VSI Job Management for OpenVMS Programming Guide

Operating System and Version: VSI OpenVMS x86-64 Version 9.2-3

Software Version: VSI Job Management V3.1 for OpenVMS

VMS Software, Inc. (VSI)
Boston, Massachusetts, USA

VSI Job Management for OpenVMS Programming Guide

Legal Notice

Table of Contents

Preface

1. About VSI

VMS Software, Inc. (VSI) is an independent software company licensed by Hewlett Packard Enterprise to develop and support the OpenVMS operating system.

2. Intended Audience

This guide is written for the following types of users:

- Data center operations teams, including system managers, system operators, and schedulers
- Production application support teams
- Application developers
- Other OpenVMS users

This guide assumes that Job Management Manager programmers are familiar with the OpenVMS operating system and with processing Digital Command Language (DCL) commands in both interactive and batch modes.

3. Document Structure

This manual is organized as follows:

- *Chapter 1, "VSI Job Management Manager API"* introduces the OpenVMS API, covering callable routines, privilege requirements, WAN support, and definitions of supported subroutines and functions for the Job Management Manager.
- *Chapter 2, "Job Management Manager Programming"* covers the topics, including example programs, date and time formats, error codes, return-buffer layouts for VSS routines, and the event log file structure.

4. Related Documents

This guide is part of the VSI Job Management for OpenVMS documentation set, which also includes the following:

- *VSI Job Management for OpenVMS Release Notes* [https://docs.vmssoftware.com/vsi-job-management-release-notes-v31/]
- *VSI Job Management for OpenVMS Installation Guide* [https://docs.vmssoftware.com/vsi-job-management-installation-guide-v31/]
- *VSI Job Management for OpenVMS Administration Guide* [https://docs.vmssoftware.com/vsi-job-management-administration-guide-v31/]
- *VSI Job Management for OpenVMS Command Reference Guide* [https://docs.vmssoftware.com/vsi-job-management-command-reference-guide-v31/]

5. VSI Encourages Your Comments

You may send comments or suggestions regarding this manual or any VSI document by sending electronic mail to the following Internet address: <docinfo@vmssoftware.com>. Users who have VSI OpenVMS support contracts through VSI can contact <support@vmssoftware.com> for help with this product.

6. OpenVMS Documentation

The full VSI OpenVMS documentation set can be found on the VMS Software Documentation webpage at https://docs.vmssoftware.com.

Chapter 1. VSI Job Management Manager API

1.1. Introduction

The library files NSCHED_SUBS.OLB and CLIB.OLB contain all the subroutines used to implement the DCL command interface to the manager. These library files are in your NSCHED$ directory.

This chapter describes the routines in NSCHED_SUBS.OLB that are officially supported by VSI; these routines make up the API to Job Management Manager. The undocumented routines are mostly lower-level routines that are called by the documented routines. Your programs should call only the routines documented in this manual.

1.2. Required Privileges

To access all callable routines, callers should have SYSPRV, SYSLCK, and CMEXEC privileges or have the images installed with those privileges. In particular, the following system privileges are required to use the API:

- SYSPRV, to access the Job Management Manager database
- SYSLCK, to notify the manager of database changes

If an image that calls the access routines will run from a nonprivileged account, you should install the image with SYSPRV and SYSLCK privileges. The VSS$PID_RUNNING routine requires the CMEXEC privilege. The VSS$INSERT routine can call VSS$PID_RUNNING, so VSS$INSERT may also require the CMEXEC privilege. You can also use the LIB$GETJPI routine on OpenVMS.

Note

The routines that perform functions involving Job Management Manager database files must open access channels to those files. To eliminate the overhead that would occur if a single image opened the database repeatedly, the access routines share a common block of memory and reuse previously opened channels. Thus, even if your program calls the access routines repeatedly, it will still only open the database once.

The DCL interface to the manager checks for valid arguments before calling these routines. Using these routines without checking whether passed arguments are correct can lead to undesired results. For example, it is possible to alter a job to depend on a nonexistent job or on itself.

All NSCHED$_ERROR symbols are defined and included in module NSCHED_MSG.OBJ, which is included in NSCHED_SUBS.OLB. Your program should declare the symbols as external longword constants (a global value longword for VSI C). Some of the callable routines also return negative numbers as an error status under certain conditions. These negative numbers are the negative of VSI BASIC error codes. See the VSI BASIC documentation for further explanation.

1.3. Wide Area Network Support

Many of the callable routines have a `<variable>(server)` argument. If you specify a string that is not blank, the routine calls a remote procedure to perform the requested operation on the remote node.

The manager asks if you want support for remote execution or wide area network support. In case your library does not have wide area network support, your copy of NSCHED_SUBS.OLB library file contains a module called NO_RPC_SUPPORT.OBJ. This module signals an NSCHED$_NORPC fatal error if you try to bind to a remote node. If your library has wide area network support, NSCHED_SUBS.OLB contains stubs for the remote routines.

You can use one of the following three formats for the string you pass as a server argument:

NODE1	Performs the operation on NODE1, using the default proxy for access.
NODE1"*username*"	Performs the operation on NODE1, using the proxy *username.* You must also have a default proxy on NODE1.
NODE1"*user passwd*"	Uses the account specified by explicit access control; no proxy is required for access.

The maximum length of the server argument string is 64 bytes.

The remote procedure calls the VSS$BIND internal routine, which prints the following type of status message to SYS$OUTPUT:

```
Connecting... connected to node xxx
```

You can suppress these messages by defining the logical NSCHED$SILENT to any value.

1.4. Definitions of Data Structures

This section describes the data structures referenced by Job Management Manager callable routines.

1.4.1. YEAR_BITMASK Data Structure

This structure is a 13 by 1 (longword) bitmask representing one calendar year. The first longword identifies which calendar year is described (for instance, 1991 or 2000). Each of the other 12 longwords represents a month of the year.

```
+-------------------------------------------------------------+
|                Flags        |      Calendar Year ID         |
+-------------------------------------------------------------+
|                      1st Month (JAN)                        |
+-------------------------------------------------------------+
|                      2nd Month (FEB)                        |
+-------------------------------------------------------------+
|                      3rd Month (MAR)                        |
+-------------------------------------------------------------+
|                      (and so forth)                         |
+-------------------------------------------------------------+
|                      12th Month (DEC)                       |
+-------------------------------------------------------------+
```

Each bit of each longword represents a day of the month. Every day in a year may be represented using this structure.

This structure is used in passing absolute dates to and from the Special Days callable routines. The structure may also be used to represent instantiations of Special Day Classes or job Special Day Restrictions.

1.4.2. ITEM_BLOCK Data Structure

This structure is a generic structure used to pass instructions and data to and from callable routines. There are many uses for this structure, but all have the same basic format:

```
+-------------------------------------------------------------+
|               Buffer Length | block ID type code            |
+-------------------------------------------------------------+
|                      Buffer Address                         |
+-------------------------------------------------------------+
|                   Return Length Address                     |
+-------------------------------------------------------------+
```

To send instructions and/or data to a callable routine, the caller forms an array of one or more of these structures and passes a pointer to the array to the callable routine. The array must be terminated by a longword of 0 (zero).

1.4.3. CLASS_SPECIFIER Data Structure

This structure is used to pass Special Day Class Specifications to and from callable routines.

```
+-------------------------------------------------------------+
|                               |        String length        |
+                               -------------------------------+
|                                                             |
\           ClassSpecifier string (cont)                      \
|           Format = (CLASSNAME=USERNAME)                     |
|                                                             |
+-------------------------------------------------------------+
```

1.4.4. ASCII Date Specifier Data Structure

This structure is used to pass ASCII Date specifications to and from callable routines.

```
+-------------------------------------------------------------+
|                               |        String length        |
+                               -------------------------------+
|                                                             |
\               ASCII Date string (cont)                      \
|                Format = (DD-MMM-YYYY)                       |
|                                                             |
+-------------------------------------------------------------+
```

1.4.5. TIME Quadword Definition

Following is the definition of the TIME quadword, used for passing binary times:

```
typedef long TIME_QUADWORD_T[2];
```

1.5. Using the Callable Interface

When you use the callable interface, keep the following in mind.

1.5.1. Using VSI BASIC

VSI BASIC uses non-contiguous class arrays (DSC$K_CLASS_NCA class arrays) rather than the contiguous class arrays used by VSI BASIC. When using the VSI Job Management for Open VMS API with VSI BASIC on OpenVMS I64 systems, BASIC performs argument validation when you pass an array in a subroutine. If you attempt to pass an incorrect class of array (contiguous instead of non-contiguous), the following error message displays at run time, followed by a stack dump:

```
BAS-F-ARGDONMAT, Arguments don't match
-BAS-I-FROSUB, In subprogram
%TRACE-F-TRACEBACK, symbolic stack dump follows
```

1.5.2. Compiling on OpenVMS Alpha Systems

When compiling a program written in C, you must use the VSI C compiler and specify the following compiler options to use the callable interface:

```
cc /NOMEMBER_ALIGNMENT/EXTERN=common
```

Depending on your coding practice, you may also need to add the **/STANDARD=VAXC** specifier.

1.5.3. Compiling Pascal Programs

When compiling a program written in Pascal, you must specify the following compiler options in order to use the callable interface:

```
PASCAL/ALIGN=vax
```

1.5.4. Linking an Object Library

When linking a program's object library, you must specify the following in the link procedure:

```
/SYSEXE
NSCHED$:NSCHED_SUBS.OLB/LIB
NSCHED$:CLIB.OLB/LIB
```

When linking a program's object library that uses TCP/IP connection services on OpenVMS systems, you must include a reference to the sharable image or object library for your specific TCP/IP stack in the link procedure. For more information, see the documentation for your TCP/IP stack.

1.5.5. Extended Job Field Lengths

Starting from Release 3.1, certain fields in the job database have extended lengths. If an application program written before Release 3.1 which will run against a database that has extended fields or will have extended fields in the foreseeable future, change your application source code as necessary. Then the application program must be recompiled and relinked after installing VSI Job Management for OpenVMS V3.1.

1.6. Callable Routines

This section describes the Job Management Manager supported callable routines. This section does not cover the lower-level routines that are called by these supported routines. Your programs should not have to call these lower-level routines directly.

Each routine description includes the following:

- Routine name
- Format
- Arguments
- Possible condition values (where appropriate)
- Possible return values

The following table briefly describes the Job Management Manager callable routines. See *Section 1.6, "Callable Routines"* in this chapter for a complete description of each routine.

Routine	**Description**
VSS$BITMASK_TO_DATE	Takes a bitmask representing all the days of a year, and returns the days whose bits are set. Caller may choose to receive this information as a null-terminated ASCII string or as a quadword. Users may have to make more than one call to get all requested information.
VSS$CHANGE_DEFAULT	Sends a message to request that a particular manager instance become the new default executor for the OpenVMS Cluster.
VSS$CHECK_BITMASK_DATE	Checks to see if certain bits are set in the YEAR_BITMASK. Takes as input a date specification and the address of a YEAR_BITMASK structure to check.
VSS$CHECK_PRIV	Checks the caller's access privileges for a job.
VSS$CREATE_ITEM	Creates a new Job Management Manager job with the given parameter values. The routine takes an item list of values to apply to the new job. Unspecified items take default parameters. The routine requires that an OpenVMS command for the job be provided in the item list. Use instead of VSS$INSERT for new development.
VSS$CREATE_SD_CLASS	Creates and defines a Special Day Class. Users may define an empty Special Day Class or provide an ItemList describing elements to be inserted into this new class. Only users with SYSPRV or OPER privileges may create Special Day Classes.
VSS$DELETE_SD_CLASS	Deletes a Special Day Class from the manager’s Special Days database. Only users with SYSPRV or OPER privileges may delete Special Day Classes.
VSS$FIND_DEPS	Returns a longword array containing the job numbers of all jobs in the specified OpenVMS Cluster which depend on a particular `job_node::job_number` pair.
VSS$GET_DEPNODES	Returns a list of nodes that have jobs that depend on a particular job in the local Job Management Manager database.
VSS$GET_HISTORY	Calculates the minimum, maximum, average, and current use of resources for a job.
VSS$GET_HIST_REC	Returns a buffer containing information about a single job run from the VSS job history log file.
VSS$GET_NEXT_START_TIME	Calculates a job’s next scheduled run time based on a given start time, schedule interval, days-of-week mask and the job’s Special Day Restrictions. This routine is much like

Routine	Description
	VSS$GET_START_TIME with the exception that this routine checks against Special Day Restrictions for the job you specify. This routine requires that the caller have READ access to the job database record. Use VSS$GET_NEXT_START_TIME instead of VSS$GET_START_TIME for new development.
VSS$GET_SD_ACTIONS	Gets information about Special Day Actions for a job.
VSS$GET_SD_CLASS_INFO	Gets information about Special Day Classes. The user may request the instantiation of the Special Day Class for a certain year or request a list of all the years for which a class has days defined.
VSS$GET_SD_RESTRICTIONS	Gets information about the Special Day Restrictions for a job. User may request "Restrict_TO" list, "Restrict_FROM" list, and the instantiation of a job's Special Day Restrictions for a specified year. ItemLists are used as the request mechanism. If the caller does not specify any items, this routine returns whether the job has any Special Day Restrictions at all.
VSS$GET_START_TIME	Calculates the next time to run, given a starting time, a day-of-the-week mask, and a valid Job Management Manager schedule interval. Use VSS$GET_NEXT_START_TIME instead of VSS$GET_START_TIME for new development.
VSS$GET_SYNC_TIME	Gets the last known time that the job completed successfully (for remote job dependencies only.)
VSS$INSERT	Creates a new job in the Job Management Manager database. Supported only for compatibility with V1.1A and earlier; for new development use VSS$CREATE_ITEM.
VSS$MODIFY	Modifies an existing job in a local or remote Job Management Manager database. Supported only for Job Management Manager compatibility with V1.1A and earlier. Use VSS$MODIFY_ITEM for new development.
VSS$MODIFY_ITEM	Modifies an existing Job Management Manager job with the given parameter values. The routine takes an item list of new field values for the job. Use instead of VSS$MODIFY for new development.
VSS$MODIFY_SD_CLASS	Modifies an existing Special Day Class. Users may add and remove absolute days from the class.
VSS$NAME_TO_NUM	Returns the corresponding job number for a given job name and user name.
VSS$NO_DEPON	Sets the dependency override mask for a particular job. If the bit for a particular job dependency is set, then that job is treated as if it has completed successfully.
VSS$NUM_TO_NAME	Given a job number, returns the last completion status, current state, job name, and user name.

Routine	Description
VSS$PID_RUNNING	Checks whether a job created by the manager is currently running in the OpenVMS Cluster, given the process ID.
VSS$SCHED_LIST	Returns a list of all nodes that are running the manager in the local OpenVMS Cluster or a remote OpenVMS Cluster.
VSS$SCHED_RUNNING	Checks whether the manager is currently running in the local OpenVMS Cluster, either on a node you specify or, if not specified, on any node in the OpenVMS Cluster.
VSS$SCRIPT	Creates a DCL command file capable of recreating a set of existing Job Management Manager jobs. Jobs to be included are specified by: • Passing a set of restrictions on the job name, job type, job group, and job owner • Specifying whether to include Job Management Manager jobs which depend on previously selected jobs
VSS$SELECT	Returns in a longword array all the jobs that match a specified criteria, including job name, group, type, user name, status, and node.
VSS$SELECT_CLASS	Selects Special Day Classes and returns their names in an array. Criteria for selection is specified using an item list sent to the routine. Names of classes which satisfy all these criteria are returned.
VSS$SET	Requests that the manager perform an action on a job. The routine sets a database flag in the job record, then notifies the manager. Defined actions include run, hold, release, delete, and abort.
VSS$SET_BITMASK_DATE	Sets or clears a bit in a YEAR BITMASK structure. Takes a date specification and the address of a YEAR BITMASK structure. Caller specifies whether the bit should be set or cleared.
VSS$SET_RESTART	Sets the restart test value symbol for the job that calls this routine.
VSS$SET_RETRY	Sets the network retry interval or network retry expiration timeout for an OpenVMS Cluster.
VSS$SET_SD_ACTIONS	Sets the Special Day Action for a job.
VSS$SET_SD_RESTRICTIONS	Creates and defines Special Day Restrictions for a job or modifies existing Special Day Restrictions for a job. Users may add or remove Special Day Classes from the job's "Restrict_TO" or "Restrict_FROM" Special Day Restriction expressions. Users may also delete any existing Special Day Restrictions for the job. Item lists are used to accomplish any and all modifications in one call.
VSS$SHOW	Returns information about a single job from the Job Management Manager database. Supported only for compatibility with V1.1A and earlier. Use VSS$SHOW_ITEM for new development
VSS$SHOW_ITEM	Returns the specified fields for an existing Job Management Manager job. The routine takes an item list that specifies which

Routine	Description
	fields to return and contains the buffers in which to return them. Use instead of VSS$SHOW for new development.
VSS$SHOW_RETRY	Performs one of two actions, based on the value of the routine's opcode argument: Fills a buffer with printable strings for the network interval, network expiration, and network alias. Reads NSCHED$:RETRY.DAT and prints to SYS$OUTPUT detailed information about retry operations in progress.
VSS$SHOW_SCHED	Returns information about a specific local or remote manager instance in a fixed-length string.
VSS$SYNC_CHECK	Returns the number of local jobs that depend on a given job.
VSS$VALIDATE_TIME	Checks a *next_scheduled_time* or *interval_time* string for proper syntax. Also checks if the start time is earlier than the current time.

VSS$BITMASK_TO_DATE

VSS$BITMASK_TO_DATE — Takes a bitmask representing all the days of a year, and returns the days whose bits are set. The calling program can choose to receive this information as a null-terminated ASCII string or as a quadword. The calling program may have to make more than one call to get all requested information.

