finish processing the command.

Wild-Card Support

SCF commands for the Kernel subsystem have the following wild-card character support:

- A trailing asterisk (*) for *#gpname* in a PROCESS object name. Refer to the considerations for each command for more specific information.
- An asterisk in place of the *cpu* value in a SERVERNET object name.

ABORT Command (Sensitive Command)

The ABORT command terminates the operation of one or more PROCESS objects as quickly as possible. The object (or objects if the process is running in more than one processor) are left in the STOPPED object state, substate ABORTED, but remain configured in the system configuration database (CONFIG).

```
ABORT [ /OUT file-spec / ] PROCESS $ZZKRN.#gpname
```

PROCESS \$ZZKRN.#*gpname*

is a generic process controlled by the Kernel subsystem manager process. You can omit PROCESS, \$ZZKRN, and #*gpname* if you have specified them with an ASSUME command.

Considerations

- When aborting a generic process configured in multiple processors, you should consider entering a TIMEOUT command (described in the *SCF Reference Manual for G-Series RVUs*) to specify a timeout value that is longer than the default 90 seconds. Several processes can be aborted within 90 seconds, but if you abort a generic process that has been configured as a group (by, for example, the CPU ALL attribute) or if you abort multiple generic processes (by using a wild card in the ABORT command), more time may be needed.
- If a generic process does not seem to enter the STOPPED state, but instead appears to be ABORTING or STOPPING, repeat the ABORT command.
- The ABORT command effectively sets the persistence count to zero.
- After you have entered an ABORT command on a generic process, \$ZPM does not restart the process until the system is loaded. \$ZPM does not restart the process if the processor is reloaded. This is always true regardless of the start mode or persistence settings.
- To restart a generic process without loading the system, use the START PROCESS command, as described under [Restarting an Aborted Generic Process on page 3-16](#).
- If the ABORT command fails with an error, the process stays in the ABORTING object state and a second ABORT command is unable to change the state to STOPPED. Although the process may still be running, it is treated like an aborted process and is not restarted when the processor is reloaded. If you want to ensure the process does not restart when the system is loaded, use the ALTER command to change the start mode to MANUAL or DISABLED.
- You cannot use the ABORT command on \$ZZKRN (the Kernel subsystem manager process).

- You should not use the ABORT command on \$ZZFOX (the FOX monitor process). Unpredictable results can occur.
- You should not use the ABORT command on \$ZM_{nn} (a QIO monitor process) while a client is active. Processor failure can occur.
- Wild-card support is limited to the trailing asterisk (*) for #*gpname*.

Examples

- To abort the #TEMP generic process, type:
-> ABORT PROCESS \$ZZKRN.#TEMP
Or type:
-> ASSUME PROCESS \$ZZKRN.#TEMP
-> ABORT
- To abort multiple instances of a generic process configured in multiple processors, type:
-> ASSUME PROCESS \$ZZKRN.#GP
-> ABORT

```
KERNEL W00028 Process \EAST.$ZZKRN.##$GP00 aborted successfully.  
KERNEL W00028 Process \EAST.$ZZKRN.##$GP01 aborted successfully.  
KERNEL W-00016 Object \EAST.$ZZKRN.##$GP02 is already in STOPPED state.  
KERNEL W-00016 Object \EAST.$ZZKRN.##$GP03 is already in STOPPED state.
```

ADD Command (Sensitive Command)

Use the ADD command to define a process object in the Kernel subsystem and add it to the system configuration database (CONFIG).

```
ADD [ / OUT file-spec / ] PROCESS $ZZKRN.#gpname
    [ , attribute-spec ]...
```

PROCESS \$ZZKRN.#*gpname*

is a generic process controlled by the Kernel subsystem manager process. You can omit PROCESS, \$ZZKRN, and #*gpname* if you have specified them with an ASSUME command.

attribute-spec

identifies the attribute names and values for the object specified in the command. The attributes are:

```
AUTORESTART n
BACKUPCPU n
CPU { ALL | FIRST | FIRSTOF (n,n1,...) | n | (n,n1,...) }
DEFAULTVOL $vol[.subvol ]
EXTSWAP { $vol | [[$vol.]subvol.]filename }
HIGHPIN { ON | OFF }
HOMETERM $device[.#subdevice]
INFILE { $device | [[$vol.]subvol.]filename }
LIBRARY [[$vol.]subvol.]filename
MEMPAGES n
NAME $name
OUTFILE { $device | [[$vol.]subvol.]filename }
PFSSIZE n
PRIMARYCPU n
PRIORITY n
PROGRAM [[$vol.]subvol.]filename
SAVEABEND { ON | OFF }
STARTMODE { KERNEL | SYSTEM | APPLICATION | MANUAL | DISABLED }
STARTUPMSG "text"
STOPMODE { SPI | STANDARD | SYSMSG }
TYPE { FOXMON | SUBSYSTEM-MANAGER | OTHER }
USERID { groupname.username | groupnum,userid }
```

AUTORESTART *n*

specifies the number of times that the \$ZPM persistence manager attempts to restart this process within a 10-minute interval after an abnormal termination (the process abends or stops by a means other than the ABORT command).

If *n* is 0 (the default), the process is not automatically restarted.

If *n* is 1 through 10, the process is restarted as many as *n* times in 10 minutes.

If *n* is greater than 0 when a processor fails, the processor is reloaded and its previously running generic processes are restarted, but the value of *n* is not

decremented. For more information about what conditions decrement the count, see [Table 3-5](#) on page 3-13.

For more information about using the AUTORESTART attribute, see [Persistence Considerations](#) on page 3-12.

BACKUPCPU *n*

specifies the processor in which this process should start its backup process. For more information, see [Controlling Where a Generic Process Starts on page 3-9](#) and [Controlling When a Generic Process Starts on page 3-10](#).

To specify this attribute, you must also specify (or have previously specified) the PRIMARYCPU attribute, but you must not specify the CPU attribute.

The variable *n* can be from 0 through the maximum number of processors (the BACKUPCPU value specified cannot be the same as the PRIMARYCPU value).

CPU { ALL | FIRST | FIRSTOF (*n,n1,...*) | *n* | (*n,n1,...*) }

specifies one or more processors in which to start this process. For more information, see [Controlling Where a Generic Process Starts on page 3-9](#).

The CPU attribute is required if you do not specify the PRIMARYCPU attribute.

ALL

specifies that an instance of the process be started in all processors, even if a processor is currently not up. If you specify ALL, you must limit the process name specified in this ADD command to 1, 2, or 3 alphanumeric characters (a two-digit processor number is appended to the process name).

FIRST

specifies that the process be started in the first available processor.

Because processors 0 and 1 are always the first processors to be loaded, you should avoid configuring too many generic processes with CPU FIRST because this can lead to an uneven load balance among the processors in the system.

FIRSTOF (*n,n1,...*)

specifies that the process be started in the first available processor in the designated group. If the processor in which a process is configured fails (for example, processor 2), the process automatically starts in the next available processor (for example, processor 3).

n

specifies that the process be started in processor *n*.

`(n,n1,...)`

specifies that an instance of the process be started in each specified processor, even if a processor is currently not up. You must limit the process name specified in this ADD command to 1, 2, or 3 alphanumeric characters (a two-digit processor number is appended to the process name).

`DEFAULTVOL $vol[.subvol]`

specifies the default volume and subvolume information sent to this process (in the startup message) when it is started.

If this attribute is not specified, the startup default volume and subvolume is \$SYSTEM.NOSUBVOL.

`EXTSWAP { $vol | [[$vol.]subvol.]filename }`

specifies the volume for or name of the swap file for the default extended data segment of this process. This option applies only to TNS objects.

If this attribute is not specified, the operating system selects a swap file or disk location if the process needs it.

`HIGHPIN { ON | OFF }`

specifies the desired PIN range for the process.

ON (the default) specifies that the process run at a high PIN if both the following are true:

- The high-PIN bit is enabled in the program file (and in the library file or files, if any).
- A high PIN is available.

The \$ZPM persistence manager tries to create this process as a high-PIN process. If no high PIN is available in the specified processor, it tries to create this process as a low-PIN process. If no low PIN is available, the process creation fails.

OFF specifies that the process run at a low PIN, regardless of any other considerations. The \$ZPM persistence manager tries to create this process as a low-PIN process. If no low PIN is available, the process creation fails.

`HOMETERM $device[.subdevice]`

specifies the home terminal to use when starting this process.

If this attribute is not specified, the process has the same home terminal (\$YMIOP.#CLCI) that \$ZPM has when it starts the process.

If you configure a generic process to use the system console (\$OSP on D-series RVUs; \$YMIOP.#CLCI or \$YMIOP.#CNSL on earlier G-series RVUs), you probably should specify HOMETERM \$ZHOME. See [Table 6-3](#).

Table 6-3. Guidelines for Configuring a HOMETERM Value

If the generic process is...	The HOMETERM value should be...
Able to handle errors returned from the home terminal	Set to the default value, \$YMIOP.#CLCI
Not able to handle errors returned from the home terminal	Set to \$ZHOME
Configured to be STARTMODE KERNEL or SYSTEM	Set to \$ZHOME
Configured to be non interactive	Set to \$ZHOME
Configured to be interactive	Set to anything but \$ZHOME
Configured to read from the home terminal	Set to anything but \$ZHOME

Refer to *HP NonStop S-Series Planning and Configuration Guide* for more information about the \$ZHOME process.

```
INFILE { $device | [[ $vol.] subvol.] filename }
```

specifies the input file or process name sent to this process (in the startup message) when it is started.

If this attribute is not specified, the process has the same infile (\$YMIOP.#CLCI) that \$ZPM has when it starts the process.

Do not specify INFILE \$ZHOME.

```
LIBRARY [[ $vol.] subvol.] filename
```

specifies a library file for this process to use when started. The file must be either a file code type 100 (TNS) or 700 (TNS/R) object file.

If you specify the SYSTEM subvolume, SCF searches for the file first on the SYSTEM subvolume and then on the current SYS_{nn} subvolume.

You cannot mix TNS (type 100) and TNS/R (type 700) object code. For example, if the PROGRAM attribute points to a TNS/R object file, then the LIBRARY modifier must also point to a TNS/R library object file.

```
MEMPAGES n
```

specifies the number of 1024-word pages of user data memory to be allocated to this process after it is started. The range is 1 through 64 pages. This option applies only to TNS objects.

If this attribute is not specified or if the value is too low, the value of MEMPAGES becomes that assigned in the program object file for this process, when compiled.

NAME *\$name*

specifies the process name of this process, as recognized by TACL. This attribute is required.

The length limitation is six characters. If you specify the CPU attribute and more than one processor, the NAME value cannot exceed three characters (after the dollar sign). This is because the two-digit processor number is appended to the process name.

OUTFILE { *\$device* | [*[\$vol.]subvol.]filename* }

specifies the output file or process name sent to this process (in the startup message) when it is started.

If this attribute is not specified, the process has the same outfile (\$YMIOP.#CLCI) that \$ZPM has when it starts the process.

If you configure a generic process to use the system console (\$OSP on D-series RVUs; \$YMIOP.#CLCI or \$YMIOP.#CNSL on earlier G-series RVUs), you probably should specify OUTFILE \$ZHOME. See [Table 6-4](#).

Table 6-4. Guidelines for Configuring an OUTFILE Value

If the generic process is...	The OUTFILE value should be...
Able to handle errors returned from the home terminal	Set to use the default value, \$YMIOP.#CLCI
Not able to handle errors returned from the home terminal	Set to \$ZHOME
Configured to be STARTMODE KERNEL or SYSTEM	Set to \$ZHOME

Refer to *HP NonStop S-Series Planning and Configuration Guide* for more information about the \$ZHOME process.

PFSSIZE *n*

specifies the size in 2048-byte pages of the process file segment (PFS) of this process. The range is 64 through 512 pages.

If this attribute is not specified, the size is calculated based on the PFSSIZE setting in the program object file for this process.

Refer to the *Guardian Application Conversion Guide* for more information about how and when to use this attribute.

PRIMARYCPU *n*

specifies the processor in which this process starts its primary process. For more information, see [Controlling Where a Generic Process Starts on page 3-9](#) and [Controlling When a Generic Process Starts on page 3-10](#).

You must specify either the CPU or PRIMARYCPU attribute, but not both in the same command.

The variable *n* can be from 0 through the maximum number of processors.

PRIORITY *n*

specifies the priority to use when starting this process. The range is 1 through 199. If this attribute is not specified, the priority is the same as the current SCF session minus 1.

If a process must be higher than 199, it must set its own priority by calling PROCESS_SETINFO_.

PROGRAM [[*\$vol.*] *subvol.*] *filename*

specifies the program object file for this process to use when started. The file must be either a file code type 100 (TNS) or 700 (TNS/R) object file. This attribute is required.

If you specify the SYSTEM subvolume, SCF searches for the file first on the SYSTEM subvolume and then on the current SYS*nn* subvolume.

You cannot mix TNS (type 100) and TNS/R (type 700) object code. For example, if the PROGRAM attribute points to a TNS/R object file, then the LIBRARY modifier must also point to a TNS/R library object file.

