
Sirius Mods Command and Parameter Reference Manual



Sirius Software, Inc.
875 Massachusetts Avenue, Suite 21
Cambridge, MA 02139

Telephone: (617) 876-6677
FAX: (617) 234-1200
E-mail: support@sirius-software.com
World Wide Web: <http://sirius-software.com>

December 10, 2009

© 2009 Sirius Software, Inc.

Proprietary Notices

The following products:

- *Fast/Backup*
- *Fast/Reload*
- *Janus Network Security*
- *Janus Open Client*
- *Janus Open Server*
- *Janus Specialty Data Store*
- *Janus SOAP*
- *Janus Sockets*
- *Janus Web Server*
- *SirFact*
- *Sirius Functions*
- *Sirius Mods*
- *UL/SPF*

are proprietary products of Sirius Software, Inc.:

Sirius Software, Inc.
875 Massachusetts Avenue, Suite 21
Cambridge, Massachusetts 02139
USA

Model 204™ is a proprietary product of Computer Corporation of America:

Computer Corporation of America
200 West St.
3rd Floor West
Waltham, MA 02451
USA

Model 204™ is a registered trademark of Computer Corporation of America.

SoftSpy™ is a proprietary product of Information Technology Systems:

Information Technology Systems
95 Wells Avenue
Newton, Massachusetts 02459-3216
USA

Contents

Proprietary Notices	iii
Contents	v
Summary of Changes	ix
Sirius Mods Version 7.5	ix
Sirius Mods Version 7.3	ix
Sirius Mods Version 7.2	ix
Sirius Mods Version 7.1	x
Sirius Mods Version 7.0	x
Sirius Mods Version 6.9	x
 Chapter 1: Overview	 1
Versions and compatibility	1
Related manuals	2
 Chapter 2: Operator Commands	 3
Issuing MODIFY or SMSG operator commands	4
BUMP <i>user_num</i>	5
CLOSE	6
MONITOR	7
RESTART <i>abend_code</i> <i>USER user_num</i> <i>TASK task_num</i>	8
SAMPLE ON OFF AUTO	10
STATUS	11
STOP	12
 Chapter 3: System Administrator Commands	 13
BUMPSNAP <i>user</i>	14
DUMP DUMPX	15
DUMPG DUMPGX	16
FILELOAD FILELOADX	17
FLOD FLODX	18
FNADDR <i>funcname</i> <i>LOADMOD loadModName</i> <i>+hexFuncNumber</i>	19
FUDAEMON	20
JANUS	21
RESTAUTH	22
RESTORE RESTOREX	23
RESTOREG RESTOREGX	24
SIRAPSY	25

SIRFACT	26
SIRFIELD	28
SIRIUS [qualifier]	30
SIRMETH	31
UNICODE	32
Display forms of UNICODE	32
Update forms of UNICODE	33
Chapter 4: User Commands	35
APPDATE	36
JAN[US]DEB[UG]	37
SIR[IUS]DEBUG	38
Chapter 5: System Parameters	39
ACTRDS	40
APSYSEC	41
CENTSPAN	42
COMMLOG	43
COMPOPT	44
CSIPID	45
CURREP31	46
CURREP64	47
DEBUGMAX	48
DEBUGPAG	49
DUMPOPTS	50
DUMPTMAX	51
DUMPTMIN	52
ENTTAB	53
FNVMASK	54
FRELPREV	55
FUNCOPTS	56
FUNMAXT	58
FUNPARG	59
FUNPGM	60
FUNTSKN	61
HGHREP31	62
HGHREP64	63
MAXBG	64
MAXDAEM	65
MAXRDS	66
MAXREP31	67
MAXREP64	68
MODPROT	69
MONPARG	70
NCMPBUF	72
PERFOPT	73

PERFOPT2	74
PUPDTH	75
RESTARTU	76
RETRVBUF	77
SCANPARM	78
SCANTIME	79
SDAEMDEV	81
SDEBGUIP	82
SDEBWRKP	83
SDMOPT	84
SESEDEFOW	85
SESEDEFTO	86
SESNPRV	87
SESNPUB	88
SESOPT	89
SESUMAX	90
SIRAPSYF	91
SIRFACT	94
SIRFUNC	96
SIRMSG	97
SIRPRMPT	98
SIRTERM	99
SIRPDLHW	100
SIRTUNE	101
SOCKMAX	102
SPANSIZE	103
SRSDEFTO	104
SRSMAX	105
SRSMAXTO	106
SRSMAXUS	107
TCPSEV	108
TCPSUBSY	109
TCPTYPE	110
TNDEV	111
WEBAUDIT	112
WEBOPT	114
Chapter 6: User Parameters	115
AUTOGCN	116
ERCNT	118
GCSTATS	119
JANDEBM	120
OBJSTAT	121
OBJSTMIN	124
RETRVOPT	125
SCRNSTBL	127
SIRCOMP	128

SIREEDIT	129
SRSPARM	131
ULTRACE	133
ULTRACEP	134
Chapter 7: Editor Commands	135
CASE Mixed/Upper command	136
Chapter 8: Terminal MODEL 6 support	137
Index	139

Summary of Changes

This section describes significant changes to the documentation. Usually, these changes correspond to enhancements made to the underlying product, although they might be simple documentation improvements.

Sirius Mods Version 7.5

The following changes correspond to changes in *Sirius Mods* since version 7.3:

- New parameters that affect *Janus SOAP ULI* “garbage collection.”: “[AUTOGCN](#)” on [page 116](#) and “[GCSTATS](#)” on [page 119](#).

Sirius Mods Version 7.3

The following changes correspond to changes in *Sirius Mods* since version 7.2:

- The UNICODE command is added (“[UNICODE](#)” on [page 32](#)).
- New parameters that affect *Model 204* retrieve key processing: “[RETRVBUF](#)” on [page 77](#) and “[RETRVOPT](#)” on [page 125](#).
- New bit settings for the FUNCOPTS parameter (“[FUNCOPTS](#)” on [page 56](#)).

Sirius Mods Version 7.2

The following changes correspond to changes in *Sirius Mods* since version 7.1:

- New report writer parameters:
 - “[CURREP31](#)” on [page 46](#)
 - “[CURREP64](#)” on [page 47](#)
 - “[HGHREP31](#)” on [page 62](#)
 - “[HGHREP64](#)” on [page 63](#)
 - “[MAXREP31](#)” on [page 67](#)
 - “[MAXREP64](#)” on [page 68](#)

Sirius Mods Version 7.1

The following changes correspond to changes in *Sirius Mods* since version 7.0:

- New SCRNSTBL user parameter (“[SCRNSTBL](#)” on page 127) specifies the maximum amount of STBL space available to an application that uses a Screen object.
- New X'02' bit for the SIRPRMPT system parameter (“[SIRPRMPT](#)” on page 98) adds the name of the job step to the command mode prompt in an Online.

Sirius Mods Version 7.0

The following changes correspond to changes in *Sirius Mods* since version 6.9:

- MAXBG parameter added (“[MAXBG](#)” on page 64).
- SDEBGUIP (“[SDEBGUIP](#)” on page 82) and SDEBWRKP (“[SDEBWRKP](#)” on page 83) parameters added. They provide default port numbers for the *Sirius Debugger* startup command.
- New X'02' bit for the FUNCOPTS system parameter (“[FUNCOPTS](#)” on page 56) causes all \$list functions that encounter CCATEMP full conditions to cancel the request with a CCATEMP full condition. This feature was also implemented via maintenance zaps in versions 6.8 and 6.9 of the *Sirius Mods*.

Sirius Mods Version 6.9

This manual was first made available.

1.1 Versions and compatibility

The *Sirius Mods* is a collection of object code enhancements to the *Model 204* database-engine nucleus. These enhancements are distributed as components of the *Sirius Mods* and make up a collection of products including those in the Janus family. The *Sirius Mods* include many products such as *Fast/Backup*, *Fast/Reload*, *Janus TCP/IP Base*, and *Janus Web Server*.

While for *Sirius Mods* customers, the commands and parameters documented in this manual appear and behave as standard *Model 204* commands and parameters, they are not part of the base *Model 204* nucleus. The base *Model 204* nucleus, itself, contains a wide variety of commands and parameters. Those commands and parameters are not documented, here, but can be found in the ***Model 204 Command Reference Manual***.

Many commands and parameters are provided to support the *Sirius Mods* products. Some of these commands and parameters are only available at sites authorized for specific products. Other commands, while available, don't really do anything unless a site is authorized for specific products. Nevertheless, all parameters and commands associated with the *Sirius Mods* are documented in this manual. Those that are restricted to sites authorized for particular products are indicated as such.

Some commands, such as the *JANUS* commands or *SIRFACT*, have extensive and complex sets of parameters and perform processing specific to a product or set of products, so they are only documented in passing here. The more complete documentation for these commands can be found in the product specific manuals. For example, the complete *JANUS* command documentation can be found in the ***Janus TCP/IP Base Reference Manual*** and the complete *SIRFACT* command documentation can be found in the ***SirFact Reference Manual***.

While this manual was first available with version 6.9 of the *Sirius Mods*, this document assumes only that a site is running *Sirius Mods* version 6.6 or later. Any documentation that requires a later version of the *Sirius Mods* will be clearly marked to indicate this. For example, a parameter that is only available in versions 6.8 and later of the *Sirius Mods* will have a sentence such as "This parameter requires version 6.8 or later of the *Sirius Mods*" in its documentation. If a feature, \$function, command or parameter is not indicated as requiring any specific version of the *Sirius Mods*, it can be assumed that it is available as documented in all versions of the *Sirius Mods* since version 6.6.

All of the products in the *UL/SPF* suite of User Language based products depend on the *Sirius Mods*. The release of *UL/SPF* is independent of the release of *Sirius Mods* on which it is running, with the exception that certain releases of *UL/SPF* require a minimum release level of the *Sirius Mods*. For example, *UL/SPF* version 6.8 requires *Sirius Mods* version 6.8 or later.

1.2 Related manuals

The person responsible for the installation of *Sirius Mods* should refer to the ***Sirius Mods Installation Guide***.

The ***Sirius Messages Manual*** contains documentation on *Sirius Mods* error messages and so might be useful to system programmers, DBA's, installers and User Language programmers.

Most sites that have the *Sirius Mods* also have access to many \$functions that are provided with *Sirius Mods* products. These are documented in the ***Sirius Functions Reference Manual***. The ***Sirius Functions Reference Manual*** does **not** contain documentation for the functions specific to particular products such as *Janus SOAP*, *Janus Sockets* or *Janus Web Server* These functions are documented in the product specific manuals.

Janus SOAP provides a wide variety of new User Language capabilities, including support for object-oriented programming constructs. In addition, a rich set of basic classes are also available with *Janus SOAP* For more information about these new capabilities and classes, see ***Janus SOAP Reference Manual***.

The product specific manuals for *Sirius Mods* or *Sirius Mods* based products include

- ***Fast/Backup User's Guide***
- ***Fast/Reload Reference Manual***
- ***Janus/Sirius Debugger User's Guide***
- ***Janus Network Security Reference Manual***
- ***Janus Omni Access Module Reference Manual***
- ***Janus Specialty Data Store Reference Manual***
- ***Janus Open Client Reference Manual***
- ***Janus Open Server Reference Manual***
- ***Janus Sockets Reference Manual***
- ***Janus TCP/IP Base Reference Manual***
- ***Janus Web Server Reference Manual***
- ***Janus SOAP Reference Manual***
- ***Sirius Performance Enhancements Reference Manual***
- ***SirDBA User's Guide***
- ***SirFact Reference Manual***
- ***SirFile User's Guide***
- ***SirLib User's Guide***
- ***SirMon User's Guide***
- ***SirPro User's Guide***
- ***SirSafe Reference Manual***
- ***SirScan User's Guide***
- ***Sir2000 Field Migration Facility Reference Manual***
- ***Sir2000 User Language Tools Reference Manual***

This chapter lists all commands available to the system operator via the MVS console MODIFY command or via the VM SMSG facility. Some standard *Model 204* commands, such as SIRIUS or STATUS, may also be issued by replying to the pending HALT message. While these are also operator commands in some sense, this chapter focuses on operator commands issued via MODIFY or SMSG. The term *operator command* in this chapter refers specifically to commands issued via MODIFY or SMSG.

Before *Sirius Mods* version 6.9, the RESTART facility had to be enabled by setting the X'01' bit in the system RESTARTU parameter before any of these commands were available. The RESTARTU system parameter must be set in the EXEC card PARM parameter under MVS or on the command which starts the Online under CMS. Before *Sirius Mods* version 6.9, MODIFY or SMSG command support could also be provided by *SirTune*, which would run as a separate load module that ran *Model 204* as a subtask. The format of the *SirTune* MODIFY/SMSG commands is documented in the ***SirTune Reference Manual***.

Under *Sirius Mods* version 6.9 and later, however, the *SirTune* data collector is integrated into the *Sirius Mods*, so it runs as part of the *Model 204* load module. Since *SirTune* is effectively part of the *Sirius Mods* as of that release, its operator commands are also documented here. The format of many of the *SirTune* and RESTART utility commands are identical, or at least similar. Any differences between the command formats are noted under the documentation for the command.

If the RESTART utility is started in an **ONLINE** (or **BATCH204**) job that is running with *SirTune*, the RESTART utility defers to *SirTune* and allows *SirTune* to provide the required functionality. Under *Sirius Mods* version 6.9 and later, if an Online is brought up under the standalone *SirTune* load module, the *SirTune* integrated with the *Sirius Mods* will defer to the standalone *SirTune*. However, this is not a recommended configuration — the standalone *SirTune* load module should not be used with *Sirius Mods* version 6.9 and later.

The RESTART utility and *SirTune* commands require a certain level of authorization. Under MVS, any user that is capable of issuing a MODIFY command is considered to be authorized to issue operator commands. Under CMS, a user must be authorized to issue these commands. This authorization is accomplished via the RESTAUTH command (“RESTAUTH” on page 22) if the RESTART utility is handling operator commands, and via the AUTHORIZE command in SIRTUNE1 if *SirTune* is handling operator commands. For information on the latter, see the ***SirTune Reference Manual***.

2.1 Issuing MODIFY or SMSG operator commands

Under MVS, you can issue operator commands using the MODIFY operator command. This command can be issued at an operator console or under SDSF or equivalent virtual console system. For example, to issue the STATUS command to *SirTune* running under job PRODONLN under SDSF's LOG screen, you can simply enter

```
/MODIFY PRODONLN,STATUS
```

or

```
/F PRODONLN,STATUS
```

Responses to MODIFY commands go to the system log and should be viewable under SDSF.

Under CMS, these commands can be issued via the SMSG CP command. For example, to issue the STATUS command to *SirTune* running on a virtual machine named PRODONLN, you can simply enter

```
SMSG PRODONLN STATUS
```

or

```
CP SMSG PRODONLN STATUS
```

Responses to SMSG commands are sent via MSGNOH, if the *SirTune/Model 204* service machine is authorized to use MSGNOH, and via MSG otherwise.

2.2 **BUMP user_num**

This operator command requests that the equivalent of the *Model 204* BUMP command be issued against the indicated user number. This command is useful if a high priority user is looping in an Online, making it impossible for anyone else to do anything.

To “bump” user number 18, for example. issue the command:

```
BUMP 18
```

Unlike the *Model 204* BUMP command, the operator BUMP command only accepts a single user number, and it does not accept userids or file names. To determine the user number of the running user, issue the MONITOR operator command (“[MONITOR](#)” on [page 7](#)). It is recommended that the MONITOR command be issued several times to ensure that a single user is indeed looping and that the performance problem is not simply the result of excess demand.

If a loop situation is caused by a *Model 204* bug, it is possible that a BUMP command will fail to force the looping user off the system. In this case, it might be necessary to issue the RESTART command (“[RESTART abend_code USER user_num | TASK task_num](#)” on [page 8](#)) to break the loop.

2.3 CLOSE

This operator command requests that *SirTune* close the sample dataset. This makes it possible to run the report generator against a sample data set that is still being updated by *SirTune*. This command has no effect on sampling activity in the *Model 204* address space.

This command is not available if the RESTART utility is responding to operator commands.

2.4 MONITOR

This operator command requests that the utility return information about what is currently happening in *Model 204*. This command is especially useful if *Model 204* appears to be hung or looping, making it impossible to log on to *Model 204* in order to determine the cause of the problem.

Information is returned for the main *Model 204* task and any subtasks (if the MP/204 feature is being used). The information returned indicates whether each task is running or waiting (corresponding to looping and hung situations, respectively), and where in the *Model 204* load module the task is running or waiting. This latter piece of information is most useful to *Model 204* internals experts.

If any *Model 204* task is indicated as running, information is also provided on the user number and userid of the running user, the current activity (evaluating, compiling, etc.), and the name of the procedure that is currently running. This information can be used to determine whether a BUMP or RESTART command should be issued.

2.5 **RESTART** *abend_code* **USER** *user_num* | **TASK** *task_num*

This RESTART utility command requests that the utility issue the equivalent of a user abend in *Model 204* for the indicated user or task. This command should be used only as a last resort when a *Model 204* Online is looping or hung and all efforts to clear up the situation have failed (see “BUMP” on page 5) In this situation, the only options left are to cancel (FORCE under CMS) the run or to issue the RESTART utility's RESTART command. The RESTART command is preferable because:

- The Online might intercept the abend, possibly take a snap dump, and then continue running.
- Even if the Online does not continue, it might at least intercept the abend and terminate “cleanly”, preventing files from being broken and eliminating the need to run recovery. Cancelling or forcing the Online ensures:
 - Any file with active updating transactions will be left broken.
 - Recovery will have to be run before the Online files can be used again.
- The RESTART command will generally result in a CCASNAP being taken rather than (or in addition to) a SYSMDUMP, SYSUDUMP, or VMDUMP. CCASNAPs are generally easier to deal with than other types of dumps.

The *abend_code* must be a 1- to 3-digit hexadecimal code that indicates the abend code to be used for the artificially generated abend. Under CMS, the abend code must be between 0C1 and 0CF. Any dumps will indicate the specified abend code.

Note: It is important to inform the support personnel that are examining the dump that the abend code was artificially generated by the RESTART utility and that the situation was actually a hung or looping Online.

To prevent accidentally terminating a user or an Online, a user number can be specified on the RESTART command. When a user number is specified on the RESTART command, no simulated abend will occur if the indicated user is not running in the Online. For example:

```
RESTART 0C4 USER 23
```

simulates a 0C4 abend if user 23 is running. If user 23 is not running, no abend will be simulated.

If no user is running (a hung Online), it might be necessary to issue the RESTART command with a task number instead. Unless running MP/204, the task number must be specified as 0. If MP/204 is running, the task number must be 0 (for the maintask) or the subtask number (from 1 to NMPSUBS) to be restarted. For example

```
RESTART 0A9 TASK 0
```

simulates a 0A9 abend on the *Model 204* maintask. Note that it is almost always preferable to specify a user number rather than a task number on the RESTART command.

