ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) Target Family Guide

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900
U.S.A.

Part Number 806-5258–10
July 2000

Contents

CHAPTER 1

ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) Target Family Guide

This guide describes how to build the ChorusOS™ 4.0.1 Simulator product for Solaris™ (SPARC™ Platform Edition).

Preface

How This Guide is Organized

ChorusOS 4.0.1 for the Simulator product specific information is provided in the following major sections:

- “Development Environment” on page 8, includes supported hosts, host operating systems and development systems.
- “ChorusOS Supported Features” on page 9, includes kernel components, POSIX components and product specific components.
- “Libraries” on page 12 lists the libraries included in the ChorusOS operating system.
- “Utilities” on page 13, includes host and target utilities.
- “Reference Hardware” on page 15, includes supported reference platform, and supported devices.
- “Building the ChorusOS Simulator” on page 16 describes how to build the ChorusOS Simulator.

- Appendix A, presents specific man pages for the simulator, and additional man pages for the MONITOR feature (these pages are not available for on-line search using the `man` command).
- Appendix B, details the list of Solaris packages in the product components, and the associated part numbers.

Related Books

See the *ChorusOS 4.0 Installation Guide for Solaris Hosts* for a description of the installation process of the ChorusOS 4.0 Simulator on a host workstation running the Solaris™ operating environment. This document also describes how to set up a boot server running the Solaris operating environment.

See the *ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) User's Guide* for information on configuring and running the simulator system image.

See the *ChorusOS 4.0 Introduction* for a complete description of the ChorusOS features.

Typographical Conventions

The following table describes the typographic changes used in this book.

TABLE 1–1 Typographical Conventions

Typeface or Symbol	**Meaning**	**Example**
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls –a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with on-screen computer output	`machine_name%` **`su`** `Password:`

TABLE 1–1 Typographical Conventions *(continued)*

Typeface or Symbol	Meaning	Example
AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type **rm** *filename*.
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You must be *root* to do this.

Shell Prompts

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE 1–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
C shell superuser prompt	`machine_name#`
Bourne shell and Korn shell prompt	`$`
Bourne shell and Korn shell superuser prompt	#

Ordering Sun Documents

Fatbrain.com, an Internet professional bookstore, stocks selected product documentation from Sun Microsystems, Inc.

For a list of documents and how to order them, visit the Sun Documentation Center on Fatbrain.com at `http://www1.fatbrain.com/documentation/sun`.

Accessing Sun Documentation Online

The docs.sun.com℠ Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is `http://docs.sun.com`.

Obtaining Technical Support

Sun Support Access offerings are available exclusively to members of the Sun Developer Connection Program. To get free membership in the Sun Developer Connection Program, go to `http://www.sun.com/developers`. For more information or to purchase Sun Support Access offerings, visit: `http://www.sun.com/developers/support` or contact the Sun Developer Connection Program office near you.

Development Environment

In the context of the ChorusOS 4.0.1 Simulator, applications are developed on a workstation (the host), and then downloaded and executed on a workstation of the same type, acting as a target.

A cross development system is needed to build the applications that execute on the target system (see Section “Utilities” on page 13).

Solaris™ Reference Host Environment

Prerequisites for the Solaris host reference configuration are the following:

- Sun SPARCstation™.
- Solaris 2.6, or Solaris 7 (32-bit).
- Sun WorkShop™ 5.0 native compiler.

Note - In order for the CC compiler to work properly, all patches related to the CC compiler must have been installed on the Solaris system.

- JDK™ 1.1.8, for the installation tool.
- JDK 1.2, for the graphical configuration tool and for Java™ applications.

Cross Compiler

This development environment component is bundled with the ChorusOS Simulator product:

- Chorus Cross Development System 5.0, SPARC ELF.

The Chorus Cross Development System is based on the Experimental GNU Compiler System egcs 1.1.2 and binutils 2.9.1 and additional patches.