Format

VSS$BITMASK_TO_DATE ***bitmask_addr, item_list, context***

Arguments

bitmask_addr

OpenVMS Usage	YEAR_BITMASK
type	unsigned longword
access	read-only
mechanism	by reference

Address of the bitmask to translate. A pointer to a structure of type YEAR_BITMASK.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies in what format to return the date information. This argument is the address of an ITEM_BLOCK (ITEM_BLOCK_T) that requests a specific format for the return information and describes the buffers into which this routine should write the results. The array is terminated by a block whose first longword is 0 (zero).

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	Meaning
VSS$K_RETURN_ASCII	Specifies that the dates should be returned in null-terminated ASCII string format. The calling program must allocate a buffer to receive this information and pass the size and address of the buffer in the ITEM_BLOCK.
VSS$K_RETURN_QUADWORD	Specifies that the dates should be returned in quadword format. The calling program must allocate a buffer to receive this quadword and pass the size and address of the buffer in the ITEM BLOCK.

context

OpenVMS usage	array of unsigned longword
type	unsigned longword
access	read/write
mechanism	by reference

Context is a structure useful to the callable routine only. If the calling program needs to make more than one call to get all the information requested, context serves as a placeholder to this routine so it can pick up where it left off. Calling programs should *never* alter the contents of context.

The calling program sets up context as an array of 3 unsigned longwords that are initialized to zero. The address of this array is passed to the callable routine.

Note

The context array *must* be zeroed before the first call to the VSS$GET* routines and *must not* be modified between calls.

Possible Return Values

Return Value	Meaning
NSCHED$_INVBITMASK	Invalid YEAR_BITMASK specification
NSCHED$_INSUFFSIZE	Insufficient buffer size
NSCHED$_INVBUFFSPEC	Invalid buffer specification
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_MORE	Call has not finished processing the Bitmask. Bitmask may contain more values. Recall the routine.
NSCHED$_NOMORE	No more bitmasks to process. Check return buffer length for number of items returned.
NSCHED$_NOITEMLIST	No *item_list* specified

VSS$CHANGE_DEFAULT

VSS$CHANGE_DEFAULT — Requests that a particular manager instance become the default executor for the OpenVMS Cluster. The default manager instance runs all jobs that do not have

a particular node specified in the "db_cluster_node" field in the manager or if load balancing is enabled, tells the selected manager instance to run the job. This routine is called by the DCL command **SCHEDULE SET DEFAULT**.

Format

VSS$CHANGE_DEFAULT ***new_default_node, return_status***

Arguments

new_default_node

OpenVMS Usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name of the node that is to become the default agent for the OpenVMS Cluster.

return_status

OpenVMS Usage	cond_value
type	unsigned longword
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

1	Success
NSCHED$_NOTVAXCLUSTER	Not an OpenVMS Cluster
NSCHED$_NOSCHED	No manager instance is available to service the request
System service error code	Error code returned by system services

Possible Return Values

This is a subroutine. The status of the call is returned in the *return_status* variable.

VSS$CHECK_BITMASK_DATE

VSS$CHECK_BITMASK_DATE — Checks to see if certain bits are set in the YEAR_BITMASK. Takes as input a date specification and the address of a YEAR_BITMASK structure to check.

Format

VSS$CHECK_BITMASK_DATE ***bitmask_addr, item_list***

Arguments

bitmask_addr

OpenVMS usage	pointer to YEAR_BITMASK
type	unsigned longword
access	read-only
mechanism	by reference

Address of the bitmask to check. A pointer to a structure of type YEAR_BITMASK.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument is the address of an ITEM_BLOCK structure that says what format the input date is in, describes the buffer containing the input date and specifies a buffer for the return code. Only one ITEM_BLOCK is passed in for this call.

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	**Meaning**
VSS$K_ASCII_INPUT	Specifies the input date is in ASCII format.
VSS$K_BINARY_INPUT	Specifies the input date is in binary format.

The following is a description of Buffer Length and Buffer Address used in the ITEM_BLOCK:

Item Code	**Meaning**
VSS$K_ASCII_INPUT	The calling program must allocate and set up a buffer containing an ASCII Date Specifier Data Structure. "Buffer Address" is the address of a buffer containing an ASCII Date Specifier Data Structure. "Buffer Length" is not used.
VSS$K_BINARY_INPUT	The calling program must allocate and set up a buffer containing a binary representation of a date specification. Buffer Address is the address of a time quadword containing the binary date. "Buffer Length" is not used.

The following is a description of "Return Length Address" used in the ITEM_BLOCK:

Return Length Address (write only) is an address of a longword to receive the function result:

- 1 - if the bit in the passed Bitmask corresponding to the passed Date is set.
- 0 - if the bit is clear.

Possible Return Values

Return Value	Meaning
NSCHED$_SUCCESS	Successful completion
NSCHED$_INVARG	Invalid argument address in ITEM_BLOCK
NSCHED$_INVBITMASK	Invalid YEAR_BITMASK specification
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_NOITEMLIST	No *item_list* specified
NSCHED$_YEARMISMATCH	Year of input date does not match year of input YEAR_BITMASK

VSS$CHECK_PRIV

VSS$CHECK_PRIV — Checks the calling program's access privilege to the given job.

Format

VSS$CHECK_PRIV ***job_name, user_name, server_node, access_type***

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name or number (decimal string) of the job for which you want information returned. The maximum length is 40 characters. Wildcard characters are not allowed. If the *job_name* is not numeric, you must specify the *user_name*.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

If the *job_name* is not a numeric string, the *user_name* field is used to discriminate between identically named jobs owned by different users. If the *job_name* is a numeric string, then the *user_name* field is ignored.If the *job_name* is not a numeric string and the *user_name* is blank, then the *user_name* defaults to the calling program's *user_name*. The maximum length for a *user_name* is 32 characters. Wildcard characters are not allowed.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Specifies the remote node on which the job resides. If the job is local, then pass a blank string ("").

access_type

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The value should be "READ", "WRITE", or "EXECUTE".

Possible Return Values

Return Value	Meaning
1	Success; caller has sufficient privilege to perform the action.
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_NODATABASE	VSI JM Manager database was not found
Negative of BASIC error code	See the VSI BASIC documentation

VSS$CREATE_ITEM

VSS$CREATE_ITEM — Creates a new Job Management Manager job with the given parameter values. The routine takes an item list of values to apply to the new job. Unspecified items take default parameters. The routine requires that an OpenVMS command for the job be provided in the item list. Use this routine to create a job; the routine VSS$INSERT is supported only for compatibility with Version 1.1 and earlier.

Format

VSS$CREATE_ITEM ***new_tag, hold_job, server_node, number_of_items, item_list***

Arguments

new_tag

OpenVMS usage	integer
type	unsigned longword
access	write-only

mechanism	by reference

The routine stores the new job number in this parameter.

hold_job

OpenVMS usage	Boolean
type	longword
access	read-only
mechanism	by value

A Boolean flag indicating whether the new job should be put on hold.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The VSI JM Manager *server_node* on which to create the job. An empty string specifies the local server.

number_of_items

OpenVMS usage	integer
type	unsigned longword
access	read-only
mechanism	by value

The number of items in the *item_list*.

item_list

OpenVMS usage	pointer to array of item blocks
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which fields are being provided for the new job. This argument is the address of an array of item blocks (*job_item*), each containing a valid item code and a descriptor containing the value for the field.

The item codes are described in the nsched_job_fields.h file in NSCHED$. The valid item codes for VSS$CREATE_ITEM are:

- NSCHED_RECORD_FLAGS

- NSCHED_LAST_STATUS
- NSCHED_SYNC_JOB_NUMS
- NSCHED_PRE_STATUS
- NSCHED_POST_STATUS
- NSCHED_JPRIORITY
- NSCHED_QPRIORITY
- NSCHED_TIMEOUTNSCHED_START_TIME
- NSCHED_SYNC_TIME
- NSCHED_SYNC_NODES
- NSCHED_PRE_FUNCTION
- NSCHED_POST_FUNCTION
- NSCHED_SCHED_INTERVAL
- NSCHED_NOSTART_TIME
- NSCHED_DOW
- NSCHED_USERNAME
- NSCHED_JOBNAME
- NSCHED_COMMENT
- NSCHED_LOGFILE
- NSCHED_OPENVMS_COMMAND
- NSCHED_MAIL_TO
- NSCHED_CLUSTER_NODE
- NSCHED_GROUP
- NSCHED_TYPE
- NSCHED_NO_DEPON
- NSCHED_RERUN_FREQUENCY
- NSCHED_RERUN_ATTEMPTS
- NSCHED_RDID
- NSCHED_WRID
- NSCHED_EXID

- NSCHED_STALL_JOB
- NSCHED_TIMEOUT_JOB
- NSCHED_EXEC_TYPE
- NSCHED_BATCH_QUEUE
- NSCHED_BATCH_CPULIM

The following table describes field indicates for the callable routines VSS$SHOW_ITEM, VSS$CREATE_ITEM, and VSS$MODIFY_ITEM. The buffer type column specifies the kind of data stored in the *item_list* string descriptor. When two types are specified (for example, String[23]/Quadword), the first is the type returned by VSS$SHOW_ITEM and the second is the type used by VSS$MODIFY_ITEM.

Field	**Buffer Type**	**Update Using VSS$MODIFY_ITEM**
0 NSCHED_TAG (job number)	Long integer	No
1 NSCHED_RECORD_FLAGS	Long integer	Yes
2 NSCHED_LAST_STATUS	Long integer	Yes
3 NSCHED_SUCCESS_COUNT	Long integer	No
4 NSCHED_FAILURE_COUNT	Long integer	No
5 NSCHED_PID	Long integer	No
6 NSCHED_UIC	String[32]	No
7 NSCHED_SYNC_JOB_NUMS	Long integer[16]	Yes
8 NSCHED_PRE_STATUS	Long integer	Yes
9 NSCHED_POST_STATUS	Long integer	Yes
10 NSCHED_JPRIORITY	Word integer	Yes
11 NSCHED_QPRIORITY	Word integer	Yes
12 NSCHED_LAST_START	String[23]	No
13 NSCHED_LAST_END	String[23]	No
14 NSCHED_TIMEOUT	String[23]/Quadword	Yes
15 NSCHED_START_TIME	String[23]	Yes
16 NSCHED_SYNC_TIME	String[23]/Quadword	Yes
17 NSCHED_SYNC_NODES	String[16][6]	Yes
18 NSCHED_STATUS_FLAG	String[1]	No
19 NSCHED_REQUEST_FLAG	String[1]	No
20 NSCHED_RESTART_PARM	String[40]	No
21 NSCHED_PRE_FUNCTION	String[40]	Yes
22 NSCHED_POST_FUNCTION	String[40]	Yes
23 NSCHED_SCHED_INTERVAL	String[23]	Yes
24 NSCHED_NOSTART_TIME	String[23]/Quadword	Yes
25 NSCHED_DOW	String[7]	Yes

Field	Buffer Type	Update Using VSS$MODIFY_ITEM
26 NSCHED_USERNAME	String[32]	Yes
27 NSCHED_JOBNAME	String[40]	Yes
28 NSCHED_COMMENT	String[80]	Yes
29 NSCHED_LOGFILE	String[50]	Yes
30 NSCHED_OPENVMS_COMMAND	String[132]	Yes
31 NSCHED_MAIL_TO	String[50]	Yes
32 NSCHED_CLUSTER_NODE	String[6]	Yes
33 NSCHED_GROUP	String[40]	Yes
34 NSCHED_TYPE	String[40]	Yes
35 NSCHED_FISCAL	String[16]	No
36 NSCHED_CURRENT_NODE	String[6]	No
37 NSCHED_NO_DEPON	Long integer	Yes
38 NSCHED_RERUN_FREQUENCY	Word integer	Yes
39 NSCHED_RERUN_ATTEMPTS	Word integer	Yes
40 NSCHED_RERUN_COUNT	Word integer	No
41 NSCHED_RDID	String[32]	Yes
42 NSCHED_WRID	String[32]	Yes
43 NSCHED_EXID	String[32]	Yes
44 NSCHED_STALL_JOB	Long integer	Yes
45 NSCHED_TIMEOUT_JOB	Long integer	Yes
46 NSCHED_EXEC_TYPE	String[1]	Yes
47 NSCHED_REMOTE_SPECIFIC	String[72]	No
48 NSCHED_BATCH_TIME	String[23]	No
49 NSCHED_BATCH_QUEUE	String[31]	Yes
50 NSCHED_BATCH_ENTRY	Long integer	No
51 NSCHED_BATCH_CPULIM	Long integer	Yes
52 NSCHED_BATCH_ CHARACTER	Long integer[4]	No

The following table contains notes on the fields in the previous table:

Field	Notes
NSCHED_TAG	Specifies the job number
NSCHED_RECORD_FLAGS	See VSSDEFS.H for the RecordFlags structure
NSCHED_UIC	Updated automatically when NSCHED_USERNAME changes
NSCHED_SYNC_JOB_NUMS	Cannot be updated without NSCHED_SYNC_NODES Updates NSCHED_SYNC_TIME when changed.
NSCHED_JPRIORITY	Must be between 0 and 16
NSCHED_QPRIORITY	Must be between 0 and 255
NSCHED_TIMEOUT	Show returns a String[23], modify requires a Quadword

Field	Notes
NSCHED_START_TIME	Can return NSCHED$_TIMBEFOR when modified; can affect NSCHED_FISCAL when modified
NSCHED_SYNC_TIME	Show returns a String[23], modify requires a Quadword Automatically set to NOW if synch jobs changed Cannot be 0
NSCHED_SYNC_NODES	Cannot be updated without NSCHED_SYNC_JOB_NUMS. Updates NSCHED_SYNC_TIME when changed
NSCHED_NOSTART_TIME	Show returns a String[23], modify requires a Quadword
NSCHED_DOW	A character string of 0's and 1's
NSCHED_USERNAME	Updates NSCHED_UIC when changed.Cannot be modified to specify a jobname/username pair which already exists.
NSCHED_JOBNAME	Cannot be modified to specify a jobname/username pair which already exists
NSCHED_LOGFILE	A blank modification string is replaced by "NL:"
NSCHED_OPENVMS_COMMAND	Cannot be changed to be blank
NSCHED_FISCAL	Automatically updated if NSCHED_START_TIME changes
NSCHED_RERUN_FREQUENCY	Automatically updated when the retry bit is set
NSCHED_RERUN_ATTEMPTS	Automatically updated when the retry bit is set
NSCHED_RERUN_COUNT	Automatically updated when the retry bit is set
NSCHED_WRID	Modification requires CMKRNL privilege
NSCHED_EXID	Modification requires SYSPRV privilege
NSCHED_STALL_JOB	Modification requires WRITE access to the SJOB or a caller holding the SJOB's EXECUTE id
NSCHED_TIMEOUT_JOB	Modification requires either the WRITE access to the TJOB or a caller holding the TJOB's EXECUTE id TJOB cannot be the same as the current job.
NSCHED_EXEC_TYPE	Valid values are ' ', 'R', and 'B' updates some NSCHED_BATCH_* fields when changed
NSCHED_REMOTE_SPECIFIC	Valid only with NSCHED_EXEC_TYPE= 'R'
NSCHED_BATCH_TIME	Valid only with NSCHED_EXEC_TYPE = 'B' VSI JM Manager

Possible Return Values

Return Value	Meaning
NSCHED$_SUCCESS	Operation completed successfully
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_INVARG	Routine was called with an invalid string argument
NSCHED$_BADITEM	Bad Item Code specified to callable routine
NSCHED$_BADVALUE	Bad *item_list* value specified
NSCHED$_FLDTOOLONG	Specified field is too long
NSCHED$_NOSUCHID	Unknown rights identifier
NSCHED$_NOSAMEJOB	Timeout Job must not be same job

Return Value	Meaning
NSCHED$_IDOUTOFRNG	Identifier value is out of range
NSCHED$_TIMBEFOR	Start time is before current time; job will run immediately
NSCHED$_DUPLNAM	Duplicate job name is not permitted
NSCHED$_ADJFLDVAL	Field value adjusted to be within limits
NSCHED$_WLDNAME	Invalid wildcard operation
NSCHED$_ONEDEPFLD	Dependency nodes and numbers cannot be updated separately
NSCHED$_INVMODEVAL	Invalid mode value; value must be DETACHED, BATCH, or REMOTE
NSCHED$_CNTMODFLD	Caller cannot modify this field
NSCHED$_FLDNOTSUPP	Field not supported in current version
NSCHED$_FLDINCTYPE	Field inconsistent with Job type

The function can also return various system service and RMS codes, and negated BASIC error codes.

VSS$CREATE_SD_CLASS

VSS$CREATE_SD_CLASS — Creates and defines a Special Day Class. Calling programs can define an empty Special Day Class or provide an item list describing elements to be inserted into this new class. Only users with SYSPRV or OPER privileges can create Special Day Classes.

Format

VSS$CREATE_SD_CLASS *class_name, reserved, item_list*

Arguments

class_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This is the name of the Special Day Class to be created. The maximum length allowed is 40 characters.

Reserved

OpenVMS usage	null_arg
type	unsigned longword
access	read-only
mechanism	by value

This argument is a placeholder reserved for future use. Argument should be null.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which elements are to be inserted into the Special Day Class. The *item_list* argument is the address of an array of ITEM_BLOCKS, each describing an element to be inserted into the Special Day Class. The array is terminated by a block whose first longword is 0 (zero).

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	Meaning
VSS$K_INSERT_ABSOLUTE_DAYS	Specifies that the indicated absolute days should be inserted into the class's absolute day list. The absolute days are represented by a YEAR_BITMASK structure that is allocated and filled in by the calling program. The size and address of this structure are passed in the ITEM_BLOCK.
VSS$K_REPLACE_ABSOLUTE_DAYS	Specifies that the indicated absolute days should replace the class's current absolute day list. The absolute days are represented by a YEAR_BITMASK structure that is allocated and filled in by the calling program. The size and address of this structure are passed in the ITEM_BLOCK.
VSS$K_INSERT_EXCLUDE_CLASS	Specifies that the indicated Special Day Class should be inserted into the class's EXCLUDE_CLASS list. The class name is represented as an SDCLASS_DESCRIPTOR structure that is allocated and filled in by the calling program. The size and address of the structure are passed in the ITEM_BLOCK.