SAVEABEND { ON | OFF }

specifies whether a saveabend file is created if this process stops abnormally. This attribute overrides the SAVEABEND setting in the PROGRAM file for this process.

ON Specifies that a saveabend file is created automatically if the process ends abnormally.

OFF Specifies that a saveabend file is not created automatically if the process ends abnormally.

If this attribute is not specified, the SAVEABEND setting in the program object file for this process is used to determine the setting of this attribute.

STARTMODE { KERNEL | SYSTEM | APPLICATION | MANUAL | DISABLED }

specifies when the process is started. The default value is DISABLED.

KERNEL The process is started early during a system load or processor reload.

SYSTEM The process is started as the final stage of system load or processor reload.

APPLICATION	The process is started after the system load or processor reload is finished.
MANUAL	The process can be started by the user any time after system load or processor reload is finished.
DISABLED	The process is not started unless you change the STARTMODE attribute to one of the values listed above.

See [Start Mode Considerations](#) on page 3-11 for a more information about using STARTMODE when configuring your own generic process.

STARTUPMSG "text"

specifies a text message to be sent to the \$ZPM persistence manager when the generic process is started. The length of this message can be 1 through 128 characters.

You can use the startup message to handle specification of the backup processor when configuring a process pair. To do that, specify the following text, including the less than (<) and greater than (>) symbols:

<BCKP-CPU>

The value you specify for the BACKUPCPU attribute is displayed on the StartupMessage line of an INFO PROCESS command (displays the configured backup processor number). For examples on how to do this, see [Creating a Generic Process as a Process Pair](#) on page 3-20.

If you do not specify the BACKUPCPU attribute, the \$ZPM persistence manager ignores (that is, it does not give an error condition for) any <BCKP-CPU> specification in the startup message.

STOPMODE { SPI | STANDARD | SYSMSG }

specifies what method the \$ZPM persistence manager should use when aborting the generic process. The default value is STANDARD.

SPI	stops the generic process by sending it a SPI STOP command (as defined in the <i>SPI Common Extensions Manual</i>).
STANDARD	stops the generic process by using the PROCESS_STOP_ procedure (as defined in the <i>Guardian Procedure Calls Reference Manual</i>).
SYSMSG	stops the generic process by sending it an internal system message.

TYPE { FOXMON | SUBSYSTEM-MANAGER | OTHER }

specifies the type of generic process being added. The default value is OTHER.

FOXMON specifies that this process is a FOX monitor process, as described in the *ServerNet/FX Adapter Configuration and Management Manual*.

Note that you cannot use the ALTER command to change the TYPE to or from FOXMON. Instead, you must delete and re-add the process.

SUBSYSTEM-MANAGER specifies that this process is a subsystem manager process, and that it participates in the \$ZPM reload check-in protocol. Note that \$ZZLAN and \$ZZFOX are exceptions; their TYPE values are OTHER and FOXMON, respectively.

OTHER specifies that this process is not a subsystem manager process or FOXMON process.

USERID { *groupname.username* | *groupnum, usernum* }

specifies the creator access ID under which this process executes. USERID must specify an existing operating system user ID.

The user ID of the current SCF session determines what user IDs can be configured:

- For the super ID (255,255), the user ID can be set to any user on the system.
- For any other super-group user (255,*n*), the user ID is set to the user ID of the current SCF session. (This is the default value).

Considerations

- For information on how to use the ADD PROCESS command, see [Adding a Generic Process](#) on page 3-18.
- If you enter a CONFIRM ON command (described in the *SCF Reference Manual for G-Series RVUs*) before the ADD command, SCF displays the following message in response to a successful ADD command:

Add accepted by KERNEL: PROCESS \system.\$ZZKRN.#newprocess

- After adding a generic process, SCF places it in the STOPPED object state. Use the START command (described on page [6-48](#)) to start the process.
- If you receive a warning message in response to an ADD command, SCF has accepted the command attributes (unless you also receive an error message in response to the same command). To correct or change an attribute (concerning the warning message), use the ALTER command.
- If \$ZZKRN cannot find a specified file on \$SYSTEM.SYSTEM, it then searches for the file on the current \$SYSTEM.SYS_{nn}. This search algorithm allows you to

change the operating system version yet keep the same attribute values for a process.

- When specifying a file name, avoid including the system name (unless the file must reside on a specific system). Omitting the system name allows the process to be system-independent.
- Wild cards are not supported for the ADD command.

Examples

- To add a process specifying only required attributes, type:

```
-> ADD PROCESS $ZZKRN.#MY-OWN-PROCESS,      &
    NAME $GT72,                                &
    PROGRAM $SYSTEM.SYSTEM.NULL,              &
    CPU 3
```

- To configure the \$ZZLAN SLISA subsystem manager process in the \$SYSTEM.ZSYSCONF.CONFIG system configuration database, type:

```
-> ADD PROCESS $ZZKRN.#ZZLAN,                  &
    AUTORESTART 10,                            &
    HOMETERM $ZHOME,                          &
    INFILE $YMIOP.#CLCI,                      &
    OUTFILE $ZHOME,                          &
    PRIMARYCPU 0,                             &
    BACKUPCPU 1,                              &
    DEFAULTVOL $SYSTEM.SYSTEM,                &
    NAME $ZZLAN,                              &
    PRIORITY 180,                             &
    PROGRAM $SYSTEM.SYSTEM.LANMAN,            &
    STARTMODE KERNEL,                        &
    STARTUPMSG "<BCKP-CPU>",                  &
    TYPE OTHER,                              &
    USERID SUPER.SUPER
```

- To configure the \$ZZWAN WAN subsystem manager process in the \$SYSTEM.ZSYSCONF.CONFIG system configuration database, type:

```
-> ADD PROCESS $ZZKRN.#ZZWAN,                  &
    AUTORESTART 10,                            &
    HOMETERM $ZHOME,                          &
    INFILE $YMIOP.#CLCI,                      &
    OUTFILE $ZHOME,                          &
    PRIMARYCPU 0,                             &
    BACKUPCPU 1,                              &
    DEFAULTVOL $SYSTEM.SYSTEM,                &
    NAME $ZZWAN,                              &
    PRIORITY 180,                             &
    PROGRAM $SYSTEM.SYSTEM.WANMGR,            &
    STARTMODE KERNEL,                        &
    STARTUPMSG "<BCKP-CPU>",                  &
    TYPE SUBSYSTEM-MANAGER,                  &
    USERID SUPER.SUPER
```

ALTER Command (Sensitive Command)

Use the ALTER command to change one or more attribute values of an object.

```
ALTER [ / OUT file-spec / ] [ object-spec ]
      [ , attribute-spec ]...
```

object-spec

specifies one of the following object type and object name combinations.

<i>object-type</i>	<i>object-name</i>
PROCESS	\$ZZKRN.# <i>gpname</i>
SUBSYS	\$ZZKRN

attribute-spec

identifies the attribute names and values for the object specified in the command.
The specific attributes are described in the following subsections.

The ALTER PROCESS command is described in the following subsection.

The ALTER SUBSYS command is described on page [6-23](#).

ALTER PROCESS Command

Use the ALTER PROCESS command to change one or more attributes of a process controlled by the Kernel subsystem manager process.

```
ALTER [ / OUT file-spec / ]
      PROCESS $ZZKRN.#gpname [ , attribute-spec ]...
```

PROCESS \$ZZKRN.#*gpname*

is the name of a process controlled by the Kernel subsystem manager process.
You can omit PROCESS, \$ZZKRN, and #*gpname* if you have specified them with an ASSUME command.

attribute-spec

identifies the attribute names and values for the object specified in the command.
The attributes are:

```
AUTORESTART n
BACKUPCPU n
CPU { ALL | FIRST | FIRSTOF (n,n1,...) | n | (n,n1,...) }
DEFAULTVOL $vol[.subvol ]
EXTSWAP { $vol | [[$vol.]subvol.]filename }
HIGHPIN { ON | OFF }
HOMETERM $device[.#subdevice]
INFILE { $device | [[$vol.]subvol.]filename }
LIBRARY [[$vol.]subvol.]filename
MEMPAGES n
NAME $name
OUTFILE { $device | [[$vol.]subvol.]filename }
PFSSIZE n
PRIMARYCPU n
PRIORITY n
PROGRAM [[$vol.]subvol.]filename
SAVEABEND { ON | OFF }
STARTMODE { KERNEL | SYSTEM | APPLICATION | MANUAL | DISABLED }
STARTUPMSG "text"
STOPMODE { SPI | STANDARD | SYSMSG }
TYPE { FOXMON | SUBSYSTEM-MANAGER | OTHER }
USERID { groupname.username | groupnum,usenum }
```

AUTORESTART *n*

specifies the number of times that the \$ZPM persistence manager attempts to restart this process within a 10-minute interval after an abnormal termination (the process abends or stops by a means other than the ABORT command).

If *n* is 0 (the default), this process is not automatically restarted.

If *n* is 1 through 10, this process is restarted as many as *n* times in 10 minutes.

If *n* is greater than 0 when a processor fails, the processor is reloaded and its previously running generic processes are restarted, but the value of *n* is not decremented. For more information about what conditions decrement the count, see [Table 3-5](#) on page 3-13.

For more information about using the AUTORESTART attribute, see [Persistence Considerations](#) on page 3-12d.

BACKUPCPU *n*

specifies the processor in which this process should start its backup process. For more information, see [Controlling Where a Generic Process Starts on page 3-9](#) and [Controlling When a Generic Process Starts on page 3-10](#).

To specify this attribute, you must also specify (or have previously specified) the PRIMARYCPU attribute, but you must not specify the CPU attribute.

The variable *n* can be from 0 through the maximum number of processors (the BACKUPCPU value specified cannot be the same as the PRIMARYCPU value).

`CPU { ALL | FIRST | FIRSTOF (n,n1,...) | n | (n,n1,...) }`

specifies one or more processors in which to start this process. For more information, see [Controlling Where a Generic Process Starts on page 3-9](#).

Specifying the CPU attribute clears any earlier configured PRIMARYCPU and BACKUPCPU attributes.

ALL

specifies that an instance of the process be started in all processors, even if a processor is currently not up. If you specify ALL, you must limit the process name specified in this ADD command to 1, 2, or 3 alphanumeric characters (a two-digit processor number is appended to the process name).

FIRST

specifies that the process be started in the first available processor.

Because processors 0 and 1 are always the first processors to be loaded, you should avoid configuring too many generic processes with CPU FIRST because this can lead to an uneven load balance among the processors in the system.

FIRSTOF (*n,n1,...*)

specifies that the process be started in the first available processor in the designated group. If the processor in which a process is configured fails (for example, processor 2), the process automatically starts in the next available processor (for example, processor 3).

n

specifies that the process be started in processor *n*.

(*n,n1,...*)

specifies that an instance of the process be started in each specified processor, even if a processor is currently not up. You must limit the process name specified in this ADD command to 1, 2, or 3 alphanumeric characters (a two-digit processor number is appended to the process name).

DEFAULTVOL *\$vol[.subvol]*

specifies the default volume and subvolume information sent to this process (in the startup message) when it is started.

If this attribute is not specified, the startup default volume and subvolume is \$SYSTEM.NOSUBVOL.

```
EXTSWAP { $vol | [[ $vol.]subvol.]filename }
```

specifies the volume for or name of the swap file for the default extended data segment of this process. This option applies only to TNS objects.

If this attribute is not specified, the operating system selects a swap file or disk location if the process needs it.

```
HIGHPIN { ON | OFF }
```

specifies the desired PIN range for the process.

ON (the default) specifies that the process run at a high PIN if both the following are true:

- The high-PIN bit is enabled in the program file (and in the library file or files, if any).
- A high PIN is available.

The \$ZPM persistence manager tries to create this process as a high-PIN process. If no high PIN is available in the specified processor, it tries to create this process as a low-PIN process. If no low PIN is available, the process creation fails.

OFF specifies that the process run at a low PIN, regardless of any other considerations. The \$ZPM persistence manager tries to create this process as a low-PIN process. If no low PIN is available, the process creation fails.

```
HOMETERM $device[.#subdevice]
```

specifies the home terminal to use when starting this process.

If this attribute is not specified, the process has the same home terminal (\$YMIOP.#CLCI) that \$ZPM has when it starts the process.

If you configure a generic process to use the system console (\$OSP on D-series RVUs; \$YMIOP.#CLCI or \$YMIOP.#CNSL on earlier G-series RVUs), you probably should specify HOMETERM \$ZHOME. See [Table 6-3](#) on page 6-9.

Refer to *HP NonStop S-Series Planning and Configuration Guide* for more information about the \$ZHOME process.

```
INFILE { $device | [[ $vol.]subvol.]filename }
```

specifies the input file or process name sent to this process (in the startup message) when it is started.

If this attribute is not specified, the process has the same infile (\$YMIOP.#CLCI) that \$ZPM has when it starts the process.

Do not specify INFILE \$ZHOME.