Graphical Debugger

This development environment component is bundled with the ChorusOS Simulator product:

- XRAY Debugger from Mentor Graphics, ELF format, version 4.4crd.

ChorusOS Supported Features

The following table shows the ChorusOS kernel and operating system optional features that are available for the Solaris Operating Environment (SPARC Platform Edition) processor family. The availability status of a feature can be one of:

Y	The feature is supported, and is configurable using the `configurator`(1CC) command, or with the *ews* GUI configuration tool. Please refer to the note at the end of the table for information about specific conditions, or restrictions, for a given supported feature. Some of the features (such as MSDOSFS, FLASH, FS_MAPPER, for example) require specific low-level drivers. These features operate only on platforms which provide these drivers.
N	The feature is not supported.

Feature Description	Feature Name	Availability
Actor management		
Dynamic actor loading management	ACTOR_EXTENDED_MNGT	Y
User-mode extension support	USER_MODE	N
Dynamic libraries	DYNAMIC_LIB	N
Compressed file management	GZ_FILE	Y
Scheduling		
POSIX round-robin scheduling class	ROUND_ROBIN	Y
Memory management		
Virtual (user and supervisor) address space	VIRTUAL_ADDRESS_SPACE	N
On-demand paging	ON_DEMAND_PAGING	N
Hot restart and persistent memory		
Hot restart	HOT_RESTART	N
Inter-thread communication		
Semaphores	SEM	Y
Event flag sets	EVENT	Y
Mutual exclusion lock supporting thread priority inversion avoidance	RTMUTEX	Y
Monitors	MONITOR	Y
Time management		
Periodic timers	TIMER	Y
Thread and actor virtual timer	VTIMER	Y
Date and time of day	DATE	Y
Real-time clock	RTC	Y
Inter-process communication		
Location-transparent inter-process communication	IPC	Y
Remote (inter-site) IPC support	IPC_REMOTE	Y
Remote IPC communications medium	IPC_REMOTE_COMM	N

Feature Description	Feature Name	Availability
Mailbox-based communications mechanism	MIPC	Y
POSIX 1003.1-compliant message queues	POSIX_MQ	Y
POSIX 1003.1-compliant shared memory objects	POSIX_SHM	N
LAP		
Local name server for LAP binding	LAPBIND	Y
LAP validity-check option	LAPSAFE	Y
Tools support		
Message logging	LOG	Y
Profiling and benchmark support	PERF	N
System Monitoring	MON	Y
System debugging	DEBUG_SYSTEM	N
C_INIT		
Basic command interpreter on target	LOCAL_CONSOLE	Y
Remote shell	RSH	Y
File system options		
Named pipes	FIFOFS	Y
MS-DOS file system	MSDOSFS	Y
NFS client	NFS_CLIENT	Y
NFS server	NFS_SERVER	Y
UFS file system	UFS	Y
I/O management		
Network packet filter	BPF	Y
Swap support	FS_MAPPER	N
Driver for IDE disk	IDE_DISK	N
`/dev/mem`, `/dev/kmem`, `/dev/null`, `/dev/zero`	DEV_MEM	Y
Support for RAM disk	RAM_DISK	Y
Support for FLASH media [1]	FLASH	N

Feature Description	Feature Name	Availability
Virtual TTY	VTTY	N
Driver for SCSI disk	SCSI_DISK	N
Support for IPC	IOM_IPC	N
Support for OSI	IOM_OSI	N
Networking		
Serial link IP	SLIP	Y [2]
POSIX 1003.1g-compliant sockets	POSIX_SOCKETS	Y
Point-to-point protocols	PPP	Y [2]
Local sockets and pipes	AF_LOCAL	Y
Administration		
ChorusOS statistics	ADMIN_CHORUSSTAT	Y
`ifconfig` administration command	ADMIN_IFCONFIG	Y
`mount` administration command	ADMIN_MOUNT	Y
`rarp` administration command	ADMIN_RARP	Y
`route` administration command	ADMIN_ROUTE	Y
`shutdown` administration command	ADMIN_SHUTDOWN	N
`netstat` administration command	ADMIN_NETSTAT	Y
JVM		
Java Virtual Machine	JVM	N