Note

The year bitmask indicated must have the year field specified to tell VSI JM Manager what year to replace. If this field is null, an error occurs.

Possible Return Values

Return Value	Meaning
NSCHED$_CLASSCREATED	Special Day Class created
NSCHED$_DUPLCLASS	Duplicate Special Day Class Name
NSCHED$_INVCHARFND	Invalid character found in Special Day Class Name
NSCHED$_LEAD-NUMNOTALL	Leading number not allowed in Special Day Class name
NSCHED$_NAMTOOLONG	Special Day Class name exceeds maximum length

Return Value	Meaning
UNKNOWN_USER_NAME	The *user_name* is unknown
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_SUCCESS	Operation completed successfully

VSS$DELETE_SD_CLASS

VSS$DELETE_SD_CLASS — Deletes a Special Day Class from the manager's Special Days database. Only users with SYSPRV or OPER privileges can delete Special Day Classes. It is possible that a Special Day Class is being used as a restriction on a job at the time the DELETE is requested. To avoid any unintentional side effects, the calling program can specify (through use of the FLAG argument) whether a class should be deleted if it has existing job restrictions dependencies. If the calling program chooses to delete the class even though it has existing dependencies, the references to the Special Day Class are deleted from the job's Special Day Restrictions as well.

Format

VSS$DELETE_SD_CLASS *class_name, reserved, flag*

Arguments

class_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This is the name of the Special Day Class to be deleted. The maximum length allowed is 40 characters.

Reserved

OpenVMS usage	null_arg
type	unsigned longword
access	read-only
mechanism	by value

This is a placeholder argument reserved for future use. This argument should be null.

Flag

OpenVMS usage	mask_longword
type	unsigned longword
access	read-only
mechanism	by reference

Longword bitmask instructing DELETE_SD_CLASS how to perform the deletion of Special Day Classes. The flag M_NODELETE_IF_JOBDEP is used to delete the Special Day Class if and only if

there are no jobs currently using this Special Day Class as a Job Special Day Restriction. If this bit is not set, the Special Day Class will be deleted and any references to this class as job restrictions will be removed.

Possible Return Values

Return Value	Meaning
NSCHED$_CLASSDELETED	Special Day Class deleted
NSCHED$_INVCHARFND	Invalid character found in Special Day Class Name
NSCHED$_JOBDEPSEXIST	Job dependencies exist for Special Day Class
NSCHED$_LEAD-NUMNOTALL	Leading number not allowed in Special Day Class name
NSCHED$_NAMTOOLONG	Special Day Class name exceeds maximum length
NSCHED$_NOPRIV	No privilege for attempted operation
NSCHED$_NOSUCHCLASS	Special Day Class does not exist
NSCHED$_SUCCESS	Operation completed successfully

VSS$FIND_DEPS

VSS$FIND_DEPS — Searches the job database on the local node or the remote OpenVMS Cluster containing *server_node*, and returns an array of longwords containing the job numbers of all jobs that depend on *job_node::job_number* (see below). If there are more jobs than can fit in the array, the routine can be called again to get remaining jobs.

Format

VSS$FIND_DEPS *job_number, job_node, tag_array(), array_size, context(), server_node, return_status*

Arguments

job_number

OpenVMS usage	longword_signed
type	longword integer (signed)
access	read-only
mechanism	by reference

The number of the job in which you are interested.

job_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The nodename of the job in which you are interested. A blank value indicates that the job is on the local node.

tag_array()

OpenVMS usage	vector_longword_signed
type	longword integer (signed)
access	write-only
mechanism	by descriptor Class A

The lowest element of the array receives the number of subsequent array elements that are filled in. The remaining elements receive job numbers for the jobs that depend on the *job_node::job_number* in the local cluster or in the OpenVMS Cluster containing *server_node* (if a *server_node* is specified).

The routine automatically calculates the lower bound of the array. The upper bound must be equal to "(array_size -- 1) + lower bound". For example, if the lower bound is 1 and the *array_size* is 50, then the upper bound is assumed to be 50.

array_size

OpenVMS usage	longword_unsigned
type	unsigned longword
access	read-only
mechanism	by reference

The number of elements in array *tag_array()*. Since the lowest element of *tag_array* receives the number of elements being returned, the most job numbers that can be filled in by each call to the routine is (array_size -- 1).

context()

OpenVMS usage	vector_longword_unsigned
type	unsigned longword
access	read/write
mechanism	by descriptor Class A

Must be initialized to 0,0 for the first call. After that, the routine can be called again if an NSCHED$_MORE status is returned, to obtain more jobs.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Specifies the remote node of the job database to search. If you wish to search the local database, specify a blank string ("").

return_status

OpenVMS usage	cond_value

type	longword integer (signed)
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

NSCHED$_NOMORE	No more items or jobs found
NSCHED$_MORE	Additional items or jobs remain
NSCHED$_NODATABASE	Job Management Manager database was not found
Negative of BASIC error code	See the VSI BASIC documentation
System service error code	Error code returned by system services
RPC or DECnet error code	An error occurred connecting to a remote node

Possible Return Values

This is a subroutine. The status of the call is returned in the *`return_status`* variable.

VSS$GET_DEPNODES

VSS$GET_DEPNODES — Gets the list of nodes with jobs that depend on the local job number passed in the argument to this call.

Format

`VSS$GET_DEPNODES` ***`job_number, number_depnodes, list_of_depnodes`***

Arguments

`job_number`

OpenVMS usage	longword_signed
type	longword integer (signed)
access	read-only
mechanism	by reference

The job number of the local job on which other jobs on the remote nodes depend.

`number_depnodes`

OpenVMS usage	longword_unsigned
type	unsigned longword
access	write-only
mechanism	by reference

The number of nodes with jobs that depend on the specified job.

list_of_depnodes

OpenVMS usage	char_string
type	character string
access	write-only
mechanism	by descriptor

The list of nodes with jobs that depend on the job passed in the *job_number* argument. Each nodename is six characters. Nodenames with less than six characters are padded with spaces.

All nodes are passed back in the string, so the string length needs to be at least (6* num_of_DEPNODES). If the passed string is too short, the information is truncated and ignored.

Possible Return Values

Return Value	Meaning
1	Success
NSCHED$_NOSYNCHDAT	The synchronization database is corrupted or missing
LIB$_STRTRU	There are more nodes than could fit in the string buffer passed by the caller
Negative of BASIC error code	See the VSI BASIC documentation

VSS$GET_HISTORY

VSS$GET_HISTORY — Returns a summary of minimum, maximum, average, and current (if the job is currently running) resource usage for all records of a job, from data in the history log. The following resources are calculated: Elapsed time, CPU time, Page faults, Direct I/O, Buffered I/O.

Format

VSS$GET_HISTORY ***job_number, op_mask, number_records, server_node, filename, ret_info***

Arguments

job_number

OpenVMS usage	longword
type	longword integer (signed)
access	read-only
mechanism	by reference

A longword for the job number for which you want the history.

op_mask

OpenVMS usage	longword
type	unsigned longword

access	read-only
mechanism	by reference

A bitmask that tells *get_history* which operation to perform.

Bit	Function
1	Include Success status records in statistics
2	Include Failure status records in statistics
4	Return times as seconds (else ddd:hh:mm:ss.hh)
8	Get current information only, no history
16	Summarize history from old log file

number_records

OpenVMS usage	longword
type	longword integer (signed)
access	write-only
mechanism	by reference

The number of history records found for that operation.

server_node

OpenVMS usage	string
type	character string
access	read-only
mechanism	by descriptor

A 6-character string that specifies a remote node from which to get information. If you want information about a local node, pass a blank string.

Filename

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid file specification; this is the file that contains the history information. If not specified, it defaults to NSCHED$:VERMONT_CREAMERY.LOG.

ret_info

OpenVMS usage	string
type	character string
access	write-only
mechanism	by descriptor

An array of 17-character strings. It returns a string that contains the elapsed time (OpenVMS time), the CPU time (OpenVMS time), page faults, direct I/O count, and buffered I/O count. At the end, three 23-character strings are added. These strings contain the earliest login time, last completion time, and current login time if a process is running.

Possible Return Values

Return Value	Meaning
NSCHED$NOHISTORY	No history records were found for the specified job
System service error code	Error code returned by system services

VSS$GET_HIST_REC

VSS$GET_HIST_REC — Returns accounting information from the history log file about a single job run. Similar to VSS$GET_HISTORY, which returns a summary of accounting information from the history file for all runs of a job.

Format

VSS$GET_HIST_REC *job_number, op_mask, context(), server_node, filename, ret_info*

Arguments

job_number

OpenVMS usage	longword
type	longword integer (signed)
access	read-only
mechanism	by reference

A longword for the job number for which you want the history.

op_mask

OpenVMS usage	longword
type	unsigned longword
access	read-only
mechanism	by reference

A bitmask that tells *get_hist_rec* which operation to perform, as follows:

Bit	Meaning
1	Include success status records in statistics
2	Include failure status records in statistics
4	Return times as seconds (else, ddd:hh:mm:ss.hh)

context()

OpenVMS usage	vector unsigned longword
type	unsigned longword
access	write
mechanism	descriptor Class A

8 bytes that contain the RFA of the last record for which information was returned. It should be initialized to 0 by the caller on the first call.

server_node

OpenVMS usage	string
type	character string
access	read-only
mechanism	by descriptor

A 6-character string that specifies a remote node from which to get information. If you want information about a local node, pass a blank string.

Filename

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid file specification; this is the file that contains the history information. If not specified, it defaults to NSCHED$:VERMONT_CREAMERY.LOG.

ret_info

OpenVMS usage	string
type	character string
access	write-only
mechanism	by descriptor

An array of character strings. It returns a string that contains the elapsed time (OpenVMS time), CPU time (OpenVMS time), page faults, direct I/O count, buffered I/O count, peak working set size (17 bytes each), login and job termination times (23 bytes each) and process ID (8 byte hex string).

If no history data is available, values are returned as spaces.

Possible Return Values

Return Value	**Meaning**
1	Success
NSCHED$_NOMORE	No more history records found
NSCHED$_NOHISTORY	No history database found
System service error code	Error code returned by system services

VSS$GET_NEXT_START_TIME

VSS$GET_NEXT_START_TIME — Calculates the next time to run, given a starting point, a valid interval, and the day-of-the-week mask.

Format

VSS$GET_NEXT_START_TIME *job_number, start_time, sched_interval, DOW_mask, job_record_flags, next_start_time, return_status*

Arguments

job_number

OpenVMS usage	longword
type	unsigned longword
access	read-only
mechanism	by reference

Number of job for which to calculate the next scheduled run time.

start_time

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by reference

An OpenVMS date expressed in the format "dd-mmm-yyyy hh:mm:ss.cc". The argument must have a full time specification (all 23 characters) with values provided for all fields, for example: 23-Mar-2025 03:35:56.00. A blank string ("") indicates that the current system time should be used.

sched_interval

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid VSI JM Manager schedule interval. This interval string must pass the VSS$VALIDATE_TIME test for intervals.

DOW_mask

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The day-of-the-week mask, expressed as a 7-character binary string of 1s and 0s representing the days of the week (Monday through Sunday) on which the job can run. For example, setting *DOW_mask* to 1000000 will indicate that the job can only run on Mondays.

job_record_flags

OpenVMS usage	longword
type	unsigned longword
access	read/write
mechanism	by reference

The job's RECORD_FLAGS field from the job record. Use the VSS$SHOW_ITEM call to retrieve the job's RECORD flags from the job record. Do not modify this field.

VSS$GET_NEXT_START_TIME may modify this field so the calling program should be sure to write this field back to the database after this call.

next_start_time

OpenVMS usage	char_string
type	character string
access	write-only
mechanism	by descriptor

The next start time for the job as calculated by VSS$GET_NEXT_START_TIME. This is a 23-character string that will contain either an OpenVMS time string in the format "dd-mmm-yyyy hh:mm:ss.cc" or the string "NEVER".

return_status

OpenVMS usage	longword
type	unsigned longword
access	write-only
mechanism	by reference

This field is used to return additional status to the calling program. This field is valid only when the return status of this routine is NSCHED$_SUCCESS. In this event, the *return_status* field will have one of the following return values:

- NSCHED$_NOTRESTRICTED
- NSCHED$_NSRTUNRESOLVED
- NSCHED$_SPECDAYHOLD
- NSCHED$_SPECDAYSKIP

Possible Return Values

Return Value	**Meaning**
1	Success

Return Value	Meaning
NSCHED$_INVSTRTIME	Invalid time string
NSCHED$_NOTRESTRICTED	Input date is not Special Day restricted
NSCHED$_NSRTUNRESOLVED	Next Scheduled Run Time for job could not be resolved
NSCHED$_SPECDAYSKIP	Job was rescheduled due to Special Day Restrictions
NSCHED$_SPECDAYHOLD	Job is in HOLD state due to Special Day Restrictions
Error codes from VSS$VALIDATE_TIME	See VSS$VALIDATE_TIME
Negative of BASIC error code	See the VSI BASIC documentation
System service error code	Error code returned by system services

VSS$GET_SD_ACTIONS

VSS$GET_SD_ACTIONS — Gets information about Special Day Actions for a job.

Format

VSS$GET_SD_ACTIONS *job_name, user_name, item_list*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name or number (as a decimal string) of the job to get Special Day Restriction information about. The maximum length allowed is 40 characters. Wildcard characters are not allowed. If *job_name* is not numeric, the calling program must specify *user_name*.

user_name

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which information is to be returned about the Special Day Actions for the job. The *item_list* argument is the address of an OpenVMS array of ITEM_BLOCKS, each of which requests an item of information and describes the buffers into which this routine should write the results. The array is terminated by a block whose first longword is 0 (zero).

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	Meaning
VSS$K_GET_ACTION	Specifies that the job's default Special Day Action should be returned. The value is returned in a longword that is allocated by the

Item Code	Meaning
	calling program and whose length and address are passed into the callable routine in the ITEM BLOCK.

Possible Return Values

Return Value	Meaning
NSCHED$_INVBUFFSPEC	Invalid buffer specification
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_NOITEMLIST	No *item_list* specified
NSCHED$_NOPRIV	No privilege for attempted operation
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_SUCCESS	Operation completed successfully

VSS$GET_SD_CLASS_INFO

VSS$GET_SD_CLASS_INFO — Gets information about Special Day Classes. The calling program can request the instantiation of the Special Day Class for a certain year or request a list of all the years for which a class has days defined. By using the *item_list* argument, a calling program can request any and all pieces of information in one call. If the calling program does not use the *item_list* argument, this routine returns only whether the Special Day Class currently exists. If the calling program requests many pieces of information in one call (multiple ITEM_BLOCKS), all the requested information may not be found. If the manager does not locate a requested piece of information, it writes a zero into the return length field specified in the item block. When requesting multiple pieces of information, the calling program should check the return length to determine which pieces of information were returned. The calling program may need to make more than one call to this routine to get all the information about a specified class. The number of calls needed depends on the size of the buffer the calling program provides to receive the information in relation to the amount of information to be passed.

Format

VSS$GET_SD_CLASS_INFO *class_name, reserved, item_list, context*

Arguments

class_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This is the name of the Special Day Class to get information about. The maximum length allowed is 40 characters.

reserved

OpenVMS usage	null_arg

type	unsigned longword
access	read-only
mechanism	by value

This is a place holding argument reserved for future use. This argument should be null.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by referenc

The *item_list* argument specifies which information is to be returned about the Special Day Class. This argument is the address of an OpenVMS array of ITEM_BLOCKS, each of which requests an item of information and describes the buffers into which this routine should write the results. The array is terminated by a block whose first longword is 0 (zero).

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	**Meaning**
VSS$K_GET_BITMASK_YEARS	Specifies that the years for which the class has days defined should be returned. The information is passed back via an array of unsigned longwords that is allocated by the calling program. The address and size of this array are passed in the ITEM_BLOCK. The calling program may have to make more than one call to VSS$GET_SD_CLASS_INFO to receive all the requested information.
VSS$K_GET_INSTANTIATION	Specifies that the include days for the Special Day Class are to be returned. This information is returned in YEAR_BITMASK form. The YEAR_BITMASK is allocated by the calling program and the size and address of the bitmask are passed into the callable routine in the ITEM_BLOCK.

Note

The year bitmask indicated must have the year field specified to tell VSI JM Manager for what year to get the instantiation. If this field is null, an error occurs.

context

OpenVMS usage	array of unsigned longwords
type	unsigned longword
access	read/write
mechanism	by descriptor

Context is a structure useful to the callable routine only. If the calling program needs to make more than one call to get all the information requested, context serves as a placeholder to this routine so it can pick up where it left off. Calling programs should never alter the contents of context.

The calling program sets up context as an array of 3 unsigned longwords that are initialized to zero. The address of this array is passed to the callable routine.

Note

The context array must be zeroed before the first call to the VSS$GET routine, and must not be modified between calls.

Possible Return Values

Return Value	Meaning
NSCHED$_CLASSEXISTS	Special Day Class exists
NSCHED$_EMPTYCLASS	No days currently specified in Special Day Class
NSCHED$_INVBITMASK	Invalid YEAR_BITMASK specification
NSCHED$_INVBUFFSPEC	Invalid buffer specification
NSCHED$_INVCHARFND	Invalid character found in Special Day Class Name
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_LEAD-NUMNOTALL	Leading number not allowed in Special Day Class name
NSCHED$_MORE	Additional Special Day Classes remain
NSCHED$_NAMTOOLONG	Special Day Class name exceeds maximum length
NSCHED$_NOMORE	No more Special Day Classes found
NSCHED$_NOSDTHATYEAR	No days in Special Day Class for the specified year
NSCHED$_NOSUCHCLASS	Special Day Class does not exist
NSCHED$_SUCCESS	Operation completed successfully

VSS$GET_SD_RESTRICTIONS

VSS$GET_SD_RESTRICTIONS — Gets information about the Special Day Restrictions for a job. The calling program can request a "Restrict_TO" list, "RESTRICT_FROM" list, and the instantiation of a job's Special Day Restrictions for a specified year. The *item_lists* argument is used as the request mechanism. If the calling program does not specify an *item_list*, this routine returns whether the job has any Special Day Restrictions. If the calling program requests many pieces of information in one call (multiple ITEM_BLOCKS), all the requested information may not be found. If the manager does not locate a requested piece of information, it writes a zero into the return length field specified in the item block. When requesting multiple pieces of information the calling program should check the return length to determine which pieces of information were returned. The calling program may need to make more than one call to this routine to get all the information about a job's Special Day Restrictions. The number of calls needed depends on the size of the buffer the calling program provides to receive the information in relation to the amount of information to be passed.