`LIBRARY [[$vol.]subvol.]filename`

specifies a library file for this process to use when started. The file must be either a file code type 100 (TNS) or 700 (TNS/R) object file.

If you specify the SYSTEM subvolume, SCF searches for the file first on the SYSTEM subvolume and then on the current SYS_{nn} subvolume.

You cannot mix TNS (type 100) and TNS/R (type 700) object code. For example, if the PROGRAM attribute points to a TNS/R object file, then the LIBRARY modifier must also point to a TNS/R library object file.

`MEMPAGES n`

specifies the number of 1024-word pages of user data memory to be allocated to this process after it is started. The range is 1 through 64 pages. This option applies only to TNS objects.

If this attribute is not specified or if the value is too low, the value of MEMPAGES becomes that assigned in the program object file for this process, when compiled.

`NAME $name`

specifies the process name of this process, as recognized by TACL. This attribute is required.

The length limitation is six characters. If you specify the CPU attribute and more than one processor, the NAME value cannot exceed three characters (after the dollar sign). This is because the two-digit processor number is appended to the process name.

`OUTFILE { $device | [[$vol.]subvol.]filename }`

specifies the output file or process name sent to this process (in the startup message) when it is started.

If this attribute is not specified, the process has the same outfile (\$YMIOP.#CLCI) that \$ZPM has when it starts the process.

If you configure a generic process to use the system console (\$OSP on D-series RVUs; \$YMIOP.#CLCI or \$YMIOP.#CNSL on earlier G-series RVUs), you probably should specify OUTFILE \$ZHOME. See [Table 6-4](#) on page 6-10.

Refer to *HP NonStop S-Series Planning and Configuration Guide* for more information about the \$ZHOME process.

`PFSSIZE n`

specifies the size in 2048-byte pages of the process file segment (PFS) of this process. The range is 64 through 512 pages.

If this attribute is not specified, the size is calculated based on the PFSSIZE setting in the program object file for this process.

Refer to the *Guardian Application Conversion Guide* for more information about how and when to use this attribute.

PRIMARYCPU *n*

specifies the processor in which this process starts its primary process. For more information, see [Controlling Where a Generic Process Starts on page 3-9](#) and [Controlling When a Generic Process Starts on page 3-10](#).

Specifying this attribute clears an earlier configured CPU attribute. You must specify either the CPU or PRIMARYCPU attribute, but not both in the same command.

The variable *n* can be from 0 through the maximum number of processors.

PRIORITY *n*

specifies the priority to use when starting this process. The range is 1 through 199. If this attribute is not specified, the priority is the same as the current SCF session minus 1.

If a process must be higher than 199, it must set its own priority by calling `PROCESS_SETINFO_`.

PROGRAM [[*\$vol.*] *subvol.*] *filename*

specifies the program object file for this process to use when started. The file must be either a file code type 100 (TNS) or 700 (TNS/R) object file. This attribute is required.

If you specify the SYSTEM subvolume, SCF searches for the file first on the SYSTEM subvolume and then on the current `SYSnn` subvolume.

You cannot mix TNS (type 100) and TNS/R (type 700) object code. For example, if the PROGRAM attribute points to a TNS/R object file, then the LIBRARY modifier must also point to a TNS/R library object file.

SAVEABEND { ON | OFF }

specifies whether a saveabend file is created if this process stops abnormally. This attribute overrides the SAVEABEND setting in the PROGRAM file for this process.

ON Specifies that a saveabend file is created automatically if the process ends abnormally.

OFF Specifies that a saveabend file is not created automatically if the process ends abnormally.

If this attribute is not specified, the SAVEABEND setting in the program object file for this process is used to determine the setting of this attribute.

STARTMODE { KERNEL | SYSTEM | APPLICATION | MANUAL | DISABLED }

specifies when the process is started. The default value is DISABLED.

KERNEL	The process is started early during a system load or processor reload.
SYSTEM	The process is started as the final stage of system load or processor reload.
APPLICATION	The process is started after the system load or processor reload is finished.
MANUAL	The process can be started by the user any time after system load or processor reload is finished.
DISABLED	The process is not started unless you change the STARTMODE attribute to one of the values listed above.

See [Start Mode Considerations](#) on page 3-11 for a more complete discussion of using STARTMODE when configuring your own generic process.

STARTUPMSG "text"

specifies a text message to be sent to the \$ZPM persistence manager when the generic process is started. The length of this message can be 1 through 128 characters.

You can use the startup message to handle specification of the backup processor when configuring a process pair. To do that, specify the following text, including the less than (<) and greater than (>) symbols:

<BCKP-CPU>

The value you specify for the BACKUPCPU attribute is displayed on the StartupMessage line of an INFO PROCESS command (displays the configured backup processor number). This is more completely described under [Creating a Generic Process as a Process Pair](#) on page 3-20.

If you do not specify the BACKUPCPU attribute, the \$ZPM persistence manager ignores (that is, it does not give an error condition for) any <BCKP-CPU> specification in the startup message.

STOPMODE { SPI | STANDARD | SYMSG }

specifies what method the \$ZPM persistence manager should use when aborting the generic process. The default value is STANDARD.

SPI stops the generic process by sending it a SPI STOP command (as defined in the *SPI Common Extensions Manual*).

STANDARD stops the generic process by using the PROCESS_STOP_ procedure (as defined in the *Guardian Procedure Calls Reference Manual*).

SYMSG stops the generic process by sending it an internal system message.

TYPE { FOXMON | SUBSYSTEM-MANAGER | OTHER }

specifies the type of generic process being altered. The default is OTHER.

FOXMON specifies that this process is a FOX monitor process, as described in the *ServerNet/FX Adapter Configuration and Management Manual*.

Note that you cannot alter to or from TYPE FOXMON without first deleting and readding the process.

SUBSYSTEM-MANAGER specifies that this process is a subsystem manager process, and that it participates in the \$ZPM reload check-in protocol. Note that \$ZZLAN and \$ZZFOX are exceptions; their TYPE values are OTHER and FOXMON, respectively.

OTHER specifies that this process is not a subsystem manager process or FOXMON process.

USERID { *groupname.username* | *groupnum, usernum* }

specifies the creator access ID under which this process executes. USERID must specify an existing operating system user ID.

The user ID of the current SCF session determines what user IDs can be configured:

- For the super ID (255,255), the user ID can be set to any user on the system.
- For any other super-group user (255,*n*), the user ID is set to the user ID of the current SCF session. This is the default.

ALTER PROCESS Considerations

- For a description of how to use the ALTER PROCESS command, see [Altering a Generic Process on page 3-25](#).

- If you enter a CONFIRM ON command (described in the *SCF Reference Manual for G-Series RVUs*) before the ADD command, SCF displays the following message in response to a successful ADD command:

```
ALTER accepted by KERNEL: PROCESS \system.$ZZKRN.#process
```

- Even if you receive a warning message in response to an ALTER command, SCF has accepted the command (unless you also receive another error message in response to the same command). If you need to correct or change an attribute in response to the warning message, use the ALTER command again.
- If \$ZZKRN cannot find a specified file on \$SYSTEM.SYSTEM, it then searches for the file on the current \$SYSTEM.SYS_{nn}. This search algorithm allows you to change the operating system version, yet keep the same attribute values for a process.
- When specifying a file name, avoid including the system name (unless the file must reside on a specific system). Omitting the system name allows the process to be system-independent.
- Wild cards are not supported for the ALTER command.

ALTER PROCESS Example

The following example alters a generic process to run in processor 4 instead of processor 3:

```
-> ALTER PROCESS $ZZKRN.#MY-OWN-PROCESS, CPU 4
```

ALTER SUBSYS Command

Use the ALTER SUBSYS command to change one or more system attributes of the Kernel subsystem.

```
ALTER [ / OUT file-spec / ]
      SUBSYS $ZZKRN [ , attribute-spec ]...
```

SUBSYS \$ZZKRN

is the name of the Kernel subsystem manager process. You can omit SUBSYS and \$ZZKRN if you have specified them with an ASSUME command.

attribute-spec

identifies the attribute names and values for the object specified in the command. The attributes are:

```
DAYLIGHT_SAVING_TIME { TABLE | USA66 | NONE }
POWERFAIL_DELAY_TIME n
RESIDENT_TEMPLATES [[ $SYSTEM. ] subvol. ] filename
SYSTEM_NAME \sysname
SYSTEM_NUMBER n
TIME_ZONE_OFFSET [ + | - ] [ h ] h [ : mm ] NONRESIDENT_TEMPLATES
[[ $SYSTEM. ] subvol. ] filename
```

```
DAYLIGHT_SAVING_TIME { TABLE | USA66 | NONE }
```

specifies the daylight-saving time algorithm to be used when the system clock is set.

See also the procedure [Changing the System Time Attributes](#) on page 2-5. Changes to this attribute take effect the next time the system is loaded.

TABLE

specifies that a table of DST (daylight-saving time) transitions is to be loaded at system-load time. To initialize the DST transitions, you use either the ADDDSTTRANSITION command (documented in the *TACL Reference Manual*) or the ADDDSTTRANSITION procedure (documented in the *Guardian Procedure Calls Reference Manual*).

You also must initialize the DST table with at least one DST transition that is less than the current date and time, and at least two DST transitions that are greater than the current date and time before the SETTIME command is entered. Refer to the *Guardian Procedure Calls Reference Manual* for considerations related to use of a DST table.

USA66

specifies that your location follows the rules set by the United States by the Uniform Time Act of 1966 for daylight-saving time. In the United States, DST begins at 2:00 a.m. on the first Sunday in April, when clocks are advanced by one hour; DST ends at 2:00 a.m. on the last Sunday in October, when clocks are set back by one hour.

NONE

specifies that neither TABLE nor USA66 are to be used when system time is set. This is the default.

```
NONRESIDENT_TEMPLATES [[ $SYSTEM. ] subvol. ] filename
```

specifies the location of the Event Management Service (EMS) nonresident template file (file code 839 or 844). The procedure for using this option is

described on page [2-1](#). Changes to this attribute take effect immediately. See also the note on page [6-25](#).

`POWERFAIL_DELAY_TIME n`

specifies the maximum time, in seconds, that the operating system is allowed to wait prior to initiating a shutdown of operations when system power failure is imminent. The range of *n* is 0 through 300 seconds; 30 seconds is the default.

See also the procedure [Changing the Power-Failure-to-Shutdown Time Interval](#) on page 2-3. Changes to this attribute take effect immediately.

NonStop S-series servers have the ability to continue to operate from battery power for some time after the main system power has been removed. This calculated time is based on the number of internal hardware devices and I/O enclosures that must be kept operational, as well as on the capabilities of the batteries.

After a power failure, if the system power has not been restored for some time, the operating system shuts down all system operations in an orderly manner. This is essential to ensure a successful recovery from the power failure.

The actual time the operating system can wait before shutdown is calculated at the time of the power failure. The operating system uses the smaller of the calculated time value and the `POWERFAIL_DELAY_TIME` value to determine how long it waits before starting to shut down system operations.

You must take care when configuring the `POWERFAIL_DELAY_TIME` attribute. The operating system continues to process data until *n* seconds expires. Unexpected errors might occur if peripherals used by the system (for example, modems, Ethernet hubs and routers, and tape drives not contained within a system enclosure) are not powered by uninterruptible power supplies (UPSs) during this time.

`RESIDENT_TEMPLATES [[$SYSTEM.] subvol.] filename`

specifies the location of the Event Management Service (EMS) resident template file (file code 839 or 844). The procedure for using this option is described on page [2-1](#). Changes to this attribute take effect immediately.

Note. If you change the location of the EMS template files using the ALTER command, the `INSTALL^TEMPLATES` program permanently changes the location of the EMS template files. As a result, when you next run DSM/SCM, even though the Build and Apply creates new EMS templates, the subsequent system load invokes the EMS templates previously specified to the `INSTALL^TEMPLATES` program. To use the `RTMPATE` and `TEMPLATE` EMS template files installed in the new `SYSnn` by DSM/SCM, you must use the following ALTER command:

```
-> ALTER, RESIDENT_TEMPLATES $SYSTEM.SYSTEM.RTEMPLATE, &
    NONRESIDENT_TEMPLATES $SYSTEM.SYSTEM.TEMPLATE
```

SYSTEM_NAME \sysname

specifies the name of the system in an Expand network. Each Expand system name must be unique within the network. The first character is alphabetic; the following six characters are alphanumeric. The default is \NONAME.

For more information, see the *HP NonStop S-Series FastPath Guide*. Changes to this attribute take effect the next time the system is loaded.

SYSTEM_NUMBER *n*

specifies the node number of the system in an Expand network. Each Expand node number must be unique within the network. The range is 0 through 254. The default is 254.

For more information, see the *HP NonStop S-Series FastPath Guide*. Changes to this attribute take effect the next time the system is loaded.

TIME_ZONE_OFFSET [+ | -] [*h*] *h* [: *mm*]

specifies an offset of standard civil time (SCT) from Greenwich mean time (GMT) in hours and minutes, where *hh* is in the range 00 through 23 hours and *mm* is in the range 0 through 59 minutes. The default is 0. SCT does not include daylight-saving time (DST).