1. Logical-to-Physical block mapping for flash file system support.
2. Requires a UART driver to be developed

Libraries

The ChorusOS operating system provides the following list of libraries:

ChorusOS embedded library [1]	`libebd.a`
ChorusOS extended library [1]	`libcx.a`
C++ library	`libC.a`
X11 related client libraries (not thread safe)	`libX11.a`, `libXaw.a`, `libXext.a`, `libXmu.a`, `libXt.a`
Specific BSD APIs (not thread safe)	`libbsd.a`
The SunRPC library	`librpc.a`
The mathematical library	`libm.a`

1. The `libebd.a`, `libcx.a`, `libm.a` and `libC.a` libraries have been made thread-safe in order to support multithreaded actors.

Utilities

Target Utilities

The following utilities may be run on the target ChorusOS operating system:

chorusStat(1CC)

cp(1CC)

cs(1CC)

date(1CC)

dd(1CC)

df(1CC)

domainname(1CC)

ftp(1CC)

hostname(1CC)

ls(1CC)

mkdir(1CC)

mkfifo(1CC)

mv(1CC)

netstat(1CC)

nfsstat(1CC)

pax(1CC)

rm(1CC)

rmdir(1CC)

touch(1CC)

uname(1CC)

arp(1M)

chat(1M)

chorusNS(1M)

chorusNSinet(1M)

chorusNSsite(1M)

disklabel(1M)

format(1M)

fsck(1M)

fsck_dos(1M)

ftpd(1M)

inetNS(1M)

inetNSdns(1M)

inetNShost(1M)

inetNSien116(1M)

inetNSnis(1M)

mkfs(1M)

mount(1M)

mount_msdos(1M)

mount_nfs(1M)

mountd(1M)

newfs(1M)

newfs_dos(1M)

nfsd(1M)

portmap(1M)

route(1M)

syncd(1M)

sysctl(1M)

telnetd(1M)

umount(1M)

ypbind(1M)

Host Utilities

The following utilities may be run on the host machine:

ChorusOSMkMf(1CC)

configurator(1CC)

configure(1CC)

ews(1CC)

mkmerge(1CC)

loader(1CC)

simudrv(1CC)

Note - The *loader* and *simudrv* host utilities are described in Appendix A.

Reference Hardware

Reference Target Platform (Hardware / Operating System)

The ChorusOS Simulator supports the following hardware:

- Sun SPARCstation™.

The ChorusOS Simulator supports the following operating systems:

- Solaris 2.6.
- Solaris 7, 32-bit mode.

Hosts for building the system image and for running the simulator must have the same operating system version (see the *ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) User's Guide*).

Reference BSP

The ChorusOS 4.0 Simulator supports the following devices:

Device Id	ChorusOS Driver
cpu (time base and decrementer)	simulator:solaris-timer
Pseudo Real-time Clock	unixrtc:-(rtc)
I/O Handler	simulator:solaris-sigio
Ethernet	sun:pseudo-ether

Building the ChorusOS Simulator

Building the Simulator System Image

The following procedures assume that the ChorusOS Simulator product has already been correctly installed on the host workstation. See the *ChorusOS 4.0 Installation Guide for Solaris Hosts.*

1. Create and change to a build directory where you will build system images:

```
$ mkdir build_dir
$ cd build_dir
```

2. **Set an environment variable to use with the** `configure`**(1CC) command as a shortcut to the base directory.**

   For example:

Set the environment variable…	To the family-specific product directory. The default value is…
DIR	/opt/SUNWconn/SEW/4.0.1/chorus-upSparc

3. **Make sure your path has been set correctly to include the directory** *install_dir*`/4.0.1/chorus-upSparc/tools/host/bin`**, where the default** *install_dir* **is** `/opt/SUNWconn/SEW`**.**

   Also make sure that your PATH includes `/usr/openwin/bin`, which contains the `imake` utility.