Format

VSS$GET_SD_RESTRICTIONS *job_name, user_name, item_list, context*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name or number (as a decimal string) of the job about which to get Special Day Restriction information. Maximum length allowed is 40 characters. Wildcard characters are not allowed. If *job_name* is not numeric, the calling program must specify *user_name*.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

If the *job_name* is not a numeric string, the *user_name* field is used to discriminate between identically named jobs owned by different users. If the *job_name* is a numeric string, then the *user_name* field is ignored. Maximum length allowed is 32 characters. Wildcard characters are not allowed.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which information is to be returned about the Special Day Restrictions for the job. The *item_list* argument is the address of an array of ITEM_BLOCKS each of which requests an item of information and describes the buffers into which this routine should write the results. The array is terminated by a block whose first longword is 0 (zero).

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	**Meaning**
VSS$K_GET_BITMASK_YEARS	Specifies that the years for which the job has restrictions defined should be returned. The information is passed back via an array of unsigned longwords that is allocated by the calling program. The address and size of this array are passed in the ITEM_BLOCK. The calling program may have to make more than one call to VSS$GET_SD_RESTRICTIONS to receive all the requested information.
VSS$K_GET_RESTRICTION_MASK	Specifies that the mask of Special Day Restrictions for a specified year is to be returned. This information is returned in YEAR_BITMASK form. The YEAR_BITMASK is allocated by the calling program, and the size and address of the bitmask is passed into the callable routine in the ITEM_BLOCK.

Item Code	Meaning
VSS$K_GET_RESTRICT_TO	Specifies that the Special Day Classes that make up the job's Special Days RESTRICT_TO expression are to be returned. The class name is passed back in an SDCLASS_DESCRIPTOR structure that is allocated by the calling program. The size and address of the structure are passed in the ITEM_BLOCK. The calling program may have to make more than one call to VSS$GET_SD_RESTRICTIONS to receive all the requested information.
VSS$K_GET_RESTRICT_FROM	Specifies that the Special Day Classes that make up the job's Special Days RESTRICT_FROM expression are to be returned. The class name is passed back in an SDCLASS_DESCRIPTOR structure that is allocated by the calling program. The size and address of the structure are passed in the ITEM_BLOCK. The calling program may have to make more than one call to VSS$GET_SD_RESTRICTIONS to receive all the requested information.

Note

The year bitmask indicated must have the year field specified to tell VSI JM Manager for what year to get the restriction mask. If this field is null, an error occurs.

context

OpenVMS usage	array of unsigned longwords
type	unsigned longword
access	read-only
mechanism	by reference

Context is a structure useful to the callable routine only. If the calling program needs to make more than one call to get all the information requested, context serves as a placeholder to this routine so it can pick up where it left off. Calling programs should never alter the contents of context. The calling program sets up context as an array of 3 unsigned longwords that are initialized to zero. The address of this array is passed to the callable routine.

Note

The context array must be zeroed before the first call to the VSS$GET routine, and must not be modified in between calls.

Possible Return Values

Return Value	Meaning
NSCHED$_INVBUFFSPEC	Invalid buffer specification
NSCHED$_INVBITMASK	Invalid YEAR_BITMASK specification
NSCHED$_INVSUFFSIZE	Insufficient buffer size

Return Value	Meaning
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_MORE	Additional Special Day Restrictions remain
NSCHED$_NOMORE	No more Special Day Restrictions found
NSCHED$_NORESTMASKS	No Special Day Restriction YEAR_BITMASK found for job
NSCHED$_NORESTRICT	No Special Day Restrictions for job
NSCHED$_NORESTTHATYEAR	No Special Day Restrictions that year for job
NSCHED$_RESFRMNOTFND	Special Days Restrict-From expression not found
NSCHED$_RESTONOTFND	Special Days Restrict-To expression not found
NSCHED$_RESTRICTEXISTS	Special Days Restrictions exist for job

VSS$GET_START_TIME

VSS$GET_START_TIME — Calculates the next time to run, given a starting point, a valid interval, and the day-of-the-week mask.

Format

VSS$GET_START_TIME *start_from, sched_interval, DOW_mask, next_start_time*

Arguments

start_from

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

An OpenVMS date expressed as "dd-mmm-yyyy hh:mm:ss.cc". The argument must be 23 characters long, and all fields must be filled in. For example: 06-FEB-2025 12:34:45.00. A blank string ("") indicates that the current system time should be used.

sched_interval

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid VSI JM Manager interval. This interval string must pass the VSS$VALIDATE_TIME test for intervals.

DOW_mask

OpenVMS usage	char_string

type	character string
access	read-only
mechanism	by descriptor

The day-of-the-week mask, expressed as a 7-character binary string of 1s and 0s representing the days (Monday through Sunday) on which the job can run. For example, 1000000 indicates that the job can run only on Mondays.

next_start_time

OpenVMS usage	char_string
type	character string
access	write-only
mechanism	by descriptor

The next starting time calculated from the other three arguments. It is a 23-character string that contains either a date "dd-mmm-yyyy hh:mm:ss.cc" or the string "NEVER".

Possible Return Values

Return Value	Meaning
1	Success
NSCHED$_INVSTRTIME	Invalid time string
Error codes from VSS$VALIDATE_TIME	See VSS$VALIDATE_TIME
Negative of BASIC error code	See the VSI BASIC documentation
System service error code	Error code returned by system services

VSS$GET_SYNC_TIME

VSS$GET_SYNC_TIME — Gets the last known time that a remote job dependency completed successfully. If flag is 0, this routine returns the time from the local cache in DEPENDENCY.DAT. This cache-time value is filled in by Job Completed messages from remote nodes. Otherwise, this routine calls the VSS$SHOW routine to do a network connect to get the value of the *sync_time* field for this job from the remote node's job database.

Format

VSS$GET_SYNC_TIME ***job_number, job_node, flag, sync_time***

Arguments

job_number

OpenVMS usage	longword_signed
type	longword integer (signed)
access	read-only
mechanism	by reference

The number of the remote job.

job_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by reference

The node for this remote job dependency.

flag

OpenVMS usage	longword_signed
type	longword OpenVMS integer (signed)
access	read-only
mechanism	by reference

If 0, gets time from local cache. Otherwise, gets value of *sync_time* field for this job from the remote node's job database.

sync_time

OpenVMS usage	quadword
type	quadword (unsigned)
access	write-only
mechanism	by reference

Receives the OpenVMS format quadword time. This time can be declared by the calling program as a string, array of two longwords, or a record, if it is passed by reference.

Note

If no cache-time is available, or if a connection to the remote node cannot be made, the routine returns 0 (December, 1888) as the *sync_time*.

Use this routine for remote job dependencies only.

Possible Return Values

Return Value	Meaning
1	Success
DECnet or RPC error	An error occurred connecting to a remote node
System service error code	Error code returned by system services

VSS$INSERT

VSS$INSERT — Creates a new job in the Job Management Manager database, either locally or on a remote system. All arguments are required. In order to run a job with override actions using the API, you

make the appropriate call to VSS$INSERT and pass it the negative of the job number you want to affect. Not all fields can be specified using this routine, which is supported only for compatibility with Version 1.1A and earlier. For new development, use VSS$CREATE_ITEM.

Format

VSS$INSERT *start_time, sched_interval, record_flags, DOW_mask, OpenVMS_command, output_file, comment, mail_address, cluster_node, user_name, job_name, group_name, type_name, maxtime, sync_array(), sync_nodes, pre_function, post_function, stall_notify, server_node, job_priority, queue_priority, hold_job, job_created, return_status*

Arguments

start_time

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

For more information on description of the VSI JM Manager starting time syntax, see the chapter *Chapter 2, "Job Management Manager Programming"*.

sched_interval

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

For more information on description of the VSI JM Manager interval syntax, see the chapter *Chapter 2, "Job Management Manager Programming"*.

record_flags

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A 32-bit binary value converted to decimal string format.

Bit	Function If Set
0	Retains the job in the database if an error occurs
1	Retains the job in the database if the job is successful
2	Sends mail to *mail_address* if an error occurs
3	Sends mail to *mail_address* if the job is successful
4	Restarts the job if the node fails while executing

Bit	Function If Set
5	Notifies the user's terminal when the job completes
11	Retries the job if it completes with an error status
12	Adds a job's schedule interval to the next starting time, instead of the time the job actually starts, to calculate the next starting time
16	The job is restricted by one or more Special Day Classes
17	Sends mail to *mail_address* if a Special Day Action occurs
19	The job's Special Day Restriction is to run only on Special Days
20	The job's Special Day Restriction is to not run on Special Days
21	The job's Special Day Action is to hold, instead of skip
22	The job's last run was a successful run, instead of a failure
128	Sends an OPCOM message on job completion
	Other bits should be 0

DOW_mask

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The day-of-the-week mask, expressed as a 7-character binary string of 1s and 0s representing the days (Monday through Sunday) on which the job can run. For example, 1000000 indicates that the job can run only on Mondays.

OPENVMS_command

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Any valid DCL command to execute. The maximum length is 132 characters. This argument is mandatory.

output_file

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid file specification in which to store output. If a blank string ("") is passed, `NL:` is the resulting output. The maximum length is 50 characters.

comment

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Character string. The maximum length is 80 characters.

mail_address

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The mail address to send mail to if *record_flags* specifies mail should be sent. If a blank string ("") is passed, the default address is the current user's name. The maximum length is 50 characters.

cluster_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name of the node within the cluster on which to run the job. The maximum length is six characters. If *cluster_node* is specified, the job can run only on this node. If *cluster_node* is blank (""), the default VSI JM Manager will assign the job to a node.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The OpenVMS *user_name* that the job runs under. The default is the current user. The maximum length is 32 characters. You need CMKRNL privilege to specify a *user_name* different from the calling program's. You must specify a *user_name* if the *job_name* is a numeric job name, not a job number.

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

An ASCII string that can be used later to refer to the job. The maximum length is 40 characters. Wildcard characters are not allowed. Implicitly qualified by *user_name*.

group_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Name of the group to which this job belongs. You can apply VSI JM Manager operations to a group of jobs at a time. The maximum length is 40 characters.

type_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name of the job type to which this job belongs. This argument is similar to *group_name*. The maximum length is 40 characters.

maxtime

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid OpenVMS delta time quadword, represented by an 8-byte string. If a job does not complete by the time indicated, mail is sent to the job's *mail_address*. A blank string ("") or quadword value of 0 indicates that there is no timeout value.

sync_array()

OpenVMS usage	vector_longword_unsigned
type	packed decimal string
access	read-only
mechanism	descriptor Class A

An array of 16 longword record numbers that this job will wait upon. The calling program must have read access to each of the dependencies. The array can start at any lower bound, but the 16 longwords must be contiguous from that bound. For example, bounds can be 0 to 15, 1 to 16, or 23 to 38. If there are less than 16 dependencies, pad the value with zeros.

sync_nodes

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A 96-character fixed-length string that holds the remote nodes for the synchronizations in the *sync_array*. For example, the first six characters hold the node for the synchronization in array element 1. If the synchronization is a local job, the six characters in the corresponding position in the string should be blank.

pre_function

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Any valid OpenVMS command, to be executed before the main command of the job. The maximum length is 40 characters.

post_function

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Any valid OpenVMS command, to be executed after the main command of the job. The maximum length is 40 characters.

stall_notify

OpenVMS usage	string
type	character string
access	read-only
mechanism	by descriptor

A valid OpenVMS delta time quadword, represented by an 8-byte string. If a job does not start within this amount of time after the job's scheduled starting time, then broadcast and mail notifications are sent. A blank string ("") or quadword value of 0 indicates that there is no stall value.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The remote node on which to insert this job. In order to use this argument, you must have chosen the wide area network (WAN) features during the VSI JM Manager installation. If you are not doing a remote insert, pass a blank string ("").

job_priority

OpenVMS usage	char_string

type	packed decimal string
access	read-only
mechanism	by descriptor

A decimal string that represents the priority level at which the job will run. It should be a numeric string between 1 and 16. You need SETPRV or ALTPRI privilege to set the priority higher than the default VSI JM Manager or user priority.

queue_priorityCAJM

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Not implemented. Pass a blank string ("").

hold_job

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A one-character string. If the value is H, then the job is placed on hold when inserted.

job_created

OpenVMS usage	longword_signed
type	longword integer (signed)
access	read/write
mechanism	by reference

This argument must pass a 0 to create a new job. The newly created job number is returned.

return_status

OpenVMS usage	cond_value
type	longword integer (signed)
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

Return Value	**Meaning**
1	Success

Return Value	Meaning
NSCHED$_DUPLNAM	Duplicate name not permitted
NSCHED$_NOSUCHUSER	No such user
NSCHED$_NOSCHED	No VSI JM Manager is available to service the request
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NODATABASE	VSI JM Manager database was not found
NSCHED$_WLDNAME	Invalid wildcard operation
NSCHED$_NOPRIV	Insufficient privilege to perform action

Possible Return Values

This is a subroutine. The status of the call is returned in the *return_status* variable.

VSS$JOB_RERUN

VSS$JOB_RERUN — Sets the retry frequency and the maximum retry attempts for a job.

Format

VSS$JOB_RERUN *job_name, user_name, frequency, attempts*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A character string containing the job number or the job name of the job for which to set the retry characteristics.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A character string containing the *user_name* of the job owner, or blanks if the *job_name* is a number.

frequency

OpenVMS usage	longword

type	unsigned longword
access	read-only
mechanism	by reference

A longword for the number of minutes between retries. Valid values are 1 through 60.

attempts

OpenVMS usage	longword
type	unsigned longword
access	read-only
mechanism	by reference

A longword for the number of attempted retries. Valid values are 1 through 10000.

Possible Return Values

Return Value	Meaning
1	Success
NSCHED$_INVARG	Invalid argument passed
NSCHED$_NOSUCHJOB	Specified job could not be found
NSCHED$_NODATABASE	VSI JM Manager database was not found
System service error code	Error code returned by system services

VSS$MODIFY

VSS$MODIFY — Modifies an existing job in the Job Management Manager database, either the local database or a remote database. Not all fields can be modified using this routine, which is supported only for compatibility with Version 1.1A and earlier. For new development, use VSS$MODIFY_ITEM.

Format

VSS$MODIFY ***start_time, schedule_interval, record_flags, DOW_mask, OpenVMS_command, output_file, comment, mail_address, cluster_node, new_user_name, new_job_name, group_name, type_name, maxtime, sync_array(), sync_nodes, job_to_modify, job_user_name, pre_function, post_function, stall_notify, server_node, job_priority, queue_priority, returned_tag, return_status***

Arguments

start_time

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

For more information on description of the VSI JM Manager starting time syntax, see the chapter *Chapter 2, "Job Management Manager Programming"*.

sched_interval

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

For more information on description of the VSI JM Manager interval syntax, see the chapter *Chapter 2, "Job Management Manager Programming"*.

record_flags

OpenVMS usage	char_string
type	packed decimal string
access	read-only
mechanism	by descriptor

A 32-bit binary value converted to decimal string format.

Bit	**Function If Set**
0	Retains the job in the database if an error occurs
1	Retains the job in the database if the job is successful
2	Sends mail to *mail_address* if an error occurs
3	Sends mail to *mail_address* if the job is successful
4	Restarts the job if the node fails while executing
5	Notifies the user's terminal when the job completes
11	Retries the job if it completes with an error status
12	Adds a job's schedule interval to the next starting time, instead of the time the job actually starts, to calculate the next starting time
16	The job is restricted by one or more Special Day Classes
17	Sends mail to *mail_address* if a Special Day Action occurs
19	The job's Special Day Restriction is to run only on Special Days
20	The job's Special Day Restriction is to not run on Special Days
21	The job's Special Day Action is to hold, instead of skip
22	The job's last run was a successful run, instead of a failure
128	Sends an OPCOM message on job completion
	Other bits should be 0

DOW_mask

OpenVMS usage	char_string
type	character string

access	read-only
mechanism	by descriptor

The day-of-the-week mask, expressed as a 7-character binary string of 1s and 0s representing the days (Monday through Sunday) on which the job can run. For example, 1000000 indicates that the job can run only on Mondays.