See also the procedure [Changing the System Time Attributes](#) on page 2-5. Changes to this attribute take effect the next time the system is loaded.

Some examples of time-zone offsets are:

TIME_ZONE_OFFSET	0:00	!London
TIME_ZONE_OFFSET +	01:00	!Paris
TIME_ZONE_OFFSET +	05:30	!Bombay
TIME_ZONE_OFFSET +	09:00	!Tokyo
TIME_ZONE_OFFSET -	05:00	!New York
TIME_ZONE_OFFSET -	8:00	!California

When you load the system the first time after changing the TIME_ZONE_OFFSET value, you must then enter a TACL SETTIME command to correct the system time, which is incorrect by the amount of the change to the TIME_ZONE_OFFSET value.

ALTER SUBSYS Considerations

- Wild cards are not supported for the ALTER SUBSYS command.
- If the ALTER command is not successful, SCF returns an error message saying the requested operation was not completely successful, plus a warning message identifying each failed operation. To correct or change an attribute in response to a warning message, enter another ALTER command.
- If \$ZZKRN cannot find a specified file on \$SYSTEM.SYSTEM, it then searches for the file on the current \$SYSTEM.SYS*nn*. This search algorithm allows you to

change the operating system version, yet keep the same attribute values for a process.

- When specifying a file name, avoid including the system name (unless the file must reside on a specific system). Omitting the system name allows the process to be system-independent.

ALTER SUBSYS Examples

- The following example shows how to specify new EMS template files:

```
-> ALTER SUBSYS $ZZKRN,                                &
    RESIDENT_TEMPLATE    $SYSTEM.TEMPLATE.TNEW,        &
    NONRESIDENT_TEMPLATE $SYSTEM.TEMPLATE.NRTNEW
```

- The following example shows how to set the time-zone offset for a location in California:

```
-> ALTER SUBSYS $ZZKRN, TIME_ZONE_OFFSET -8:00
```

CONTROL Command (Sensitive Command)

Use the CONTROL command to power down a NonStop S-series server.

- △ **Caution.** Refer to the *HP NonStop S-Series Operations Guide* for the system power-off procedure and when to use this command.

You must be in interactive mode to use this sensitive command.

```
CONTROL [ /OUT file-spec/ ] SUBSYS $ZZKRN , SHUTDOWN
```

SUBSYS \$ZZKRN

is the name of the Kernel subsystem. You can omit SUBSYS and \$ZZKRN if you have specified them with an ASSUME command.

SHUTDOWN

specifies that the system perform an orderly shutdown of its hardware, prior to unplugging it from the external power supply.

Consideration

Entering this command generates an operator message that reports the command, the time the command was entered, and the group name and user name of the person issuing the command.

Example

The following command powers off a NonStop S-series server:

```
-> CONTROL SUBSYS $ZZKRN , SHUTDOWN
```

DELETE Command (Sensitive Command)

Use the DELETE command to remove a process object from the Kernel subsystem and from the system configuration database (CONFIG). Only objects that were added with the ADD command can be deleted.

```
DELETE [ / OUT file-spec / ] PROCESS $ZZKRN.#gpname
```

PROCESS \$ZZKRN.#*gpname*

is the name of a process controlled by the Kernel subsystem manager process. You can omit PROCESS, \$ZZKRN, and #*gpname* if you have specified them with an ASSUME command.

Considerations

- [Deleting a Generic Process](#) on page 3-29 describes how to use the DELETE command. Specifically, you must put a generic process in the STOPPED object state before deleting it.
- If you enter a CONFIRM ON command (described in the *SCF Reference Manual for G-Series RVUs*) before the ADD command, SCF displays the following message in response to a successful ADD command:

```
Delete accepted by KERNEL: PROCESS \system.$ZZKRN.#process
```

- It is recommended that you enter a TIMEOUT command (described in the *SCF Reference Manual for G-Series RVUs*) to specify a timeout value that is larger than the default 90 seconds when deleting a generic process configured in multiple processors. Several processes can likely be deleted within the 90 second default. But if you delete a generic process that has been configured as a group (by, for example, the CPU ALL attribute), or if you start multiple generic processes (by using a wild card in the ABORT command), more time may be needed.
- You cannot use the DELETE command on the \$ZZKRN Kernel subsystem manager process itself.
- Wild-card support is limited to the trailing asterisk (*) for #*gpname*. However, you cannot use an asterisk when deleting a subsystem manager. To delete a process configured with TYPE SUBSYSTEM-MANAGER, you must specify the complete #*gpname*.

Examples

- The following example shows how to delete a generic process named #MY-OWN:
-> DELETE PROCESS \$ZZKRN.#MY-OWN

- The following example shows how to delete all generic processes whose names begin with \$ZZKRN.#MY*:

```
-> CONFIRM ON  
-> DELETE PROCESS $ZZKRN.#MY*  
Delete accepted by KERNEL: PROCESS \EAST.$ZZKRN.#MYPROC
```

INFO Command

Use the INFO command to display configuration information, including attribute values, for the specified object. For a generic process, this configuration information displays the attribute values set by its ADD command.

This is a nonsensitive command.

```
INFO [ / OUT file-spec / ] [ object-spec ] [ , DETAIL ]
```

The value of *object-spec* is one of the following object type and object name combinations:

<i>object-type</i>	<i>object-name</i>
PROCESS	\$ZZKRN[. <i>#gpname</i>]
SUBSYS	\$ZZKRN

The INFO PROCESS command is described in the following subsection. The INFO SUBSYS command is described on page [6-38](#).

INFO PROCESS Command

Use the INFO PROCESS command to display configuration information about the specified process.

```
INFO [ / OUT file-spec / ]  
PROCESS $ZZKRN[.#gpname ] [ , DETAIL ]
```

PROCESS \$ZZKRN

is the name of the Kernel subsystem manager process. You can omit PROCESS and \$ZZKRN if you have specified them with an ASSUME command.

This form of the command causes the display to show the current values for the Kernel subsystem manager process.

PROCESS \$ZZKRN.*#gpname*

is the name of a generic process controlled by the \$ZZKRN Kernel subsystem manager. You can omit PROCESS, \$ZZKRN, and *#gpname* if you have specified them with an ASSUME command.

This form of the command causes the display to show the configured values for the specified generic process.

DETAIL

causes SCF to display detailed information about the specified process.

INFO PROCESS Consideration

Wild-card support is limited to the trailing asterisk (*) for *#gpname*.

INFO PROCESS Summary Display Format

The format of the summary display for the INFO PROCESS command (without the DETAIL option) is described here. See also the examples on page [6-35](#).

-> INFO PROCESS \$ZZKRN.*#gpname*

```
NONSTOP KERNEL - Info PROCESS \system.$ZZKRN.#gpname
```

Symbolic Name	*Name	*Autorestart	*Program
<i>gpname</i>	<i>\$proc</i>	<i>n</i>	
<i>\$vol.subv.file</i>			

An asterisk (*) before an attribute name indicates that its attribute value can be changed with the SCF ALTER command.

The following terms appear in the preceding display:

Symbolic Name	The symbolic name of a generic process, as specified in the ADD command. For the Kernel subsystem manager, this name is ZZKRN.
Name	The name of the process, as specified by the NAME attribute of the SCF ADD or ALTER command, and as recognized by TACL. If this generic process is configured in more than one processor, its name ends in <i>nn</i> , representing a two-digit processor number.
Autorestart	The persistence count of the generic process; that is, the number of times the \$ZPM persistence manager attempts to restart the process in ten minutes if it goes down abnormally. This is configured by the AUTORESTART attribute of the SCF ADD or ALTER command. If a generic process with a configured AUTORESTART value greater than 0 is aborted by a processor failure, the process is restarted when the processor reloads, and its Autorestart value is not decremented.
Program	The location of the program file for the generic process, as specified in the ADD or ALTER command.

INFO PROCESS Detailed Display Format

The format of the display for the INFO PROCESS command (with the DETAIL option) is described here. See also the examples on page [6-36](#), page [6-36](#), and page [6-37](#).

-> INFO PROCESS \$ZZKRN.#*gpname*, DETAIL

```

NONSTOP KERNEL - Detailed Info Process \EAST.$ZZKRN.#gpname

*AutoRestart.....n
*BackupCPU.....n
*CPUList.....n
*DefaultVolume.....$vol.subv
*ExtSwap.....$vol.subv.file
*HighPIN.....{ ON | OFF }
*HomeTerminal.....$term[. #subdev ]
*InFile.....$vol.subv.file
*Library.....$vol.subv.file
*MemPages.....n
*Name.....$process
*OutFile.....$vol.subv.file
*PFSSize.....n
*PrimaryCPU.....n
*Priority.....n
*Program.....$vol.subv.file
*SaveAbend.....{ ON | OFF }
*StartMode.....KERNEL
*StartupMessage....."text"
*StopMode.....STANDARD
*Type.....OTHER
*UserId.....SUPER.n          (255,n)

```

An asterisk (*) before an attribute name indicates that its attribute value can be changed with the SCF ALTER command.

The following terms appear in the preceding display. Refer to the ADD command (starting on page [6-6](#)) for more information about each attribute and value.

AutoRestart	<p>The persistence count of the generic process; that is, the number of times the \$ZPM persistence manager attempts to restart the process in ten minutes if it goes down abnormally. This is configured by the AUTORESTART attribute of the SCF ADD or ALTER command.</p> <p>If a generic process with a configured AUTORESTART value greater than 0 is aborted by a processor failure, the process is restarted when the processor reloads, and its Autorestart value is not decremented.</p>
BackupCPU	The number of the backup processor in which this process starts its backup process.
CPUList	The processor or processors (from 0 through ALL) in which this process starts. This value is specified by the CPU attribute in the most recent ADD or ALTER command for this process.

DefaultVolume	The default volume and subvolume information sent to this process (in the startup message) when it starts.
ExtSwap	The extended swap file name or disk location for this process to use when it starts.
HighPIN	The desired PIN range for the process.
HomeTerminal	The name of the home terminal used when this process starts.
InFile	The input file information sent to this process (in the startup message) when it starts.
Library	The library file name this process uses when it starts.
MemPages	The number of 1024-word pages of user data memory allocated to this process after it starts.
Name	The name of the process, as specified by the NAME attribute of the SCF ADD or ALTER command, and as recognized by TACL. If this generic process is configured in more than one processor, its name ends in <i>nn</i> , representing a two-digit processor number.
OutFile	The output file information sent to this process (in the startup message) when it starts.
PFSSize	The size, in kilobytes, of the process file segment (PFS) of the process.
PrimaryCPU	The number of the primary processor in which this process starts its primary process.
Priority	The priority of the process when it was configured, or, if it is not currently running, the priority level it will have when it is next started.
Program	The location of the program file for the generic process, as specified in the ADD or ALTER command.
SaveAbend	Whether or not a saveabend file is created if this process stops abnormally.
StartMode	Whether or not the \$ZPM persistence manager starts this process automatically at system load or initial processor reload. A process with a MANUAL or DISABLED start mode is not automatically started at system load or initial processor reload.
StartupMessage	The startup text message sent to this process when it starts.

StopMode	What method the \$ZPM persistence manager uses when aborting this generic process.
Type	Whether the process is a subsystem manager process, a user-created generic process, or a FOX monitor process.
UserId	The user ID under which this process executes. The default value is the user ID of the current SCF session.