2.6 **SAMPLE ON | OFF | AUTO**

This command either places *SirTune* sampling back into automatic mode (SAMPLE AUTO), where sampling is controlled by INCLUDE/EXCLUDE statements in SIRTUNEI, or it initiates (SAMPLE ON) or terminates (SAMPLE OFF) collection of sample data. This command has no effect on the collection of compilation data: Compilation data is collected regardless of the sampling state.

For example, to immediately start collecting data, the following command should be issued:

```
SAMPLE ON
```

To immediately stop collecting data, issue:

```
SAMPLE OFF
```

After a SAMPLE ON or SAMPLE OFF command is issued, *SirTune* is in “manual” sampling mode.

This command is not available if the RESTART utility is responding to operator commands.

2.7 STATUS

This command requests the current status of the RESTART utility or *SirTune*. If the response is from the RESTART utility, it indicates the number of restarts that have been performed (hopefully 0) since the Online started. If the response is from *SirTune*, it indicates *SirTune*'s current sampling mode (AUTO or MANUAL), state (ON or OFF), and the number of samples collected to this point.

2.8 STOP

This command requests that *SirTune* stop collecting both sampling and compilation data and close the sample dataset. After this command is issued, sampling cannot be restarted for the run.

Since the sample dataset is closed by a STOP command, it is possible to use this dataset to generate reports after a STOP command, even while the ONLINE/BATCH204 job continues to run.

This command is not available if the RESTART utility is responding to operator commands.

System Administrator Commands

This chapter lists all available *Model 204* commands provided by the *Sirius Mods* that can only be issued by system administrators.

3.1 BUMPSNAP user

The BUMPSNAP command is identical to the *Model 204* BUMP command with the exception that a CCASNAP is taken for the user or users being BUMPed. This can be useful in diagnosing a hung user situation where the error is believed to be a *Model 204* internals problem. Whether or not a CCASNAP or SYSMDUMP/VMDUMP is actually taken depends on many factors, including the settings of the SNAPCTL system parameter, the presence of DD cards or FILEDEFS for CCASNAP or dump datasets, and the setting of SNAPLIM and the current number of CCASNAPs already taken.

The syntax of the BUMPSNAP command is identical to that of the BUMP command, so the *Model 204 Command Reference Manual* should be consulted for more details.

The BUMPSNAP command, like the BUMP command, allows BUMPing and SNAPPING of more than one user with a single command. Generally, this is not what is desired with a BUMPSNAP command, so use caution when issuing a BUMPSNAP command that doesn't specify a user number (but instead specifies something more generic like a userid or a subsystem name or file), since this could result in a large number of snap dumps being taken. For most purposes, the BUMPSNAP command should simply be followed by the user number of the user for which a snap is desired.

The BUMPSNAP command can be issued as either of these:

- An operator command, that is, as a reply to the HALT message under OS/390
- A command typed on the Online virtual console under CMS

3.2 DUMP | DUMPX

The DUMP command works exactly like the *Model 204* DUMP command, with the exception that at *Fast/Backup* authorized sites, DUMP invokes *Fast/Backup* rather than the standard *Model 204* dump processing. If for some reason a *Fast/Backup* site wishes to use the standard *Model 204* DUMP command processing (most likely for diagnosing a problem with *Fast/Backup*), a DUMPX command can be issued. The DUMPX command has syntax identical to the DUMP command.

See the ***Model 204 Command Reference Manual*** for more information about the DUMP command, and see the ***Fast/Backup User's Guide*** for more information about *Fast/Backup*.

3.3 DUMPG | DUMPGX

The DUMPG command works exactly like the *Model 204* DUMPG command, with the exception that at *Fast/Backup* authorized sites, DUMPG invokes *Fast/Backup* rather than the standard *Model 204* dump processing. If for some reason a *Fast/Backup* site wishes to use the standard *Model 204* DUMPG command processing (most likely for diagnosing a problem with *Fast/Backup*), a DUMPGX command can be issued. The DUMPGX command has syntax identical to the DUMPG command.

See the ***Model 204 Command Reference Manual*** for more information about the DUMPG command, and see the ***Fast/Backup User's Guide*** for more information about *Fast/Backup*.

3.4 FILELOAD | FILELOADX

The FILELOAD command works exactly like the *Model 204* FILELOAD command, with the exception that at *Fast/Reload FLOD Compiler* authorized sites, FILELOAD invokes the *Fast/Reload FLOD Compiler* rather than the standard *Model 204* dump processing. If for some reason a *Fast/Reload FLOD Compiler* site wishes to use the standard *Model 204* FILELOAD command processing (most likely for diagnosing a problem with the *Fast/Reload FLOD Compiler*), a FILELOADX command can be issued. The FILELOADX command has syntax identical to the FILELOAD command.

See the ***Model 204 File Manager's Guide*** for more information about the FILELOAD command, and see the ***Fast/Reload Reference Manual*** for more information about *Fast/Reload FLOD Compiler*.

Note that while the *Fast/Reload FLOD Compiler* FILELOAD command does not support all the statements supported by the standard FILELOAD command, if such an unsupported statement is detected, the *Fast/Reload FLOD Compiler* automatically “hands off” the request to the standard FILELOAD command. This handing off is done by saving the FILELOAD program in a temporary procedure, the number of which is controlled by the FRELPREV system parameter (“FRELPREV” on page 55).

3.5 FLOD | FLODX

The FLOD command works exactly like the *Model 204* FLOD command, with the exception that at *Fast/Reload FLOD Compiler* authorized sites, FLOD invokes the *Fast/Reload FLOD Compiler* rather than the standard *Model 204* dump processing. If for some reason a *Fast/Reload FLOD Compiler* site wishes to use the standard *Model 204* FLOD command processing (most likely for diagnosing a problem with the *Fast/Reload FLOD Compiler*), a FLODX command can be issued. The FLODX command has syntax identical to the FLOD command.

See the ***Model 204 File Manager's Guide*** for more information about the FLOD command, and see the ***Fast/Reload Reference Manual*** for more information about *Fast/Reload FLOD Compiler*.

Note that while the *Fast/Reload FLOD Compiler* FLOD command does not support all the statements supported by the standard FLOD command, if such an unsupported statement is detected, the *Fast/Reload FLOD Compiler* automatically “hands off” the request to the standard FLOD command. This handing off is done by saving the FLOD program in a temporary procedure, the number of which is controlled by the FRELPREV system parameter (“FRELPREV” on page 55).

3.6 FNADDR funcname | LOADMOD loadModName | +hexFuncNumber

The FNADDR command is used to determine the location in the *Model 204* load module of a specified \$function, or the location of the *Fast/Unload* load module, if the *Fast/Unload User Language Interface* is active.

`FNADDR funcname | LOADMOD loadModName | + hexFuncNumber`

FNADDR command syntax

The function name specified after FNADDR should not include the initial leading dollar sign (or pound in England). For example, this finds the location of the \$SUBSTR function:

```
FNADDR SUBSTR
```

This command is mostly useful for diagnosis of *Model 204* internals problems, especially those related to \$functions. It can also be useful in determining whether a particular \$function is local, a Sirius \$function, or part of the *Model 204* base.

If **LOADMOD** is specified, it is followed by the name of the load module to locate. Currently, the only load module name supported is FUNLOAD, so the following command can be used for diagnosis by Sirius Software when the *Fast/Unload User Language Interface* is active:

```
FNADDR LOADMOD FUNLOAD
```

If **+ hexFuncNumber** is specified, the number is that used in QTBL for invocations of the \$function.

The FNADDR command is available in *Sirius Mods* version 6.2 and later. The LOADMOD qualifier of FNADDR is available in *Sirius Mods* version 6.8 and later. The hex function number form of FNADDR is available in *Sirius Mods* version 7.2 and later.

3.7 FUDAEMON

This command, only useful under CMS, indicates that a thread is to act as a *Fast/Unload* task for *Fast/Unload User Language Interface* requests. It is necessary under CMS if *Fast/Unload User Language Interface* is to be used, because of the lack of true multi-tasking support by CMS. The thread on which the FUDAEMON command is issued (presumably an IODEV 3) simulates an MVS task running *Fast/Unload*.

FUDAEMON

FUDAEMON command syntax

3.8 JANUS

The JANUS command is used to specify the configuration of Janus ports and, in a few cases, system-wide Janus configuration settings. While the JANUS command can be issued by customers that are not authorized for any Janus products, only web ports can be defined at these sites, and only at sites with a minimal number of ports (3).

The JANUS command can be issued as either of these:

- An operator command (as of *Sirius Mods* version 6.2), that is, as a reply to the HALT message under OS/390
- A command typed on the Online virtual console under CMS

See the ***Janus TCP/IP Base Reference Manual*** for the complete JANUS command syntax.

3.9 RESTAUTH

The RESTART utility allows users outside the *Model 204* address space to issue certain commands while the **ONLINE** or **BATCH204** job is running. Users can be authorized to do this with the RESTAUTH command. Since under MVS it is impossible to determine the originator of a MODIFY command, any user that is capable of issuing a MODIFY command is considered to be authorized by the RESTART utility. Since generally, most users don't have this privilege, this is not a problem.

Under CMS, however, RESTART utility commands are issued via the VM SMSG facility, a facility available to all users. Fortunately, the issuer of a SMSG command to the RESTART utility can be determined so a user must be authorized to issue RESTART utility commands via the RESTAUTH command. RESTAUTH is a *Model 204* command that can only be issued by a system manager or system administrator.

`RESTAUTH userlist`

RESTAUTH command syntax

The RESTAUTH command should be followed by a list of userids that are authorized to issue RESTART utility commands. For example:

```
RESTAUTH SKIPPER GILLIGAN
```

authorizes CMS users SKIPPER and GILLIGAN to issue RESTART utility commands (via SMSG). Wildcards can also be used in the userids. An asterisk matches any group of characters and a question mark matches any single character. For example

```
RESTAUTH PROF* *ANN GING?? *HOWE*
```

authorizes any CMS userid that begins with the letters 'PROF', ends with the letters 'ANN', begins with the letters 'GING' and is exactly six characters long, or has the letters 'HOWE' anywhere in it to issue RESTART utility commands (via SMSG).

3.10 RESTORE | RESTOREX

The RESTORE command works exactly like the *Model 204* RESTORE command with the exception that at *Fast/Backup* authorized sites RESTORE invokes *Fast/Backup* rather than the standard *Model 204* restore processing. If for some reason a *Fast/Backup* site wishes to use the standard *Model 204* RESTORE command RESTOREX command can be issued. The RESTOREX command has syntax identical to the RESTORE command.

See the ***Model 204 Command Reference Manual*** for more information about the RESTORE command and see the ***Fast/Backup User's Guide*** for more information about *Fast/Backup*.

3.11 RESTOREG | RESTOREGX

The RESTOREG command works exactly like the *Model 204* RESTOREG command, except that at *Fast/Backup* authorized sites RESTOREG invokes *Fast/Backup* rather than the standard *Model 204* restore processing. If for some reason a *Fast/Backup* site wishes to use the standard *Model 204* RESTOREG command processing (most likely for diagnosing a problem with *Fast/Backup*), a RESTOREGX command can be issued. The RESTOREGX command syntax is identical to the RESTOREG command.

See the ***Model 204 Command Reference Manual*** for more information about the RESTOREG command, and see the ***Fast/Backup User's Guide*** for more information about *Fast/Backup*.

3.12 SIRAPSY

This SIRAPSY command allows the manual setting of the resident/non-resident attribute of a pre-compiled subsystem procedure. This is useful in overriding *Model 204*'s default behavior if the decisions it makes based on RESTHRSH and RESLTHR are not appropriate. The SIRAPSY command can only be run against an active subsystem so a logical place for SIRAPSY commands might be in a subsystem start/init proc.

```
SIRAPSY DIS | DISPLAY | RES | NORES subsystem proc
```

SIRAPSY command syntax

where

subsystem is the name of the subsystem to which the command applies.

proc is the name of the procedure to which the command applies.

For example

```
SIRAPSY RES SIRMOM MOPR-MENU
```

sets SIRMOM procedure MOPR-MENU to become resident next time it is loaded.

```
SIRAPSY NORES SIRMOM MOPR-SYSOVR
```

sets SIRMOM procedure MOPR-SYSOVR so that it will never be set to resident. SIRAPSY NORES will not drop the residency attribute of a procedure once it is made resident.

This command is only available with the *Sirius Performance Enhancements V2*. and can only be issued by a system manager or system administrator.

3.13 SIRFACT

The SIRFACT command invokes the *SirFact* post hoc debugging facility. SIRFACT has a set of several subcommands that lets you tailor, trap early, and otherwise maximize the error information dumped for *Model 204* debugging purposes.

SIRFACT commands can also be issued as operator commands, that is, on the Online virtual console under VM or as the response to the HALT message under OS/390.

The SIRFACT command is described in detail in the ***SirFact Reference Manual***.

SIRFACT subcomm operands

SIRFACT command syntax

where

subcomm One of the subcommands listed below.

operands One or more operands specific to the SIRFACT subcommand invoked. These are described in detail in the ***SirFact Reference Manual***.

The SIRFACT subcommands are summarized below and described in detail in the ***SirFact Reference Manual***. You invoke a subcommand with the “SIRFACT” prefix (for example, SIRFACT CANCEL, SIRFACT MAXDUMP).

CANCEL Indicates which return codes from \$functions should result in request cancellation. Can be abbreviated “CAN”.

DISPLAY Shows the currently active SIRFACT subcommands. Can be abbreviated “DISP.”.

DUMP Indicates which request cancellations should cause a *SirFact* dump to be taken and where the dump is to go.

IGNORE Indicates which request cancellation error messages are not to produce *SirFact* dumps. Can be abbreviated “IGN.”.

MAXDUMP Places limits on the number of *SirFact* dumps that will be taken. The default system limit is 0 so a SIRFACT MAXDUMP **must** be issued to get **any** SIRFACT dumps. Can be abbreviated “MAXD”.

NOCANCEL Indicates that certain return codes from \$functions should not result in request cancellation. Cancels out the effect of a SIRFACT CANCEL command. Can be abbreviated “NOCAN”.

NODUMP	Indicates that certain request cancellations should not result in <i>SirFact</i> dumps being produced. Cancels out the effect of a SIRFACT DUMP command.
NOIGNORE	Indicates that certain request cancellation error messages are to produce <i>SirFact</i> dumps. Cancels out the effect of a SIRFACT IGNORE command. Can be abbreviated “NOIGN”.
QUIESCE	Facilitates updates of APSY subsystem procedures while the subsystem is up and in use.
RECNDUMP	Establishes the number of record numbers from each found set or LIST to be dumped. Can be abbreviated “RECND.”.
RESUME	Stops the effect of a SIRFACT QUIESCE command and returns the subsystem to normal operation.
SNAP	Requests that a <i>SirFact</i> dump be taken for another thread.

3.14 SIRFIELD

The SIRFIELD command allows you to control access to and update of fields in a file. This control is provided in any *Model 204* load module (BATCH204, ONLINE, IFAM4, IFAM1, etc.) with the *Sir2000 Field Migration Facility*.

The SIRFIELD command is used to assign special attributes to a field name; the purpose of these attributes is to provide additional checking of uses of the field, and to ease a migration period for applications using the field. The general form of the SIRFIELD command is:

`[IN file] SIRFIELD name subcommand operands`

SIRFIELD command syntax

where

IN file specifies the file to which the SIRFIELD command applies. This is only necessary if *file* is not the default file. *IN file* can be specified with any SIRFIELD subcommand but, for readability, is not presented in subsequent syntax diagrams.

name is alias name or fieldname in the current file (it must already be defined). If it contains blanks or special characters, the entire name must be enclosed in apostrophes, for example, SIRFIELD 'DATE OF BIRTH' ... Note that certain SIRFIELD subcommands accept either a fieldname or an alias, while other subcommands accept only a fieldname or only an alias.

subcommand Indicates the operation being performed. It is one of the words *ALIAS*, *DELETE*, *FORMAT*, *DISPLAY*, *RELATE*, or *SET*. Meanings of these are described below. The subcommands are described briefly below and in detail in the *Sir2000 Field Migration Facility Reference Manual*.

operands The operands specific to the operation. These are described in the *Sir2000 Field Migration Facility Reference Manual*.

The SIRFIELD command may not be issued against a file that has “CCA” as the first three characters of the file name.

The SIRFIELD subcommands are summarized below and described in detail in the *SirFact Reference Manual*. You invoke a subcommand with the “SIRFACT” prefix (for example, SIRFACT CANCEL, SIRFACT MAXDUMP).

ALIAS Provide an additional name that programs can use to refer to a field.

DELETE Removes an alias name.

DISPLAY Shows the attributes that SIRFIELD has associated with a field.

FORMAT Ensures that only values of the specified format(s) are stored in the field, and associates the format, CENTSPAN, and SPANSIZE with the field for use in applications or in the SIRFIELD RELATE subcommand.

RELATE Causes any update of a field to be reflected by the equivalent update to another field.

SET Specifies special processing options for a field or alias name.

This command will perform no useful processing unless a site is authorized for the *Sir2000 Field Migration Facility*.

3.15 SIRIUS [qualifier]

The SIRIUS command is used to determine basic release and authorization for an Online.

`SIRIUS [ALL | COMPLETE | DEBUG | EXPWARN | MAINT | SNAPLMOD]`

SIRIUS command syntax

If no qualifier is specified after SIRIUS, the command displays the site ID (as declared by Sirius Software), the *Model 204* and *Sirius Mods* versions being run, the jobname, CPU id all products for which the site is authorized and the used areas of the Sirius Software patch space.

One of the following optional qualifiers can also be used:

- ALL** The information described above is provided, as well as a list of all products for which the site is not authorized.
- COMPLETE** The information provided with the *ALL* qualifier is provided as well as a list of all pseudo-products for which the site is not authorized. Pseudo-products are not really products but control access to certain commands and facilities that are not made available to everyone. If the output from a SIRIUS COMPLETE is intriguing, contact Sirius Software for more information.
- DEBUG** See [“SIR\[IUS\]DEBUG” on page 38.](#)
- EXPWARN** Only products with impending expiration dates will be displayed.
- MAINT** The information of the unqualified SIRIUS command is returned, except that no product authorization or expiration information is indicated, and patch space usage is presented in an abbreviated form.
- SIRIUS MAINT, combined with the DISPLAY EW ALL command, provides what should be a definitive “footprint” of the versions of *Model 204*, *Sirius Mods*, and *Fast/Unload* in use, and of the maintenance applied to *Model 204* and *Sirius Mods*.
- SNAPLMOD** This form of the SIRIUS command dumps, to CCASNAP, the contents of the *Model 204* load module; it may be requested by Sirius Software to assist in maintenance. This form is restricted to System Managers.