OPENVMS_command

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Any valid DCL command to execute. The maximum length is 132 characters.This argument is mandatory.

output_file

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid file specification in which to store output. If a blank string ("") is passed, `NL:` is the resulting output. The maximum length is 50 characters.

comment

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Character string. The maximum length is 80 characters.

mail_address

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The mail address to send mail to if *record_flags* specifies mail should be sent. If a blank string ("") is passed, the default address is the current user's name. The maximum length is 50 characters.

cluster_node

OpenVMS usage	char_string

type	character string
access	read-only
mechanism	by descriptor

The name of the node within the cluster on which to run the job. The maximum length is six characters. If *cluster_node* is specified, the job can run only on this node. If *cluster_node* is blank (""), the default VSI JM Manager will assign the job to a node.

new_user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The new OpenVMS *user_name* that the job runs under. The default is the current user. The maximum length is 32 characters. You need CMKRNL privilege to specify a *user_name* different from the calling program's. You must specify a *user_name* if the *job_name* is a numeric job name, not a job number.

new_job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

An ASCII string that can be used later to refer to the job. The maximum length is 40 characters. Wildcard characters are not allowed. Implicitly qualified by *user_name*.

group_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Name of the group to which this job belongs. You can apply VSI JM Manager operations to a group of jobs at a time. The maximum length is 40 characters.

type_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name of the job type to which this job belongs. This argument is similar to *group_name*. The maximum length is 40 characters.

maxtime

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A valid OpenVMS delta time quadword, represented by an 8-byte string. If a job does not complete by the time indicated, mail is sent to the job's *mail_address*. A blank string ("") or quadword value of 0 indicates that there is no timeout value.

sync_array()

OpenVMS usage	vector_longword_unsigned
type	packed decimal string
access	read-only
mechanism	descriptor Class A

An array of 16 longword record numbers that this job will wait upon. The calling program must have read access to each of the dependencies. The array can start at any lower bound, but the 16 longwords must be contiguous from that bound. For example, bounds can be 0 to 15, 1 to 16, or 23 to 38. If there are less than 16 dependencies, pad the value with zeros.

sync_nodes

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A 96-character fixed-length string that holds the remote nodes for the synchronizations in the *sync_array*. For example, the first six characters hold the node for the synchronization in array element 1. If the synchronization is a local job, the six characters in the corresponding position in the string should be blank.

job_to_modify

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name or number of the job to modify. The maximum length is 40 characters.

job_user_name

OpenVMS usage	char_string
type	character string

access	read-only
mechanism	by descriptor

The user name of the job. If this argument is blank (""), the default id the calling program's *user_name*.

pre_function

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Any valid OpenVMS command, to be executed before the main command of the job. The maximum length is 40 characters.

post_function

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Any valid OpenVMS command, to be executed after the main command of the job. The maximum length is 40 characters.

stall_notify

OpenVMS usage	string
type	character string
access	read-only
mechanism	by descriptor

A valid OpenVMS delta time quadword, represented by an 8-byte string. If a job does not start within this amount of time after the job's scheduled starting time, then broadcast and mail notifications are sent. A blank string ("") or quadword value of 0 indicates that there is no stall value.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The remote node on which to insert this job. In order to use this argument, you must have chosen the wide area network (WAN) features during the VSI JM Manager installation. If you are not doing a remote insert, pass a blank string ("").

job_priority

OpenVMS usage	char_string
type	packed decimal string
access	read-only
mechanism	by descriptor

A decimal string that represents the priority level at which the job will run. It should be a numeric string between 1 and 16. You need SETPRV or ALTPRI privilege to set the priority higher than the default VSI JM Manager or user priority.

queue_priority

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Not implemented. Pass a blank string ("").

return_tag

OpenVMS usage	longword_signed
type	longword integer (signed)
access	write-only
mechanism	by reference

Holds the job entry number of the job being modified.

return_status

OpenVMS usage	cond_value
type	longword integer (signed)
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

Value	**Meaning**
1	Success
NSCHED$_DUPLNAM	Duplicate name not permitted
NSCHED$_NOSUCHUSER	No such user
NSCHED$_NOSCHED	No VSI JM Manager is available to service the request
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NODATABASE	VSI JM Manager database was not found
NSCHED$_WLDNAME	Invalid wildcard operation

Value	Meaning
NSCHED$_NOPRIV	Insufficient privilege to perform action
Error codes from VSS$VALIDATE_TIME	See VSS$VALIDATE_TIME
Negative of BASIC error code	See the VSI BASIC documentation
System service error code	Error code returned by system services
RPC or DECnet error code	An error occurred connecting to a remote node

Possible Return Values

This is a subroutine. The status of the call is returned in the *return_status* variable.

VSS$MODIFY_ITEM

VSS$MODIFY_ITEM — Modifies an existing Job Management Manager job with the given parameter values. The routine takes an *item_list* of new field values for the job. Use this routine to modify a job; the routine VSS$MODIFY is supported only for compatibility with Version 1.1 and earlier.

Format

VSS$MODIFY_ITEM ***job_name, job_owner, server_node, number_of_items, item_list***

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job name to change. The name may not contain wildcard characters.

job_owner

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job owner for the job. The name may not contain wildcard characters.

server_node

OpenVMS usage	char_string
type	character string
access	read-only

mechanism	by descriptor

The VSI JM Manager *server_node* on which to modify the job. An empty string specifies the local server.

number_of_items

OpenVMS usage	integer
type	unsigned longword
access	read-only
mechanism	by value

The number of items in the *item_list*.

item_list

OpenVMS usage	pointer to array of item blocks
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which fields are to be modified for the job. This argument is the address of an array of item blocks (*job_item*) each containing a valid item code and a descriptor containing the value for the field.

The item codes are described in the nsched_job_fields.h file in NSCHED$. The valid item codes for VSS$MODIFY ITEM are:

- NSCHED_RECORD_FLAGS
- NSCHED_LAST_STATUS
- NSCHED_SYNC_JOB_NUMS
- NSCHED_PRE_STATUS
- NSCHED_POST_STATUS
- NSCHED_JPRIORITY
- NSCHED_QPRIORITY
- NSCHED_TIMEOUT
- NSCHED_START_TIME
- NSCHED_SYNC_TIME
- NSCHED_SYNC_NODES
- NSCHED_PRE_FUNCTION
- NSCHED_POST_FUNCTION

- NSCHED_SCHED_INTERVAL
- NSCHED_NOSTART_TIME
- NSCHED_DOW
- NSCHED_USERNAME
- NSCHED_JOBNAME
- NSCHED_COMMENT
- NSCHED_LOGFILE
- NSCHED_OPENVMS_COMMAND
- NSCHED_MAIL_TO
- NSCHED_CLUSTER_NODE
- NSCHED_GROUP
- NSCHED_TYPE
- NSCHED_NO_DEPON
- NSCHED_RERUN_FREQUENCY
- NSCHED_RERUN_ATTEMPTS
- NSCHED_RDID
- NSCHED_WRID
- NSCHED_EXID
- NSCHED_STALL_JOB
- NSCHED_TIMEOUT_JOB
- NSCHED_EXEC_TYPE
- NSCHED_BATCH_QUEUE
- NSCHED_BATCH_CPULIM

See VSS$CREATE_ITEM for tables that list field indicates and field descriptions for the callable routines VSS$SHOW_ITEM, VSS$CREATE_ITEM, and VSS$MODIFY_ITEM.

Possible Return Values

Return Value	Meaning
NSCHED$_SUCCESS	Operation completed successfully
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_INVARG	Routine was called with an invalid string argument

Return Value	Meaning
NSCHED$_BADITEM	Bad Item Code specified to callable routine
NSCHED$_FLDTOOLONG	Specified field is too long
NSCHED$_BADVALUE	Bad *item_list* value specified
NSCHED$_NOSUCHID	Unknown rights identifier
NSCHED$_NOSAMEJOB	Timeout Job must not be same job
NSCHED$_IDOUTOFRNG	Identifier value is out of range
NSCHED$_TIMBEFOR	Start time is before current time; job will run immediately
NSCHED$_DUPLNAME	Duplicate job name is not permitted
NSCHED$_ADJFLDVAL	Field value adjusted to be within limits
NSCHED$_WLDNAME	Invalid wildcard operation
NSCHED$_ONEDEPFLD	Dependency nodes and numbers cannot be updated separately
NSCHED$_INVMODEVAL	Invalid mode value; value must be DETACHED, BATCH, or REMOTE
NSCHED$_CNTMODFLD	Caller cannot modify this field
NSCHED$_FLDNOTSUPP	Field not supported in current version

The function can also return various system service and RMS codes, and negated BASIC error codes.

VSS$MODIFY_SD_CLASS

VSS$MODIFY_SD_CLASS — Modifies an existing Special Day Class. Users can add and remove absolute days from the class.

Format

VSS$MODIFY_SD_CLASS *class_name, reserved, item_list*

Arguments

class_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This is the name of the Special Day Class to be modified. The maximum length allowed is 40 characters.

reserved

OpenVMS usage	null_arg
type	unsigned longword
access	read-only

mechanism	by value

This is a placeholder argument and is reserved for future use. This argument should be null.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by value

The *item_list* argument specifies which modifications are to be made to the Special Day Class. This argument is the address of an array of ITEM_BLOCKS each describing a modification to be made to the Special Day Class. The array is terminated by a block whose first longword is 0 (zero).

Item Code	Meaning
VSS$K_INSERT_ABSOLUTE_DAYS	Specifies that the indicated absolute days should be inserted into the class's absolute day list. The absolute days are represented by a YEAR_BITMASK structure that is allocated and filled in by the calling program. The size and address of this structure are passed in the ITEM_BLOCK.
VSS$K_REMOVE_ABSOLUTE_DAYS	Specifies that the indicated absolute days should be removed from the class's absolute day list. The absolute days are represented by a YEAR_BITMASK structure that is allocated and filled in by the calling program. The size and address of this structure are passed in the ITEM_BLOCK.
VSS$K_REPLACE_ABSOLUTE_DAYS	Specifies that the indicated absolute days should replace the class's current absolute day list. The absolute days are represented by a YEAR_BITMASK structure that is allocated and filled in by the calling program. The size and address of this structure are passed in the ITEM_BLOCK.

Note

The year bitmask indicated must have the year field specified to tell VSI JM Manager what year to replace. If this field is null, an error occurs.

Possible Return Values

Return Value	Meaning
NSCHED$_CLASSMODIFIED	Special Day Class modified
NSCHED$_INVBITMASK	Invalid YEAR_BITMASK specification
NSCHED$_INVBUFFSPEC	Invalid buffer specification
NSCHED$_INVCHARFND	Invalid character found in Special Day Class Name
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK

Return Value	Meaning
NSCHED$_LEAD-NUMNOTAL	Leading number not allowed in Special Day Class name
NSCHED$_NAMTOOLONG	Special Day Class name exceeds maximum length
NSCHED$_NOITEMLIST	No *item_list* specified
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_NOSUCHCLASS	Special Day Class does not exist
NSCHED$_NYI	
NSCHED$_SUCCESS	Operation completed successfully

VSS$NAME_TO_NUM

VSS$NAME_TO_NUM — Looks up a job's entry number when given its *job_name* and *user_name*.

Format

VSS$NAME_TO_NUM *job_name, user_name, server_node, job_number, return_status*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name of the job for which to look up the job number. The maximum length is 40 characters. Wildcard characters are not allowed.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

If this argument is blank (""), the default is the calling program's *user_name*. The maximum length is 32 characters.

server_node

OpenVMS usage	char_string
type	character string
access	read-only

mechanism	by descriptor

The remote node on which the job resides. If the job is local, pass a blank string ("").

job_number

OpenVMS usage	longword_signed
type	longword integer (signed)
access	write-only
mechanism	by reference

Holds the job number value for the specified job.

return_status

OpenVMS usage	cond_value
type	longword integer (signed)
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

Value	Meaning
1	Success
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NODATABASE	VSI JM Manager database was not found
Negative of BASIC error code	See the VSI BASIC documentation

Possible Return Values

This is a subroutine. The status of the call is returned in the *return_status* call.

VSS$NO_DEPON

VSS$NO_DEPON — Overrides the specified dependencies for a particular job. The 16 low-order bits (0 to 15) in the *no_dep_mask* longword mask correspond to each job dependency in the job's synchronization array. The synchronization array is a 16-element array containing the job numbers on which the job is dependent (or 0 if there are no dependent jobs). Use VSS$SHOW to show the dependencies of a job. The order of the dependencies as shown by VSS$SHOW shows their placement in the synchronization array. For example, to override the dependencies in synchronization array elements 3 and 15, you would use a longword mask value of "22 + 214 = 16388". Setting a bit in the mask makes the manager consider the corresponding job dependency as satisfied for the next run of the job.

Format

VSS$NO_DEPON *job_name, user_name, no_dep_mask*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name or number of the job. Wildcard characters are not allowed. If the *job_name* is numeric, you must specify the *user_name*.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name of the user who owns the job. If this argument is blank ("") and the *job_name* is numeric, then the job is looked up by number. If not, the calling program's *user_name* is used as the default. Wildcard characters are not allowed.

no_dep_mask

OpenVMS usage	mask_longword
type	unsigned longword
access	read-only
mechanism	by reference

If any of the first 16 bits (0 to 15) are set, *no_dep_mask* indicates that the corresponding job dependency in the synchronization array is satisfied for the next run. For example, to override the dependencies in synchronization array positions 1 and 8:

```
no_dep_mask = 20 + 27 = 129
```

Possible Return Values

Return Values	Meaning
1	Success
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NOSCHED	No VSI JM Manager is available to service the request
NSCHED$_NODATABASE	VSI JM Manager database was not found
Negative of BASIC error code	See the VSI BASIC documentation

VSS$NUM_TO_NAME

VSS$NUM_TO_NAME — Takes the job number and node of the job, and returns the last completion status, current state, and a string with the *job_name* and *user_name*.

Format

VSS$NUM_TO_NAME *job_number, server_node, job_status, job_state, ret_buffer*

Arguments

job_number

OpenVMS usage	longword
type	longword integer (signed)
access	read-only
mechanism	by descriptor

The job number that you want to reference.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The remote node on which the job resides. If the job is local, pass a blank string ("").

job_status

OpenVMS usage	longword
type	longword integer (signed)
access	write-only
mechanism	by descriptor

Returned latest completion status for the specified job. This will be an OpenVMS NSCHED completion status.

job_state

OpenVMS usage	character
type	character string
access	read-only
mechanism	by descriptor

The job state is one of the following:

R	Running
J	Job-slot wait
D	Dependency wait

S	Scheduled
H	Hold
Q	In a BATCH queue awaiting execution

ret_buffer

OpenVMS usage	string
type	character string
access	write-only
mechanism	by descriptor

A 72-character string, with the first 40 characters being the job name and the last 32 characters being the *user_name*. If the string is short of characters, it will be padded with blank spaces to make up the difference.

Possible Return Values

Return Value	Meaning
NSCHED$_SUCCESS	Operation completed successfully
System service error code	Error code returned by system services

VSS$PID_RUNNING

VSS$PID_RUNNING — Checks whether a job created by the manager is currently running on a node in the cluster.

Format

VSS$PID_RUNNING *process_ID*

Arguments

process_id

OpenVMS usage	process_id
type	unsigned longword
access	read-only
mechanism	by reference

The process identification (PID) of the job you want to check.

Possible Return Values

Return Value	Meaning
1	Running
0	Not running
System service error code	Error code returned by system services

Note

You need SYSLCK and CMEXEC privileges to use VSS$PID_RUNNING. This routine works only for processes created by VSI JM Manager and relies on an exec mode lock that DOO_COMMAND.EXE makes. The process ID of a running job is stored in the VSI JM Manager database and returned by the VSS$SHOW routine. You can also use the LIB$GETJPI routine on OpenVMS.

VSS$RESET_SYNC

VSS$RESET_SYNC — Resets the time by which dependencies must be met for this job to run. If the sync time is currently set to the current time or a time in the future, this routine clears all the currently satisfied dependencies so that they must be met again before the job will run. If the sync time is currently set to some time in the past, this routine may cause unsatisfied dependencies to be considered satisfied if the dependencies have completed successfully in the past.

Format

VSS$RESET_SYNC *job_number, reserved, sync_time*

Arguments

job_number

OpenVMS usage	longword
type	longword integer (signed)
access	read-only
mechanism	by reference

A longword for the job number for which you want to set the *sync_time*.

reserved

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This is a character string that is reserved for future use. It may not be omitted.

sync_time

OpenVMS usage	quadword
type	quadword (unsigned)
access	read-only
mechanism	by descriptor

OpenVMS format quadword time. This is the time after which all of the job's dependencies must have completed before the job will run. This argument resets any dependencies overridden by VSS$NO_DEPON.

Possible Return Values

Return Values	Meaning
1	Success
NSCHED$_INVARG	Invalid argument passed
NSCHED$_NOSUCHJOB	Specified job could not be found
NSCHED$_NODATABASE	NSCHED$:VSS.DAT not found
System service error code	Error code returned by system services

VSS$SCHED_LIST

VSS$SCHED_LIST — Returns a list of nodes in the local or remote cluster that are currently running the manager.

Format

VSS$SCHED_LIST *server_node, number_of_scheds, list_of_nodes, return_status*

Arguments

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The remote node where this routine is executed.

number_of_scheds

OpenVMS usage	longword_unsigned
type	unsigned longword
access	write-only
mechanism	by reference

The number of VSI JM Manager copies currently running in the OpenVMS Cluster. Checks a remote cluster if a server is specified.

list_of_nodes

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

List of nodes that are running VSI JM Manager. Each node takes up exactly 6 characters in the list.

return_status

OpenVMS usage	cond_value
type	longword integer (signed)
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

Value	Meaning
1	Success
NSCHED$_NOSCHED	No VSI JM Manager is available to service the request
System service error	Error code returned by system services
RPC or DECnet error code	An error occurred connecting to a remote node

Possible Return Values

This is a subroutine. The status of the call is returned in the *return_status* variable.

VSS$SCHED_RUNNING

VSS$SCHED_RUNNING — Checks whether the manager is running anywhere in the OpenVMS Cluster or on a particular local node.

Format

VSS$SCHED_RUNNING ***nodename***

Arguments

nodename

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The node must be a local node. If this argument is blank (""), the routine checks if VSI JM Manager is running on any node on the OpenVMS Cluster.

Possible Return Values

Return Value	Meaning
1	VSI JM Manager is currently running
0	VSI JM Manager is not currently running
System service error code	Error code returned by system services

VSS$SCRIPT

VSS$SCRIPT — Creates a DCL command file capable of recreating a set of existing local Job Management Manager jobs. For example, you can create the command file and then copy it to another machine and run it on that machine to recreate the jobs.

VSS$SCRIPT Overview

There are two ways to include a job in the command file:

- The user can provide four types of selection criteria to identify the primary set of jobs to include. These criteria are: the job name, the job owner, the job group and the job type. Wildcard matching can be used with each of these criteria.
- The user can specify that jobs which are dependencies or dependents of the primary jobs to also be included. Dependency jobs are those which must complete before a primary job can run. Dependent jobs are those which will run after the primary job completes.