INFO PROCESS Examples

- The following example gives current values for the \$ZZKRN Kernel subsystem manager process. As shown in the Name column in the display, \$ZZKRN is also the name recognized by TACL for the \$ZZKRN Kernel subsystem manager process.

```
-> INFO PROCESS $ZZKRN.#ZZKRN
```

```
NONSTOP KERNEL - Info PROCESS \EAST.$ZZKRN.#ZZKRN

Symbolic Name      *Name      *Autorestart *Program
ZZKRN              $ZZKRN    10          $SYSTEM.SYSTEM.OZKRN
```

- The following example gives current values for the #TEMP generic process:

```
-> INFO PROCESS $ZZKRN.#TEMP
```

```
NONSTOP KERNEL - Info PROCESS \EAST.$ZZKRN.#TEMP

Symbolic Name      *Name      *Autorestart *Program
TEMP               $TEMP      0           $SYSTEM.SYSTEM.TEMP
```

- The following example displays all the generic processes currently configured on system \WEST:

```
-> INFO PROCESS \WEST.$ZZKRN.*
```

```
NONSTOP KERNEL - Info PROCESS \WEST.$ZZKRN.*

Symbolic Name      *Name *Autorestart *Program
CEV-SERVER-MANAGER-P0 $ZCVP0 5 $SYSTEM.SYSTEM.CEVSMX
CEV-SERVER-MANAGER-P1 $ZCVP1 5 $SYSTEM.SYSTEM.CEVSMX
OSM-APPSRV R      $ZO SM 10 $SYSTEM.SYSTEM.APPSRVR
OSM-CIMOM          $ZCMOM 5 $SYSTEM.SYSTEM.CIMOM
OSM-CONFLH-RD      $ZOLHI 0 $SYSTEM.SYSTEM.TACL
OSM-OEV            $ZOE V 10 $SYSTEM.SYSTEM.EVTMGR
QIOMON             $ZMnn 10 $SYSTEM.SYSTEM.QIOMON
ROUTING-DIST       $TSMRD 0 $SYSTEM.SYSTEM.TACL
SP-EVENT           $ZSPE 10 $SYSTEM.SYSTEM.ZSPE
TSM-SNMP           $TSMS 5 $SYSTEM.SYSTEM.SNMPAGT
TSM-SRM            $ZTSM 5 $SYSTEM.SYSTEM.SRM
ZLOG               $ZLOG 5 $SYSTEM.SYSTEM.EMSACOLL
ZTCP0              $OSMM0 0 $SYSTEM.SYSTEM.TACL
ZTCP1              $OSMM1 0 $SYSTEM.SYSTEM.TACL
ZZKRN              $ZZKRN 10 $SYSTEM.SYSTEM.OZKRN
ZZLAN              $ZZLAN 10 $SYSTEM.SYSTEM.LANMAN
ZZSTO              $ZZSTO 10 $SYSTEM.SYSTEM.TZSTO
ZZWAN              $ZZWAN 10 $SYSTEM.SYSTEM.WANMGR
```

- The following example gives detailed configuration information about the \$ZZKRN subsystem manager process.

```
-> INFO PROCESS $ZZKRN.#ZZKRN, DETAIL
```

```
NONSTOP KERNEL - Detailed Info PROCESS \SUN.$ZZKRN.#ZZKRN

*AutoRestart.....10
*BackupCPU.....1
*CPU.....Not Specified
*DefaultVolume.....$SYSTEM.SYSTEM
*ExtSwap.....Not Specified
*Highpin.....ON
*HomeTerminal.....$ZHOME
*InFile.....Not Specified
*Library.....Not Specified
*MemPages.....Not Specified
*Name.....$ZZKRN
*OutFile.....$ZHOME
*PFSSize.....Not Specified
*PrimaryCPU.....0
*Priority.....180
*Program.....$SYSTEM.SYSTEM.OZKRN
*SaveAbend.....ON
*StartMode.....KERNEL
*StartupMessage.....<BCKP-CPU>
*StopMode.....STANDARD
*Type.....OTHER
*UserId.....SUPER.SUPER ( 255,255 )
```

- The following example gives detailed configuration information about the #TEMP generic process. Note that because #TEMP was configured to start in all

processors (the CPU value is ALL), the name is the configured name (\$GP) plus two digits representing the processor number part of the name.

-> INFO PROCESS \$ZZKRN.#TEMP, DETAIL

```

NONSTOP KERNEL - Detailed Info PROCESS \EAST.$ZZKRN.#TEMP

*AutoRestart.....10
*BackupCPU.....Not Specified
*CPU.....ALL
*DefaultVolume.....$SYSTEM.NOSUBVOL
*ExtSwap.....Not Specified
*Highpin.....ON
*HomeTerminal.....Not Specified
*InFile.....Not Specified
*Library.....Not Specified
*MemPages.....Not Specified
*Name.....$GP08
*OutFile.....Not Specified
*PFSSize.....Not Specified
*PrimaryCPU.....Not Specified
*Priority.....148
*Program.....$SYSTEM.SYSTEM.O999GP
*SaveAbend.....OFF
*StartMode.....APPLICATION
*StartupMessage.....Not Specified
*StopMode.....STANDARD
*Type.....OTHER
*UserId.....SUPER.SUPER ( 255,255 )

```

- The following example gives detailed configuration information about the \$ZHOME process.

-> INFO PROCESS \$ZZKRN.#ZHOME, DETAIL

```

NONSTOP KERNEL - Detailed Info PROCESS \SUN.$ZZKRN.#ZHOME

*AutoRestart.....10
*BackupCPU.....Not Specified
*CPU.....Not Specified
*DefaultVolume.....$SYSTEM.NOSUBVOL
*ExtSwap.....Not Specified
*Highpin.....ON
*HomeTerminal.....$YMIOP.#CLCI
*InFile.....$YMIOP.#CLCI
*Library.....Not Specified
*MemPages.....Not Specified
*Name.....$ZHOME
*OutFile.....$YMIOP.#CLCI
*PFSSize.....Not Specified
*PrimaryCPU.....0
*Priority.....199
*Program.....$SYSTEM.SYSTEM.ZHOME
*SaveAbend.....OFF
*StartMode.....KERNEL
*StartupMessage.....Not Specified
*StopMode.....STANDARD
*Type.....OTHER
*UserId.....SUPER.SUPER ( 255,255 )

```

INFO SUBSYS Command

Use the INFO SUBSYS command to display currently configured values for the system parameters managed by the \$ZZKRN Kernel subsystem manager.

```
INFO [ / OUT file-spec / ] SUBSYS $ZZKRN
```

SUBSYS \$ZZKRN

is the name of the Kernel manager process. You can omit SUBSYS and \$ZZKRN if you have specified them in an ASSUME command.

INFO SUBSYS Considerations

Wild cards are not supported for the INFO SUBSYS command.

INFO SUBSYS Display Format

The format of the display for the INFO PROCESS command is described here. See also the example on page [6-40](#).

Note that if you use the ALTER command (described on page [6-15](#)) to change the system name, system number, or a time attribute, this change does not take effect until the next system load.

The INFO SUBSYS command lists any pending changed parameters at the bottom of the display.

```
NONSTOP KERNEL - Info SUBSYS $ZZKRN

*DAYLIGHT_SAVING_TIME..... { USA66 | TABLE | NONE }
*NONRESIDENT_TEMPLATES..... $vol.subvol.file
*POWERFAIL_DELAY_TIME..... n
*RESIDENT_TEMPLATES..... $vol.subvol.file
  SUPER_SUPER_IS_UNDENIABLE..... { ON | OFF }
*SYSTEM_NAME..... \system
*SYSTEM_NUMBER..... n
  SYSTEM_PROCESSOR_TYPE..... NSR-x
*TIME_ZONE_OFFSET..... [-]hh:mm
Pending Changes (will take effect at next system load)

*DAYLIGHT_SAVING_TIME..... {USA66 | TABLE | NONE}
*SYSTEM_NAME..... \system
*SYSTEM_NUMBER..... n
*TIME_ZONE_OFFSET..... [-]hh:mm
```

An asterisk (*) before an attribute name indicates that its attribute value can be changed with the SCF ALTER command.

No asterisk before a name indicates that the parameter corresponds to an ALLPROCESSORS paragraph entry in the CONFTEXT file. For detailed information regarding these parameters, refer to the *System Generation Manual for G-Series RVUs*.

The following terms apply to the preceding display:

DAYLIGHT_SAVING_TIME	The daylight-saving time algorithm used when the system clock is set: TABLE, USA66, or NONE.
NONRESIDENT_TEMPLATES	The location of the EMS nonresident template file. If \$ZZKRN cannot find an EMS template file on \$SYSTEM.SYSTEM, it then searches for the file on the current \$SYSTEM.SYS _{nn} .
POWERFAIL_DELAY_TIME	The maximum number of seconds after a power failure that the operating system waits before initiating an orderly shutdown of the system.
RESIDENT_TEMPLATES	The location of the EMS resident template file. If \$ZZKRN cannot find an EMS template file on \$SYSTEM.SYSTEM, it then searches for the file on the current \$SYSTEM.SYS _{nn} .
SUPER_SUPER_IS_UNDENIABLE	Whether or not Safeguard security ignores explicit denials of access authority to the super ID (255,255).
SYSTEM_NAME	The name of the system.
SYSTEM_NUMBER	The Expand node number of the system.
SYSTEM_PROCESSOR_TYPE	The type of processor used in the system.
TIME_ZONE_OFFSET	The time offset for the system from Greenwich mean time (GMT).

INFO SUBSYS Example

The following example displays information about current changes to the operating system:

-> INFO SUBSYS \$ZZKRN

```
NONSTOP KERNEL - Info SUBSYS  \EAST.$ZZKRN

Current Settings

*DAYLIGHT_SAVING_TIME ..... USA66
*NONRESIDENT_TEMPLATES..... $SYSTEM.SYSTEM.TEMPLATE
*POWERFAIL_DELAY_TIME..... 30
*RESIDENT_TEMPLATES..... $SYSTEM.SYSTEM.RTMPLATE
  SUPER_SUPER_IS_UNDENIABLE..... OFF
*SYSTEM_NAME..... \EAST
*SYSTEM_NUMBER..... 254
  SYSTEM_PROCESSOR_TYPE ..... NSR-W
*TIME_ZONE_OFFSET..... -8:00
```


NAMES Command

Use the NAMES command to display a list of the subordinate object types and names associated with the specified object.

This is a nonsensitive command.

```
NAMES [ / OUT file-spec / ] [ object-spec ]
```

The value of *object-spec* is one of the following object type and object name combinations.

<i>object-type</i>	<i>object-name</i>
null	{ \$ZZKRN \$ZSNET }
PROCESS	\$ZZKRN[. # <i>gpname</i>]
SERVERNET	\$ZSNET
SUBSYS	\$ZZKRN

These versions of the NAMES command are discussed on the following pages:

Command	Page
NAMES null Command	6-41
NAMES PROCESS Command	6-43
NAMES SERVERNET Command	6-45
NAMES SUBSYS Command	6-46

NAMES null Command

Use the NAMES null command to display a list of subordinate object types and names associated with \$ZZKRN or \$ZSNET. The NAMES null command is described next.

```
NAMES [ / OUT file-spec / ] { $ZZKRN | $ZSNET }
```

```
{ $ZZKRN | $ZSNET }
```

is the \$ZZKRN Kernel subsystem manager or the \$ZSNET ServerNet manager process. You can omit \$ZZKRN or \$ZSNET if you have specified either of them with an ASSUME command.

NAMES null Display Formats

The format of the display for the NAMES \$ZZKRN command is described here. See also Example 1 on page [6-43](#).

```

NONSTOP KERNEL - Names  \system.$ZZKRN
Subsys
$ZZKRN

Process
$ZZKRN.#gpname1           $ZZKRN.#gpname2
$ZZKRN.#gpname3           $ZZKRN.#gpname4
...
```

The following terms appear in the preceding display:

- Subsys** The name of the \$ZZKRN Kernel subsystem manager process.
- Process** An alphabetized two-column list of the PROCESS objects associated with the \$ZZKRN Kernel subsystem manager process.

The format of the display for the NAMES \$ZSNET command is described here. See also Example 2 on page [6-43](#).

```

NONSTOP KERNEL Names  \system.$ZSNET
SERVERNET
$ZSNET
$ZSNET.fabric.cpu0       $ZSNET.fabric.cpu0
$ZSNET.fabric.cpu1       $ZSNET.fabric.cpu1
$ZSNET.fabric.cpu2       $ZSNET.fabric.cpu2
$ZSNET.fabric.cpu3       $ZSNET.fabric.cpu3
$ZSNET.fabric.cpu4       $ZSNET.fabric.cpu4
$ZSNET.fabric.cpu5       $ZSNET.fabric.cpu5
$ZSNET.fabric.cpu6       $ZSNET.fabric.cpu6
$ZSNET.fabric.cpu7       $ZSNET.fabric.cpu7
$ZSNET.fabric.cpu8       $ZSNET.fabric.cpu8
$ZSNET.fabric.cpu9       $ZSNET.fabric.cpu9
$ZSNET.fabric.cpu10      $ZSNET.fabric.cpu10
$ZSNET.fabric.cpu11      $ZSNET.fabric.cpu11
$ZSNET.fabric.cpu12      $ZSNET.fabric.cpu12
$ZSNET.fabric.cpu13      $ZSNET.fabric.cpu13
$ZSNET.fabric.cpu14      $ZSNET.fabric.cpu14
$ZSNET.fabric.cpu15      $ZSNET.fabric.cpu15
```

The following terms appear in the preceding display:

- SERVERNET** A number-ordered two-column list of the SERVERNET objects associated with the \$ZZKRN Kernel subsystem manager process, in the form of the \$ZSNET process name, an X fabric or Y fabric designation, and a processor number.