3.16 SIRMETH

The SIRMETH command provides system managers with the ability to allow specific subsystems or even all users, whether or not they are system managers, to do certain things that previously only system managers and pre-compiled subsystem procedures could do. These things are:

- Set system globals and strings.
- Set subsystem globals and strings.
- Set a subsystem context.

```
SIRMETH {ALLOW | DISALLOW} -
        {SYSTEMSET | SUBSYSTEMSET |
        SUBSYSTEMCONTEXT csubsys | ALL} -
        [SUBSYSTEM subsys [NONPRE] ]
```

For more information about the SIRMETH command, see the *Janus SOAP Reference Manual*.

3.17 UNICODE

The UNICODE command is used to manage the Unicode tables, which specify translations between EBCDIC and Unicode/ASCII. The command also lets you replace individual Unicode characters by designated character strings, and it has varied options for displaying translation table codepages and code point mappings, as well as displaying any translation customizations you have specified.

The general form of the UNICODE command is:

UNICODE subcommand operands

UNICODE command syntax

where

subcommand A term that indicates which operation is being performed. `List`, `Difference`, and `Display` are subcommands that only produce an information display; `Table` produces a character translation update.

operands The operands specific to the operation.

The UNICODE subcommands are described below in separate sections according to type (display or update). Only the update forms of UNICODE require System Administrator (or User 0) privileges.

Subcommand and operand keywords of the UNICODE command may be entered in any combination of uppercase or lowercase letters. The term “UNICODE” that starts the command must be entered entirely in uppercase letters.

The command descriptions that follow use an initial capital letter to indicate a keyword, and they use all-lowercase letters to indicate a term that is substituted for a particular value in the command.

The UNICODE command is available as of *Sirius Mods* version 7.3. For UNICODE examples and more information about Sirius Unicode support, see the ***Janus SOAP Reference Manual***.

3.17.1 Display forms of UNICODE

The UNICODE subcommands that produce information displays are described below. In the descriptions:

- *h2* is two hexadecimal digits.
- *hex4* is four hexadecimal digits, excluding FFFE, FFFF, and the surrogate areas (D800 through and including DFFF).

The display forms of the UNICODE command are:

UNICODE List Codepages

Obtains a list of all codepages.

UNICODE Difference Codepages name1 And name2 [Range E=h2 To E=h2]

Obtains a list of the differences between two codepages for the EBCDIC range specified. The default range is 00 to FF.

UNICODE Difference Xtab name1 And Codepage name2 [Range E=h2 To E=h2]

Obtains a list of the differences between a JANUS XTAB table and a codepage for the EBCDIC range specified. The default range is 00 to FF.

UNICODE Display Codepage name

Obtains, in commented form, the maps (see the `Map` update subcommand in “Update forms of UNICODE”) of the specified codepage.

UNICODE Display Table Standard

Obtains, in command form, a display of any current replacements and current maps and/or translations (see the `Trans` update subcommands in “Update forms of UNICODE”) that differ from the base.

3.17.2 Update forms of UNICODE

The updating forms of the UNICODE command begin with the keyword `Table` and have the following format:

```
UNICODE Table Standard subcommand
```

The *subcommand* values are described below.

For the updating subcommands:

- The user must be a System Administrator (or user 0).
- These commands should only be invoked during *Model 204* initialization, because other users running at the same time as the change may obtain inconsistent results, including the results of `UNICODE Display` (described in the previous section).

The *subcommand* values of the updating form of the UNICODE command follow:

Base Codepage name

Replace the current translation tables with those derived from the named codepage.

Trans E=h2 To U=hex4

Specify one-way translation from EBCDIC point *h2* to Unicode point *hex4*.

Trans E=h2 Invalid

Specify that the given EBCDIC point is not translatable to Unicode.

Trans E=h2 Base

Remove any customized translation or mapping specified for the given EBCDIC point, thus returning to the base codepage translation for the point.

Trans U=hex4 To E=h2

Specify one-way translation from Unicode point *hex4* to EBCDIC point *h2*.

Trans U=hex4 Invalid

Specify that the given Unicode point is not translatable to EBCDIC.

Trans U=hex4 Base

Remove any customized translation or mapping specified for the given Unicode point, thus returning to the base codepage translation for the point.

Trans All Base

Remove any customized translation or mapping specified from all Unicode and EBCDIC points.

Map E=h2 Is U=hex4

Specify mapping from EBCDIC point *h2* to Unicode point *hex4*, and from Unicode point *hex4* to EBCDIC point *h2*.

Map U=hex4 Is E=h2

Same as `Map E=h2 Is U=hex4`.

Rep U=hex4 'str'

Specify replacement for Unicode point *hex4* by the Unicode string *str*. *str* may be a series of the following:

- Non-ampersand EBCDIC characters (which must be translatable to Unicode)
- `&` ; (for an ampersand)
- A character reference of the form `&#xhhhh;`

The length of the resulting Unicode replacement string is limited to 127 characters. No character in the replacement string may be the `U=hex4` value in any `Rep` subcommand.

Norep U=hex4

Specify that there is no replacement string for Unicode point *hex4*.

Norep All Specify that there is no replacement string for any Unicode point.

CHAPTER 4 ***User Commands***

This chapter lists all available *Model 204* commands provided by the *Sirius Mods* that can be issued by general users.

4.1 **APPDATE**

The APPDATE command allows you to control the operation of date and time oriented User Language \$functions, in the following two ways:

1. You can specify a system-wide or user-level clock that is used to obtain date and time values for User Language \$functions. For application testing, this is preferred to the *Model 204* SYSDATE parameter, which is much less flexible and which greatly complicates the ability to do things such as share procedure files or read-only data files in the testing environment.

The APPDATE command affects only date and time oriented User Language \$functions; it does not affect any other date or time in the *Model 204* environment.

2. You can use the DATE_ERR clause to set switches that control the default system behaviour when errors are encountered in date and time oriented User Language \$functions. The choices are to produce an error message along with request cancellation, produce a warning message, or silently continue with the request. This control is available at the system and user level, and can also be set for the duration of a User Language request, via the \$SIR_DATE_ERR function. At the system and user level, you can also control whether procedure names and line numbers are available for error messages.

This command will perform no useful processing unless a site is authorized for the *Sir2000 User Language Tools*. The APPDATE command has an extensive set of parameters and is more completely documented in the ***Sir2000 User Language Tools Reference Manual***.

4.2 JAN[US]DEB[UG]

Invokes the *Janus Web Server* debug facility. This facility makes it possible to run a web request on 3270 threads so that 3270 debugging tools could be used to debug these requests. The need for this facility is largely obviated by the *Janus Debugger* in *Sirius Mods* 6.9 and later.

This command will perform no useful processing unless a site is authorized for *Janus Web Server*. For more information, see the ***Janus Web Server Reference Manual***.

4.3 SIR[IUS]DEBUG

Starts and stops the *Sirius Debugger* (for debugging 3270, BATCH2, or other non-web applications), and can also be used for debugging programs served by *Janus Web Server* threads. Connects to a Debugger Client on a workstation, from where debugging is orchestrated via the Client GUI.

The commands `SIRIUSDEBUG`, `SIRDEBUG`, and `SIRIUS DEBUG` are interchangeable. The general form of the command is:

```
SIRIUSDEBUG { [ON|OFF] [SUSPEND|RESUME] [STATUS] }
```

SIRIUSDEBUG command syntax

Where:

- The ON subcommand starts a *Sirius Debugger* session (it requires additional parameters). It can also be used for debugging *Janus Web Server* threads.
- The OFF subcommand stops a debugging session.
- The SUSPEND and RESUME subcommands discontinue and continue a debugging session.
- The STATUS subcommand gets a status report about the worker threads for the *Janus Debugger*, the *Sirius Debugger*, or both.

This command will perform no useful processing unless a site is authorized for the *Sirius Debugger* (and for *Janus Web* threads, *Janus Debugger*). For more information, see the ***Janus/Sirius Debugger User's Guide***.

This chapter lists all available *Model 204* parameters provided by the *Sirius Mods* that can only be reset by system administrators. These include parameters that must be set in the CCAIN stream or on the EXEC card (or *Model 204* command under CMS). Note that most user parameters (which are documented in [“User Parameters” on page 115](#)) can also be set on the IODEV cards in the CCAIN stream.

5.1 ACTRDS

Default value 0

Parameter type System

Where set System manager resettable

Related products *Janus Network Security*

Introduced Before *Sirius Mods 6.7*

The ACTRDS system parameter is a numeric parameter that indicates the number of special-purpose public-key/private-key encryption/decryption subtasks to be used. Since public-key/private-key encryption/decryption is extremely CPU intensive, it can be useful to have the encryption/decryption performed in a special-purpose subtask, freeing the main *Model 204* task for other work. This parameter can be reset by a system manager but will not cause more than MAXRDS (“MAXRDS” on page 66) tasks to be used at any time. Since CMS does not support true multi-tasking, this parameter has no effect under CMS. Also, since public-key/private-key encryption is only used by *Janus Network Security*, this parameter has no effect at sites not authorized for *Janus Network Security*.

Setting this parameter is especially useful at sites where:

- The Online is CPU constrained in the main task.
- There are usually or often idle CPUs available.
- The MP/204 feature is not available to move main task work to idle CPUs.

In an MP/204 environment, it's probably not worth setting ACTRDS as it would probably make more sense to add an additional MP/204 subtask to the Online by increasing the value of the AMPSUBS system parameter. In a non-MP/204 environment, it probably doesn't make much sense (though is also largely harmless) to set an ACTRDS value greater than 1.

5.2 APSYSEC

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>SirSafe</i>
Introduced	<i>Sirius Mods 6.7</i>

The APSYSEC parameter makes it possible for a system manager to START, STOP, DEBUG, or TEST any subsystem, without have to add the system manager to the sclass authorized to do these things. Setting this parameter has no effect at sites not authorized for *SirSafe*.

The APSYSEC parameter is a bitmask parameter where the bits mean:

X'01' System managers are allowed to START, STOP, DEBUG, or TEST any subsystem. This saves the effort of adding a system manager to a privileged sclass in every subsystem in an Online so that the system manager could at least start and stop the subsystems — a common thing for system managers to need to do.

In fact, if no users other than system managers need to start or stop subsystems, this can eliminate the need to even have sclasses in a subsystem to allow starting or stopping of the subsystem. In some cases, eliminating this need can reduce the subsystem definition to a single default sclass, which has performance benefits — no sclass lookup is required when a user enters a subsystem, and no sclass-specific compiles are done for the procedures in the subsystem.

Because of the overhead associated with multiple sclasses in a subsystem (not huge, but possibly measurable), some sites take the risk of adding START and STOP privileges (and perhaps TEST and DEBUG) to the one and only sclass for a subsystem. This, of course, means that any user can start and stop the subsystem, which might not be ideal from a control or security perspective.

The APSYSEC parameter allows such a site to keep one sclass but remove the risk. START and STOP privileges can be removed from the default/only sclass for a subsystem, and only system managers (or users running subsystems that give them system manager privileges) can start or stop subsystems.

To continue using *Model 204's* traditional fine-grained control of START, STOP, TEST, and DEBUG privileges, the APSYSEC X'01' bit should *not* be set.

5.3 CENTSPAN

Default value	-50
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

The CENTSPAN system parameter is a numeric parameter that indicates the default CENTSPAN value to be used for the processing of dates containing two-digit years, in a wide variety of \$functions. The default setting for CENTSPAN of -50 means that start of the assumed range for two-digit years (in affected \$functions) is 50 years before the current date.

Some of the \$functions affected by this CENTSPAN setting are:

- \$sir_date2n
- \$sir_date2nd
- \$sir_date2nm
- \$sir_date2ns
- \$sir_dateChg
- \$sir_dateChk
- \$sir_dateCnv
- \$sir_dateDif
- \$web_date2n
- \$web_date2nd
- \$web_date2nm
- \$web_date2ns

For more information about CENTSPAN, see the ***Sirius Functions Reference Manual***.

5.4 COMMLOG

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

This is a bitmask parameter that affects the type of login performed by daemon threads when logged in via a \$comm function (\$command, \$commndl, or \$commbg), or via a New method for a daemon class object.

The bits in this parameter mean:

- X'01'** A login that bypasses external authorizer (RACF, ACF2, Top Secret) validation is performed.
- X'02'** A trusted login is performed via the external authorizer.

If COMMLOG is set to zero, the *Sirius Mods* attempts to determine if the external authorizer (if any) supports trusted login (RACF, ACF2, and Top Secret all support trusted login):

- If the authorizer does, the *Sirius Mods* sets COMMLOG to X'02' and then does all trusted logins via the external authorizer.
- If the authorizer does not, the flag is set to X'01', and all logins for \$comm and daemon objects do not use the external authorizer.

Generally, this parameter should be left as 0, though it might make sense to set it to X'01' for efficiency — there is considerable overhead in an external authorizer login, even a trusted one.

The downside of setting this parameter to X'01' is that no external authorizer logging will be performed for the daemon login (this might be viewed as a benefit), and things that required external authorizer validation (such as sequential file access) will not be correctly controlled by the external authorizer.

5.5 COMPOPT

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This bitmask parameter facilitates migration to mixed-case User Language. Its bit settings have the following meanings:

- X'01'** If on, all procedures start out in *Sirius Case ToUpper* mode, whether or not they begin with a mixed-case Begin statement. *Sirius Case ToUpper* mode translates all unquoted tokens to uppercase, so User Language statements, keywords, variable names, etc. may be written in mixed case. By setting the COMPOPT X'01' bit, a site is essentially enabling mixed-case User Language almost everywhere.
- X'02'** If on, *Sirius Case Leave* compiler directives are to be ignored: if *Sirius Case ToUpper* is in effect, it remains in effect even if a *Sirius Case Leave* directive is encountered. Setting the COMPOPT X'02' bit along with the X'01' bit enables mixed-case User Language everywhere, thus ensuring consistent language processing throughout an Online.
- X'04'** If on, image or image-item names, either literal or in variables, are to be automatically converted to uppercase before being used in methods or \$functions. Since mixed-case User Language is accomplished by translating unquoted tokens to uppercase, this case conversion for image or image-item names is the runtime equivalent of the compiler mixed-case support.

Setting the COMPOPT X'04' bit enables image and image-item names that appear as literals in User Language programs to be entered in mixed case. The only time this might be a problem is if there are true mixed-case image or image-item names in an application. A true mixed-case image or image-item name is one that is written in mixed case, either inside quotation marks (image and image-item names can *indeed* be quoted) or without *Sirius Case ToUpper* in effect. In general, neither of these is too likely, so true mixed-case image or image-item names are not likely in most applications.

5.6 CSIPID

Default value	'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter is the identifier of the CSI TCP/IP server under VSE. This is a two-character identifier that must match the `ID=` parameter on the `VSE // EXEC` statement for the TCP/IP partition with which Janus will communicate.

The default value of `00` is also the CSI TCP/IP default.

5.7 CURREP31

Default value	0
Parameter type	System
Where set	Not resettable
Related products	<i>SirTune</i>
Introduced	<i>Sirius Mods 7.2</i>

This parameter displays the amount of 31-bit virtual storage currently being used by report writers (currently, only the Dataset class SirtuneReport method).

See also:

- [“CURREP64” on page 47](#)
- [“HGHREP31” on page 62](#)
- [“HGHREP64” on page 63](#)
- [“MAXREP31” on page 67](#)
- [“MAXREP64” on page 68](#)

5.8 CURREP64

Default value	0
Parameter type	System
Where set	Not resettable
Related products	<i>SirTune</i>
Introduced	<i>Sirius Mods 7.2</i>

This parameter displays the amount of 64-bit virtual storage currently being used by report writers (currently, only the Dataset class SirtuneReport method).

See also:

- [“CURREP31” on page 46](#)
- [“HGHREP31” on page 62](#)
- [“HGHREP64” on page 63](#)
- [“MAXREP31” on page 67](#)
- [“MAXREP64” on page 68](#)

5.9 **DEBUGMAX**

Default value 0

Parameter type System

Where set User 0 CCAIN parameters

Related products *Janus Debugger* and *Sirius Debugger*

Introduced *Sirius Mods 6.8*

This parameter sets the number of internal debugging control blocks that will be allocated. One of these control blocks is required for each active *Janus Debugger* and *Sirius Debugger* session.

Specify a **DEBUGMAX** value of at least the number of seats authorized by the sum of the authorization keys for the *Janus Debugger* and the *Sirius Debugger*. You can set it higher, however, if you anticipate getting a key in the future that adds seats.

Note: The maximum number of concurrent debug sessions may never exceed the seat count of the key(s), no matter what value you specified for **DEBUGMAX**.

5.10 DEBUGPAG

Default value	100
Parameter type	System
Where set	System manager resettable
Related products	<i>Janus Debugger</i> and <i>Sirius Debugger</i>
Introduced	<i>Sirius Mods</i> 6.8

This numeric parameter specifies the upper limit of the CCATEMP page allowance for a debugging session for the *Janus Debugger* or *Sirius Debugger*. The Debugger uses CCATEMP pages for its audit trail and source code lines.

If you are going to debug large programs (more than 1000 lines in a single request), increase the DEBUGPAG setting to, say, 250. Its maximum is 25000.

The default DEBUGPAG setting was changed from 10 to 100 in *Sirius Mods* version 6.9.

5.11 DUMPOPTS

Default value	X'00'
Parameter type	System
Where set	System manager resettable
Related products	<i>Fast/Backup</i>
Introduced	Before <i>Sirius Mods</i> 6.7

This bitmask parameter sets system-wide DUMP options. This bits have the following meanings:

- X'80'** Indicates that if the DUMPTMIN (“[DUMPTMIN](#)” on page 52) or DUMPTMAX (“[DUMPTMAX](#)” on page 51) values are reached, the dump should be cancelled. If this bit is not set, threads needing to update pages while *Fast/Backup* is waiting for a tape mount will also wait on the tape mount.

See also “[DUMPTMAX](#)” on page 51 and “[DUMPTMIN](#)” on page 52.

5.12 DUMPTMAX

Default value	0
Parameter type	System
Where set	System manager resettable
Related products	<i>Fast/Backup</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter sets the maximum number of CCATEMP pages any individual DUMP can use.

If a thread is performing an update on a file that is being dumped, and the thread needs to update a page that has not been dumped yet, that page needs to be dumped before the page can be updated. Normally, *Fast/Backup* will dump such a page almost immediately, and there will be no noticeable delay for the updating transaction. However, if *Fast/Backup* is waiting for a tape volume switch, perhaps waiting for a tape mount, an updating transaction in this scenario would also have to wait for the tape mount. The result is not good for the updating request and potentially not good for other users in the Online, as resources held by the updating user are not released and, if sub-transaction checkpoints are being used, checkpoints are timed out.