VSS$SCRIPT has the following call format:

```
long vss$script ( StringDescPtr name_desc, StringDescPtr owner_desc,
StringDescPtr group_desc, StringDescPtr type_desc,
StringDescPtr outfile_desc, char graph_type,
char version_11 )
```

Job selection parameters include:

- *name_desc* – include all jobs with this job name or number. (Must be a name, number or *.)
- *owner_desc* – include all jobs owned by this user. (Must be a name, * or blank.)
- *group_desc* – include all jobs in this job group. (Must be a name, * or blank.)
- *type_desc* – include all jobs of this job type. (Must be a name, * or blank.)
- *outfile_desc* – the name of the command file to be produced.
- *graph_type* – include jobs connected to primary jobs through the specified type of dependencies. Valid types are: (--1) No dependencies/dependents, (0) All dependencies/dependents, (1) Only dependencies, (2) Only dependents.
- *version_11* – Boolean flag specifying if the target system is running a version of Scheduler at or before Version 1.1. The script will subsequently not include qualifiers from Version 2.x or later.

Running VSS$SCRIPT

VSS$SCRIPT cannot be run on remote Job Management Manager servers.

If a job number is specified in the selection criteria, the username, group name, and type name qualifiers must be blank.

If an unnamed job is specified in a command file, but is not created in that file, the job's current job number is used as the job name. Since this job is unlikely to have the same job number on the target machine, SCRIPT returns a warning status code.

Any jobs not included in the command file are assumed to exist on the target machine.

Remote dependent jobs are not included in the command file. If such jobs are specified, they are assumed to exist on the remote server.

Dependencies include only synchronization jobs. Timeout and stall jobs are not considered to be dependencies. If such jobs are specified, they are assumed to exist on the target machine.

RDID, WRID and EXID identifiers are assumed to exist on the target machine.

Format

VSS$SCRIPT *job_name, job_owner, job_group, job_type, output_file_name, graph_type, version_11*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job name to select. The name may contain wildcard characters.

job_owner

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job owner to select. The name may contain wildcard characters.

job_group

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job group to select. The group may contain wildcard characters.

job_type

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job type to select. The type may contain wildcard characters.

output_file_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name of the DCL command file to create.

graph_type

OpenVMS usage	integer
type	byte (unsigned)
access	read-only
mechanism	by value

The kinds of dependent jobs to include. There are four valid graph types (defined for C programs in GET_JOB_STREAM.H):

NO_GRAPH	Ignore dependencies
WHOLE_GRAPH	Include all dependencies
JUST_PARENTS	Include all ancestors of selected jobs
JUST_CHILDREN	Include all descendants of selected jobs

version_11

OpenVMS usage	byte (unsigned)
type	Boolean
access	read-only
mechanism	by value

A Boolean field which specifies whether the target machine is running CA JM Manager Version 1.1 or earlier.

Possible Return Values

Return Value	**Meaning**
NSCHED$_CANTOPNDB	Privilege violation, could not open the VSI JM Manager database
NSCHED$_INVARG	Routine was called with an invalid string argument
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_SUCCESS	Operation completed successfully
System service error code	Error code returned by system service
BASIC error code	Error code returned by individual routine

VSS$SELECT

VSS$SELECT — Selects jobs from the database and returns their number in an array. Selection is based on any combination of job name, group name, type name, user name, job state, node, and scheduled run time. Wildcard characters are allowed where appropriate. If there are more selected jobs than can fit in the array, the routine saves its place and can be called again to get the remaining jobs.

Format

VSS$SELECT *job_name, group_name, type, user_name, stat_mask, node, tag_array(), array_size, context(), server_node, return_status*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The job name specification for which to search. The maximum number of characters is 40. All wildcard characters are allowed.

group_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The group name specification for which to search. The maximum number of characters is 40. All wildcard characters are allowed.

type

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Type of class name specification for which to search. The maximum number of characters is 40. All wildcard characters are allowed.

user_name

OpenVMS usage	char_string
type	character string
access	read-only

mechanism	by descriptor

The *user_name* specification for which to search. The maximum number of characters is 32. All wildcard characters are allowed.

stat_mask

OpenVMS usage	mask_longword
type	unsigned longword
access	read-only
mechanism	by reference

The five low-order bits of this longword specify the states of the jobs for which to search. If the bit for a state is set, the call does not return jobs of that state.

Bit	**Job State**	
0	Held	Least significant bit
1	Scheduled	
2	Job-slot wait	
3	Running	
4	Dependency wait	Most significant bit
5	Queued	

To search for jobs in all states, use a value of 31 for *stat_mask*.

node

OpenVMS usage	string
type	character string
access	read-only
mechanism	by descriptor

Specifies the cluster node on which the job is restricted to run. This returns any job that is scheduled to run or currently running on the specified node.

If the VSI JM Manager that is currently running on the specified node is the default VSI JM Manager, then this argument also returns any jobs that do not have a node restriction.

To specify the default node, use a blank string ("") for the node. To match any node, use an asterisk (*) wildcard for the node.

tag_array()

OpenVMS usage	longword_signed
type	longword
access	write-only
mechanism	by descriptor Class A

The lowest element of the array receives the number of array elements filled in. The remaining elements receive job numbers that match all the selection criteria specified in the other arguments.

The routine automatically calculates the lower bound of the array. The upper bound must be equal to "*array_size --1* + lower bound". For example, if the lower bound is 1 and the *array_size* is 50, then the upper bound is assumed to be 50.

array_size

OpenVMS usage	longword_unsigned
type	unsigned longword
access	read-only
mechanism	by reference

The number of elements in array *tag_array()*. Since the lowest element of *tag_array()* receives the number of elements being returned, the most job numbers that can be filled in by each call to the routine is *array_size* -- 1.

context()

OpenVMS usage	vector_longword_unsigned
type	unsigned longword
access	read/write
mechanism	by descriptor Class A

The first two longwords in this array form a quadword representing the scheduled date of the jobs for which to begin searching. Regardless of the scheduled date, if you are selecting jobs, this quadword should be "(0,0) (=17-NOV-1858)". If you want only jobs scheduled after a certain time, fill in the quadword with that time.

Note

Jobs that are not currently scheduled can have the year 5999 in their next scheduled time field. Jobs that are either waiting for a process slot or waiting for other jobs to complete before running can have scheduled times that are in the past.

The third type longword specifies the job number of the job for which to begin searching. Do not modify this longword. It is used by the routine to keep track of the calls received, in case the routine needs more than one call to return all selected jobs. If the longword is modified, unpredictable results may occur and the routine may return NSCHED$_NOSUCHJOB.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Specifies the remote node that has the VSI JM Manager database you want to search. To search the local database, specify a blank string ("").

Note

When the *server_node* argument is supplied, a maximum of 49 jobs can be returned for a remote node. This is due to a restriction in the network message protocol.

return_status

OpenVMS usage	cond_value
type	longword integer (signed)
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

Value	Meaning
NSCHED$_NOMORE	No more items or jobs found
NSCHED$_MORE	Additional items or jobs remain
NSCHED$_NODATABASE	VSI JM Manager database not found
Negative of BASIC error code	See the VSI BASIC documentation
System service error code	Error code returned by system services
RPC or DECnet error code	Error code returned by RPC or DECnet

Possible Return Values

This is a subroutine. The status of the call is returned in the *return_status* call.

VSS$SELECT_CLASS

VSS$SELECT_CLASS — Selects Special Day Classes and returns their names in an array. Criteria for selection is specified using an item list sent to the routine. Names of classes which satisfy all these criteria are returned. Wildcard characters are allowed where appropriate.

Format

VSS$SELECT_CLASS ***initial_call, num_selectors, selectors, return_array_size, number_classes_returned, return_array***

Arguments

initial_call

OpenVMS usage	char
type	character
access	read-only
mechanism	by reference

Indicates whether or not this is the first call made to the VSS$SELECT_CLASS routine. Since the routine may find more classes than can fit in the return array, it may need to be called multiple times to return all class names. For the first call of a given run, this parameter must be true (non-zero). For subsequent calls, this parameter must be zero.

num_selectors

OpenVMS usage	longword_unsigned
type	unsigned longword
access	read-only
mechanism	by reference

The number of entries in the selectors array.

selectors

OpenVMS usage	array of ClassSelectorItems
type	unsigned longword
access	read-only
mechanism	by reference

This array contains selection criteria which specify which classes to include in the array.

See HCISELECT.H and the documentation for valid selector types.

The ClassSelectorItem is declared as follows:

```
typedef struct ClassSelectorItem
{
int which_item; /* Index of specified item */
struct dsc$descriptor_s item_desc;/* Buffer to hold data */
}
```

The *which_item* argument can take the following values:

Name	Value	Meaning
K_CSELECT_CLASS_NAME	4	Select classes with this name. All wildcard characters are allowed.
K_CSELECT_CLASS_YEAR	5	Select classes in this year. All wildcard characters are allowed.

return_array_size

OpenVMS usage	longword_unsigned
type	unsigned longword
access	read-only
mechanism	by reference

The size of the array in which to return the class names.

number_classes_returned

OpenVMS usage	longword_unsigned
type	unsigned longword
access	write-only
mechanism	by reference

The number of class names actually stored in the return array. The address of this longword should be passed into this routine.

return_array

OpenVMS usage	array of SDCLASS_DESC_T structures
type	unsigned longword
access	write-only
mechanism	by reference

The array in which to return the class names. This array is to be allocated by the calling program. The size and address of this array are to be passed into this routine.

Possible Return Values

Return Value	Meaning
NSCHED$_NOMORE	No more special day class names found
NSCHED$_MORE	Additional special day class names remain
NSCHED$_NODATABASE	VSI JM Manager database not found
Negative of BASIC error code	See the VSI BASIC documentation
System service error code	Error code returned by system services
RMS error code	Error code returned by RMS

VSS$SET

VSS$SET — Sets a request flag for a single job. Requests include ABORT, DELETE, HOLD, RUN, and RELEASE. You can also use this routine to set the request flag for a remote job. The routine notifies the appropriate manager instance to perform the necessary action.

Format

VSS$SET ***job_name, user_name, request, server_node, return_status***

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The job name or job number (as a string) of the job for which you want to set the request. If the *job_name* is not a numeric string, then you must specify the *user_name*. The maximum length is 40 characters. Wildcard characters are not allowed.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The user who owns the job. If you do not specify a *user_name*, the default value is the routine's calling program. If the *job_name* is not a numeric string, you must specify the *user_name*.

request

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A 1-character field specifying the action to be requested. Only one action can be requested at a time. If additional requests are made before VSI JM Manager processes the current requests, the previous entries are erased. Valid requests are:

Value	Action
A	Abort this job
D	Delete this job from the database
H	Put this job on hold
U	Release (unhold) this job
N	Run this job now, regardless of whether it is scheduled to run

Other values are accepted and written to the database, but are likely to be cleared by VSI JM Manager without notice.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Specifies the remote node where this job resides.

return_status

OpenVMS usage	cond_value
type	longword integer (signed)

access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

Value	Meaning
1	Success
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NOSCHED	No VSI JM Manager is available to service the request
NSCHED$_NOTDONE	Job is already running
NSCHED$_NODATABASE	VSI JM Manager database was not found
Negative of BASIC error code	See the VSI BASIC documentation
System service error code	Error code returned by system services
RPC or DECnet error code	An error occurred connecting to a remote node

Possible Return Values

This is a subroutine. The status of the call is returned in the *return_status* variable.

VSS$SET_BITMASK_DATE

VSS$SET_BITMASK_DATE — Sets or clears a bit in a YEAR_BITMASK structure. Takes a date specification and the address of a YEAR_BITMASK structure. Calling program specifies whether the bit should be set or cleared.

Format

VSS$SET_BITMASK_DATE *bitmask_addr, item_list, action*

Arguments

bitmask_addr

OpenVMS usage	YEAR_BITMASK
type	unsigned longword
access	read-write
mechanism	by reference

Address of the bitmask to modify. A pointer to the structure of YEAR_BITMASK.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword

access	read-only
mechanism	by reference

The *item_list* argument specifies in what format the date information is sent; This argument is the address of an ITEM_BLOCK that says what format the input date is in and describes the buffer containing the input date. Only one ITEM_BLOCK is passed in for this call.

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	Meaning
VSS$K_ASCII_INPUT	Specifies the input date is in ASCII format. The calling program must allocate and set up a buffer containing a null-terminated ASCII string representation of a date specification. The size and address of this buffer must be passed in the ITEM_BLOCK.
VSS$K_BINARY_INPUT	Specifies the input date is in binary format. The calling program must allocate and set up a buffer with a binary number representation of a date specification. The size and address of this buffer must be passed in the ITEM_BLOCK.

action

OpenVMS usage	longword
type	unsigned longword
access	read-only
mechanism	by reference

Specifies whether the bit indicated by the input date specification should be set or cleared. This field must have one of the following values:

Code	Meaning
VSS$K_SET_BIT	Specifies the bit should be set (to 1)
VSS$K_CLEAR_BIT	Specifies the bit should be cleared (set to 0)

Possible Return Values

Return Value	Meaning
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_SUCCESS	Operation completed successfully
NSCHED$_YEARMISMATCH	Year of input date does not match year of input YEAR_BITMASK

VSS$SET_RESTART

VSS$SET_RESTART — Sets the restart test value symbol for a job that is currently running. This routine must be called from within a program that is running as a Job Management Manager job; otherwise, the routine returns NSCHED$_NOSUCHJOB. This routine is called by the DCL command **SCHEDULE SET RESTART *restart_string***

Format

VSS$SET_RESTART *restart_string*

Arguments

restart_string

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The test value symbol that is made available to the job if it is restarted after a system problem. The maximum length is 40 characters, and embedded spaces are not allowed.

Possible Return Values

Return Value	**Meaning**
1	Success
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NODATABASE	VSI JM Manager database was not found

VSS$SET_RETRY

VSS$SET_RETRY — Sets the network interval or network expiration field for an OpenVMS Cluster. Network interval is the time interval used by the network between attempts to retry operations such as sending job done messages or cleaning up the databases of job dependencies. Network expiration is the delta time after which the manager gives up attempts to retry operations.

Format

VSS$SET_RETRY *field_number, new_value, server_node*

Arguments

field_number

OpenVMS usage	longword_unsigned
type	longword
access	read-only
mechanism	by reference

Specifies whether to set the network interval or network expiration value.

2	Set the network interval. The *new_value* is a string of the form "dddd hh:mm:ss.hh"

3	Set the network expiration value. The *new_value* is a string of the form "dddd hh:mm:ss.hh"
Other	Return a status of NSCHED$_INVQUALVAL

new_value

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

An ASCII delta time for the network interval or network expiration value. The *field_number* determines the use of this argument.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A node in the remote OpenVMS Cluster being modified or a blank string ("") for the local OpenVMS Cluster.

Possible Return Values

Return Value	**Meaning**
1	Success
NSCHED$_INVSTRTIME	Invalid time string
NSCHED$_INVQUALVAL	Invalid qualifier value
RPC or DECnet error	An error occurred connecting to a remote node

VSS$SET_SD_ACTIONS

VSS$SET_SD_ACTIONS — Sets the Special Day Action for a job.

Format

VSS$SET_SD_ACTIONS *job_name, user_name, item_list*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only

mechanism	by descriptor

The name or number (as a decimal string) of the job about which to get Special Day Restriction information. Maximum length allowed is 40 characters. Wildcard characters are not allowed. If *job_name* is not numeric, the calling program must specify *user_name*.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

If *job_name* is a name, *user_name* serves to discriminate between identically named jobs with different owners. If *job_name* is not numeric, the calling program must specify *user_name*. Maximum length allowed is 32 characters. Wildcard characters are not allowed.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which modifications are to be made to the job's Special Day Actions. This argument is the address of an OpenVMS array of ITEM_BLOCKS (ITEM_BLOCK_T) each describing a modification to be made to the job's Special Day Actions. The array is terminated by a block whose first longword is 0 (zero).

Valid Special Day Actions are VSS$K_SKIP (0) and VSS$K_HOLD (1).

Item Code	**Meaning**
VSS$K_SET_ACTION	Specifies that the job's [default] Special Day Action should be set. The value is specified in a longword that is allocated by the calling program and whose length and address are passed into the callable routine in the ITEM BLOCK.

Possible Return Values

Return Value	**Meaning**
NSCHED$_INAPP_VALUE	Inappropriate value
NSCHED$_INVBUFFSPEC	Invalid buffer specification
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_NOITEMLIST	No *item_list* specified
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_SUCCESS	Operation completed successfully

VSS$SET_SD_RESTRICTIONS

VSS$SET_SD_RESTRICTIONS — Creates and defines Special Day Restrictions for a job or modifies existing Special Day Restrictions for a job. Calling programs can add or remove Special Day Classes from the job's RESTRICT_TO or RESTRICT_FROM Special Day Restriction expressions. Calling programs can also delete any existing Special Day Restrictions for the job. Item lists are used to accomplish any and all modifications in one call.

Format

VSS$SET_SD_RESTRICTIONS *job_name, user_name, item_list*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name or number (as a decimal string) of the job to get Special Day Restriction information about. Maximum length allowed is 40 characters. Wildcard characters are not allowed. If *job_name* is not numeric, the calling program must specify *user_name*.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

If *job_name* is a name, *user_name* serves to discriminate between identically named jobs with different owners. If *job_name* is not numeric, the calling program must specify *user_name*. Maximum length allowed is 32 characters. Wildcard characters are not allowed.

item_list

OpenVMS usage	array of ITEM_BLOCKS
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which modifications are to be made to the Special Day Restrictions for the job. This argument is the address of an OpenVMS array of ITEM_BLOCKS each of which describes a modification to be made to the job's Special Day Restrictions. The array is terminated by a block whose first longword is zero.