NAMES null Examples

1. To list the objects associated with \$ZZKRN, type:

-> NAMES \$ZZKRN

```

NONSTOP KERNEL - Names PROCESS \SUN.$ZZKRN
Subsys
$ZZKRN

Process
$ZZKRN.#CEV-SERVER-MANAGER-P0      $ZZKRN.#CEV-SERVER-MANAGER-P1
$ZZKRN.#CLCI-TACL                  $ZZKRN.#CHK
$ZZKRN.#OSM-APPSRVR                $ZZKRN.#OSM-CIMOM
$ZZKRN.#OSM-CONFLH-RD              $ZZKRN.#OSM-OEV
$ZZKRN.#QIOMON                     $ZZKRN.#ROUTING-DIST
$ZZKRN.#TCPIP-ZTC02                $ZZKRN.#TSM-SNMP
$ZZKRN.#SP-EVENT                   $ZZKRN.#TSM-SRM
$ZZKRN.#ZLOG                       $ZZKRN.#ZTCP0
$ZZKRN.#ZTCP1                      $ZZKRN.#ZHOME
$ZZKRN.#ZZKRN                     $ZZKRN.#ZZLAN
$ZZKRN.#ZZSTO                      $ZZKRN.#ZZWAN

```

2. To list the objects associated with \$ZSNET, type:

-> NAMES \$ZSNET

```

NONSTOP KERNEL Names \EAST.$ZSNET
SERVERNET
$ZSNET
$ZSNET.X.0      $ZSNET.Y.0
$ZSNET.X.1      $ZSNET.Y.1
$ZSNET.X.2      $ZSNET.Y.2
$ZSNET.X.3      $ZSNET.Y.3
$ZSNET.X.4      $ZSNET.Y.4
$ZSNET.X.5      $ZSNET.Y.5
$ZSNET.X.6      $ZSNET.Y.6
$ZSNET.X.7      $ZSNET.Y.7
$ZSNET.X.8      $ZSNET.Y.8
$ZSNET.X.9      $ZSNET.Y.9
$ZSNET.X.10     $ZSNET.Y.10
$ZSNET.X.11     $ZSNET.Y.11
$ZSNET.X.12     $ZSNET.Y.12
$ZSNET.X.13     $ZSNET.Y.13
$ZSNET.X.14     $ZSNET.Y.14
$ZSNET.X.15     $ZSNET.Y.15

```

NAMES PROCESS Command

Use the NAMES PROCESS command to display a list of generic processes associated \$ZZKRN.

```
NAMES [ / OUT file-spec / ] PROCESS $ZZKRN[. #gpname ]
```

```
PROCESS $ZZKRN[. #gpname ]
```

is the name of a generic process controlled by the \$ZZKRN Kernel subsystem manager. You can omit PROCESS, \$ZZKRN, and #*gpname* if you have specified them with an ASSUME command.

NAMES PROCESS Consideration

The trailing asterisk (*) wild-card character is supported for #*gpname*.

NAMES PROCESS Display Format

The format of the display for the NAMES PROCESS command is described here. See also the examples on page [6-44](#).

```
NONSTOP KERNEL - Names PROCESS \system.$ZZKRN[. #gpname ]
Process
$ZZKRN.#gpname1                                $ZZKRN.#gpname2
$ZZKRN.#gpname3                                $ZZKRN.#gpname4
```

The following term appears in the preceding display:

Process	An alphabetized two-column list of the PROCESS objects associated with the \$ZZKRN Kernel subsystem manager process.
---------	--

NAMES PROCESS Examples

1. The following example shows how to list the PROCESS objects associated with the Kernel subsystem manager process:

```
-> NAMES PROCESS $ZZKRN
```

```
NONSTOP KERNEL - Names PROCESS \EAST.$ZZKRN
Process
$ZZKRN.#CEV-SERVER-MANAGER-P0      $ZZKRN.#CEV-SERVER-MANAGER-P1
$ZZKRN.#CLCI-TACL                  $ZZKRN.#CHK
$ZZKRN.#OSM-APPSRVr                $ZZKRN.#OSM-CIMOM
$ZZKRN.#OSM-CONFLH-RD              $ZZKRN.#OSM-OEV
$ZZKRN.#QIOMON                     $ZZKRN.#ROUTING-DIST
$ZZKRN.#TCPIP-ZTC02                $ZZKRN.#TSM-SNMP
$ZZKRN.#SP-EVENT                   $ZZKRN.#TSM-SRM
$ZZKRN.#ZLOG                       $ZZKRN.#ZTCP0
$ZZKRN.#ZTCP1                      $ZZKRN.#ZHOME
$ZZKRN.#ZZKRN                      $ZZKRN.#ZZLAN
$ZZKRN.#ZZSTO                      $ZZKRN.#ZZWAN
```

- The following example shows how to display the name for the WAN subsystem manager process:

```
-> NAMES PROCESS $ZZKRN.#ZZWAN
```

```
NONSTOP KERNEL - Names PROCESS \EAST.$ZZKRN.#ZZWAN
Process
$ZZKRN.#ZZWAN
```

- The following example shows how to list the PROCESS object associated with a user-created generic process:

```
-> NAMES PROCESS $ZZKRN.#GP
```

```
NONSTOP KERNEL - Names PROCESS \EAST.$ZZKRN.#GP
Process
$ZZKRN.#GP
```

- The following command produces the same display as in NAMES PROCESS example [3](#):

```
-> NAMES PROCESS $ZZKRN.#G*
```

NAMES SERVERNET Command

Use the NAMES SERVERNET command to display a list of SERVERNET objects associated with the \$ZSNET process.

```
NAMES [ / OUT file-spec / ] SERVERNET $ZSNET
```

SERVERNET \$ZSNET

is the name of the ServerNet manager process.

NAMES SERVERNET Display Format and Example

The format of the display for the NAMES SERVERNET command is described here and produced by the following command:

```
-> NAMES SERVERNET $ZSNET
```

```
NONSTOP KERNEL - Names SERVERNET \system.$ZSNET
SERVERNET
$ZSNET
$ZSNET.X.0          $ZSNET.Y.0
$ZSNET.X.1          $ZSNET.Y.1
$ZSNET.X.2          $ZSNET.Y.2
$ZSNET.X.3          $ZSNET.Y.3
$ZSNET.X.4          $ZSNET.Y.4
$ZSNET.X.5          $ZSNET.Y.5
$ZSNET.X.6          $ZSNET.Y.6
$ZSNET.X.7          $ZSNET.Y.7
$ZSNET.X.8          $ZSNET.Y.8
$ZSNET.X.9          $ZSNET.Y.9
$ZSNET.X.10         $ZSNET.Y.10
$ZSNET.X.11         $ZSNET.Y.11
$ZSNET.X.12         $ZSNET.Y.12
$ZSNET.X.13         $ZSNET.Y.13
$ZSNET.X.14         $ZSNET.Y.14
$ZSNET.X.15         $ZSNET.Y.15
```

The following term appears in the preceding display:

SERVERNET	A list of the SERVERNET objects associated with the \$ZZKRN Kernel subsystem manager, in the form of the \$ZSNET process name, an X fabric or Y fabric designation, and a processor number. All possible processors are listed, regardless of how many are up.
------------------	--

NAMES SUBSYS Command

Use the NAMES SUBSYS command to display a list of PROCESS objects associated with the \$ZZKRN process.

```
NAMES [ / OUT file-spec / ] SUBSYS $ZZKRN
```

```
SUBSYS $ZZKRN
```

is the name of the ServerNet manager process. You can omit SUBSYS and \$ZZKRN if you have specified them in an ASSUME command.

NAMES SUBSYS Consideration

Wild cards are not supported for the NAMES SUBSYS command.

NAMES SUBSYS Display Format

The format of the display for the NAMES SUBSYS command is described here.

```

NONSTOP KERNEL - Names SUBSYS \system.$ZZKRN
Subsys
$ZZKRN

Process
$ZZKRN.#gpname1                $ZZKRN.#gpname2
$ZZKRN.#gpname3                $ZZKRN.#gpname4

```

The following terms appear in the preceding display:

Subsys The name of the \$ZZKRN Kernel subsystem manager process.

Process An alphabetized two-column list of the PROCESS objects associated with the \$ZZKRN Kernel subsystem manager process.

NAMES SUBSYS Example

The following example shows how to list the SUBSYS objects associated with the Kernel subsystem manager process:

-> NAMES SUBSYS \$ZZKRN

```

NONSTOP KERNEL - Names SUBSYS \BLUE.$ZZKRN
Subsys
$ZZKRN

Process
$ZZKRN.#SSM                    $ZZKRN.#TEMP
$ZZKRN.#XYZZZ                 $ZZKRN.#ZZKRN

```

START Command (Sensitive Command)

Use the START command to initiate the operation of an object.

```
START [ / OUT file-spec / ] [ object-spec ]
```

The value of *object-spec* is one of the following object type and object name combinations.

<i>object-type</i>	<i>object-name</i>
PROCESS	<code>\$ZZKRN.#<i>gpname</i></code>
SERVERNET	<code>\$ZSNET.{X Y}. {<i>cpu</i> *}</code>

The START PROCESS command is described in the following subsection.
The START SERVERNET command is described on page [6-49](#).

START PROCESS Command

Use the START PROCESS command to start running a generic process.

```
START [ / OUT file-spec / ] PROCESS $ZZKRN.#gpname
```

PROCESS \$ZZKRN.#*gpname*

is the name of a generic process controlled by the \$ZZKRN Kernel subsystem manager. You can omit PROCESS, \$ZZKRN, and #*gpname* if you have specified them with an ASSUME command.

START PROCESS Considerations

- [Starting a Generic Process](#) on page 3-23 describes how to use the START PROCESS command.
- It is recommended that you enter a TIMEOUT command (described in the *SCF Reference Manual for G-Series RVUs*) to specify a timeout value that is larger than the default 90 seconds when starting a generic process configured in multiple processors. Several processes can likely be started within the 90 second default. But if you start a generic process that has been configured as a group (by, for example, the CPU ALL attribute), or if you start multiple generic processes (by using a wild card in the ABORT command), more time may be needed.
- The START PROCESS command initiates execution of generic processes. Executing this command directs the \$ZPM persistence manager to create one or many processes, depending on the values configured for the CPU, PRIMARYCPU, and BACKUPCPU attributes.
- If the start mode is not DISABLED but the processor in which the generic process is configured is down, the START command puts the process into the STOPPED

object state, substate STOPPED. When the processor comes up, the generic process starts.

- If the start mode is not DISABLED and the processor in which the generic process is configured is up, a successful START command puts the process into the STARTED object state.
- A successful completion of the START command indicates that this process (or processes, if started in more than one processor) has been started (if its processor is up) and the startup message has been sent to it.
- The START command sets the persistence count to the configured value.
- Wild-card support is limited to the trailing asterisk (*) for *#gpname*.
- The START PROCESS command is not supported for the \$ZZKRN Kernel subsystem manager process. (The \$ZPM persistence manager ensures that \$ZZKRN remains up at all times.)

START PROCESS Example

The following example shows how to start the process \$ZZKRN.#TEMP:

```
-> START PROCESS $ZZKRN.#TEMP
```

START SERVERNET Command

Use the START SERVERNET command to start a ServerNet fabric.

```
START [ / OUT file-spec / ]
      SERVERNET $ZSNET.{X|Y}. {cpu | *}
```

*\$ZSNET.{X|Y}. {cpu | *}*

is the name of a ServerNet fabric (X or Y) and the processor number (*cpu*). You can omit SERVERNET and \$ZSNET if you have specified them with an ASSUME command.

START SERVERNET Considerations

- The asterisk (*) wild-card character specifies all available processors.
- The START SERVERNET command tells a specific processor to begin using a specific ServerNet fabric.
- Refer to the *HP NonStop S-Series Hardware Installation Guide* for complete information about using this command when adding an enclosure or upgrading memory.

START SERVERNET Display Format

The format of the display for the START SERVERNET command is described here. See also Example [1](#) on page 6-51.

NONSTOP KERNEL - Start SERVERNET \$ZSNET.XorY.cpu																	
XorY-FABRIC																	
TO	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
FROM																	
0	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
1	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
2	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
3	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
4	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
5	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
6	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
7	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
8	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
9	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
10	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
11	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
12	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
13	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
14	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
15	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st

XorY

indicates whether the X or Y fabric is being started.

cpu

is a processor number or an asterisk (*) for starting all processors.

st

indicates the status of the path between the two processors and has one of the following values:

DIS (disabled) A ServerNet fabric is down at the “TO” location. As a result, the path from the “FROM” processor to the “TO” processor is down for receiving, which means that the “TO” processor cannot receive from any other processor or from I/O devices on that fabric. DIS overrides UP and DN.

DN (down) The path from the “FROM” processor to the “TO” processor is down because the path is failing. The “FROM” processor cannot communicate with the “TO” processor on that fabric.

<-DOWN (across a row) The “FROM” processor is down or nonexistent.

Ennn The ServerNet fabric unexpectedly returned file-system error nnn regarding the path from the “FROM” processor to the “TO” processor.

ERROR *nnn* (across a row) The “FROM” processor unexpectedly returned file-system error *nnn* to the ServerNet fabric.

Refer to the *Guardian Procedure Errors and Messages Manual* for information about the file-system error.

UNA (unavailable) The link from the “FROM” processor to the “TO” processor is down because the “TO” processor is down or nonexistent. UNA overrides all other values.

UP The path from the “FROM” processor to the “TO” processor is up.

START SERVERNET Examples

- The following example shows the results of starting the ServerNet X fabric in processor 0. Note that only the first four processors are up in the system.