In such a situation, *Fast/Backup* will try to copy the page to be updated to CCATEMP, allowing the updating thread to continue with its update. After the tape mount is satisfied, the page is copied from CCATEMP, and the CCATEMP used by the cached page is freed. Obviously, if there is a lot of updating for a file being dumped, and the dump ends up waiting on a tape mount, this could result in a **lot** of CCATEMP being used, leaving insufficient CCATEMP for other applications.

The DUMPTMAX parameter indicates the maximum number of CCATEMP pages that an individual DUMP will be allowed to use to cache pages that are about to be updated. If this number of pages is exceeded in the scenario described above, the updating thread will have to wait for the tape mount, or the dump will be terminated with an error, depending on the setting of the DUMPOPTS parameter (“[DUMPOPTS](#)” on page 50).

The default value for DUMPTMAX of 0 means that no limit is placed on the use of CCATEMP pages by *Fast/Backup*

See also “[DUMPTMIN](#)” on page 52 and “[DUMPOPTS](#)” on page 50.

5.13 DUMPTMIN

Default value	0
Parameter type	System
Where set	System manager resettable
Related products	<i>Fast/Backup</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter sets the minimum number of free CCATEMP pages that must be available for DUMP to use one for pre-imaging when waiting for second or subsequent tape volume mounts.

If a thread is performing an update on a file that is being dumped, and the thread needs to update a page that has not been dumped yet, that page needs to be dumped before the page can be updated. Normally, *Fast/Backup* will dump such a page almost immediately, and there will be no noticeable delay for the updating transaction. However, if *Fast/Backup* is waiting for a tape volume switch, perhaps waiting for a tape mount, an updating transaction in this scenario would also have to wait for the tape mount. The result is not good for the updating request, and potentially not good for other users in the Online, as resources held by the updating user are not released, and if sub-transaction checkpoints are being used, checkpoints are timed out.

In such a situation, *Fast/Backup* will try to copy the page to be updated to CCATEMP, allowing the updating thread to continue with its update. After the tape mount is satisfied, the page is copied from CCATEMP, and the CCATEMP used by the cached page is freed. Obviously, if there is a lot of updating for a file being dumped, and the dump ends up waiting on a tape mount, this could result in a **lot** of CCATEMP being used, leaving insufficient CCATEMP for other applications.

The DUMPTMIN parameter indicates the minimum number of CCATEMP pages that must be free, that is, available to other applications, before *Fast/Backup* will use one to cache a page that is about to be updated. If this number of pages is not available in the scenario described above, the updating thread will have to wait for the tape mount, or the dump will be terminated with an error, depending on the setting of the DUMPOPTS parameter (“DUMPOPTS” on page 50).

The default value for DUMPTMIN of 0 means that no limit is placed on the use of CCATEMP pages by *Fast/Backup*

See also “DUMPTMAX” on page 51 and “DUMPOPTS” on page 50.

5.14 ENTTAB

Default value '< lt & amp " quot'.

Parameter type System

Where set System manager resettable

Related products All

Introduced Before *Sirius Mods* 6.7

This parameter is a string that sets the default entity translation table for various character entity translation facilities available to User Language programs. These facilities include the \$ent function, translation performed as a result of the \$ent_print function, and entity translation performed as a result of the ampersand (&) special character at the start of an expression inside an Html/Text block.

5.15 FNVMASK

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Sirius Performance Enhancements V3</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This is a single-byte bitmask parameter. If set to a non-zero value, FNVMASK indicates the bytes in filenames that are not required to match those in the filenames on the physical file pages. Bytes in filenames corresponding to the ON bits in FNVMASK do not have to match the comparable bytes in the physical filenames.

For example, if FNVMASK is set to X'10', the fourth byte of a filename does not have to match the physical filename. That is, you might be able to open a file with physical name FOOBAR as FOODAR, or FOOLAR, or FOOTAR. If FNVMASK were set to X'34', bytes 3, 4, and 5 wouldn't need to match, so FOOBAR could be opened as FOOLER, or FOCKER, or FOLDER, to name jut a few.

FNVMASK facilitates sharing of data between Onlines, making it possible for two files with the same physical name to be opened in the same Online by opening them under two different names that only differ in the bytes indicated by FNVMASK. This can be especially useful in moving procedures from one file to a like-named file to be used in a different Online.

Note: FNVMASK in no way alters the file-enqueuing behavior of *Model 204* — no file, opened for update by one Online can be opened by another, regardless of the name under which it is opened.

Use of FNVMASK does not weaken page trailer validation, as page trailer validation at disk reads will validate the name against all bytes of the physical file name, regardless of FNVMASK.

5.16 FRELPREV

Parameter type System

Where set System manager resettable

Related products *Fast/Reload*

Introduced Before *Sirius Mods 6.7*

This parameter should be set to 0 or to a negative number whose absolute value is greater than the value of the NORQS system parameter.

Fast/Reload logs FLOD and FILELOAD programs to a temporary procedure, so if it decides to pass the FLOD or FILELOAD program to standard FLOD or FILELOAD, the statements it has scanned are not lost. Ordinarily, the temporary procedure used for this purpose is -1. If this is inconvenient, the temporary procedure number to be used can be changed by setting the FRELPREV system parameter. Setting FRELPREV to a positive number turns off the copying of FLOD/FILELOAD programs to a temporary procedure, which makes it impossible to have *Fast/Reload* automatically pass off to standard FLOD or FILELOAD.

Since this parameter only affects *Fast/Reload*, there is probably little reason to reset this parameter in an Online.

5.17 FUNCOPTS

Default value	X'00'
Parameter type	System
Where set	System manager resettable
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

This parameter can be used to customize the behavior of certain \$functions in a particular Online. This is a bitmask parameter where the bits have the following meanings:

- X'80'** Causes \$procopn to use the current file as its default file context if none is specified. The current file at a \$procopn is usually the last file referenced in the request, whether via a Find statement, or a For Each Record statement, or some other \$function. The behavior indicated by setting FUNCOPTS X'80' is inconsistent with all other file-related \$functions, which use their compile-time file/group context as their run-time file/group context. This setting is made available to allow backward-compatibility with pre-*Sirius Mods* 4.0. Any site running without this bit set, should not set it.
- X'40'** Allows any user to issue the \$PRIORITY function to change another user's priority. If neither this nor the X'20' bit is set, only a System Manager or System Administrator can use the \$PRIORITY function.
- This setting is available as of *Sirius Mods* version 7.3.
- X'20'** Allows a procedure invoked via the NEWSDESCMD facility to use the \$PRIORITY function to change a user's priority. If neither this nor the X'40' bit is set, only a System Manager or System Administrator can use the \$PRIORITY function.
- This setting is available as of *Sirius Mods* version 7.3.
- X'02'** Causes all \$list functions that encounter CCATEMP full conditions to act as if the LISTFC \$sirparm were set to 1, that is, to cancel the request with a CCATEMP full condition. The default behavior of most \$list functions is to return a -3 on a CCATEMP full condition. Since it is not likely that much, if any, User Language code is prepared to deal with a CCATEMP full condition, it is probably far better for a request to be cancelled in a CCATEMP full situation than to hand it an unexpected return code and carry on. This is especially true if the request that hits such an error is, itself, the cause of the CCATEMP full situation. This bit setting has no effect before *Sirius Mods* version 6.8.

Most system methods (as opposed to \$functions) that encounter a CCATEMP full situation already cause a request cancellation, so this parameter has no effect on system methods.

- X'01'** Allows \$bump to bump a user with the same userid as the issuing user, even if the user is not a system manager. If this bit is not set, only system managers are allowed to bump other threads.

5.18 FUNMAXT

Default value	0
Parameter type	System
Where set	System manager resettable
Related products	<i>Fast/Unload User Language Interface</i>
Introduced	<i>Sirius Mods 6.7</i>

This is a numeric parameter (with valid values from 0 to 36000) that indicates the maximum amount of time, in seconds, a *Fast/Unload User Language Interface* request is to be given to complete. The timer starts from the initiation of the request, either via \$Funload, or via the `FastUnload` and `FastUnloadTask` methods in the `RecordSet` class. The default value of FUNMAXT, 0, means that there will be no time limit placed on *Fast/Unload User Language Interface* requests.

The purpose of FUNMAXT is to prevent user requests being “hung up” indefinitely while queuing for busy *Fast/Unload* tasks or for unintentionally long-running requests.

FUNMAXT can be overridden for specific requests by using either of these:

- The `MaxTime` named parameter on the `FastUnload` and `FastUnloadTask` methods in the `RecordSet` class

```
* Make sure request completes in one minute
%rc = %recset:funload(%inList, %outList, -
                    %reportList, 'NEBUFF=10', -
                    maxtime=60)
```

- The new sixth parameter on \$Funload:

```
* Make sure request completes in one minute
%rc = $funload('LABEL', %inList, %outList, -
              %reportList, 'NEBUFF=10', 60)
```

A reasonable strategy would be to set FUNMAXT to a fairly low value, then selectively set it higher for requests that need more time. Of course, it can be very difficult to ensure that short-running requests complete quickly, if the Online also has long-running requests that might tie up all the *Fast/Unload* tasks. The odds are better if there are more *Fast/Unload* tasks (FUNTSKN bigger), but even with more tasks, these potential problems remain:

- If there are enough long running requests, all tasks might be tied up, anyway.
- Some of the *Fast/Unload* tasks might have trouble getting dispatched because there are more of them than CPUs to run them.

5.19 FUNPARM

Default value	X'00'
Parameter type	System
Where set	System manager resettable
Related products	<i>Fast/Unload User Language Interface</i>
Introduced	<i>Sirius Mods 6.7</i>

This is a bitmask parameter that controls the behavior of the *Fast/Unload User Language Interface*, where the bits mean:

- X'01'** Indicates that a synchronous *Fast/Unload* request is not to be allowed while an updating transaction is active. This is to prevent a *Fast/Unload* request that might take a long time to complete from being run while a user has resources enqueued for an updating transaction. These resources would, of course, include the blocking of checkpoints if subtransaction checkpoints are not being used.

If the X'01' bit is set, and a thread attempts a synchronous *Fast/Unload User Language Interface* request (either via \$Funload or the FastUnload method of the RecordSet class) in the middle of an updating transaction, the transaction will be cancelled with a message like the following:

```
CANCELLING REQUEST: MSIR.0561: $FUNLOAD: Synchronous request during
update transaction in line 43, procedure FUNTEST,
file ALEXPROC
```

5.20 FUNPGM

Default value	'FUNLOAD'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Fast/Unload User Language Interface</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter is a string of length eight or less that contains the name of the load module that is to be loaded at the start of the run, to run *Fast/Unload User Language Interface* requests. It defaults to the default *Fast/Unload* load module name of FUNLOAD. The indicated program must be on an accessed disk under CMS, or in the STEPLIB concatenation or system link list under MVS.

This parameter has no effect unless FUNTSKN ([“FUNTSKN” on page 61](#)) is set to a value greater than 0 and, like FUNTSKN, if a site is not authorized for *Fast/Unload User Language Interface*, this parameter will have no effect.

5.21 FUNTSKN

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Fast/Unload User Language Interface</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter must have a numeric value between 0 and 64, inclusive. If set to 0 (the default), the *Fast/Unload User Language Interface* is not able to initiate requests because there can be no *Fast/Unload* tasks started to run them. A number greater than 1 indicates the maximum number of *Fast/Unload* tasks that are to be attached to the Online so the maximum number of *Fast/Unload User Language Interface* requests that can run concurrently — if this limit is reached, additional requests are queued until a request completes.

The advantage of setting this value to a larger number is the potential for increased concurrency. The disadvantage is that if more requests want to run simultaneously than the number of available processors to run them, some of the requests could suffer severe performance degradation, to the point that they might complete significantly faster if they run serially rather than concurrently. In general, unless a site has many idle processors, it's probably unwise to set this parameter greater than 2.

If FUNTSKN is set to a non-zero value, the *Model 204* will try to load the program indicated by the FUNPGM parameter (“FUNPGM” on page 60) at Online initialization.

If a site is not authorized for *Fast/Unload User Language Interface*, this parameter will have no effect.

5.22 HGHREP31

Default value	0
Parameter type	System
Where set	Not resettable
Related products	<i>SirTune</i>
Introduced	<i>Sirius Mods 7.2</i>

This parameter displays the greatest amount of 31-bit virtual storage used in the run by report writers (currently, only the Dataset class SirtuneReport method).

See also:

- [“CURREP31” on page 46](#)
- [“CURREP64” on page 47](#)
- [“HGHREP64” on page 63](#)
- [“MAXREP31” on page 67](#)
- [“MAXREP64” on page 68](#)

5.23 HGHP64

Default value	0
Parameter type	System
Where set	Not resettable
Related products	<i>SirTune</i>
Introduced	<i>Sirius Mods 7.2</i>

This parameter displays the greatest amount of 64-bit virtual storage used in the run by report writers (currently, only the Dataset class SirtuneReport method).

See also:

- [“CURREP31” on page 46](#)
- [“CURREP64” on page 47](#)
- [“HGHP31” on page 62](#)
- [“MAXREP31” on page 67](#)
- [“MAXREP64” on page 68](#)

5.24 MAXBG

Default value	Number of daemon (SDAEMDEV) threads divided by two, or number of daemon threads minus 10, whichever is greater
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	<i>Sirius Mods 7.0</i>

This parameter specifies the maximum number of background or independent daemon requests that may be running at once. Before *Sirius Mods 7.0*, the only way to create a background daemon request was with the \$commbg function. Under *Sirius Mods 7.0* and later, independent daemon requests can also be created using the Daemon class RunIndependently method. Daemons running either as a result of RunIndependently or \$commbg are both counted the same way against the MAXBG limit.

If a \$commbg request would exceed the MAXBG limit (or there are no daemon threads immediately available to run the request) the request is enqueued and run later when daemon threads are available and the number of running background/independent requests drops below MAXBG. If a RunIndependently request would exceed the MAXBG limit, the request is cancelled.

The default value of MAXBG is the number of daemon (SDAEMDEV) threads divided by two or the number of threads minus 10, whichever is greater. For example, if there are 10 daemon threads defined in the Online, MAXBG defaults to 5. If there are 40 daemon threads, MAXBG defaults to 30. While there was no explicit MAXBG parameter before *Sirius Mods 7.0*, the default MAXBG was used “under the covers” by \$commbg, anyway.

Setting MAXBG to a value greater than or equal to the number of daemon threads allows background/independent requests to use up every available daemon thread, if they want. Setting MAXBG to 0 indicates that the default value of MAXBG, calculated from the number of daemon threads, is to be used. This means that the lowest explicit value of MAXBG that will be honored is 1.

5.25 MAXDAEM

Default value	1
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	<i>Sirius Mods 6.7</i>

This parameter indicates the maximum number of sdaemon threads that can be used by an individual thread. Such threads are allocated either as the result of a function that begins with \$comm or via the New method for Daemon objects.

Since sdaemon threads are a relatively scarce resource, it is not a good idea for a thread to use more than a handful of them at a time. Unfortunately, with Daemon objects, especially such objects embedded in another class, it is possible for a single runaway request to cause great problems by quickly consuming every daemon thread in an Online.

5.26 MAXRDS

Default value	1
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Janus Network Security</i>
Introduced	Before <i>Sirius Mods 6.7</i>

The MAXRDS system parameter is a numeric parameter that indicates the number of special-purpose public-key/private-key encryption/decryption subtasks to be used. Since public-key/private-key encryption/decryption is extremely CPU intensive, it can be useful to have the encryption/decryption performed in a special-purpose subtask, freeing the main *Model 204* task for other work.

Since CMS does not support true multi-tasking, this parameter has no effect under CMS. Also, since public-key/private-key encryption is only used by *Janus Network Security*, this parameter has no effect at sites not authorized for *Janus Network Security*.

Setting this parameter is especially useful at sites where:

- The Online is CPU constrained in the main task.
- Usually or often idle CPUs are available.
- The MP/204 feature is not available to move main task work to idle CPUs.

In an MP/204 environment, it is probably not worth setting MAXRDS. It would probably make more sense to add an additional MP/204 subtask to the Online by increasing the value of the AMPSUBS system parameter.

In a non-MP/204 environment, it probably doesn't make much sense (though is also largely harmless) to set an MAXRDS value greater than 1.

To actually use the tasks attached because of the MAXRDS parameter, the ACTRDS parameter ([“ACTRDS” on page 40](#)) must also be set.

5.27 MAXREP31

Default value 1024 if NUSERS=1; 0 otherwise

Parameter type System

Where set System manager resettable

Related products *SirTune*

Introduced *Sirius Mods 7.2*

This parameter indicates the maximum amount of 31-bit virtual storage to be used by report writers (currently, only the Dataset class SirtuneReport method). Report writers can use a lot of virtual storage which can adversely effect other users in an Online. This is why the default is 0 for any multi-user run. With the default, no report writer will be able to run in a multi-user Online.

Generally, report writers will be invoked in single-user runs, in which case the default of 1024 (megabytes) should be adequate for most purposes. If you find that you are running out of 31-bit storage, you can try to get the report writer to use 64-bit storage by setting MAXREP64 (“MAXREP64” on page 68). This will only work on *Model 204 V6R3* and later, because earlier releases of *Model 204* do not support 64-bit addressing. For more information on utilizing 64-bit storage, see the documentation for the specific report writer (for example, the *SirTune Reference Manual*).

The maximum allowed value for MAXREP31 is 2048, because that is the maximum amount of storage that may be addressed using 31-bit addressing. If MAXREP31 must be set to a value greater than 1024 to get reports to run, contact Sirius Software technical support, since this would suggest that the report writer is getting close to some architectural limits and might need to be restructured.

See also:

- “CURREP31” on page 46
- “CURREP64” on page 47
- “HGHREP31” on page 62
- “HGHREP64” on page 63
- “MAXREP64” on page 68

5.28 MAXREP64

Default value 0

Parameter type System

Where set System manager resettable

Related products *SirTune*

Introduced *Sirius Mods 7.2*

This parameter indicates the maximum amount of 64-bit virtual storage to be used by report writers (currently, only the Dataset class SirtuneReport method). Report writers can use a lot of virtual storage which can adversely effect other users in an Online. This is why the default is 0 for any multi-user run.

Setting MAXREP64 to a non-zero value will cause report writers to try to use 64-bit storage which could relieve pressure on 31-bit storage. This will only work on *Model 204 V6.3* and later, because earlier releases of *Model 204* do not support 64-bit addressing. For more information on utilizing 64-bit storage, see the documentation for the specific report writer (for example the *SirTune Reference Manual*).

For single-user runs, it probably makes sense to make MAXREP64 very large (like 100000), as no report writer is likely to use that much, anyway, and the system-wide MEMLIMIT controls should prevent a job from using more virtual storage than it should.