4. **Configure the build directory, using the** `configure`**(1CC) command.**

```
$ configure -b $DIR/kernel \
$DIR/os \
$DIR/tools \
-s $DIR/src/nucleus/bsp/up \
$DIR/src/nucleus/bsp/up/upSparcSolaris \
$DIR/src/nucleus/bsp/drv \
$DIR/src/iom
```

Note - The above command configures the build directory to include components installed during a "Default Install". It does not include optional components, such as the X library, code examples or components related to add-on packages, that you may choose to install separately on Solaris host workstations. For example, in order to include everything in your build environment:

```
$ configure -b $DIR/kernel \
$DIR/os \
$DIR/opt/X11 \
$DIR/tools \
-s $DIR/src/nucleus/bsp/up \
$DIR/src/nucleus/bsp/up/upSparcSolaris \
$DIR/src/nucleus/bsp/drv \
$DIR/src/iom \
$DIR/opt/examples
```

If you are building from the source distribution, see the *ChorusOS 4.0 Production Guide*.

As a result of configuration, *build_dir* contains a `Makefile`, which is used to generate the build environment, and a `Paths` file which specifies paths to files required by and created in the build environment.

5. **Generate the build environment:**

```
$ make
```

6. **Build a system image:**

```
$ make chorus
```

The resulting system image file is located in the build directory, *build_dir* and is called `chorus.RAM`.

Note - You can also make a smaller system image that includes only the operating system kernel:

```
$ make kernonly
```

Please refer to the *ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) User's Guide* for information on configuring and running your simulator system image.

APPENDIX **A**

ChorusOS 4.0.1 Simulator Additional Man Pages

The following man pages are not available for on-line use using the `man` command. They will be integrated with the package of man pages in a later major product release.

section 1CC: Host and Target Utilities

loader(1CC)
simudrv(1CC)

section 4CC: Files

simudrv.conf(4CC)
site_number.conf(4CC)

section 2K: Kernel System Calls

monitor(2K), *monitorInit(2K)*, *monitorGet(2K)*, *monitorNotify(2K)*, *monitorNotifyAll(2K)*, *monitorRel(2K)*, *monitorWait(2K)*

section 5FEA: ChorusOS Features and APIs

MONITOR(5FEA)

NAME

loader – boot simulator image

SYNOPSIS

loader simulator_image instance_number

DESCRIPTION

The `loader` command is a host utility.

The `loader` boots the system image specified by *simulator_image*, assigning it the specified *instance_number* required for network communication. This is done by allocating 32 MB of virtual memory for the *simulator_image* on the host, copying the system image to the allocated memory, configuring the UDP port associated with the system image based on information in the site configuration file, *site_number.conf*(4CC) , and finally activating the system image entry point.

As the simulator boots, system messages are displayed on standard output.

OPERANDS

The following operands are supported:

simulator_image — Path from the build directory to the simulator system image to boot, for example: `chorus.RAM`.

instance_number — Number between 1 and 254 inclusive, used by the `loader` to locate site configuration information for the *simulator_image* in the site configuration file. For example, the IP address of the simulator is derived from the *instance_number* (see *site_number.conf*(4CC)).

The following entry in the `sysadm.ini`(4CC) file for the *simulator_image* makes it possible to use the same system image, each with its own IP address:

```
ifconfig ifeth0 TAG.TGT.IPA.TAG netmask \
netmask broadcast broadcast
```

The `TAG.TGT.IPA.TAG` string is replaced by the IP address defined for *instance_number* when the system image is booted with the `loader`. *netmask* and *broadcast* must be replaced with the corresponding parameters for the subnetwork where the simulator runs.