-> START SERVERNET \$ZSNET.X.0

NONSTOP KERNEL - Start SERVERNET \$ZSNET.X.0																	
X-FABRIC																	
T0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
FROM																	
0	UP	UP	UP	UP	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
1	UP	UP	UP	UP	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
2	UP	UP	UP	UP	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
3	UP	UP	UP	UP	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
4	<-DOWN																
5	<-DOWN																
6	<-DOWN																
7	<-DOWN																
8	<-DOWN																
9	<-DOWN																
10	<-DOWN																
11	<-DOWN																
12	<-DOWN																
13	<-DOWN																
15	<-DOWN																

- The following example shows how to start the ServerNet X fabric in all configured processors in the system.

-> START SERVERNET \$ZSNET.X.*

STATUS Command

Use the STATUS command to display current status information about an object.

This is a nonsensitive command.

```
STATUS [ / OUT file-spec / ] [ object-spec ] [ , DETAIL ]
```

The value of *object-spec* is one of the following object type and object name combinations.

<i>object-type</i>	<i>object-name</i>
PROCESS	\$ZZKRN[.# <i>gpname</i>]
SERVERNET	\$ZSNET
SUBSYS	\$ZZKRN

These versions of the STATUS command are discussed on the following pages:

Command	Page
STATUS PROCESS Command	6-52
STATUS SERVERNET Command	6-56
STATUS SUBSYS Command	6-60

STATUS PROCESS Command

The STATUS PROCESS command displays current status about a process. The command takes the following form.

```
STATUS [ / OUT file-spec / ]
PROCESS $ZZKRN[ .#gpname ] [ , DETAIL ]
```

PROCESS \$ZZKRN

is the name of the Kernel subsystem manager process. You can omit PROCESS and \$ZZKRN if you have specified them with an ASSUME command.

This form of the command causes the display to show current status information for the Kernel subsystem manager.

PROCESS \$ZZKRN.#*gpname*

is the name of a generic process controlled by the \$ZZKRN Kernel subsystem manager. You can omit PROCESS and \$ZZKRN.#*gpname* if you have specified them with an ASSUME command.

This form of the command causes the display to show current status information for the specified process.

DETAIL

causes SCF to display detailed information about the specified process.

STATUS PROCESS Consideration

Wild-card support is limited to the trailing asterisk (*) for #gpname.

STATUS PROCESS Summary Display Format

The format of the summary display for the STATUS PROCESS command (without the DETAIL option) is described here. See also Example 1 on page 6-55.

NONSTOP KERNEL -	Status Process	\system.\$ZZKRN.[#gpname]				
Symbolic Name	Name	State	Sub	Primary	Backup	Owner
			PID	PID	PID	ID
gpname	\$process	state	sub	n,m	n,m	
nnn,nnn						

The following terms appear in the preceding display:

- Symbolic Name The symbolic name of a generic process, as specified in the ADD command. For the Kernel subsystem manager, this name is ZZKRN.
- Name The name of the process, as specified by the NAME attribute of the SCF ADD or ALTER command, and as recognized by TACL. If this generic process is configured in more than one processor with some form of the CPU attribute, its name ends in nn, representing a two-digit processor number.
- State The current object state of the PROCESS object. Possible states are ABORTING, STARTED, and STOPPED.
- Sub The ABT (ABORTED) substate, if the PROCESS object has been stopped by the ABORT command.
- Primary PID The number of the primary processor and process identification number (PIN) for this process.
- Backup PID The number of the backup processor and process identification number (PIN) for this process, if it exists.
- Owner ID The owner group number and user number of this process.

STATUS PROCESS Detailed Display Format

The format of the detailed display for the STATUS PROCESS command (with the DETAIL option) is described here. See also Example 2 on page [6-55](#).

```

NONSTOP KERNEL - Detailed Status Process \system.$ZZKRN[. #gpname ]
Backup PID.....n,m
Creation Time.....dd mmm yyyy, hh:mm:ss:ff
Name.....$name
Owner ID.....n,m
Primary PID.....n,m
Priority.....n
State.....state
Substate.....substate

```

The following terms appear in the preceding display:

Backup PID	The number of the backup processor and process identification number (PIN) for this process, if it exists.
Creation Time	The date and time when the process was created; this means the time it became a process in the operating system, for example, as a result of an SCF START command.
Name	The name of the process, as specified by the NAME attribute of the SCF ADD or ALTER command, and as recognized by TACL. If this generic process is configured in more than one processor with some form of the CPU attribute, its name ends in <i>nn</i> , representing a two-digit processor number.
Owner ID	The owner group number and user number of this process.
Primary PID	The number of the primary processor and process identification number (PIN) for this process.
Priority	The current priority of the process or, if it is not currently running, the priority level it will have when it is next started. The process may have been configured with a different priority level than this display indicates. This is possible because the priority could have been changed by an operating system or TACL command, and such a change is reflected by SCF STATUS, but not the SCF INFO command. To find out the configured priority of the process, use the INFO PROCESS, DETAIL command (see page 6-31).
State	The current object state of the PROCESS object. Possible states are ABORTING, STARTED, and STOPPED.
Substate	ABORTED, if the PROCESS object has been aborted.

STATUS PROCESS Examples

1. The following example displays summary current status information for the \$ZZKRN Kernel subsystem manager:

-> STATUS PROCESS \$ZZKRN

```
NONSTOP KERNEL - Status PROCESS \EAST.$ZZKRN.#ZZKRN

Symbolic Name      Name      State      Sub Primary  Backup  Owner
                  PID          PID          PID      ID
ZZKRN              $ZZKRN STARTED    0 ,11    1,11    255,255
```

2. The following example displays detailed current status information for \$ZZKRN:

-> STATUS PROCESS \$ZZKRN, DETAIL

```
NONSTOP KERNEL - Detailed Status PROCESS \SUN.$ZZKRN.#ZZKRN

Backup PID..... None
Creation Time..... JAN 21,2000 11:41:59
Name..... $ZZKRN
OwnerID..... 255, 255
Primary PID..... 0 , 15
Priority..... 180
State..... STARTED
Substate.....
```

3. The following example displays summary current status information about a user-configured generic process that is configured in two processors, when one processor is down:

-> STATUS PROCESS \$ZZKRN.#GP

```
NONSTOP KERNEL - Status PROCESS \EAST.$ZZKRN.#GP

Symbolic Name      Name      State      Sub Primary  Backup  Owner
                  PID          PID          PID      ID
GP                 $GP00 STARTED    0 ,16    None    255,255
GP                 $GP01 STOPPED    None     None
```

4. The following example displays detailed current status information about the same user-configured generic process. Because \$GP01 is stopped, the STATUS

command cannot display information about the creation time, owner, or priority for the process.

```
-> STATUS PROCESS $ZZKRN.#GP, DETAIL
```

```
NONSTOP KERNEL - Detailed Status PROCESS \EAST.$ZZKRN.#GP

Backup PID..... None
Creation Time..... JAN 21,2000 11:51:50
Name..... $GP00
OwnerID..... 255, 255
Primary PID..... 0 , 16
Priority..... 148
State..... STARTED
Substate.....

NONSTOP KERNEL - Detailed Status PROCESS \EAST.$ZZKRN.#GP

Backup PID..... None
Creation Time.....
Name..... $GP01
OwnerID.....
Primary PID..... None
Priority.....
State..... STOPPED
Substate.....
```

STATUS SERVERNET Command

The STATUS SERVERNET command displays the current object status of all ServerNet fabrics on the system. The command takes the following form.

```
STATUS [ / OUT file-spec / ] SERVERNET $ZSNET
```

SERVERNET \$ZSNET

is the name of the ServerNet manager process. You can omit SERVERNET and \$ZSNET if you have specified them in an ASSUME command.

STATUS SERVERNET Considerations

- The DIS status overrides the display of UP or DN status.
- The UNA status overrides all other displayed values.

STATUS SERVERNET Display Format

The format of the display for the STATUS SERVERNET command is described here.
See also the example on page [6-59](#).

NONSTOP KERNEL - Status SERVERNET																	
X-FABRIC																	
TO	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
FROM																	
0	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
1	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
2	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
3	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
4	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
5	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
6	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
7	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
8	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
9	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
10	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
11	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
12	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
13	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
14	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
15	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
Y-FABRIC																	
TO	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
FROM																	
0	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
1	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
2	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
3	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
4	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
5	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
6	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
7	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
8	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
9	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
10	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
11	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
12	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
13	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
14	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
15	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st

The STATUS SERVERNET command displays a matrix for the ServerNet X fabric and Y fabric. Each matrix shows the status of the path between all pairs of processors. The status (*st*) can be:

DIS	(disabled) A ServerNet fabric is down at the “TO” location. As a result, the path from the “FROM” processor to the “TO” processor is down for receiving, which means that the “TO” processor cannot receive from any other processor or from I/O devices on that fabric. DIS overrides UP and DN.
DN	(down) The path from the “FROM” processor to the “TO” processor is down because the path is failing. The “FROM” processor cannot communicate with the “TO” processor on that fabric.
<-DOWN	(across a row) The “FROM” processor is down or nonexistent.
ERROR <i>nnn</i>	(across a row) The “FROM” processor unexpectedly returned file-system error <i>nnn</i> to the ServerNet fabric. Refer to the <i>Guardian Procedure Errors and Messages Manual</i> for information about the file-system error.
UNA	(unavailable) The link from the “FROM” processor to the “TO” processor is down because the “TO” processor is down or nonexistent. UNA overrides all other values.
UP	The path from the “FROM” processor to the “TO” processor is up.

STATUS SERVERNET Example

The following example displays the status of the ServerNet network in a four-processor system:

-> STATUS SERVERNET \$ZSNET

NONSTOP KERNEL - Status SERVERNET																
X-FABRIC																
TO	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
FROM																
0	UP	DN	UNA	UP	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
1	UP	UP	UNA	UP	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
2	<-DOWN															
3	UP	UP	UNA	UP	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
4	<-DOWN															
5	<-DOWN															
6	<-DOWN															
7	<-DOWN															
8	<-DOWN															
9	<-DOWN															
10	<-DOWN															
11	<-DOWN															
12	<-DOWN															
13	<-DOWN															
15	<-DOWN															
Y-FABRIC																
TO	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
FROM																
0	UP	UP	UNA	DIS	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
1	UP	UP	UNA	DIS	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
2	<-DOWN															
3	DN	DN	UNA	DIS	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA	UNA
4	<-DOWN															
5	<-DOWN															
6	<-DOWN															
7	<-DOWN															
8	<-DOWN															
9	<-DOWN															
10	<-DOWN															
11	<-DOWN															
12	<-DOWN															
13	<-DOWN															
14	<-DOWN															
15	<-DOWN															

The display shows the following:

- Processor 2 is down, as shown by the UNA status in column 2 and the DOWN in row 2 for both fabrics (shown in bold).
- A single point-to-point link (the X fabric from processor 0 to processor 1) is down, as shown by the DN status in the X fabric matrix (shown in bold).
- The Y fabric is totally down in processor 3, as shown by the DIS status in column 3 of the Y fabric and the DN status in row 3 of the Y fabric.

STATUS SUBSYS Command

The STATUS SUBSYS command displays current status information about the Kernel subsystem manager. The command takes the following form.

```
STATUS [ / OUT file-spec / ] SUBSYS $ZZKRN [ , DETAIL ]
```

SUBSYS \$ZZKRN

is the name of the Kernel subsystem manager process. You can omit SUBSYS and \$ZZKRN if you have specified them with an ASSUME command.

DETAIL

causes SCF to display detailed information about the specified process.

STATUS SUBSYS Summary Display Format

The format of the summary display for the STATUS SUBSYS command (without the DETAIL option) is described here. See also Example 1 on page [6-61](#).

```
NONSTOP KERNEL - Status SUBSYS \system.$ZZKRN

Name           State      Processes
                (conf/strd)
$ZZKRN         state      (   n/m   )
```

The following terms appear in the preceding display:

Name	\$ZZKRN, the name of the Kernel subsystem manager process.
State	The current object state of \$ZZKRN is always STARTED.
Processes (conf/strd)	The number of generic process records or groups configured and the number with at least one instance of the process started. To display the names of these processes, use the INFO PROCESS (on page 6-31) or NAMES PROCESS command (on page 6-43).

STATUS SUBSYS Detailed Display Format

The format of the detailed display for the STATUS SUBSYS command (with the DETAIL option) is described here. See also Example 2 on page [6-61](#).

```
NONSTOP KERNEL - Detailed Status SUBSYS \system.$ZZKRN

Primary PID..... n,m      Backup PID ..... n,m
Subsystem Owner..... n,m    Subsystem State ... state
Processes Configured. n      Processes Started.. n
```

The following terms apply to the preceding display:

Primary PID	The number of the primary processor and process identification number (PIN) for the \$ZZKRN process.
Backup PID	The number of the backup processor and process identification number (PIN) for the \$ZZKRN process.
Subsystem Owner	The owner group number and user number of the \$ZZKRN process is always the super ID (255,255).
Subsystem State	The current object state of \$ZZKRN is always STARTED.
Processes Configured	The number of generic process currently configured.
Processes Started	The number of generic processes that currently have at least one instance of the generic process started.