See also:

- [“CURREP31” on page 46](#)
- [“CURREP64” on page 47](#)
- [“HGHREP31” on page 62](#)
- [“HGHREP64” on page 63](#)
- [“MAXREP31” on page 67](#)

5.29 MODPROT

Default value	X'01'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This User 0 system parameter is a bitmask parameter that controls the load module protection facility. The load module protection facility protects the reentrant parts of the *Model 204* load module from update after initialization. Since most of *Model 204* is reentrant, this facility protects most of the *Model 204* load module from accidental corruption, thus increasing the Online's reliability.

The meaning of the bits in this parameter are:

- X'01'** Protect the reentrant parts of the *Model 204* load module. This bit should always be left on.
- X'02'** Protect FUNU. FUNU contains locally maintained \$functions so cannot be assumed to be reentrant. However, if FUNU is known to be reentrant, this bit can be set.
- X'04'** Protect PTCH. PTCH contains early warning code. While this code is generally reentrant, this is not guaranteed to be the case. Still, it is probably generally safe to set this bit.

The default value for MODPROT is X'01', which means that most of the *Model 204* load module is protected, but FUNU and PTCH are not.

Setting MODPROT will not affect *ZAP. A *ZAP of a protected part of the *Model 204* load module will temporarily turn off the module protect for the area being *ZAP'ed.

5.30 **MONPARM**

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>SirMon</i>
Introduced	Before <i>Sirius Mods 6.7</i>

Associated Sirius product: *SirMon* This is a bitmask parameter that affects certain performance and monitoring facilities. The meaning of the bits are:

X'01' Indicates that the *SirMon* slicing enhancement is to be **disabled**. By default, scheduler slicing at *SirMon* authorized sites is enhanced so that a user that exceeds its disk I/O or CPU limits is more quickly detected than it would be, ordinarily. Without this enhancement, runaway threads can run for a long time before they are determined to have exceeded their resource limits. There is no particularly good reason to turn this enhancement off.

X'02' Unused.

X'04' Disables the *Model 204* “free checkpoint” feature. With this feature, every user that commits an updating transaction after a checkpoint times out checks to see if it is now possible to perform a checkpoint. Such a checkpoint is called a free checkpoint.

The advantage of this feature is that, if a checkpoint timeout was caused by a single, log-updating transaction, a checkpoint will be taken as soon as that transaction is done. The disadvantage is that on a busy system, this can take a bad situation (a missed checkpoint) and make it worse by gobbling up CPU at the end of every updating transaction, trying to determine if a checkpoint is now possible.

X'08' Requests collection of UTI (User Think Interval) statistics. These statistics provide a picture of how long users are spending between screens.

This setting is especially useful for detecting the use of screen-scraping scripts running against an Online — extremely small think times are indicative of scripts.

The collected UTI stats begin with the letters UTI and can be found among the *SirMon* system stats.

Setting this bit also causes the DEV63 through DEV74 stats in the journal to be misused to indicate the number of screen responses in a time range:

DEV63 UTI <= 0.1 second

DEV64 0.1 < UTI <= 0.25

DEV65 0.25 < UTI <= 0.5

DEV66 0.5 < UTI <= 1

DEV67 1 < UTI <= 2

DEV68 2 < UTI <= 5

DEV69 5 < UTI <= 10

DEV70 10 < UTI <= 15

DEV71 15 < UTI <= 20

DEV72 20 < UTI <= 30

DEV73 30 < UTI <= 50

DEV74 50 < UTI

- X'08'** Requests collection of UTI (User Think Interval) statistics.
- X'20'** Prevents users from setting an account ID. This is mainly useful if the account field is being used to assist in 3270 screen-scraping script instrumentation (MONPARM X'80' bit set).
- X'40'** Makes the default account ID blank rather than equal to the userid. This is mainly useful if the account field is being used to assist in 3270 screen-scraping script instrumentation (MONPARM X'80' bit set).
- X'80'** Enables screen-scraping instrumentation support. Taking advantage of this fairly advanced facility requires modification of any 3270 screen-scraping scripts. If there is interest in instrumenting 3270 scripts, contact Sirius Software technical support.

5.31 NCMPBUF

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This numeric parameter indicates the number of compression buffers to be allocated at initialization. It must have a value between 0 and 32767.

Since datastream compression and decompression requires a compression buffer, this parameter limits the number of simultaneous compression or decompression operations. If not set, no buffers are allocated and no compression or decompression is allowed. If all compression buffers are in use, a compression or decompression request waits for the next available buffer.

Since decompression and (especially) compression are very CPU-intensive operations, there is probably little benefit to having many such operations running at once. Adding to this the fact that compression buffers are quite large (over 300K bytes), suggests that it is best to use a relatively small value for NCMPBUF (like 10 or less).

Compression and decompression are used by:

- The *Janus Web Server* requests for which compression is requested (via the COMPRESS setting on JANUS DEFINE or JANUS WEB, or via the \$WEB_SET function).
- The \$inflate, \$deflate, \$gzip, and \$gunzip functions.

For more information about the compression feature, see the \$deflate, \$inflate, \$gzip, and \$gunzip descriptions in the ***Sirius Functions Reference Manual***, and see the description of the COMPRESS parameter of the JANUS DEFINE command in the ***Janus TCP/IP Base Reference Manual***.

5.32 PERFOPT

Default value X'00000000'

Parameter type System

Where set User 0 CCAIN parameters

Related products *Sirius Performance Enhancements, Sirius Performance Enhancements V2, and Sirius Performance Enhancements V3*

Introduced Before *Sirius Mods 6.7*

This is a bitmask parameter with 32 available bits. These bits are used to enable certain performance enhancements for *Sirius Performance Enhancements* customers. Many of the enhancements have been incorporated into the core *Model 204* product, so the bits associated with these enhancements no longer have any effect (so can either be set or not).

The bits that still have an effect are:

X'00000800' Optimizes LOGONENQ processing by maintaining a virtual storage hash table of logged on usersids.

5.33 **PERFOPT2**

Default value X'00000000'

Parameter type System

Where set User 0 CCAIN parameters

Related products *Sirius Performance Enhancements, Sirius Performance Enhancements V2, and Sirius Performance Enhancements V3*

Introduced Before *Sirius Mods 6.7*

This is a bitmask parameter with 32 available bits. These bits are used to enable certain performance enhancements for *Sirius Performance Enhancements* customers. All of the enhancements have been incorporated into the core *Model 204* product or have been made available to all *Sirius Mods* customers, so the bits in PERFOPT2 currently do nothing.

5.34 PUPDTH

Default value	4091
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>SirFact</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This numeric parameter can have any value between 1 and 999999. It indicates the number of hash slots to allocate to detect procedure updates.

The hash table is only used by *SirFact* to determine if a procedure needs to be recompiled because it or some included procedure has been changed. The procedure update hash table makes it possible to quickly determine that a procedure has not changed without actually scanning the procedure dictionary (an expensive operation).

Each hash cell requires 8 bytes of virtual storage. The default value of 4091 is probably fine for most sites.

5.35 RESTARTU

Default value	X'00'
Parameter type	System
Where set	EXEC card PARM value under MVS, M204CMS command parameters under CMS
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

This parameter enables the restart utility, which makes it possible to communicate with a looping (or hung) Online with the MODIFY command under MVS or with the SMSG facility under CMS. Using MODIFY or SMSG commands, one can determine which user is running in the Online and BUMP a user, or even force a user restart.

The restart utility is enabled by setting the X'01' bit of the RESTARTU parameter. For more information about the restart utility, see [“Operator Commands” on page 3](#).

The *SirTune* data collector, in addition to collecting performance data for an Online, provides functionality identical to that of the restart facility. Under *Sirius Mods* 6.9 and later, the *SirTune* data collector is integrated into the *Sirius Mods*. As such, in these releases, if the *SirTune* data collector is active, the RESTARTU parameter is ignored.

5.36 RETRVBUF

Default value	288
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	<i>Sirius Mods 7.3</i>

This parameter specifies the size of the buffer for the 3270 terminal PF key that retrieves previously input command lines. The larger the retrieve buffer, the more commands you can save for retrieval.

The minimum and default setting of RETRVBUF is 288 bytes; its maximum is 32767. The parameter setting is always rounded down to the nearest 8-byte multiple, so the largest buffer allowed is actually 32760 bytes.

Note: The retrieve buffer is a virtual storage area allocated for any full screen thread with a retrieve key set, so the cost of a large buffer is higher virtual storage usage.

5.37 SCANPARM

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>SirScan</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter is a bitmask parameter that controls certain *SirScan* behavior.

The SCANPARM bits are:

- X'01'** Allow non-system-manager users to browse journal entries for public requests on *Janus Web Server* threads. Public requests are those that are not protected from access by userid.
- X'02'** Issue a redundant “MSIR.0361 Processing request *method URL*” message after login. Normally, MSIR.0361 is issued on web threads before login, since the method and URL are known before the userid and password, if a login is even required for the request.

Unless the X'02' SCANPARM bit is set, users browsing journal entries for *Janus Web Server* threads based on userid do not see the MSIR.0361 message (since it occurs before the login), so they have a difficult time ascertaining the method and URL being processed in a particular request.

If a site is not authorized for *SirScan*, this parameter will have no effect.

5.38 SCANTIME

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>SirScan</i>
Introduced	Before <i>Sirius Mods</i> 6.7

This parameter indicates the maximum amount of time between *SirScan* “heartbeat” messages, and must have a value between 0 and 3600, inclusive. These heartbeat messages are RK type audit messages, and they contain identifying information about the thread such as userid, terminal ID, and IP address and port number for Janus server threads.

SCANTIME indicates the maximum number of seconds between journal messages for a thread before a heartbeat message is issued. If a message is about to be sent to the journal for a thread, and it has been more than SCANTIME seconds since the last heartbeat message, then the heartbeat message is first sent to the journal. This means that if a journal message is found for a thread, the userid and other identifying information can **always** be found by looking backwards in the journal no more than SCANTIME seconds.

The default value for SCANTIME of 0 means that heartbeat messages are not logged. If SCANTIME is set to a positive value up to its maximum value of 3600, the *SirScan* heartbeat messages will be logged to the journal.

If SCANTIME is set to a non-zero value, the SIRSCAN subsystem detects this and, on the scan specification screen, gives users the option of reading an extra SCANTIME seconds on every interval collected from the journal. The extra SCANTIME seconds read are before the start of the requested interval. While the data from the extra SCANTIME seconds worth of journal entries is not formatted, it is scanned for the heartbeat messages, and any information in those messages is saved for each thread. In this way, all messages for all threads after the start of the requested interval can be deterministically associated with a userid, terminal ID, and IP address and Janus port for Janus server requests.

The cost of getting this deterministic behavior is the overhead of scanning an extra SCANTIME seconds worth of journal. If scanning is being done in auto-refresh mode, each refresh (attempt to move past the current bottom) will cause an extra SCANTIME seconds to be scanned. This means that the setting of SCANTIME involves a trade-off between the number of heartbeat messages logged to the journal and the overhead of scanning an extra SCANTIME seconds of journal on every *SirScan* browse request. The bigger the value of SCANTIME, the fewer heartbeat messages will be logged, but the more journal records would need to be scanned on each *SirScan* browse request. A small value of SCANTIME will, of course, have the opposite effect.

Since journal messages tend to be bunched (when a thread gets one audit message logged it is likely to get several around the same time), there is probably not a significantly higher cost to a relatively small SCANTIME, like 10 or even 5, than to a much larger one, like 60. On the other hand, the amount of extra work performed by *SirScan* associated with the extra SCANTIME second scan for each browse request will be roughly proportional to the size of SCANTIME. All this would suggest using a fairly small SCANTIME: a value of 10 might be a good starting point.

If you are setting SCANTIME for the first time in an Online, it is prudent to anticipate a 3% increase in journal usage, though the actual increase will almost undoubtedly be much less than that (probably less than even 1%).

Note: The heartbeat message can even be useful outside of *SirScan*. Since the heartbeat messages are regular RK type messages, they can be viewed with AUDIT204 as well as *SirScan*.

This means that if one is using AUDIT204 for a previous run's journal, and one wishes to find entries for a particular userid in a particular time interval, one can simply request from AUDIT204 all records for all users in the desired interval plus SCANTIME seconds before. By doing so, one can be certain that any messages formatted by AUDIT204 will be preceded by a message identifying the user associated with that message.

If a site is not authorized for *SirScan*, this parameter will have no effect.

5.39 SDAEMDEV

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

This parameter specifies the IODEV number to be used for SDAEMON (pronounced ess-demon) threads. These are special background threads (analogous to IODEV=3 threads) that operate without terminals and perform a variety of tasks, from running requests on behalf of users via \$comm functions (see the ***Sirius Functions Reference Manual***) or Daemon class requests (see the ***Janus SOAP Reference Manual***), to *Janus Web Server* and other Janus server requests, to running as *Janus Debugger* or *SirFact* worker threads. The value of the SDAEMDEV parameter must be an odd number between 1 and 53, and must be different from the TNDEV system parameter (“TNDEV” on page 111).

Note: Any IODEV number selected for SDAEMDEV will be unavailable for its normal function. The recommended setting for SDAEMDEV is 15 *unless* a site is using BTAM TTY terminals, which is exceedingly unlikely these days.

Once an SDAEMDEV value is defined, IODEV cards must be added for the threads that will run as SDAEMONS, just as IODEV cards are added for any other thread type supported by *Model 204*. For more information about IODEV cards, see the ***Model 204 Command Reference Manual***.

The correct number of IODEV cards depends on too many factors to provide any simple rules, but most sites should have at least 10, and then as many more as the maximum expected number of simultaneous Janus connections.

For more information about setting up SDAEMON threads, see the ***Sirius Mods Installation Guide***

5.40 SDEBGUIP

Default value 0

Parameter type System

Where set User 0 CCAIN parameters

Related products *Sirius Debugger, Janus Debugger*

Introduced *Sirius Mods 7.0*

Referenced by the SIRIUS DEBUG ON command which starts a debugging session, a non-zero value of this parameter provides a default for the number of the workstation port to which the Debugger Client listens.

If SDEBGUIP is 0, there is no default workstation port value, and a SIRIUS DEBUG ON command must explicitly specify a workstation port number for connections between the Online and the Debugger Client GUI.

The maximum valid value is 65535.

5.41 SDEBWRKP

Default value 0

Parameter type System

Where set User 0 CCAIN parameters

Related products *Sirius Debugger, Janus Debugger*

Introduced *Sirius Mods 7.0*

Referenced by the SIRIUS DEBUG ON command which starts a debugging session, a non-zero value of this parameter provides a default for the number of the Janus port to which the Online listens that is used for spawning Debugger worker threads.

If SDEBWRKP is 0, there is no default port value, and a SIRIUS DEBUG ON command must explicitly specify a port number for spawning *Model 204* Debugger worker threads.

The maximum valid value is 65535.

5.42 **SDMOPT**

Default value	X'03'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	<i>Sirius Mods 6.7</i>

This User 0 system parameter is a bitmask parameter that affects sdaemon processing. The meaning of the bits are:

- X'01'** Server tables are not to be cleared at user login, logout, or UTABLE commands. While clearing of user tables is “nice” in some sense, since daemons tend to do a lot of logging in and out and a lot of UTABLE commands, this niceness can have a high CPU cost. Other than being “nice” there is little benefit to clearing all the server tables at these points.
- X'02'** Do optimized file and group opens and closes when entering or exiting a subsystem. Since daemons that use APSY subsystems have a high ratio of subsystem exit/entry to subsystem processing, reducing this overhead can be a great CPU savings.

The default for SDMOPT of X'03' means that all the sdaemon optimizations are enabled. There should be no reason to disable any optimizations unless one is suspected of causing a bug.

5.43 SESDEFOW

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

The parameter which must have a numeric value between 0 and 43200, sets the default open wait time value for a session if one is not specified in the \$session_open call. The default value for SESDEFOW of 0 indicates that a \$session_open call will not wait if the request session is in-use, that is open by another thread. If a session is still in-use after the open wait time, a return code of 2 is set by \$session_open to indicate the problem. The SESDEFOW setting can be overridden in the \$session_open call.

5.44 **SESDEFTO**

Default value	900
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

The parameter which must have a numeric value between 0 and 43200, sets the default timeout value for a session, if one is not specified in the \$session_create call. The default value for SESDEFTO of 900 indicates a default timeout of 15 minutes. If a session is not re-opened within the timeout value of the \$session_create, it is considered *timed-out* and liable to deletion under control of the SESOPT flags (“[SESOPT](#)” on page 89). The SESDEFTO setting can be overridden in many of the \$session functions.

5.45 SESNPRV

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

The parameter which must have a numeric value between 0 and 1048575, sets the maximum number of private sessions that can be active. Setting this value allocates about 160 bytes of virtual storage for each possible session. The default value for SESNPRV of 0 means that no private sessions can be created.

Private sessions are created with the `$session_create` function. For more information on this and related functions see the ***Sirius Functions Reference Manual***.

5.46 **SESNPUB**

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

The parameter which must have a numeric value between 0 and 1048575, sets the maximum number of public sessions that can be active. Setting this value allocates about 160 bytes of virtual storage for each possible session. The default value for SESNPUB of 0 means that no public sessions can be created.

Public sessions are created with the `$session_create` function. For more information on this and related functions see the ***Sirius Functions Reference Manual***.

5.47 SESOPT

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter controlled certain aspects of session processing, but is no longer used. It is only present for backward compatibility.

5.48 **SESUMAX**

Default value	1
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

The parameter which must have a numeric value between 1 and 1048575, sets the maximum number of private sessions that can be active for a userid. Setting this value allocates about 160 bytes of virtual storage for each possible session. The default value for SESUMAX of 1 means that a userid can only have a single session associated with it at a time.

Sessions are created with the `$session_create` function. For more information on this and related functions see the ***Sirius Functions Reference Manual***.

5.49 SIRAPSYF

Default value	X'00'
Parameter type	System
Where set	System manager resettable
Related products	<i>SirFact</i>
Introduced	Before <i>Sirius Mods</i> 6.7

This parameter controls a number of APSY subsystem maintenance features. The bits defined for the features are described below.

Note: No SIRAPSYF features are enabled unless you also set the X'80' bit on the SIRFACT parameter (see [“SIRFACT” on page 94](#)).

X'01' Allows procedure compilations to be saved (pre-compiled) for unlocked procedure group members. If an outer or an inner procedure in an unlocked file in a procedure group is changed, or if an outer procedure is added to an unlocked file in a procedure group, the procedure is recompiled and that compilation is saved.

Also allows the pre-compiling of a procedure with a pre-compile prefix that was not present in the procedure group when the subsystem was started.

In version 7.1 or higher of the *Sirius Mods*, if a pre-compiled procedure has multiple Begin/End brackets, the first bracket will be saved as the pre-compilation and subsequent brackets will be re-compiled every time the procedure is run.