ENVIRONMENT VARIABLES

The following host environment variables are supported:

CHORUS_SITE_NUMBER_FILE — Contains the path to the site configuration file, *site_number.conf*(4CC).

If this environment variable is set, it overrides the default path, `/usr/local/chorus/simu_admin/site_number.conf`.

FILES

`/usr/local/chorus/simu_admin/site_number.conf`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

site_number.conf(4CC)

ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) User's Guide

NAME

simudrv – Ethernet pseudo-driver start/stop script

SYNOPSIS

/bin/sh simudrv [start|stop]

DESCRIPTION

The `simudrv` command is a host utility.

The `simudrv` is used to start and stop the Ethernet pseudo-driver on the host. The Ethernet pseudo-driver provides a logical Ethernet adaptor allowing the simulator to communicate through with the host, and with other systems when IP forwarding and routing is enabled on the host. `simudrv` requires an Ethernet pseudo-driver configuration file, *simudrv.conf*(4CC), indicating the IP address associated with the pseudo-driver and the hostname of the host system.

`simudrv` must be run on the host as a root process.

OPERANDS

The following operands are supported:

start Start the Ethernet pseudo-driver

stop Stop the Ethernet pseudo-driver

FILES

`/usr/local/chorus/simu_admin/simudrv.conf`

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

simudrv.conf(4CC)

ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) User's Guide

NAME

simudrv.conf – Ethernet pseudo-driver configuration file

SYNOPSIS

`/usr/local/chorus/simu_admin/simudrv.conf`

DESCRIPTION

The ChorusOS simulator architecture includes an Ethernet pseudo-driver that handles communication between instances of the simulator running on the host and the host IP layer.

The Ethernet pseudo-driver configuration file specifies both the host name and the IP address associated with the Ethernet pseudo-driver.

The *ChorusOS Simulator User's Guide* includes a description of how to plan a simulator-related network.

EXAMPLES

An example of a typical Ethernet pseudo-driver configuration file:

```
#
# simudrv.conf: Ethernet pseudo-driver configuration file
#
#IP Address host
2.1.1.1  server1
```

This indicates that on a system named `server1` (according to `uname -n`), the Ethernet pseudo-driver uses IP address `2.1.1.1`.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

simudrv(1CC)

NAME

site_number.conf – simulator site configuration file

SYNOPSIS

`/usr/local/chorus/simu_admin/site_number.conf`

DESCRIPTION

The ChorusOS simulator architecture makes it possible to run multiple instances of the same system image. Each simulator instance is assigned an instance number between 1 and 254 inclusive at boot time. The instance number acts as an index to entries in the site configuration file, each of which defines:

- The hostname of the host system on which the simulator instance runs
- The UDP port number used by the simulator instance for remote IPC
- The IP address assigned to the simulator instance, used to replace the `TAG.TGT.IPA.TAG` string passed as a parameter to `ifconfig`(1M) in an entry used to configure the network as part of the `sysadm.ini`(4CC) system initialization script.

The site configuration file is read by the *loader*(1CC) at boot time.

EXAMPLES

An example of a typical site configuration file:

```
#
# site_number.conf: site configuration file
#
#inst   host            UDP port        IP Address
1       server1         2052            2.1.1.2
2       server1         2053            2.1.1.3
11      server2         2052            3.7.12.11
12      server2         2100            3.7.12.12
```

This indicates, for example, that on a system named `server2` (according to `uname -n`), the simulator with instance number `11` uses UDP port `2052` for Chorus remote IPC and IP address `3.7.12.11` for TCP/IP communication through the Ethernet pseudo-driver.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

loader(1CC)

NAME

monitor, monitorInit, monitorGet, monitorNotify, monitorNotifyAll, monitorRel, monitorWait – initialize a monitor; acquire a monitor; release a monitor; wait within a monitor for notification; notify a thread waiting within a monitor; notify all threads waiting within a monitor

#include <sync/chMonitor.h>

KnError **monitorInit**(KnMonitor **monitor*);

KnError **monitorGet**(KnMonitor **monitor*);

KnError **monitorNotify**(KnMonitor **monitor*);

KnError **monitorNotifyAll**(KnMonitor **monitor*);

KnError **monitorRel**(KnMonitor **monitor*);

KnError **monitorWait**(KnMonitor **monitor*, KnTimeVal **timeout*);

FEATURES

MONITOR

DESCRIPTION

A monitor is a synchronization object used to protect shared procedures and data against simultaneous access. Once the monitor is acquired by a thread, the thread can suspend its ownership and wait until it is notified or a timeout occurs.