STATUS SUBSYS Examples

1. The following example displays the status of the \$ZZKRN subsystem manager process. Of 12 processes that have been configured, 10 are started.

-> STATUS SUBSYS \$ZZKRN

```
NONSTOP KERNEL - Status SUBSYS \EAST.$ZZKRN

Name           State      Processes
                (conf/strd)
$ZZKRN         STARTED   ( 12/10 )
```

2. The following example displays the detailed status of the same \$ZZKRN subsystem manager process as in the preceding example:

-> STATUS SUBSYS \$ZZKRN, DETAIL

```
NONSTOP KERNEL - Detailed Status SUBSYS \EAST.$ZZKRN

Primary PID..... 1 , 51   Backup PID ..... 0 , 51
Subsystem Owner..... 255, 255 Subsystem State ... STARTED
Processes Configured. 12     Processes Started.. 10
```

STOP Command (Sensitive Command)

The STOP command terminates the activity of a ServerNet fabric in a normal manner. Upon the successful completion of the STOP command, the ServerNet fabric is left in the STOPPED object state.

You must be in interactive mode to use this sensitive command.

```
STOP [ / OUT file-spec / ] SERVERNET $ZSNET.{X|Y}. {cpu|*}
```

```
SERVERNET $ZSNET.{X|Y}. {cpu|*}
```

stops a ServerNet fabric (X or Y) in one or all processors. The variable *cpu* is the processor number.

Considerations

- The asterisk (*) wild-card character in a STOP SERVERNET command specifies all available processors.
- You cannot stop both ServerNet fabrics at the same time. If you attempt to do this, the operating system keeps up the last path from processor 0 to processor 1 in the second fabric brought down (shown in Example [2](#) on page 6-65).
- Refer to the *HP NonStop S-Series Hardware Installation Guide* for complete information about using this command when adding an enclosure or upgrading memory.
- Because of the serious effects of this command, the operating system responds to a STOP SERVERNET command with a confirmation request.

Display Format

The format of the display for the STOP SERVERNET command is described here.

NONSTOP KERNEL - Stop SERVERNET \$ZSNET.XorY.cpu																
XorY-FABRIC																
T0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
FROM																
0	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
1	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
2	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
3	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
4	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
5	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
6	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
7	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
8	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
9	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
10	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
11	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
12	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
13	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
14	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st
15	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st	st

XorY

indicates whether the X or Y fabric is being stopped.

cpu

is a processor number or an asterisk (*) for stopping all processors.

st

indicates the status of the path between the two processors and has one of the following values:

- DIS

(disabled) A ServerNet fabric is down at the “TO” location. As a result, the path from the “FROM” processor to the “TO” processor is down for receiving, which means that the “TO” processor cannot receive from any other processor or from I/O devices on that fabric. DIS overrides UP and DN.
- DN

(down) The path from the “FROM” processor to the “TO” processor is down because the path is failing. The “FROM” processor cannot communicate with the “TO” processor on that fabric.
- <-DOWN

(across a row) The “FROM” processor is down or nonexistent.
- Ennn

The ServerNet fabric unexpectedly returned file-system error nnn regarding the path from the “FROM” processor to the “TO” processor.

ERROR *nnn* (across a row) The “FROM” processor unexpectedly returned file-system error *nnn* to the ServerNet fabric.

Refer to the *Guardian Procedure Errors and Messages Manual* for information about the file-system error.

UNA (unavailable) The link from the “FROM” processor to the “TO” processor is down because the “TO” processor is down or nonexistent. UNA overrides all other values.

UP The path from the “FROM” processor to the “TO” processor is up.

UP* The path from the “FROM” processor to the “TO” processor was left up in order not to bring down the last path between these two processors (STOP command only).

Examples

1. The following example shows how to stop the Y fabric in processor 0:

```
-> STOP SERVERNET $ZSNET.Y.0
```

This command brings down the connection from processor 0 to the ServerNet Y fabric. As a result, the display shows the column for processor 0 to be UNA, while the row for processor 0 is DN. These changes are shown in bold.

```
NONSTOP KERNEL - Stop SERVERNET $ZSNET.Y.0
Y-FABRIC
  TO    0    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
FROM
  0      UNA DN  DN  DN  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  1      UNA UP   UP   UP   UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  2      UNA UP   UP   UP   UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  3      UNA UP   UP   UP   UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  4      <-DOWN
  5      <-DOWN
  6      <-DOWN
  7      <-DOWN
  8      <-DOWN
  9      <-DOWN
 10      <-DOWN
 11      <-DOWN
 12      <-DOWN
 13      <-DOWN
 14      <-DOWN
 15      <-DOWN
```

2. The following example shows the results of taking down the Y fabric on a system when the X fabric is already down. Note that because the X fabric is already down, the "UP*" status (shown here in bold) is maintained for the point-to-point link from processor 0 to processor 1 so as to not bring down the last path between the two processors.

```
-> STOP SERVERNET $ZSNET.Y.*
```

```
NONSTOP KERNEL - Stop SERVERNET $ZSNET.Y.*
Y-FABRIC
  TO    0    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
FROM
  0      DN   UP* DN   DN   UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  1      DN   DN   DN   DN   UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  2      DN   DN   DN   DN   UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  3      DN   DN   DN   DN   UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA  UNA
  4      <-DOWN
  5      <-DOWN
  6      <-DOWN
  7      <-DOWN
  8      <-DOWN
  9      <-DOWN
 10      <-DOWN
 11      <-DOWN
 12      <-DOWN
 13      <-DOWN
 14      <-DOWN
 15      <-DOWN
```

UP* indicates that the path was not brought down because it is the last path up between these two processors

VERSION Command

Use the VERSION command to display the operating system version level of the specified object.

This is a nonsensitive command.

```
VERSION [ / OUT file-spec / ] [ object-spec ] [ , DETAIL ]
```

The value of *object-spec* is one of the following object type and object name combinations.

<i>object-type</i>	<i>object-name</i>
<i>null</i>	\$ZZKRN
PROCESS	\$ZZKRN
SERVERNET	\$ZSNET
SUBSYS	\$ZZKRN

These versions of the VERSION command are discussed on the following pages:

Command	Page
VERSION null and VERSION PROCESS Commands	6-66
VERSION SERVERNET Command	6-68
VERSION SUBSYS Command	6-69

VERSION null and VERSION PROCESS Commands

The VERSION null command and the VERSION PROCESS command both display the version level of the \$ZZKRN subsystem manager process.

```
VERSION [ / OUT file-spec / ] [ PROCESS ] $ZZKRN [ , DETAIL ]
```

VERSION null and VERSION PROCESS Summary Display Format

The format of the summary display for the VERSION null and VERSION PROCESS commands (without the DETAIL option) is described here. See also Example 1 on page [6-67](#).

```
VERSION [ PROCESS ] \system.$ZZKRN: KERNEL (MGR) - T1085vff - (ddmmmyy) -  
(ddmmmyy)
```

VERSION null and VERSION PROCESS Detailed Display Format

The format of the detailed display for the VERSION null and VERSION PROCESS commands (with the DETAIL option) is described here. See also Example 2 on page [6-68](#).

```
Detailed VERSION PROCESS \system.$ZZKRN
SYSTEM \system
  KERNEL (MGR) - T1085vff - (ddmmmyy) - (ddmmmyy)
  GUARDIAN - T9050 - (tos)
  SCF KERNEL - T9082vff - (ddmmmyy) (ddmmmyy)
  KERNEL PM - T1084vff - (ddmmmyy) - (ddmmmyy)
```

The following terms appear in the preceding two displays:

\system

is the system \$ZZKRN is running on.

vff

is the product version, for example, F40.

tos

is the operating system version of the Guardian kernel for this RVU, for example, N40.

ddmmmyy

are the RVU and compile dates, respectively, for the product.

VERSION null and VERSION PROCESS Examples

1. The following example shows how to display version information about the \$ZZKRN subsystem manager process:

-> VERSION PROCESS \$ZZKRN

```
VERSION PROCESS \EAST.$ZZKRN: KERNEL (MGR) - T1085F40 - (31AUG99) -
(31AUG99)
```

- The following example shows how to display detailed version information about the \$ZZKRN subsystem manager process:

```
-> VERSION $ZZKRN, DETAIL
```

```
Detailed VERSION \EAST.$ZZKRN
SYSTEM \EAST
  KERNEL (MGR) - T1085F40 - (31AUG99) - (28MAY99)
  GUARDIAN - T9050 - (N40)
  SCF KERNEL - T9082F40 - (01AUG97) (23JUL97)
  KERNEL PM - T1084F40 - (31AUG99) - (28MAY99)
```

VERSION SERVERNET Command

The VERSION SERVERNET command displays the version level of the \$ZSNET ServerNet manager process.

```
VERSION [ / OUT file-spec / ] SERVERNET $ZSNET [ , DETAIL ]
```

VERSION SERVERNET Summary Display Format

The format of the summary display for the VERSION SERVERNET command (without the DETAIL option) is described here. See also Example 1 on page [6-69](#).

```
VERSION \system.$ZSNET: SERVERNET (MGR) - T1085vff - (ddmmmyy) - (ddmmmyy)
```

VERSION SERVERNET Detailed Display Format

The format of the detailed display for the VERSION SERVERNET command (with the DETAIL option) is described here. See also Example 2 on page [6-69](#).

```
Detailed VERSION \system.$ZSNET
SYSTEM \system
  SERVERNET (MGR) - T1085vff - (ddmmmyy) - (ddmmmyy)
  GUARDIAN - T9050 - (tos)
  SCF KERNEL - T9082vff - (ddmmmyy) (ddmmmyy)
  KERNEL PM - T1084vff - (ddmmmyy) - (ddmmmyy)
```

The following terms appear in the preceding two displays.

\system

is the system \$ZSNET is running on.

vff

is the product version; for example, F40.

tos

is the operating system version of the Guardian kernel for this RVU, for example, N40.

ddmmmyy

are the RVU and compile dates, respectively, for the product.

VERSION SERVERNET Examples

1. The following example shows how to display the version of the \$ZSNET ServerNet manager process:

```
-> VERSION SERVERNET $ZSNET
```

```
VERSION \EAST.$ZSNET: SERVERNET (MGR) - T1085F40 - (31AUG99) - (28MAY99)
```

2. The following example shows how to display detailed version information about the \$ZZKRN Kernel subsystem manager:

```
-> VERSION SERVERNET $ZSNET, DETAIL
```

```
Detailed VERSION \EAST.$ZSNET
SYSTEM \EAST
  SERVERNET (MGR) - T1085F40 - (31AUG99) - (28MAY99)
  GUARDIAN - T9050 - (N40)
  SCF KERNEL - T9082F40 - (01AUG97) (23JUL97)
  KERNEL PM - T1084F40 - (31AUG99) - (28MAY99)
```

VERSION SUBSYS Command

The VERSION SUBSYS command displays the version level of the \$ZZKRN Kernel subsystem manager process.

```
VERSION [ / OUT file-spec / ] SUBSYS $ZZKRN [ , DETAIL ]
```

VERSION SUBSYS Summary Display Format

The format of the summary display for the VERSION SUBSYS command (without the DETAIL option) is described here. See also Example 1 on page [6-70](#).

```
VERSION SUBSYS \system.$ZZKRN: KERNEL (MGR) - T10859vff - (ddmmmyy) - (ddmmmyy)
```

VERSION SUBSYS Detailed Display Format

The format of the detailed display for the VERSION SUBSYS command (with the DETAIL option) is described here. See also Example 2 on page [6-70](#).

```
Detailed VERSION SUBSYS \system.$ZZKRN
SYSTEM \system
  KERNEL (MGR) - T1085vff - (ddmmmyy) - (ddmmmyy)
  GUARDIAN - T9050 - (tos)
  SCF KERNEL - T9082vff - (ddmmmyy) (ddmmmyy)
  KERNEL PM - T1084vff - (ddmmmyy) - (ddmmmyy)
```

The following terms appear in the preceding two displays:

\system

is the system \$ZZKRN is running on.

vff

is the product version, for example, F40.

tos

is the operating system version of the Guardian kernel for this RVU, for example, N40.

ddmmmyy

are the RVU and compile dates, respectively, for the product.

VERSION SUBSYS Examples

1. The following example shows how to display the version of the \$ZZKRN subsystem manager process:

```
-> VERSION SUBSYS $ZZKRN
```

```
VERSION SUBSYS \EAST.$ZZKRN: KERNEL (MGR) - T1085F40 - (31AUG99) - (28MAY99)
```

2. The following example shows how to display detailed version information about the \$ZZKRN subsystem manager process:

```
-> VERSION SUBSYS $ZZKRN, DETAIL
```

```
Detailed VERSION SUBSYS \EAST.$ZZKRN
SYSTEM \EAST
  KERNEL (MGR) - T1085F40 - (31AUG99) - (28MAY99)
  GUARDIAN - T9050 - (N40)
  SCF KERNEL - T9082F40 - (01AUG97) (23JUL97)
  KERNEL PM - T1084F40 - (31AUG99) - (28MAY99)
```