If this bit is not set, using unlocked files to facilitate the updating of procedures in a running subsystem has an efficiency cost because procedure compilations are not saved.

This setting has no effect on procedures in subsystems that use a procedure file instead of a procedure group, and it has no effect on subsystems that use a procedure group but not unlocked files.

X'02' Detects changes to included procedures that reside in a pre-compiled procedure in a subsystem procedure group. If such an included procedure is changed, the pre-compiled procedure is recompiled.

This setting has no effect on procedures in subsystems that use a procedure file instead of a procedure group, and it has no effect on subsystems that use a procedure group but not unlocked files.

- X'04'** Tracks in a bitmap the CCATEMP pages allocated to pre-compiled procedures in a subsystem. When the subsystem is stopped, this bitmap is used to free the pages rather than chaining through them, which requires considerable CCATEMP I/O. Although the bitmap method has more (but probably not measurable) overhead while saving compilations, it can make the STOP SUBSYSTEM process significantly faster.

The bitmap is subsystem-wide and not procedure-specific. It does not reduce the time required for discarding the CCATEMP pages that are associated with a compilation that is being replaced.

Usage notes:

- These SIRAPSYF features along with the SIRFACT X'40' and X'80' bits are designed to simplify the updating of procedures and the pre-compiling of these updated procedures while their subsystem is in use.

Note: These settings do **not** eliminate the lock on outer procedures in locked procedure files. They are designed to suit a procedure group and the placement of updated procedures in unlocked file(s).

The SIRFACT QUIESCE and RESUME subcommands supplement these features by preventing subsystem users from interfering with procedure update operations. For more information, see the *SirFact Reference Manual*.

- For both the X'01' and X'02' bits, an inner or outer procedure is considered changed if the actual procedure is modified or if a new version of the procedure is added to an earlier file in the procedure group.
- When using temporary procedure groups, a request compilation is not saved if any of the outer or inner procedures came from a file not in the subsystem's permanent group. Furthermore, if the outer procedure is found in a file not in the subsystem's permanent group, it will always be recompiled. If an inner procedure (but not the outer) is found in a file not in the subsystem's permanent group, whether the procedure is recompiled depends on the X'02' bit setting:
 - If the bit is off, the procedure might or might not be recompiled.
 - If the bit is on, the procedure is always recompiled.

Hence, it is recommended that where temporary procedure groups are to be used, the X'02' bit is to be set.

- The SIRAPSYF features you specify apply on a system-wide basis. To specify an override for an individual subsystem, you can specify in CCASYS a special deferred update DD name for a procedure group that defines its particular SIRAPSYF option:

SUBSYSTEMGMT		Subsystem File Use Update Mode						
Subsystem Name: SALES		From: -						
File/Group Name	File Location	Group Y/N	Auto Y/N	Mandatory Y/N	Procs NUMLK	Deferred Name	Ordered-index Deferred Name	
PR SALESPRC		N	Y	Y				
1: PAYROLL		N	Y	Y				
2: EMPLOYEE		N	Y	Y				
3: CLIENTS	DALLAS	N	Y	Y				
4: PRODUCTS	DALLAS	N	Y	Y				
5: SALESGRP		Y	Y	Y		TAPE*07		
===>								
1=HELp	2=	3=QUIt	4=OPEration	5=PR0cedure	6=			
7=BACKward	8=F0Rward	9=USErdef	10=	11=SYSclass	12=END			

SIRAPSYF override

As shown for SALESGRP above, you must:

- In SUBSYSTEMGMT, specify the special deferred update DD name under “Deferred Name” on the Subsystem File Use screen. Or, if you use an ad hoc procedure, specify this special name for the APSFDN field in the SCLS records for the subsystem.
- Begin the deferred update DD name with the characters “TAPE*” and append the two hexadecimal digits that indicate the subsystem-specific bit settings you want for the *SirFact* APSY enhancements. “TAPE*07” would set the SIRAPSYF X'07' features for the group.

Setting a deferred index update file name is completely harmless in systems that do not have the *SirFact* APSY facility or do not have the facility enabled (by specifying the SIRFACT X'80' bit).

5.50 SIRFACT

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>SirFact</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter consists of several bits that can control the collection of compilation data and the trapping of certain User Language coding errors. The bits defined for the *SirFact* parameter are:

- X'01'** Collect quad offset to procedure line number mapping information to CCATEMP. This makes it possible for request cancelling errors to indicate the procedure and line number where the error occurred. It provides identical functionality as the ULDEBUG parameter, but without the QTBL space overhead.
- X'02'** Collect quad offset to procedure line number mapping information to server tables even if it is also being collected to CCATEMP. There's probably no real good reason to set this bit.
- X'04'** Don't do comment initialized global dummy string substitution. See the ***SirFact Reference Manual*** for more information on comment initialized globals.
- X'08'** Cancel FOR RECORD NUMBER or FRN statements where the record number is a null string.
- X'10'** Cancel FOR RECORD NUMBER or FRN statements where the record number is not a valid number.
- X'20'** Cancel FOR RECORD NUMBER or FRN statements where the record number is not found.
- X'40'** When a procedure is included (whether as part of an APSY subsystem or directly from *Model 204* command mode), copy it to CCATEMP. After the copy, release the share enqueue on the procedure. As a result, a user who includes a procedure does not prevent others from updating the procedure. See the ***SirFact Reference Manual*** for more information on subsystem procedure enqueues.

The copy of the procedure to CCATEMP incurs some extra (barely measurable) overhead.

X'80' Enable the *SirFact* APSY maintenance enhancements you specify with the SIRAPSYF parameter (see “[SIRAPSYF](#)” on page 91).

Also, regardless of the SIRAPSYF settings, release the share enqueue on a procedure when the last line of the procedure is read, not when the line after the last line is attempted to be read. See the ***SirFact Reference Manual*** for more information on subsystem procedure enqueues.

5.51 SIRFUNC

Default value	X'00'
Parameter type	System
Where set	EXEC card PARM value under MVS, M204CMS command parameters under CMS
Related products	All
Introduced	<i>Sirius Mods 6.9</i>

This is a bitmask parameter that controls \$function settings where the meanings of the bits are:

X'01' Ignore the linked in FUND or FUNDLE module. This makes it possible to access the Sirius version of many of the functions in FUND, without relinking the *Model 204* load module. The Sirius \$function alternatives available in \$SIRMODS. 6.9 and later are:

- \$abs
- \$arccos
- \$arcsin
- \$arctan
- \$arctan2
- \$cos
- \$cosh
- \$cotan
- \$erf
- \$erfc
- \$ixpi
- \$log
- \$log10
- \$exp
- \$gamma
- \$lgamma
- \$max
- \$min
- \$pi
- \$rxpi
- \$rxpr
- \$sin
- \$sinh
- \$sqrt
- \$tan
- \$tanh

5.52 SIRMSG

Default value	0
Parameter type	System
Where set	System manager resettable
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This numeric parameter must have a value of 0, 1, or 2 indicates the case (mixed vs.upper) of Sirius messages (messages that begin with MSIR.). The values for SIRMSG are:

- 0** Means that messages are left as mixed case unless DBCSENV is set to a non-zero value, in which case the are translated to upper case. This value means that mixed case messages are translated to upper case in any Online using DBCS characters where lower case characters might end up being displayed as katakana. Unless a site is using DBCS, this default setting should be fine.
- 1** Forces translation of Sirius messages to upper case
- 2** Always leaves Sirius messages in mixed case, even if DBCSENV is set to a non-zero value. This is useful at DBCS sites where it is known that lower case characters will be correctly displayed.

5.53 SIRPRMPT

Default value	X'00'
Parameter type	System
Where set	System manager resettable
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

This parameter is a bitmask parameter that affects the command mode prompt in an Online. The meaning of the bits in SIRPRMPT are:

- X'01'** Add the jobname (or the VM's userid under CMS) to the command mode prompt. This makes it easy to determine which Online one is connected to in command mode, possibly averting accidents like someone typing E0J in a production Online when she means to be terminating a test Online
- X'02'** Add the name of the job step to the command mode prompt. Note that if the X'01' bit is on, the X'02' bit is ignored. The X'02' bit should not be used under CMS. This setting was made available in version 7.1 of the *Sirius Mods*.

5.54 SIRTERM

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	<i>Sirius Mods 6.7</i>

This is a bitmask parameter with the following meanings for the bits:

- X'01'** Enable MODEL 6 support (“[Terminal MODEL 6 support](#)” on page 137). Model 6 support allows terminals with essentially arbitrary numbers of rows and columns, making it easier to better utilize a workstation's screen size.
- X'02'** Always issue a Write Structured Field Query for terminals connecting to *Model 204* through VTAM. This allows *Model 204* to dynamically determine the screen geometry of any terminal connecting to it through VTAM, without having to issue a RESET MODEL 6 command.

The downside of this setting is that it could add a small amount of time to the initial connection process and a slight amount of extra network traffic.

If a terminal is using a non-standard screen geometry via model 6 support, the *Model 204* editor and command line will correctly use the available screen space. Many *UL/SPF* subsystems such as SirScan, SirMon, and SirPro will also take advantage of the additional available screen space.

5.55 SIRPDLHW

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter can be either 0 or 1, and if 1 enables more accurate calculation of PDL usage high water marks. The PDL high water marks reported by *Model 204* are typically a little lower than the true high water marks. There ***might*** be measurable (but not large) CPU overhead associated with this feature, so it ***might*** not be worth enabling this in extremely CPU-constrained environments. It is almost certainly worth setting in a development environment, where it is more important to get accurate table usage statistics than to save a tiny bit of CPU.

5.56 SIRTUNE

Default value	X'01'
Parameter type	System
Where set	EXEC card PARM value under MVS, M204CMS command parameters under CMS
Related products	<i>SirTune</i>
Introduced	<i>Sirius Mods 6.9</i>

This is a bitmask parameter that controls *SirTune*. *SirTune* is integrated with the *Sirius Mods* as of *Sirius Mods* version 6.9. The meaning of the bits are:

X'01' Enables the *SirTune* data collector. When this is set, the *SirTune* data collector will look for the SIRTUNED dataset. If present, the *SirTune* data collector will be enabled, and compilation and sample data will be collected to SIRTUNED. In addition, the MODIFY/MSG commands will also be available ([“Operator Commands” on page 3](#)).

The default value of X'01' for SIRTUNE causes the *SirTune* data collector to be enabled **if possible**. Things that prevent it from being enabled are the absence of the SIRTUNED dataset and the fact that *Model 204* is running under an older *SirTune* load module. In the first case, the *SirTune* data collector silently disables itself. In the second case, it issues an informational message and then disables itself. In no case should the default value for SIRTUNE prevent an Online from coming up, so there is probably little reason to set this parameter.

Setting this parameter in an Online not authorized for *SirTune* will have no effect.

For more information about the *SirTune* data collector, see the ***SirTune Reference Manual***.

5.57 SOCKMAX

Default value	10
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Janus Sockets</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter specifies the maximum total number of sockets a user may use at any time (one of these is reserved for the server socket number 1, whether the user is a server socket program or not). The default for SOCKMAX is 10, which allows 9 client socket numbers to be in use.

The minimum value for SOCKMAX is 2, and the maximum value is 88.

SOCKMAX affects storage allocation for each user who is using any sockets — each such user has allocated $6 + (6 * \text{SOCKMAX})$ bytes (rounded up to a multiple of 4). In addition, each open socket over a *Janus Sockets* port increases the storage requirement by 960 bytes plus the value of the IBSIZE and OBSIZE parameters.

Note the distinction between this *Model 204* system parameter and SOCKPMAX.

5.58 SPANSIZE

Default value	90
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This is a numeric parameter that must have a value between 10 and 100. It sets the default value for the range in number of years that two-digit years are allowed to encompass. By limiting the valid range of two digit years, protection is provided against misinterpretation of ambiguous two-digit dates such as, for example, '10' being misinterpreted as 1910, when in fact, it means 2010.

This setting affects many \$functions, and it can be overridden by many of these same \$functions. For more information about how SPANSIZE is used, see the ***Sir2000 User Language Tools Reference Manual*** or the ***Sir2000 Field Migration Facility Reference Manual***.

5.59 SRSDEFTO

Default value 900

Parameter type System

Where set System manager resettable

Related products *Janus Web Server*

Introduced Before *Sirius Mods 6.7*

This parameter sets the default timeout value (in seconds) to be used for saved record sets if none is specified on the `$web_save_reset` function. If SRSDEFTO exceeds SRSMAXTO, SRSMAXTO ([“SRSMAXTO” on page 106](#)) will effectively act as the default timeout value.

The default for SRSDEFTO is 900, which means 15 minutes. Setting SRSDEFTO to 0 means no record sets will be saved unless an explicit timeout is specified in the `$web_save_reset` function.

The SRSDEFTO parameter has no effect unless SRSMAX ([“SRSMAX” on page 105](#)) is set to a non-zero value.

Valid values for SRSDEFTO are between 0 and 2592000 (720 hours).

5.60 SRSMAX

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Janus Web Server</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter sets the maximum number of total saved record sets in the system. SRSMAX defaults to 0, and it must be set to a positive value less than or equal to 16777215 to allow use of the saved record set feature of *Janus Web Server*. If SRSMAX is 0, \$web_save_recset will never save a record set.

The number of record sets that can actually be saved at a given time might be somewhat less than the SRSMAX value if many of the saved record sets are associated with groups with large numbers of members.

The saved record set feature will use approximately 64*SRSMAX bytes of virtual storage.

5.61 SRSMAXTO

Default value	3600
Parameter type	System
Where set	System manager resettable
Related products	<i>Janus Web Server</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter sets the maximum length of time (in seconds) that a saved record set can be saved without being referenced. The actual maximum length of time a record set will be saved can be set in the `$web_save_recset` function, but it will be set to SRSMAXTO if the `$web_save_recset` value exceeds SRSMAXTO. Similarly, if a timeout is not specified on the `$web_save_recset`, the timeout value is derived from SRSDEFTO (“[SRSDEFTO](#)” on page 104), but if SRSDEFTO is greater than SRSMAXTO, SRSMAXTO will be used as the timeout.

The default for SRSMAXTO of 3600 means a maximum timeout of one hour.

Resetting SRSMAXTO to 0 temporarily disables the saved record set facility, and all enqueues associated with saved record sets are freed. This can be useful in clearing up an enqueueing problem caused by a saved record set.

Resetting SRSMAXTO will dynamically adjust the timeout for any record sets saved with a higher timeout value. So if there is a saved record set with a timeout of 1800 seconds when SRSMAXTO is reset to 300, that record set's timeout is updated to 300. If that record set had been saved more than 300 seconds before the reset, it would be immediately freed.

The SRSMAXTO parameter has no effect unless SRSMAX (“[SRSMAX](#)” on page 105) is set to a non-zero value.

Valid values for SRSMAXTO are between 0 and 2592000 (720 hours).

5.62 SRSMAXUS

Default value	0
Parameter type	System
Where set	System manager resettable
Related products	<i>Janus Web Server</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter sets the maximum number of saved record sets per user. This limit is a per-userid limit, and it applies even if the userid is being used from multiple web browsers. The default for SRSMAXUS of 0 means that there is no per-user limit for saved record sets.

SRSMAXUS can be used to ensure that a single user, either as the result of a programming error or simply through running frequent queries, does not monopolize the saved record set facility.

The SRSMAXUS parameter has no effect unless SRSMAX ([“SRSMAX” on page 105](#)) is set to a non-zero value.

5.63 TCPSERV

Default value	'TCPIP'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter indicates the name of the TCP/IP server address space (MVS) or virtual machine (CMS).

5.64 TCPSUBSY

Default value	' '	(four blanks)
Parameter type	System	
Where set	User 0 CCAIN parameters	
Related products	All	
Introduced	Before <i>Sirius Mods 6.7</i>	

This parameter indicates the subsystem name to be used by Janus. It only has meaning when using the MVS IUCV interface (a deprecated API), and the Interlink interface. The default of blank causes the default value of 'VMCF' to be used in the unlikely event that MVS IUCV is being used and 'ACSS' for Interlink.

5.65 TCPTYPE

Default value	'UNKNOWN'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter specifies the type of TCP/IP network to which *Model 204* is connected. Generally, you can leave this parameter at its default value and let Janus automatically detect the type of TCP/IP you have on your machine.

If you specify type **IBM**, Janus detects and sets the type of IBM interface it will use:

- Under CMS, this is always IUCV.
- Under MVS, Janus will select the IBM interface in the following order: BPX (Unix System Services), HPNS, then IUCV.
- If multiple types are installed, Janus will select the interface in the following order for MVS: BPX, HPNS, IUCV, Interlink, then KNET. For CMS the order is IUCV, then KNET.

You can always override auto-detection by explicitly supplying a TCPTYPE parameter. The only reason to do this would be if you have multiple TCP/IP stacks installed and you want to force use of the non-preferred stack.

The valid TCPTYPE values are:

IBM	Specifies IBM TCP/IP. This means to auto-detect the “best” IBM TCP/IP interface. For CMS, this is always IUCV. For MVS, Janus will look for BPX, HPNS, then IUCV.
BPX	Specifies IBM BPX (Unix System Services) (MVS only).
HPNS	Specifies IBM HPNS (High Performance Native Sockets) (MVS only). Note that despite its name, HPNS's performance is not nearly as good as BPX's.
INTERLNK	Specifies InterLink TCP/IP (MVS only).
CSI	Specifies Connectivity Systems TCP/IP (VSE only).

Most of the communication with the TCP/IP address space is accomplished via a PST. Because of this, you may need to increase NSUBTKS by 1 before using Janus.

5.66 TNDEV

Default value	0
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Janus Sockets</i>
Introduced	<i>Sirius Mods 6.9</i>

This parameter specifies the IODEV number to be used for Janus Telnet Server connections. When a connection is made to a Janus Telnet Server, the IODEV number of the SDAEMON thread is dynamically switched to the TNDEV value. If TNDEV is not set to a non-zero value, no Janus Telnet Server (TNSERV) ports can be defined, and no telnet connections can be made directly to *Model 204*.

The value of the TNDEV parameter must be an odd number between 1 and 53, and it must be different from the SDAEMDEV system parameter (“[SDAEMDEV](#)” on page 81).

Note: No IODEV cards are allowed for the TNDEV IODEV number — the IODEV number can only be used dynamically by Janus Telnet Server.

The recommended setting for TNDEV is 21, an IODEV number no longer in use by *Model 204*.

For more information about the Janus Telnet Server, see the ***Janus Sockets Reference Manual***.