The following is a description of item codes used in the ITEM_BLOCK:

Item Code	Meaning
VSS$K_ADD_RESTRICT_TO	Specifies that the indicated class is to be inserted into the list of classes that make up the job's RESTRICT_TO Special Day Restriction. The class name is passed back in an SDCLASS_DESCRIPTOR structure that is allocated by the calling program. The size and address of the structure are passed in the ITEM_BLOCK.
VSS$K_REMOVE_RESTRICT_TO	Specifies that the indicated class is to be removed from the list of classes that make up the job's RESTRICT_TO Special Day Restriction. The class name is passed back in an SDCLASS_DESCRIPTOR structure that is allocated by the calling program. The size and address of the structure are passed in the ITEM_BLOCK.
VSS$K_ADD_RESTRICT_FROM	Specifies that the indicated class is to be inserted into the list of classes that make up the job's RESTRICT_FROM Special Day Restriction. The class name is passed back in an SDCLASS_DESCRIPTOR structure that is allocated by the calling program. The size and address of the structure are passed in the ITEM_BLOCK.
VSS$K_REMOVE_RESTRICT_FROM	Specifies that the indicated class is to be removed from the list of classes that make up the job's RESTRICT_TO Special Day Restriction. The class name is passed back in an SDCLASS_DESCRIPTOR structure that is allocated by the calling program. The size and address of the structure are passed in the ITEM_BLOCK.
VSS$K_DELETE_RESTRICT_TO	Specifies that the RESTRICT_TO Special Day Restriction expression for this job should be deleted. There is no data passed in this type of ITEM_BLOCK.
VSS$K_DELETE_RESTRICT_FROM	Specifies that the RESTRICT_FROM Special Day Restriction expression for this job should be deleted. There is no data passed in this type of ITEM_BLOCK.

Possible Return Values

Return Value	Meaning
NSCHED$_COMBNOTALL	Combined "Restrict-To" and "Restrict-From" expression not allowed
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_JOBRESTMODIF	Job Restrictions successfully modified
NSCHED$_NOITEMLIST	No *item_list* specified
NSCHED$_NOSUCHCLASS	Special Day Class does not exist
NSCHED$_NOTONLIST	Special Day Class not on job's "Restrict-To" or "Restrict-From" list
NSCHED$_RESFRMNOTFND	Special Days Restrict-From expression not found
NSCHED$_RESTONOTFND	Special Days Restrict-To expression not found
NSCHED$_SUCCESS	Operation completed successfully

VSS$SHOW

VSS$SHOW — Fetches information for a single job from the Job Management Manager database. The job can be a local or remote job. The information is returned in a 1050-character fixed-length string. VSS$SHOW does not fetch all information about a job, and is supported only for compatibility with Version 1.1A and earlier. For new development, use VSS$SHOW_ITEM.

Format

VSS$SHOW *job_name, user_name, server_node, return_status, return_buffer*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The name or number (as a decimal string) of the job for which you want information returned. The maximum length is 40 characters. Wildcard characters are not allowed. If the *job_name* is not numeric, you must specify the *user_name*.

user_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

If the *job_name* is a name, the *user_name* serves to discriminate between identically named jobs owned by different users. If the *job_name* is not numeric, you must specify *user_name*. The maximum length is 32 characters. Wildcard characters are not allowed.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

Specifies the remote node where the job resides. If the job is local, then pass a blank string ("").

return_status

OpenVMS usage	cond_value
type	longword integer (signed)
access	write-only

mechanism	by reference

A longword that receives one of the possible return values.

return_buffer

OpenVMS usage	char_string
type	character string
access	write-only
mechanism	by descriptor

A 1050-byte string or record that contains all the information about the job in the database.

For more information on string or record arrangement and description of the individual fields, see the chapter *Chapter 2, "Job Management Manager Programming"*.

Possible Return Values

Return Value	Meaning
1	Success
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_NODATABASE	Job Management Manager database was not found
Negative of BASIC error code	See the VSI BASIC documentation

VSS$SHOW_ITEM

VSS$SHOW_ITEM — Returns the specified fields for an existing Job Management Manager job. The routine takes an item list specifying which fields to return and containing the buffers in which to return them.

VSS$SHOW_ITEM Overview

The structure contains three fields:

- A long integer specifying one field index (these values are described in the VSS$CREATE_ITEM Field Indicates section). For example, field NSCHED_TAG has index 0.
- A return status code. If the routine completed successfully, this status code reflects warning or informational messages regarding a specific field. This code may sometimes indicate that the field was inappropriately selected, and so was ignored. If the routine failed, an error code in this field may indicate which field's value caused the failure.
- A string descriptor. For VSS$SHOW_ITEM, this contains the buffer in which to return the requested field's current value.

Note

The type of data stored in this field differs in some cases between VSS$SHOW_ITEM and VSS$MODIFY_ITEM. For example, VSS$SHOW_ITEM returns the timeout time field as a character

string (for example, 10-Jan-2000 10:00, whereas VSS$MODIFY_ITEM expects an OpenVMS quadword value to update this field.

For example, the JobItem structure is declared as follows:

```
#include <descrip>
typedef struct dsc$descriptor_s StringDesc, *StringDescPtr; typedef struct
 {
int which_item; /* Index of specified item */
long int status; /* Status code for item */
StringDesc item_desc; /* Buffer in which to pass data */
} JobItem;
```

Format

VSS$SHOW_ITEM *job_name, job_owner, server_node, number_of_items, item_list*

Arguments

job_name

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job name to select. The name may not contain wildcard characters.

job_owner

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

This field contains the job owner for the job. The name may not contain wildcard characters.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The VSI JM Manager *server_node* on which the job resides. An empty string specifies the local server.

number_of_items

OpenVMS usage	integer

type	unsigned longword
access	read-only
mechanism	by value

The number of items in the *item_list*.

item_list

OpenVMS usage	pointer to array of item blocks
type	unsigned longword
access	read-only
mechanism	by reference

The *item_list* argument specifies which fields are to be returned and contains the buffers in which to return them. This argument is the address of an array of item blocks (*job_item*) each containing a valid item code and a descriptor containing the value for the field.

The item codes are described in the NSCHED_JOB_FIELDS.H file in NSCHED$. The valid item codes for VSS$SHOW_ITEM are:

- NSCHED_TAG
- NSCHED_RECORD_FLAGS
- NSCHED_LAST_STATUS
- NSCHED_SUCCESS_COUNT
- NSCHED_PID
- NSCHED_UIC
- NSCHED_SYNC_JOB_NUMS
- NSCHED_PRE_STATUS
- NSCHED_POST_STATUS
- NSCHED_JPRIORITY
- NSCHED_QPRIORITY
- NSCHED_LAST_START
- NSCHED_LAST_END
- NSCHED_TIMEOUT
- NSCHED_START_TIME
- NSCHED_SYNC_TIME
- NSCHED_SYNC_NODES
- NSCHED_STATUS_FLAG

- NSCHED_REQUEST_FLAG
- NSCHED_RESTART_PARM
- NSCHED_PRE_FUNCTION
- NSCHED_POST_FUNCTION
- NSCHED_SCHED_INTERVAL
- NSCHED_NOSTART_TIME
- NSCHED_DOW
- NSCHED_USERNAME
- NSCHED_JOBNAME
- NSCHED_COMMENT
- NSCHED_LOGFILE
- NSCHED_OPENVMS_COMMAND
- NSCHED_MAIL_TO
- NSCHED_CLUSTER_NODE
- NSCHED_GROUP
- NSCHED_TYPE
- NSCHED_FISCAL
- NSCHED_CURRENT_NODE
- NSCHED_NO_DEPON
- NSCHED_RERUN_FREQUENCY
- NSCHED_RERUN_ATTEMPTS
- NSCHED_RERUN_COUNT
- NSCHED_RDID
- NSCHED_WRID
- NSCHED_EXID
- NSCHED_STALL_JOB
- NSCHED_TIMEOUT_JOB
- NSCHED_EXEC_TYPE
- NSCHED_REMOTE_SPECIFIC

- NSCHED_BATCH_TIME
- NSCHED_BATCH_QUEUE
- NSCHED_BATCH_ENTRY
- NSCHED_BATCH_CPULIM
- NSCHED_BATCH_CHARACTER

See VSS$CREATE_ITEM for tables that list field indicates and field descriptions for the callable routines VSS$SHOW_ITEM, VSS$CREATE_ITEM, and VSS$MODIFY_ITEM.

Possible Return Values

Return Value	Meaning
NSCHED$_SUCCESS	Operation completed successfully
NSCHED$_NOSUCHJOB	No such job in database
NSCHED$_BADITEM	Bad Item Code specified to callable routine
NSCHED$_NOPRIV	Insufficient privilege to perform action
NSCHED$_FLDINCTYPE	Field inconsistent with Job type
NSCHED$_FLDNOTSUPP	Field not supported in current version

VSS$SHOW_RETRY

VSS$SHOW_RETRY — Performs one of two actions based on the value of the routine's *opcode* argument – fills a buffer with printable strings for the network interval, network expiration, and network alias; reads NSCHED$:RETRY.DAT and prints to SYS$OUTPUT detailed information about retry operations in progress.

Format

VSS$SHOW_RETRY ***opcode, network_info, server_node***

Arguments

opcode

OpenVMS usage	longword_unsigned
type	unsigned longword
access	read-only
mechanism	by reference

Specifies one of two actions:

Value	Action
0	Fills the *network_info* buffer with printable strings for the network interval and network expiration. If you also specify the *server_node*, the routine returns this information for the remote OpenVMS Cluster to which the *server_node* belongs.

Value	Action
1	Reads NSCHED$:RETRY.DAT and prints to SYS$OUTPUT detailed information about retry operations in progress. Does not fill the *network_info* buffer. Does not perform any operation if you specify the *server_node*.
Other	Undefined.

network_info

OpenVMS usage	char_string
type	character string
access	write-only
mechanism	by descriptor

Receives the following 32 bytes of information:

First 6 bytes	Network alias for this OpenVMS Cluster
Next 13 bytes	ASCII value for the network interval, in the form "dddd hh:mm:ss"
Next 13 bytes	ASCII value for network expiration, in the form "dddd hh:mm:ss"

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A node in the OpenVMS Cluster for which the information specified by opcode is to be returned. A blank string ("") specifies the local OpenVMS Cluster.

Possible Return Values

Return Value	Meaning
1	Success
LIB$_STRU	Alternate success status, with string truncated
NSCHED$_NOPRIV	Insufficient privilege to perform action
System service error code	Error code returned by system services
RPC or DECnet error code	An error occurred connecting to a remote node

VSS$SHOW_SCHED

VSS$SHOW_SCHED — Returns information about a Job Management Manager server running on a particular node. The information returned is the same information displayed by the DCL command **SCHEDULE SHOW STATUS**. The information is returned in a 100-character fixed-length string. For more information on description of the buffer and its fields, see the chapter *Chapter 2, "Job Management Manager Programming"*.

Format

VSS$SHOW_SCHED *nodename, server_node, sched_buffer*

Arguments

nodename

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The specific node to return information about. A blank string ("") returns information about the default VSI JM Manager node.

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The remote node of which *nodename* is a cluster member. If the node is local, specify a blank string ("").

sched_buffer

OpenVMS usage	char_string
type	character string
access	write-only
mechanism	by descriptor

A 100-byte string or record that contains all the information about VSI JM Manager running on the specified node.

Return Value	Meaning
1	Success
NSCHED$_NOSCHED	No VSI JM Manager is available to service the request
NSCHED$_NOPRIV	Insufficient privilege to perform action
System service error code	Error code returned by system services
RPC or DECnet error code	An error occurred connecting to a remote node

VSS$SYNC_CHECK

VSS$SYNC_CHECK — Returns the number of local jobs that depend on the job specified by the routine's *job_number* and *job_node* arguments. If you specify the *server_node* argument, this operation is performed for remote job databases.

Format

VSS$SYNC_CHECK *job_number, job_node, server_node*

Arguments

job_number

OpenVMS usage	longword_signed
type	longword integer (signed)
access	read-only
mechanism	by reference

The job number for which to return the number of local jobs that depend on this job.

job_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The *job_number* node. If *job_number* is a local job, specify a blank string ("").

server_node

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

A node in a remote OpenVMS Cluster. If you want information about dependencies in your local OpenVMS Cluster, specify a blank string ("").

Possible Return Values

The number of jobs that depend on this node or job (a longword).

Note

This routine does not return status. Instead, the routine calls LIB$SIGNAL if an error occurs within the routine.

VSS$VALIDATE_TIME

VSS$VALIDATE_TIME — Checks that the Job Management Manager starting time and interval strings are valid. The manager has its own syntax for dates and times. All time strings should be validated by this routine before they are used. For more information on description of the valid VSI Job Management Manager date and time syntax, see the chapter *Chapter 2, "Job Management Manager Programming"*.

Format

VSS$VALIDATE_TIME *time_string_type, time_string, return_status*

Arguments

time_string_type

OpenVMS usage	longword_unsigned
type	unsigned longword
access	read-only
mechanism	by reference

Specifies whether the *time_string* is for a schedule interval or for the next scheduled starting time.

1	The ***time_string*** is a schedule interval
2	The ***time_string*** is the next scheduled time
Other	Invalid. Returns an error code of 4

time_string

OpenVMS usage	char_string
type	character string
access	read-only
mechanism	by descriptor

The *time_string* to check for validity.

return_status

OpenVMS usage	cond_value
type	longword integer (signed)
access	write-only
mechanism	by reference

A longword that receives one of the possible condition values returned.

Possible Condition Values

Value	Meaning
1	The string is valid. For a starting time, this means that the time is the same or later than the current time
2	The starting time is valid, but is earlier than current time (only returned for the starting time)
SYSTEM-F-IVTIME	The string is not valid
4	Unsupported validation type
System service error code	Error code returned by system services

Value	Meaning
Negative of BASIC error code	See the VSI BASIC documentation

For more information on description of the valid VSI Job Management Manager starting time and interval syntax, see the chapter *Chapter 2, "Job Management Manager Programming"*.

Possible Return Values

This is a subroutine. The status of the call is returned in the `return_status` call.

Chapter 2. Job Management Manager Programming

2.1. Program Examples

The installation procedure places example programs that use callable routines in the NSCHED$ directory. You can edit these examples and use them as a basis for your own programs, where applicable.

The following files are included with the Job Management Manager kit:

- EXAMPLE_CREATE.C

  This file provides an example of how to create a job using the API routines provided by the manager. It provides an example of initializing a job item, and the source code has comments in it explaining the example. The program is written in C and calls the routine VSS$CREATE_ITEM. See comments in EXAMPLE_CREATE.C for instructions on how to compile, link, and run the program. For more information on the available callable routines and how to use them, see *Chapter 1, "VSI Job Management Manager API"*.

- SBQUEUE.COM

  This command file scans the SLS$SYSBAK directory for *_SBK.COM and (*.SBK from prior SLS versions) files that are set up for automatic scheduling. Files found are queued for execution with a **`SUBMIT/AFTER=time`** command. This file should be placed in the SLS$CUSTOM directory.

- CONSOLE_SCAN.TEMPLATE

  This file provides a Console Management scan profile for Job Management Manager jobs. It includes jobs rescheduled because of NOT_ON Special Day Restrictions and jobs being held due to ON_ONLY Special Day Restrictions.

  To use this file, issue the following commands:

  ```
  $ CONSOLE EDIT /INTEFACE=CHAR
  CM EDIT> @CONSOLE_SCAN.TEMPLATE
  CM EDIT> EXIT
  $ CONSOLE RECONFIG
  ```

  Note

  You must have VSI Console Management installed to use these commands.

- NSCHED_JOB_FILES.H

  This file provides information about fields used for the VSS$CREATE_ITEM, VSS$MODIFY_ITEM, and VSS$SHOW_ITEM callable routines.

2.2. Job Management Manager Date and Time Specification

This section describes the date and time syntax for the Job Management Manager starting time and schedule interval.

Note

If you do not specify a time, the default is "00:00:00" unless otherwise noted.

Note that several Job Management Manager commands use qualifiers such as **/MAX_TIME** and **/STALL_NOTIFY** that require that you specify time intervals using OpenVMS delta time. For more information on OpenVMS delta time, see the *VSI Job Management for OpenVMS Administration Guide* [https://docs.vmssoftware.com/vsi-job-management-administration-guide-v31/].

2.2.1. Valid Schedule-Interval Syntax

The maximum length of a schedule-interval string is 14 characters.

Use This Syntax	To Choose This Interval…
M [dd] [hh:mm:ss.cc]	Monthly on the given day. (The calendar is not fiscal.) The default day is 1, if not specified. If the number of the specified day is greater than the number of days in the next scheduled month, then the job will be scheduled for the last day of the month (for example, if the specified day is 31 and the next month has only 30 days).
D [hh:mm:ss.cc]	Daily at hh:mm:ss.cc.
H [mm:ss.cc]	Hourly at mm:ss.cc after each hour.
+DDDD [hh:mm:ss.cc]	Job Management Manager delta time. The + symbol is required. To specify no days, you must use 0.
all blank	No interval.
NONE	No interval.
0	No interval. The manager runs the job continuously.
Fiscal Interval	See Fiscal Starting Time Syntax section

2.2.2. Valid Next-Scheduled-Time Syntax

Use This Syntax…	To Choose This Time…
NOW	Schedule the job to run now. You cannot abbreviate this syntax.
NEVER	No starting time. You cannot abbreviate this syntax.
dd-mmm-yy hh:mm:ss.cc	Absolute date. You can omit trailing fields.
+DDDD [hh:mm:ss.cc]	Job Management Manager delta time. The + symbol is required. To specify no days, you must use 0.
TOMORROW [hh:mm:ss.cc]	Tomorrow plus delta time. The shortest abbreviation is TOM.

Use This Syntax…	To Choose This Time…
Fiscal Starting	Time See the Fiscal Starting Time section.

2.2.3. Fiscal Starting Time Syntax

Syntax:

```
F [Yyy] [Qqq] [Mmm] [Www] [Dddd] [hh:mm:ss]
```

Spaces do not matter, except that there must be a space before the time expression (hh:mm:ss).