Monitors are `KnMonitor` structures allocated in memory.

monitorInit() initializes the monitor whose address is *monitor*. The monitor is initialized as unlocked.

Statically allocated monitors can be initialized using the K_KNMONITOR_INITIALIZER macro, which initializes the monitor as unlocked. This macro is used as follows:

```
KnMonitor myMonitor = K_KNMONITOR_INITIALIZER
```

monitorGet() is used by a thread to acquire a monitor. If the monitor is unlocked, it becomes locked by the thread and the caller continues its execution normally. If the monitor is already locked by the current thread, execution also continues normally. If the monitor is locked by another thread, the caller is blocked until the monitor is released.

monitorWait() is used by a thread which has acquired a monitor to relinquish its lock on it, to lie dormant until another thread notifies it using **monitorNotify()** or **monitorNotifyAll()**, or until the amount of time specified by timeout has elapsed, and finally to re-acquire its lock on the monitor.

monitorNotify() is used by a thread which has acquired the monitor specified by *monitor* to notify a thread waiting within **monitorWait()** to resume. The calling thread must then call **monitorRel()** so that the waiting thread may actually resume.

monitorNotifyAll() notifies all threads waiting within **monitorWait()** to resume.

monitorRel() is used by a thread which has acquired a monitor to release it. If threads are blocked behind the monitor, one of them is awakened.

A blocking **monitorGet()** is NONABORTABLE (see `threadAbort`(2K)). **monitorWait()** is ABORTABLE, that is, when a **threadAbort()** is addressed to a waiting thread, it behaves as if its time-out had expired.

RESTRICTIONS

A user application and a supervisor application may not share a monitor.

Conversely, two applications running in the same mode (user or supervisor) may share a monitor by mapping it in both address spaces. Such shared monitors must be dynamically allocated monitors. In supervisor mode, the same address may be used by both applications, but care must be taken to keep the monitor's region allocated because the system may crash otherwise.

RETURN VALUES

Upon successful completion, `0` is returned. Otherwise a negative error code is returned.

ERRORS

K_EFAULT	Some of the data provided are outside the address space of the current actor.
K_EINVAL	`waitLimit` is not a valid `KnTimeVal`.
K_EINVAL	The calling thread is not the current owner of the monitor on *monitorRel*, *monitorNotify*, *monitorNotifyAll*, *monitorWait*.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

`mutexGet`(2K), `mutexInit`(2K), `mutexRel`(2K), `CORE`(5FEA), *JVM*(5FEA), *MONITOR*(5FEA), `MUTEX`(5FEA)

NAME

MONITOR – monitors

FEATURE SUMMARY

Monitors are a way of synchronizing concurrent threads. A monitor is a set of functions in which only one thread may execute at a time. It is possible for a thread running inside a monitor to suspend its execution so that another thread may enter the monitor. The initial thread waits for the second one to notify it (for example, that a resource is now available) and then to exit the monitor. By extension to object-oriented languages such as the Java™ language, monitor objects are associated with the set of functions. The functions take a monitor object as argument. Only one thread at a time uses a given monitor object. In this context, the term "monitor" often refers to the monitor object itself.