5.67 WEBAUDIT

Default value	X'00000000'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	All
Introduced	<i>Sirius Mods 6.9</i>

This parameter makes it possible to reduce the quantity of useless audit trail data associated with *Janus Web Server* applications. The WEBAUDIT parameter is a collection of bits that, when set, reduce the quantity of data sent to the *Model 204* journal or make it possible to reduce this quantity. The bits in WEBAUDIT are:

- X'00000001'** Allow NOAUDIT and NOAUDITW keywords on MSGCTL command. Specifying `MSGCTL msg_no NOAUDIT` prevents `msg_no` from ever going to the audit trail, while `MSGCTL msg_no NOAUDITW` prevents `msg_no` from going to the audit trail, if issued on a *Janus Web Server* thread.
- X'00000002'** Suppress APSY load since-last statistics for *Janus Web Server* threads. The resource usage associated with the APSY load such as CPU, DKPRs, disk I/Os, etc. will be included in the EVAL since-last statistics for the request. Since APSY load resource utilization is typically extremely minor relative to request evaluation, APSY load since-last statistics tend to be fairly useless.
- X'00000004'** Suppress APSY load since-last statistics for all threads. The resource usage associated with the APSY load such as CPU, DKPRs, disk I/Os, etc. will be included in the EVAL since-last statistics for the request. Since APSY load resource utilization is typically extremely minor relative to request evaluation, APSY load since-last statistics tend to be fairly useless.
- X'00000008'** Suppress “NO USERID” logout messages (M204.0353) for *Janus Web Server* threads. These messages are sometimes issued before a user logon, even though there is no userid logged on the thread, so they are singularly useless.
- X'00000010'** Suppress “NO USERID” logout messages (M204.0353) for all threads. These messages are sometimes issued before a user logon, even though there is no userid logged on the thread, so they are singularly useless.

The MSGCTL NOAUDIT enhancement should be used with caution. Preventing certain messages from going to the journal could make diagnosing certain problems more difficult. It is **not**, however, a new capability or one that is only available to *Janus Web Server* customers. Even without *Janus Web Server*, it is possible to issue a MSGCTL command for any message with the AUDITRK parameter. It is then possible to suppress RK messages from going to the audit trail by making sure the X'20' bit is not set in SYSOPT. The one unfortunate side-effect of this technique is that it also suppresses RK messages that one might actually want to go to the audit trail, but this can be overcome by MSGCTL'ing these messages to NOAUDITRK. Thanks to Warren Kring for this technique.

In any case, the MSGCTL NOAUDIT and NOAUDITW provide a neater and more fine-grained way of reducing audit trail messages for *Janus Web Server* or other threads. Some good candidates for MSGCTL NOAUDIT or NOAUDITW are:

- M204.0099** MINIMUM SERVSZ FOR THESE TABLES = ...
- M204.0131** (Checkpoint completed or timed out messages)
- M204.0353** (login/logout message that is essentially a shortened version of M204.0352 that also gets logged to the journal)
- M204.0608** FILE CLOSED: ...
- M204.0619** GROUP FILE OPENED: ...
- M204.0620** FILE file OPENED (-- NO UPDATES ALLOWED)
- M204.0621** (Informational message indicating recovery status of file being opened)
- M204.0622** (Informational message indicating the last update applied by the last roll-forward for a file being opened)
- M204.0821** GROUP FILE CLOSED: ...
- M204.0858** GROUP group OPENED (-- NO UPDATES ALLOWED)
- M204.1203** FILE file WAS LAST UPDATED ON ...
- M204.1238** (Informational message indicating the time the file was last recovered.)

Under *Sirius Mods* version 6.9 and later, there is also a NOAUD parameter on Janus port definitions that allow even more information to be suppressed from the audit trail on a port basis.

5.68 **WEBOPT**

Default value	X'00'
Parameter type	System
Where set	User 0 CCAIN parameters
Related products	<i>Janus Web Server</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This parameter is a bitmask parameter where the X'01' bit means that RACF calls are to be performed in a special RACF subtask. If using *Janus Web Server* with RACF doing login validation, it is possible to get very high RACF call loads. Since these calls often involve synchronous I/Os that stop the *Model 204* main task, they're generally not very good for *Model 204* performance. Under the WEBOPT X'01' setting, these calls are done in a subtask, dramatically reducing their impact on Online performance.

CHAPTER 6 *User Parameters*

User parameters are *Model 204* parameters that

- can be set using the *Model 204* RESET command by any logged-in user
- affect only the thread on which they are issued

At logout, user parameters are set back to the values they had when the thread was defined, that is, during Online initialization.

6.1 AUTOGCN

Default value	0
Parameter type	User
Where set	User resettable
Related products	<i>Janus SOAP</i>
Introduced	<i>Sirius Mods 7.5</i>

The *Janus SOAP ULI* performs garbage collection automatically at user logout: that is, at logout it removes user-created objects that were not discarded by the user or by the *Janus SOAP ULI* during and after each request. This garbage collection process can also be invoked explicitly (by `%(object):GarbageCollect`), and it may be time consuming. The AUTOGCN User parameter lets you invoke garbage collection automatically during a user session whenever normal post-request cleanup “leaves behind” a number of objects that meets or exceeds the AUTOGCN value.

At the end of a request, normal *Janus SOAP ULI* object cleanup discards all objects — if there are no global or session objects defined in the request. However, if the request created global or session objects, normal cleanup discards only objects that are not global and not session and have no references by other request object variables. Global and session objects and any other objects they reference or that are still referenced by other request object variables are **not** discarded.

Garbage collection sorts through these non-discarded objects and removes those that are not global or session objects or are not accessible, directly or indirectly, from a global or session object. An orphan cycle is a prime example of such an unreachable object. Orphans reference each other and may be complex, associated with many resources. By invoking garbage collection automatically, the AUTOGCN User parameter lets you keep the number of non-discarded, referenced but unreachable, objects below a threshold value.

If $AUTOGCN > 0$, garbage collection runs whenever the number of referenced, non-global, non-session objects created by the current request and not discarded after the current request's normal cleanup is at least the AUTOGCN threshold number.

If $AUTOGCN = 0$, automatic garbage collection is not invoked until user logout.

If garbage collection runs, whether AUTOGCN-invoked or via a `GarbageCollect` method call, you can be notified with a Sirius message if you set the GCSTATS User parameter to one of its non-default values. GCSTATS (“GCSTATS” on page 119) produces a message that indicates the number of objects the garbage collection discarded and how long it took to do so.

The request in the following example creates 50 self-referential, orphan objects which are not discarded by normal request cleanup because the request also creates a global object. The request does not trigger automatic garbage collection, however, because the AUTOGCN threshold is 70.

```

R GCSTATS X'03'
R AUTOGCN 70
Begin

class Linked
  public
    variable next is object Linked
  end public
end class

%cycle is object Linked
%gl is object linked global
%i is float

%gl = new

for %i from 1 to 50
  %cycle = new
  %cycle:next = %cycle
end for

print 'End of request'
End

```

If the request runs a second time, the result is the same as the first time, and now one hundred orphans remain in storage. If the request runs a third time but AUTOGCN is set to 30, the 50 orphans that are newly created and not discarded exceed the AUTOGCN threshold, and automatic collection runs, discarding the orphans from all three requests, but not discarding the global object. The third request produces the following result:

```

End of request
*** MSIR.0995: (implicit) Garbage collection completed in
0ms realtime with 0ms CPU time. Discarded 150/151 objects.

```

Note: If the second and third times the request ran it was modified to exclude the creation of the global object, the requests' orphans would still not be discarded, because the presence of the global object in the initial request affects object cleanup for the life of the user thread, as long as the global is not explicitly discarded.

For more information about garbage collection, see the *Janus SOAP Reference Manual*.

6.2 **ERCNT**

Default value	0
Parameter type	User
Where set	Not resettable
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

This parameter simply displays the current error count — the number of counting error messages issued. If this count exceeds ERMX, the user is restarted.

6.3 GCSTATS

Default value	X'00'
Parameter type	User
Where set	User resettable
Related products	<i>Janus SOAP</i>
Introduced	<i>Sirius Mods 7.5</i>

This bitmask parameter controls the reporting of *Janus SOAP ULI* garbage collection statistics. The statistics, provided in an MSIR error message, include the number of objects discarded by the garbage collection process and the amount of time taken.

These are the GCSTATS bit settings:

X'00' Do not send garbage collection statistics. This is the default.

X'01' Send statistics to the audit trail.

X'03' Send statistics to the audit trail and to the terminal.

The message that is sent is similar to the following, which reports the discarding of 10001 of the 10002 objects still occupying system resources for this thread and session after this request's normal cleanup:

```
*** MSIR.0995: (explicit) Garbage collection completed in
16ms realtime with 16ms CPU time. Discarded 10001/10002
objects.
```

The `(explicit)` tag in the message above signifies that garbage collection was initiated explicitly by a `%(object):GarbageCollect` call. An `(implicit)` tag displays if the garbage collection is invoked for a user by the system because more than the AUTOGCN ("[AUTOGCN](#)" on page 116) threshold number of "orphaned" objects remained after the request's normal object cleanup.

For more information about garbage collection, see the *Janus SOAP Reference Manual*.

6.4 JANDEBM

Default value	X'01'
Parameter type	User
Where set	User resettable
Related products	<i>Janus Web Server</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This is a bitmask parameter that controls the display of messages and data that are destined for the browser while debugging a *Janus Web Server* request with the JANUSDEBUG command (which is described further in “[JAN\[US\]DEB\[UG\]](#)” on page 37).

These are the JANDEBM bit settings:

- X'01'** Display *Model 204* messages (M204.xxx, MSIR.xxx, and USER.xxx) on the debugging user's terminal.
- X'02'** Display application generated data, that is, data from PRINT, HTML, TEXT, or WRITE TERMINAL statements on the debugging user's terminal. Typically this data is HTML, XML, or plain text.

6.5 OBJSTAT

Default value	X'00'
Parameter type	User
Where set	User and \$RESETN resettable
Related products	<i>Janus SOAP</i>
Introduced	<i>Sirius Mods 6.7</i>

This bitmask parameter controls the display of journal messages that contain user statistics about *Janus SOAP ULI* object usage per request. The messages specify server table usage (VTBL and STBL) and object-swapping counts per object class and summed for all classes.

The meanings of the bits in the OBJSTAT parameter are:

- X'01'** Display post-compilation object statistics to the journal.
- X'02'** Display post-evaluation object statistics to the journal.
- X'10'** Display post-compilation object statistics to both the terminal and the journal. If this bit is on, it doesn't matter whether the X'01' bit is on or off — post-compilation statistics will still go to the journal.
- X'20'** Display post-evaluation object statistics to both the terminal and the journal. If this bit is on, it doesn't matter whether the X'02' bit is on or off — post-evaluation statistics will still go to the journal.

The layout for the OBJSTAT messages is:

- One message for each class of objects in the program, with this format:

```
MSIR.0884: OBJECT classname: objects/VTBL/STBL - x/y/z,
count/pages swapped a/b
```

- A single message that reports the sum of the counts of the preceding individual classes, with this format:

```
MSIR.0884: *TOTAL*: objects/VTBL/STBL - x/y/z,
count/pages swapped a/b
```

Where:

- classname** Identifies the name of the class for which space for object instances is allocated. Sirius system classes are prefixed with **SYSTEM:** or

COLLECTION:, as applies. Sirius enumerations are excluded because they do not take up server space outside of what is used for the enumeration variables themselves.

- x** The number of object instances for which server space is allocated for the indicated class for the request. Note that this is different from the number of object variables in a request. Object variables also take up table space, but they are fixed in size (and relatively small) and number at compile time. The actual objects, on the other hand, can be quite large, and they can often have a very large number of instances created at evaluation time.

Unless a *Sirius* compiler directive is specified, this number reflects the number of objects that *Janus SOAP ULI* has determined is the minimum required. Server and, in some cases for system classes, space might be required for three or four objects.

- y** The VTBL units (32 bytes per unit) occupied by the allocated object instances. Each instance of an object in a given class requires the same amount of space: a fixed amount for an object header, and a non-fixed amount for each member variable (that is, every variable in the Public or Private — but not Public Shared or Private Shared — blocks).

The VTBL units reported here do **not** include the VTBL space that is occupied by each local object reference in the main program routine. These references include object variables in declarations (including method parameter declarations and explicit and implicit method objects) and “work” object variables. These work variables are used to hold intermediate results in the evaluation of statements containing method concatenations. They may be reused within the class, and they typically do not add significantly to the VTBL space used. Object references use 2 VTBL units each, excepting XmlNodeNodes, which use approximately 8.

- a** The number of object swaps. This is the number of times objects of this class were swapped into or out of the server during the request. A swap out counts as one, and a swap in as one, so a swap in and out counts as two. Accessing a global/session object counts as a swap in, since objects do not stay in the server between requests, so they need to be swapped into/out of CCATEMP.

Swaps happen only during evaluation, so this number should always be zero in the post-compilation statistics.

- b** The number of 6144-byte pages swapped into or out of the server. This number should always be an integral multiple of **count**, and the multiplier should be one, except for very large objects (greater than 6144 bytes).

The following is an example of typical post-compilation OBJSTAT output:

```
MSIR.0884: OBJECT LONG: objects/VTBL/STBL - 2/7/0, count/pages
           swapped 0/0
MSIR.0884: OBJECT PT: objects/VTBL/STBL - 2/18/0, count/pages
           swapped 0/0
MSIR.0884: OBJECT CELL: objects/VTBL/STBL - 2/20/0, count/pages
           swapped 0/0
MSIR.0884: OBJECT BOX: objects/VTBL/STBL - 2/75/144, count/pages
           swapped 0/0
MSIR.0884: COLLECTION SYSTEM:ARRAYLIST OF OBJECT PT:
           objects/VTBL/STBL - 3/25/0, count/pages swapped 0/0
MSIR.0884: OBJECT SYSTEM:STRINGLIST: objects/VTBL/STBL - 3/18/0,
           count/pages swapped 0/0
MSIR.0884: *TOTAL*: objects/VTBL/STBL - 14/163/144, count/pages
           swapped 0/0
```

And the following is an example of typical post-evaluation OBJSTAT output:

```
MSIR.0884: OBJECT LONG: objects/VTBL/STBL - 2/7/0, count/pages
           swapped 571/571
MSIR.0884: OBJECT PT: objects/VTBL/STBL - 2/18/0, count/pages
           swapped 310/310
MSIR.0884: OBJECT CELL: objects/VTBL/STBL - 2/20/0, count/pages
           swapped 528/528
MSIR.0884: OBJECT BOX: objects/VTBL/STBL - 2/75/144, count/pages
           swapped 0/0
MSIR.0884: COLLECTION SYSTEM:ARRAYLIST OF OBJECT PT:
           objects/VTBL/STBL - 3/25/0, count/pages swapped 0/0
MSIR.0884: OBJECT SYSTEM:STRINGLIST: objects/VTBL/STBL - 3/18/0,
           count/pages swapped 0/0
MSIR.0884: *TOTAL*: objects/VTBL/STBL - 14/163/144, count/pages
           swapped 1409/1409
```

As can be seen from this example, the object statistics can take up a fair amount of space and, in many cases, be quite uninteresting — classes without any swapping are typically of not much interest in post-evaluation statistics. The OBJSTMIN parameter (“OBJSTMIN” on page 124) can be used to limit the quantity of object statistics logged to the journal to those statistics that are actually of interest.

Regardless of the setting of OBJSTAT and OBJSTMIN, the since-last, user, and system OBJSWAP statistics are incremented for every object swap. It is probably worth looking at these stats before deciding whether to try to track down the cause of object swapping using OBJSTAT — if the OBJSWAP numbers are relatively low, it’s probably not worth much effort looking at detailed object-swapping statistics.

Note: Before *Sirius Mods* version 6.9, OBJSTAT was a system manager resettable system parameter.

For more information about object swapping, see the *Janus SOAP Reference Manual*.

6.6 OBJSTMIN

Default value	0
Parameter type	User
Where set	User and \$RESETN resettable
Related products	<i>Janus SOAP</i>
Introduced	<i>Sirius Mods 6.8</i>

This numeric parameter indicates the minimum number of object swaps required for a class, before its post-evaluation statistics are displayed at the user terminal or the journal. Post evaluation object statistics are requested by setting the X'02' or X'20' bits in the OBJSTAT parameter (“[OBJSTAT](#)” on page 121). The OBJSTMIN parameter has no effect on post-compilation statistics — it is assumed that the reason for requesting post-compilation stats is to get server space-usage information, so the number of server swaps for a class (which is always zero, anyway, during compilation) is irrelevant.

The default value for OBJSTMIN of 0 means that post-evaluation statistics will be provided for all classes in a request, since even an unused class would meet the minimum swap requirement of 0.

Regardless of the setting of OBJSTAT and OBJSTMIN, the since-last, user, and system OBJSWAP statistics are incremented for every object swap. It is probably worth looking at these stats before deciding whether to try to track down the cause of object swapping using OBJSTAT — if the OBJSWAP numbers are relatively low, it's probably not worth much effort looking at detailed object-swapping statistics.

Note: Before *Sirius Mods* version 7.0, OBJSTMIN was a system manager resettable system parameter.

For more information about object swapping, see the ***Janus SOAP Reference Manual***.

6.7 RETRVOPT

Default value	X'00'
Parameter type	User
Where set	User resettable
Related products	All
Introduced	<i>Sirius Mods 7.3</i>

This is a bitmask parameter that controls some characteristics of the *Model 204* retrieve key, the 3270 terminal PF key that retrieves previously input command lines.

The meaning of the RETRVOPT bits are:

X'01' The PF key that is 1 higher than the RETRVKEY setting is mapped to a forward retrieval operation, which returns the command in the retrieve buffer that follows the one that is currently displayed. For example, if this X'01' bit is set, and RETRVKEY is set to 6, PF7 will be a forward retrieve key.

If the RETRVKEY setting is 24, specifying this X'01' bit maps the forward operation to the PF1 key.

X'02' The forward and backward retrieve keys will **not** wrap. With this bit set, if you retrieve all the commands stored in the retrieve buffer, your next retrieval is a blank (instead of wrapping to the beginning of the retrieval ring, as was the behavior prior to the availability of the RETRVOPT parameter).

The retrieve buffer is a virtual storage area allocated for any full screen thread that has a retrieve key set, and its size is controlled by the RETRVBUF parameter ([“RETRVBUF” on page 77](#)).

It is recommended that you set RETRVOPT X'02' in tandem with RETRVOPT X'01'. Otherwise, once you reach the end of the retrieve buffer, you will be unable to get at the retrieved data until you press the Enter key or some other PF key.

X'04' Do not add to the retrieve buffer any text typed in response to a screen-full prompt. For example, if you issue a *VIEW*, then type *C* or *K* to stop the output, the *C* or *K* is added to the retrieve buffer — unless RETRVOPT X'04' is set.

A command you enter when on the penultimate page of a multi-page output (like from a *VIEW* command) that gets processed after the next page of data is **not** saved in the retrieve buffer.