Defaults:

Year	Current fiscal year
Day	1
Time	00:00:00

Each individual date parameter is contained within the scope of the previous date parameter. For example:

```
Year = current fiscal year.
F Q3 M2 = month 2 in quarter 3.
F Y97 M2 = month 2 in the year.
```

F Y97 M9 is valid, but F Y97 Q3 M9 is an invalid fiscal starting time, because there are only 3 months in a quarter.

The manager works with 5-week months and 53-week years. It will not allow F W53 if the year has only 52 weeks, or F M2 W5 because the month has only 4 weeks.

Examples:

The following examples are based on a current fiscal year of FY96.

FY 97 Q2 12:	Fiscal year 97, day 1 of the second quarter at noon.
Fm12w 5 D7	Current fiscal year, month 12 (for example, JUN), fifth week of the month, day 7 of the week (Saturday) at 00:00:00.
F Q4 w 14 D7 19:00	Current year, week 14 in the fourth quarter, day 7 (Saturday) at 7 P.M. (the last day of the fiscal year). Allow Q4 to have 14 weeks rather than the normal 13 weeks.
Fy97 D236	Day 236 of fiscal year 97.

Examples of Invalid Expressions:

The following expressions would be rejected by VSS$validate_time:

FM2W6	Month 2 of the year has only 4 weeks.
F M2W3 D9	The week has only 7 days.

FY90W3412:00	The time must be separated by a space.

2.2.4. Fiscal Interval

Syntax:

```
F [Y] [Q[qq]] [M[mm]] [W[ww]] [+|-] [D[ddd]] [hh:mm:ss.cc]
```

Specific date parameters are not required. The scope of each parameter is contained within the previous parameter, as in the fiscal starting time.

Spaces do not matter, except in two cases:

- A specific date parameter must follow directly after its specifier with no spaces. (For example, Q4 is correct, but Q 4 is incorrect.)
- A space must precede the time expression if the time expression is used.

Defaults:

Day	1
Sign	+
Time	00:00:00

Examples:

The following are examples of valid fiscal intervals:

F W	Weekly on day 1 (Sunday) at midnight.
FQD5	Quarterly on the fifth day of each quarter.
Fq4m3 -d3	Yearly, three days before the end of the third month of Q4.
FQ -d1 12:00	The last day of each quarter at noon.
F M w3 +D2	The second day of the third week of each month.
FM-D7	The first day of the last week in each fiscal month.

Invalid Examples:

If you specify an interval that does not exist for some time periods, then the manager will reduce the interval so that it fits the time period. For example, FMW5 becomes FMW4 for 4-week months, and FQ W14 becomes FQ W13 for 13-week quarters.

2.3. NSCHED$_ERROR Return Status Codes for Routines

This section describes the NSCHED$_ERROR codes returned by Job Management Manager routines.

Some Error Code Descriptions contain a letter indicating the following types:

- I – Informational

- W – Warning Status
- E – Error Status
- F – Fatal Status

NSCHED$_ERROR Code	**Description**
1	Success
NSCHED$_ADJFLDVAL	Field valueadjusted to be within limits
NSCHED$_BADITEM	Bad Item Code specified to callable routine
NSCHED$_BADVALUE	Bad *item_list* value specified
NSCHED$_CNTMODFLD	Caller cannot modify this field
NSCHED$_CANTOPNDB	Privilege violation, could not open the Job Management Manager database (F)
NSCHED$_CLASSCREATED	Special Day Class created
NSCHED$_CLASSDELETED	Special Day Class deleted
NSCHED$_CLASSEXISTS	Special Day Class exists
NSCHED$_CLASSMODIFIED	Special Day Class modified
NSCHED$_COMBNOTALL	Combined Restrict-To and Restrict-From expression not allowed
NSCHED$_DUPLCLASS	Duplicate Special Day Class Name
NSCHED$_DUPLNAM	Duplicate job name is not permitted (F)
NSCHED$_EMPTYCLASS	No days current specified in Special Day Class
NSCHED$_FLDINCTYPE	Field inconsistent with Job type
NSCHED$_FLDNOTSUPP	Field not supported in current version
NSCHED$_FLDTOOLONG	Specified field is too long
NSCHED$_IDOUTOFRNG	Identifier value is out of range
NSCHED$_INAPP_VALUE	Inappropriate Value
NSCHED$_INSUFFSIZE	Insufficient Buffer Size
NSCHED$_INVARG	Routine was called with an invalid string argument (F)
NSCHED$_INVBITMASK	Invalid YEAR_BITMASK Specification
NSCHED$_INVBUFFSPEC	Invalid buffer specification
NSCHED$_INVCHARFND	Invalid character found in Special Day Class Name
NSCHED$_INVMODEVAL	Invalid mode value; value must be DETACHED, BATCH or REMOTE
NSCHED$_INVQUALVAL	Invalid qualifier value (W)
NSCHED$_INVSTRTIME	Invalid time string (W)
NSCHED$_INVTYPECODE	Invalid type code in ITEM_BLOCK
NSCHED$_JOBDEPSEXIST	Job dependencies exist for Special Day Class
NSCHED$_JOBRESTMODIF	Job Restrictions successfully modified
NSCHED$_LEADNUMNOTALL	Leading number not allowed in Special Day Class name
NSCHED$_MORE	Additional items or jobs remain (I)

NSCHED$_ERROR Code	Description
NSCHED$_NAMTOOLONG	Special Day Class name exceeds maximum length
NSCHED$_NODATABASE	Job Management Manager database was not found (F)
NSCHED$_NOITEMLIST	No *item_list* specified
NSCHED$_NOMORE	No more items or jobs found (I)
NSCHED$_NOPRIV	Insufficient privilege to perform action (F)
NSCHED$_NORESTMASKS	No Special Day Restriction YEAR_BITMASK found for job
NSCHED$_NORESTRICT	No Special Day Restrictions for job
NSCHED$_NORESTTHATYEAR	No Special Day Restrictions that year for job
NSCHED$_NORPC	Product was built without wide area network support (F) s codes
NSCHED$_NOSAMEJOB	Timeout Job must not be same job
NSCHED$_NOSCHED	No manager instance available to service request (W)
NSCHED$_NOSDTHATYEAR	No days in Special Day Class for the specified year
NSCHED$_NOSUCHCLASS	Special Day Class does not exist
NSCHED$_NOSUCHID	Unknown rights identifier
NSCHED$_NOSUCHJOB	No such job in database (F)
NSCHED$_NOSUCHUSER	No such user (F)
NSCHED$_NOSYNCHDAT	The synchronization database is corrupted or missing (F)
NSCHED$_NOTDONE	Job is already running (I)
NSCHED$_NOTONLIST	Special Day Class not on job's Restrict-To or Restrict-From List
NSCHED$_NOTRESTRICTED	Input date is not Special Day Restricted
NSCHED$_NOTVAXCLUSTER	Not an OpenVMS Cluster (W)
NSCHED$_NSRTUNRESOLVED	Next Scheduled Run Time for job could not be resolved
NSCHED$_ONEDEPFLD	Dependency nodes and numbers cannot be updated separately
NSCHED$_RESFRMNOTFND	Special Days Restrict-From expression not found
NSCHED$_RESTONOTFND	Special Days Restrict-To expression not found
NSCHED$_RESTRICTEXISTS	Special Days Restrictions exist for job
NSCHED$_SPECDAYHOLD	Job is in HOLD state due to Special Day Restrictions
NSCHED$_SPECDAYSKIP	Job was rescheduled due to Special Day Restrictions
NSCHED$_SUCCESS	Operation completed successfully
NSCHED$_TIMBEFOR	Start time is before current time; job will run immediately
NSCHED$_WLDNAME	Invalid wildcard operation (W)
NSCHED$_YEARMISMATCH	Year of input date does not match year of input YEAR_BITMASK

2.4. Return-Buffer Layout for the VSS$SHOW and VSS$SHOW_SCHED Routines

This section describes the format and contents of the job data buffer returned by the Return-Buffer Layout for the VSS$SHOW and VSS$SHOW_SCHED routines.

2.4.1. Return-Buffer Layout for the VSS$SHOW Routine

The size of the return buffer for the VSS$SHOW() subroutine is 1050 bytes. The following table lists the size and format of the buffer's elements:

Element	Number of Bytes	Format
status_flag	1	Character
last_start	23	ASCII date and time
last_end	23	ASCII date and time
last_status	8	Hexadecimal string
sched_interval	23	ASCII date and time
next_q	23	Next scheduled time, as an ASCII date and time
record_flags	8	Numeric string
DOW	7	Days of week as 1111111
UIC	12	String of the form [g,m]
success_count	8	Numeric string
failure_count	8	Numeric string
PID	8	Hexadecimal string
OPENVMS_cmd	132	String
logfile	50	String
user_name	32	String
comment	80	String
request_flag	1	Character
mail_to	50	String
cluster_node	6	String
job_name	40	String
group	40	String
type	40	String
fiscal	16	Fiscal starting time
timeout	23	ASCII date and time
sync(16)	8	Numeric strings
sync_nodes	96	Node name list
tag	8	This job's primary key in database
current_node	6	Node on which we are currently running

Element	Number of Bytes	Format
sync_time	23	Job's dependency time
prefunc	40	String
postfunc	40	String
nostart	23	ASCII date and time
pre_status	8	Hexidecimal string
post_status	8	Hexidecimal string
jpriority	4	Decimal string (0 to 16)
qpriority	4	Decimal string (0 to 255)

2.4.1.1. Description of Fields

status_flag

H = on hold

R = running

D = waiting for a job dependency

S = scheduled to run

J = waiting for job slot to run in

Q = queued

last_start

Last time the job ran, in ASCII format.

last_end

Last finish time of job, in ASCII format.

last_status

Exit status of last run, as a hexadecimal string.

sched_interval

M dd hh:mm:ss	Run the job monthly (calendar) on the specified day.
D hh:mm:ss.cc	Run the job daily at the time specified by hh:mm:ss.cc
H mm:ss.cc	Run the job hourly at mm:ss.cc past the hour.
+dddd hh:ss:mm	Run the job at specified delta time (dddd hh:mm:ss after last completion).
0	Run continuously (restart the job when it completes).
Fiscal Interval	See section, Fiscal Interval for a description.

next_q

The next time the job is scheduled to run, in ASCII format (dd-mmm-yyyy hh:mm:ss.cc).

record_flags

A 32-bit binary value converted to decimal string format.

Bit	Function (If Set)
0	Retain the job in database on error.
1	Retain the job in database on success.
2	Send mail to *mail_address* on error.
3	Send mail to *mail_address* on success.
4	Restart the job if the CPU fails while executing.
5	Notify the user's terminal when the job completes.
11	Retry the job if it completes with an error status.
12	Add the interval to the next scheduled time instead of the time job actually starts running, to calculate the next time to run.
16	The job is restricted by one or more Special Day Classes.
17	Send mail to *mail_address* if a Special Day Action occurs.
19	The job's Special Day Restriction is to run only on Special Days.
20	The job's Special Day Restriction is to not run on Special Days.
21	The job's Special Day Action is to hold, instead of skip.
22	The job's last run was a successful run, instead of a failure.

DOW

A 7-digit binary string specifying the days of the week to run job. Positions run Monday through Sunday. A job that runs on Monday and Saturday would have 1000010 as its DOW field.

UIC

A string of the form [g,m].

success_count

The number of times the job has succeeded, in decimal string form.

failure_count

The number of times the job has failed, in decimal string form.

PID

If the job is currently running, the field has the process identification (PID of the created process, as a hexadecimal string.

OPENVMS_cmd

The OpenVMS command associated with the job.

logfile

The file spec specified as SYS$OUTPUT for the job.

user_name

The username that the job will run under.

comment

A comment associated with the job.

request_flag

An unprocessed operation that is currently requested of the manager for the job.

mail_to

The address where mail is to be sent.

cluster_node

The node that the job is scheduled to run on.

job_name

The name of the job, if present.

group

The name of the job's group, if present.

type

The name of the job's type, if present.

fiscal_start

The fiscal starting time expression, if any. This is the user's starting time expression before it was converted to a regular calendar date time. The maximum length is 16 characters.

timeout

The delta time format string that specifies an interval after the *start_time* during which the job must be converted; otherwise, mail is sent to *mail_to*.

sync 1 through 16

Decimal strings that represent job numbers of jobs that must complete successfully before the one can run. These are sorted in ascending order; empty elements contain zeros.

sync_nodes

The nodes associated with the job dependencies for the job. This string is composed of 16 individual nodenames (six characters eachthat identify the remote node the dependency resides on.

If the sync is local, then the corresponding segment of the string is filled with six blank spaces. Each nodename is padded out to six characters. If the sync in the sync array element 4 is remote, then the nodename is specified in characters 19 to 24 of string SYNC_NODES.

tag

The job number of the job, in decimal string format.

current_node

The node on which the job is running, if it is currently running.

sync_time

A 23-character ASCII date-and-time string. It holds the time when the job last ran *or* the last time its synchronization list was modified, whichever is later.

Use the time to check whether dependencies have been satisfied. The last end time of jobs it depends on must be later than *sync_time*; otherwise, the job is still waiting on those dependencies.

pre_func

The prefunction OpenVMS command that executes before the main job command.

post_func

The postfunction OpenVMS command that executes after the main job command.

nostart_time

Delta-time format string that specifies elapsed time since a job's scheduled run time by which a job must have actually started. If the job does not start in this time, the owner and *mail_destination* (if different) receive notification.

pre_status

The latest completion status of <argument>(pre_function, as a hexadecimal string.

post_status

The latest completion exit status of post_function, as a hexadecimal string.

jpriority

The run priority of the job, a decimal string from 1 to 16.

qpriority

A decimal string from 0 to 255. Not implemented.

2.4.2. Return-Buffer Layout for VSS$SHOW_SCHED Routine

The size of the return buffer for the VSS$SHOW_SCHED routine is 100 bytes. The following table lists the size and format of the buffer's elements:

Element	Number of Bytes	Format
sched_node	7	String
sched_ident	9	String
sched_start	20	ASCII date and time

Element	Number of Bytes	Format
sched_jobcount	5	Decimal string
sched_jobmax	6	Decimal string
sched_log	6	Decimal string
sched_prio	5	Decimal string
sched_rating	7	Decimal string
sched_ASCII_flags	42	String

2.4.2.1. Description of Fields

sched_node

The name of the node that the returned information is about.

sched_ident

The version of the manager running on the node. For example: V1.0-00.

sched_start

Date and time when the manager last started up on the node.

sched_jobcount

The number of Job Management Manager jobs currently running on the node.

sched_jobmax

The maximum number of jobs that can run at the same time on the node.

sched_log

The event classes that are being logged on the node.

sched_prio

The default priority at which the manager on the node will run jobs.

sched_rating

If load balancing is enabled, the system rating for load balancing calculations.

sched_ASCII_flags

Displays Job Management Manager settings currently in use for the node. The only setting currently implemented is load balancing. If debug is turned on, SCHED_ASCII_FLAGS returns the string " D". Otherwise, it returns a blanks string.

2.5. Job Management Manager Event Log File Layout and Organization

This section shows the layout and organization of the Job Management Manager event log file.

The manager maintains an event log. You can log thirty-two classes of events (11 implemented). Each class includes one or more event types. Using the command **SCHED SET LOGGING**, you can turn logging on or off on a class-by-class basis.

The event log is an indexed RMS file with variable-length records. There are two keys. The primary key is job number (which is 0 for nonjob events). The alternate key is the quadword OpenVMS time that the event was logged.

2.5.1. BASIC Map

The following is a map (in BASIC) that defines the records:

```
MAP (M$LOGREC)              &
LONG L_TAG,               ! CA JM Manager job number &
WORD L_EVENT_TYPE,        ! 32,000 belonging to 32 classes &
LONG L_PID,               ! PID &
QUADWORD L_TIME,          ! Time that the record was inserted &
STRING L_NODE = 6%,       ! Node where the event happened &
STRING L_USER = 32%,      ! OpenVMS user name &
STRING L_DATA = 100%      ! Free-form data
! records are varying length)
```

2.5.2. Job-end Records with Accounting Information

The following is an alternate description for job-end records with accounting information:

```
MAP (M$LOGREC)              &
LONG L_TAG,               ! CA JM Manager job number &
WORD L_EVENT_TYPE,        ! 32,000 belonging to 32 classes &
LONG L_PID,               ! PID &
QUADWORD L_TIMEX,         ! Current time &
STRING L_NODE = 6%,       ! Our node &
STRING L_USER = 32%,      ! OpenVMS user name &
STRING FILL$ = 4%,        ! Filler &
LONG L_FINALSTS,          ! Exit status code &
LONG FILL,                ! Filler &
LONG FILL,                ! Filler &
QUADWORD L_TERMTIME,      ! Time of process termination &
STRING L_ACCOUNT = 8%,    ! Account name for process &
STRING L_USERNAME = 12%,  ! OpenVMS user name &
LONG L_CPUTIM,            ! CPU time used by the process &
LONG L_PAGEFLTS,          ! Page faults incurred by process &
LONG L_PGFLPEAK,          ! Peak paging file usage &
LONG L_WSPEAK,            ! Peak working set size &
LONG L_BIOCNT,            ! Buffered I/O count for process &
LONG L_DIOCNT,            ! Direct I/O count for process &
LONG L_VOLUMES,           ! Count of volumes mounted b
                          ! process &
QUADWORD L_LOGIN,         ! Time process logged in &
LONG L_OWNER              ! Process identification of owner
```

The string function EVENT_TYPE (included in NSCHED_SUBS) takes one longword argument, an event number. EVENT_TYPE returns a 12-byte ASCII string that describes the event, such as **JOB START** or **JOB FINISH**. To convert from an event number to an event class, divide the event number by 1000, discarding the remainder. To see the defined event classes, enter the following commands:

```
$ SCHED SET LOG -1      ! turn on all logging bits
$ SCHED SHOW LOGGING
```