API

The MONITOR feature API is summarized in the following table:

monitorGet()	Obtains the lock on the given monitor.
monitorInit()	Initializes the given monitor.
monitorNotify()	Notifies one thread waiting in **monitorWait()**.
monitorNotifyAll()	Notifies all threads waiting in **monitorWait()**.
monitorRel()	Releases a lock on a given monitor.
monitorWait()	Causes a thread that owns the lock on the given monitor to suspend itself until it receives notification from another thread.

ATTRIBUTES

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

monitorGet(2K), *monitorInit*(2K), *monitorNotify*(2K), *monitorNotifyAll*(2K), *monitorRel*(2K), *monitorWait*(2K), `mutexGet`(2K), `mutexInit`(2K), `mutexRel`(2K), `MUTEX`(5FEA)

APPENDIX **B**

ChorusOS 4.0.1 Simulator Product Packages and Part Numbers

The tables below list the Solaris packages available in this release of the ChorusOS 4.0.1 Simulator for the Solaris Operating Environment (SPARC Platform Edition) product, and indicates the part number for each distinct product component.

Binary Product — for Solaris Host

Part Number	**CLX401-SSS0**
Package Name	**Description**
SUNWewbs	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) BSP source
SUNWewcd	Sun Embedded Workshop PDF Format Common Documentation
SUNWewch	Sun Embedded Workshop HTML Format Common Documentation
SUNWewcp	Sun Embedded Workshop PostScript Format Common Documentation
SUNWewds	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) XRAY Debugger
SUNWewgs	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) GUI Tools

Part Number	CLX401-SSS0
Package Name	**Description**
SUNWewis	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) IOM source
SUNWewks	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) Kernel
SUNWewm	Sun Embedded Workshop On-Line Manual Pages
SUNWewos	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) OS
SUNWewps	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) Examples
SUNWewqs	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) Simu Tools
SUNWewsd	Sun Embedded Workshop PDF Format Target Specific Documentation
SUNWewsh	Sun Embedded Workshop HTML Format Target Specific Documentation
SUNWewsp	Sun Embedded Workshop PostScript Format Target Specific Documentation
SUNWewts	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) Build Tools
SUNWewxs	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) X11 Library
SUNWewzs	Sun Embedded Workshop for the Solaris Operating Environment (SPARC Platform Edition) egcs Toolchain
SUNWewcab [1]	ChorusOS 4.0.1 Common Documentation Collection
SUNWewsab [1]	ChorusOS 4.0.1 Target Family Documentation Collection
SUNWewmab [1]	ChorusOS 4.0 Reference Manual Collection

1. Answerbook packages cannot be installed using the graphical installer. See the Sun document *Installing and Administering an AnswerBook2 Server* for a complete description of the AnswerBook2 documentation installation process.

Source Add-on for Solaris Host

Part Number	**CLX401-SSS0-S**
Package Name	**Description**
SUNWewhs	Sun Embedded Workshop for Solaris Operating Environment (SPARC Platform Edition) OS source
SUNWewls	Sun Embedded Workshop for Solaris Operating Environment (SPARC Platform Edition) Kernel source

Documentation for Solaris Host

Part Number	**CLX401-SAA0-D1N**
Package Name	**Description**
SUNWewcd	Sun Embedded Workshop PDF Format Common Documentation
SUNWewch	Sun Embedded Workshop HTML Format Common Documentation
SUNWewcp	Sun Embedded Workshop PostScript Format Common Documentation
SUNWewm	Sun Embedded Workshop On-Line Manual Pages
SUNWewsd	Sun Embedded Workshop PDF Format Specific Documentation
SUNWewsh	Sun Embedded Workshop HTML Format Specific Documentation
SUNWewsp	Sun Embedded Workshop PostScript Format Specific Documentation
SUNWewcab [1]	ChorusOS 4.0.1 Common Documentation Collection

Part Number	CLX401-SAA0-D1N
Package Name	**Description**
SUNWewsab [1]	ChorusOS 4.0.1 Target Family Documentation Collection
SUNWewmab [1]	ChorusOS 4.0 Reference Manual Collection