RETRVOPT X'04' also prevents **N**, **NP**, or **NEW** commands (which clear the screen) from being added to the retrieve buffer.

X'10' If the X'01' bit is also set, RETRVOPT X'10' maps the forward retrieve key to the PF key that is 12 greater than the current retrieve key (setting of RETRVKEY). This will be the shifted version of the current retrieve key if that current key is less than PF13, and it will be the non-shifted version if that key is greater than PF12.

For example, if RETRVOPT is set to X'17' and RETRVKEY is 12, the forward retrieve key is PF24. And if RETRVOPT is set to X'17' and RETRVKEY is 21 (Shift+PF9), the forward retrieve key is PF9.

6.8 SCRNSTBL

Default value	6144
Parameter type	User
Where set	User resettable
Related products	All
Introduced	<i>Sirius Mods 7.1</i>

This is a numeric parameter (with valid values 0 or greater) that indicates the maximum amount of STBL space available to *Janus SOAP* Screen objects. Screen objects use STBL space and not FSCB space, and any instance of a screen object must allocate the maximum allowable STBL space, even if some of this space may not be used by a particular application. An application that exceeds its screen STBL allocation is cancelled.

Some *Sirius UL/SPF* products, like *SirScan*, also make use of screen objects. The SCRNSTBL default value has been set to be sufficient for those products.

6.9 SIRCOMP

Default value	X'00'
Parameter type	User
Where set	User resettable
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This is a bitmask parameter that controls some User Language compilation options. The meaning of the SIRCOMP bits are:

- X'01'** Html/Text statements should compile literals onto an internal \$list rather than STBL. Of course, for a pre-compiled procedure, this \$list will be shared among all users of the procedure. This reduces STBL requirements for requests with large quantities of literal text inside Html/Text blocks, at the cost of slightly more disk buffer pool usage. Access to these string literals stored in CCATEMP is highly optimized, so it might have little or no CPU cost compared to storing them in STBL — in fact, there might be some CPU benefits in storing them in CCATEMP.
- X'02'** If a compilation that was started inside a procedure (the Begin statement was in a procedure) is still active when the last Included procedure is closed (return to command prompt on an interactive thread), the compilation is terminated with an error.
- X'04'** If a compilation that was started inside a procedure is still active when the procedure that contained the Begin is closed, the compilation is terminated with an error.

The X'02' and X'04' settings are intended to minimize the problems or annoyances associated with a forgotten End statement in a User Language procedure, or, more commonly, unmatched quotes resulting in an End statement being “swallowed” as part of a literal (producing the well-known `M204.1248: LOOKING FOR CLOSE QUOTE` message). A compilation ended because of the SIRCOMP X'02' or X'04' setting also automatically closes any unclosed quotes or unclosed comment blocks.

The difference between the X'02' bit and the X'04' bit is that the X'02' bit (when the X'04' bit is not set) allows an Including procedure to contain the End statement for a Begin that was in an Included procedure. Unless this unusual structure is used, the X'04' bit is probably a better choice for most sites. If the X'04' bit is set, the setting of the X'02' bit has no effect.

6.10 SIREEDIT

Default value	X'00'
Parameter type	User
Where set	User resettable
Related products	All
Introduced	Before <i>Sirius Mods 6.7</i>

This is a bitmask parameter that controls various enhancements to the *Model 204* full-screen editor. The meaning of the bits are:

X'01' Switch to *LOWER mode upon entry to the editor, and revert to prior setting upon exit. This allows command mode to work in *UPPER mode so that commands are automatically uppcased, while allowing the editor to work in mixed-case mode.

Since most *Model 204* commands take entirely uppercase input, *UPPER mode is almost always preferable in command mode. On the other hand, most modern User Language programs use mixed case, so *LOWER mode is preferable inside the editor. The SIREEDIT X'01' bit allows one to get the best of both worlds with no effort.

This bit has no effect if the user is already in *LOWER mode on entry to the editor.

X'02' Change the LINEND character for the current procedure to carriage return (X'0D'). This allows procedures to include semicolons as data, and it also makes it easier to edit a procedure that has been uploaded from an HTML generator.

X'04' Always force the current 3270 Mod 2 behavior (24 lines by eighty characters), even if the editor is invoked by a thread using a Mod 5 session. This bit takes precedence over the following bit.

X'08' Allow Mod 5 sessions in DBCS environments. Ordinarily the widescreen (Mod 5) support is disabled under DBCS environments. When this bit is set, even DBCS sessions are eligible for widescreen support.

X'10' Provide case-insensitive locate in non-DBCS environments. Thus, the editor command

```
/this is
```

will locate both of the following lines, regardless of the setting of *LOWER/*UPPER or the SIREDDIT X'01' bit:

```
This is sometimes hard to find due to mixed
case, while "this is" is all lower case.
```

- X'20'** Provide case-insensitive search strings for the replace command, while still retaining the case as entered on the replacement string, assuming that either *LOWER is in effect or the X'01' bit of SIREDDIT is set. Under these conditions, and using the text from the previous example, the command:

```
r/this is/This Is/* all
```

will change occurrences on both lines, resulting in:

```
This Is sometimes hard to find due to mixed
case, while "This Is" is all lower case.
```

In addition to the SIREDDIT parameter, the *Sirius Mods* provide some additional full-screen editor commands (["Editor Commands" on page 135](#)).

6.11 SRSPARM

Default value	X'00'
Parameter type	User
Where set	User and \$RESETN resettable
Related products	All
Introduced	Before <i>Sirius Mods</i> 6.7

This bitmask parameter makes it possible to change the default behavior of the \$web_rest_recset and \$web_rest_list functions. By default, they act as if MOVE were specified (see the *Sirius Functions Reference Manual* for more information on these functions). The default behavior of all of these functions can, however, be changed on a thread basis by setting the SRSPARM parameter (which can be reset with \$RESETN).

SRSPARM is a bitmask user parameter where the bits mean:

- X'01'** Make default for \$WEB_SAVE_RECSET and \$WEB_SAVE_LIST “ERROR”.
- X'02'** Make default for \$WEB_SAVE_RECSET and \$WEB_SAVE_LIST “CANCEL”. If this bit is set, the X'01' bit is irrelevant.
- X'04'** Make default for \$WEB_SAVE_RECSET and \$WEB_SAVE_LIST “UERROR”.
- X'08'** Make default for \$WEB_SAVE_RECSET and \$WEB_SAVE_LIST “UCANCEL”. If this bit is set, the X'04' bit is irrelevant.
- X'10'** Make default for \$WEB_SAVE_RECSET and \$WEB_SAVE_LIST “COPY”.
- X'20'** CANCEL request on a COPY style \$WEB_SAVE_RECSET for an exclusively locked record set. If this bit is not set, this error would simply be reflected with an error code of 7. This operation is not allowed, because there cannot be two threads that have the same records locked in exclusive mode, which is what would be the case after a COPY type save of an exclusively locked record set.
- X'40'** Make default for \$WEB_REST_RECSET and \$WEB_REST_LIST “COPY”.
- X'80'** CANCEL request on a COPY style \$WEB_REST_RECSET for an exclusively locked record set. If this bit is not set, this error would simply be reflected with an error code of 7. This operation is not allowed, because there cannot be two threads that have the same records locked in exclusive mode, which is what would be the case after a COPY type restore of an exclusively locked record set.

In general, table full problems with saved record sets are best handled (as are most system-wide resource shortages) with request cancellation, because it does not make sense to code applications in a way that would be able to detect and correct for these shortages.

6.12 ULTRACE

Default value	X'01'
Parameter type	User
Where set	User and \$RESETN resettable
Related products	<i>SirFact</i>
Introduced	Before <i>Sirius Mods 6.7</i>

This bitmask parameter controls the behavior of the User Language Trace statement. The bits mean:

- X'01'** Send TRACE output to terminal. This is the default.
- X'02'** Send TRACE output to audit trail.
- X'04'** Send TRACE output to a CCATEMP wrap-around trace table. See [“ULTRACEP” on page 134](#).

The default setting for ULTRACE (X'01') means that the Trace statement acts much like a Print statement. The bits can be combined so that output can be sent to both the terminal and the audit trail, or to the CCATEMP wrap-around trace table and to the terminal and to the audit trail.

If a Trace statement is executed when ULTRACE is set to 0, the request is cancelled.

For more information about the Trace statement, see the ***SirFact Reference Manual***.

6.13 ULTRACEP

Default value	2
Parameter type	User
Where set	User and \$RESETN resettable
Related products	<i>SirFact</i>
Introduced	Before <i>Sirius Mods</i> 6.7

This numeric parameter indicates the maximum number of CCATEMP pages that will be allocated for a user for holding trace data. The CCATEMP wrap-around trace table is a user-specific trace table that consists of a set of CCATEMP pages whose maximum size is specified by the ULTRACEP user parameter.

If ULTRACEP pages are already in use when a Trace statement is issued, and output is being routed to the trace table (ULTRACE X'04' set — see “[ULTRACE](#)” on page 133), the trace data on the oldest page is discarded, and the oldest page is re-used for the new trace data. For more information on the Trace statement, see the ***SirFact Reference Manual***.

The wrap-around trace table is dumped in *SirFact* dumps, and it can be viewed with the following command when using the FACT subsystem to examine a dump:

```
D[ISPLAY] T[RACE][.{* | N}]
```

where *N* specifies that the last *n* entries are to be displayed. Specifying *D T* will display all trace entries, and specifying *D T.20* will display the last 20.

The current contents of the wrap-around trace table can also be examined with the `$trace2list` function or the `StringList` class `AppendTrace` method. For more information about *SirFact* dumps, the `$trace2list` function, and the `AppendTrace` method, see the ***SirFact Reference Manual***.

CHAPTER 7 *Editor Commands*

Editor commands are commands that are available in the full-screen *Model 204* editor for *Sirius Mods* customers.

7.1 CASE Mixed/Upper command

You may issue either of the following commands in the *Model 204* Full Screen Editor:

CASE Mixed This command indicates that terminal input is accepted “as is,” without translation to uppercase. This is similar to invoking the *LOWER command prior to entering the Full Screen Editor. The second word of this command, “Mixed”, may be abbreviated to any prefix.

CASE Upper This command indicates that terminal input is translated to uppercase. This is similar to invoking the *UPPER command prior to entering the Full Screen Editor. The second word of this command, “Upper”, may be abbreviated to any prefix.

This command will not affect the case translation setting (*UPPER/*LOWER) that was in effect “outside” the editor; that is, when you exit the editor, the case translation will be as it was prior to entering the editor.

In addition to the CASE command, the initial case translation setting within the editor can be automatically set to *LOWER, using the '1' bit of the SIREDDIT parameter (“[SIREDDIT](#)” on page 129).

Neither the CASE command nor the *UPPER/*LOWER setting affect the input on the editor command line.

The matching of strings (for example, “/subroutine”) can be done in a case-insensitive manner, using the '2' bit of the SIREDDIT parameter (“[SIREDDIT](#)” on page 129).

Terminal MODEL 6 support

The *Sirius Mods* adds support for terminal models beyond the standard Mod 2 (24 X 80), Mod 3 (32 X 80), Mod 4 (43 X 80), and Mod 5 (27 X 132). The new terminal models are supported by setting the terminal model to 6:

RESET MODEL 6

There's really no such thing as a Model 6 terminal, but setting the terminal model to 6 tells *Model 204* to issue a Write Structured Field Query to the terminal to have the terminal indicate its geometry (number of rows and columns) to *Model 204*. In this way, *Model 204* can dynamically set a terminal's geometry, whether it's one of the standard geometries (Mod 2, 3, 4, or 5) or not. Many terminal emulators allow alternate 3270 sizes to be set. This makes it possible to set the terminal geometry to match the optimal combination of font size and physical screen size for a particular workstation, rather than trying to set the emulator font size to work well with one of a limited number of screen geometries.

Unfortunately, the standard User Language screen definitions don't allow the defining of fields that extend beyond column 79. However, \$scrHide, \$scrSize, and \$scrWide (described in the *Sirius Functions Reference Manual*) make it possible for User Language screens to take advantage of columns beyond column 79. In addition, these functions make it possible to dynamically modify screen definitions to allow a single screen definition to work with an arbitrary variety of screen sizes. While these functions are a bit awkward to use and somewhat limited, they are not unreasonable for building dynamic scrolling screens — scrolling screens being particularly suited for larger screen geometries.

To facilitate User Language applications for varying screen sizes, the VIEW command for the MODEL parameter has been enhanced to show the screen geometry in addition to the model number for model 6 terminals:

```
> V MODEL
MODEL      6 34*142    TERMINAL MODEL
```

Issuing \$view for the above terminal returns a “6 34*142”, from which a User Language application can readily determine that the screen has 34 rows and 142 columns.

To enable model 6 support, the SIRTERM system parameter (“[SIRTERM](#)” on page 99) must be set in the CCAIN stream.

If a terminal is using a non-standard screen geometry via model 6 support, the *Model 204* editor and command line will correctly use the available screen space. Many UL/SPF subsystems such as SirScan, SirMon, and SirPro will also take advantage of the additional available screen space.

Index

\$

\$SIR_DATE_ERR ... 36

A

ACTRDS system parameter ... 40
 APPDATE user command ... 36
 APSYSEC system parameter ... 41
 Authorizing RESTART users ... 22
 AUTOGCN user parameter ... 116

B

Backing up a Model 204 file ... 15
 Backing up CCAGRP ... 16
 BUMP operator command ... 5
 Bumping and snapping a user ... 14
 BUMPSNAP system administrator
 command ... 14

C

CASE editor command ... 136
 CENTSPAN system parameter ... 42
 Clock control ... 36
 CLOSE operator command ... 6
 COMMLOG system parameter ... 43
 COMPOPT system parameter ... 44
 Configuring Janus ... 21
 CSIPID system parameter ... 45
 CURREP31 system parameter ... 46
 CURREP64 system parameter ... 47

D

Date handling ... 36
 DATE_ERR ... 36
 DEBUGMAX system parameter ... 48
 DEBUGPAG system parameter ... 49
 Determining applied ZAPs ... 29
 Determining location of a \$function ... 19
 Determining location of Fast/Unload load
 module ... 19
 Determining Sirius Mods version ... 29
 DUMP system administrator command ... 15

DUMPG system administrator command ... 16
 DUMPGX system administrator
 command ... 16
 Dumping a Model 204 file ... 15
 Dumping CCAGRP ... 16
 DUMPOPTS system parameter ... 50
 DUMPTMAX system parameter ... 51
 DUMPTMIN system parameter ... 52
 DUMPX system administrator command ... 15

E

ENTTAB system parameter ... 53
 ERCNT user parameter ... 118

F

FILELOAD system administrator
 command ... 17
 FILELOADX system administrator
 command ... 17
 FLOD system administrator command ... 18
 FLODX system administrator command ... 18
 FNADDR system administrator command ... 19
 FNVMASK system parameter ... 54
 FUDAEMON system administrator
 command ... 20
 FUNCOPTS system parameter ... 56
 FUNMAXT system parameter ... 58
 FUNPARAM system parameter ... 59
 FUNPGM system parameter ... 60
 FUNTSKN system parameter ... 61

G

Garbage collection ... 116
 GCSTATS user parameter ... 119

H

HGHREP31 system parameter ... 62
 HGHREP64 system parameter ... 63

I

IBSIZE parameter, JANUS DEFINE ... 102

J

JANDEB user command ... 37
JANDEBM user parameter ... 120
JANDEBUG user command ... 37
JANUS system administrator command ... 21
JANUSDEB user command ... 37
JANUSDEBUG user command ... 37

M

MAXBG system parameter ... 64
MAXDAEM system parameter ... 65
MAXRDS system parameter ... 66
MAXREP31 system parameter ... 67
MAXREP64 system parameter ... 68
MODEL parameter, VIEW command ... 137
MODPROT system parameter ... 69
MONITOR operator command ... 7
MONPARAM system parameter ... 70

N

NCMPBUF system parameter ... 72
NSUBTKS parameter, Model 204 ... 110

O

OBJSTAT user parameter ... 121
OBJSTMIN user parameter ... 124
OBSIZE parameter, JANUS DEFINE ... 102

P

PERFOPT system parameter ... 73
PERFOPT2 system parameter ... 74
PUPDTH system parameter ... 75

R

RESTART operator command ... 8
RESTARTU system parameter ... 76
RESTAUTH system administrator
command ... 22
RESTORE system administrator
command ... 23
RESTOREG system administrator
command ... 24
RESTOREGX system administrator
command ... 24

RESTOREX system administrator
command ... 23

Restoring a Model 204 file ... 23
Restoring CCAGRP ... 24
RETRVBUF system parameter ... 77
RETRVOPT user parameter ... 125

S

SAMPLE operator command ... 10
SCANPARAM system parameter ... 78
SCANTIME system parameter ... 79
SCRNSTBL user parameter ... 127
SDAEMDEV system parameter ... 81
SDEBGUIP system parameter ... 82
SDEBWRKP system parameter ... 83
SDMOPT system parameter ... 84
SESDEFOW system parameter ... 85
SESDEFTO system parameter ... 86
SESNPRV system parameter ... 87
SESNPUB system parameter ... 88
SESOPT system parameter ... 89
SESUMAX system parameter ... 90
Setting parameters for Janus ... 21
SIRAPSY system administrator command ... 25
SIRAPSYF system parameter ... 91
SIRCOMP user parameter ... 128
SIRDEBUG user command ... 38
SIREDDIT user parameter ... 129
SIRFACT system administrator command ... 26
SIRFACT system parameter ... 94
SIRFIELD system administrator
command ... 28
SIRFUNC system parameter ... 96
SIRIUS DEBUG user command ... 38
SIRIUS system administrator command ... 30
SIRIUSDEBUG user command ... 38
SIRMETH system administrator
command ... 31
SIRMSG system parameter ... 97
SIRPDLHW system parameter ... 100
SIRPRMPT system parameter ... 98
SIRTERM system parameter ... 99
SIRTUNE system parameter ... 101
SOCKMAX system parameter ... 102
SPANSIZE system parameter ... 103
SRSDEFTO system parameter ... 104
SRSMAX system parameter ... 105
SRSMAXTO system parameter ... 106
SRSMAXUS system parameter ... 107

SRSPARM user parameter ... 131
STATUS operator command ... 11
STBL space ... 127
STOP operator command ... 12
SYSDATE parameter ... 36

T

TCPSERV system parameter ... 108
TCPSUBSY system parameter ... 109
TCPTYPE system parameter ... 110
Time handling ... 36
TNDEV system parameter ... 111

U

ULTRACE user parameter ... 133
ULTRACEP user parameter ... 134
UNICODE system administrator
command ... 32

W

WEBAUDIT system parameter ... 112
WEBOPT system parameter ... 114

