ChorusOS 4.0.1 x86/Pentium Target Family Guide

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

Part Number 806-5562–10
July 2000

Copyright 2000 Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, California 94303-4900 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, docs.sun.com, AnswerBook, AnswerBook2, Sun Embedded Workshop, ChorusOS, Solstice, JDK and Solaris are trademarks, registered trademarks, or service marks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

Federal Acquisitions: Commercial Software — Government Users Subject to Standard License Terms and Conditions.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2000 Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, Californie 94303-4900 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées du système Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, docs.sun.com, AnswerBook, AnswerBook2, Sun Embedded Workshop, ChorusOS, Solstice, JDK et Solaris sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Contents

CHAPTER 1

ChorusOS 4.0.1 x86/Pentium Target Family Guide

This guide describes how to run the ChorusOS™ 4.0.1 product for the x86/Pentium processor family.

Preface

How This Guide is Organized

ChorusOS x86/Pentium 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 and POSIX components.
- "Libraries" on page 12.
- "Utilities" on page 13, includes host and target utilities.
- "Reference Hardware" on page 16, includes supported reference platforms, supported devices, and validated reference platforms.
- "How to Build and Boot a System Image on the Target" on page 19.
- Appendix A, presents additional man pages for the MONITOR and JVM features (these man 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 product 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 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:`
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[SM] 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

The ChorusOS product provides a host-target development environment. Applications are developed on a workstation (the host), and then downloaded and executed on a specific board (the target).

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

Solaris™ (SPARC™ Platform Edition) 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 for x86/Pentium product:

- Chorus Cross Development System 5.0, target x86/Pentium 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 for x86/Pentium 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 x86/Pentium 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	Y
Dynamic libraries	DYNAMIC_LIB	Y
Compressed file management	GZ_FILE	Y
Scheduling		
POSIX round-robin scheduling class	ROUND_ROBIN	Y
Memory management		

Feature Description	Feature Name	Availability
Virtual (user and supervisor) address space	VIRTUAL_ADDRESS_SPACE	Y
On-demand paging	ON_DEMAND_PAGING	Y
Hot restart and persistent memory		
Hot restart	HOT_RESTART	Y
Inter-thread communication		
Semaphores	SEM	Y
Event flag sets	EVENT	Y
Mutual exclusion lock supporting thread priority inversion avoidance	RTMUTEX	Y [1]
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	Y
Mailbox-based communications mechanism	MIPC	Y
POSIX 1003.1-compliant message queues	POSIX_MQ	Y
POSIX 1003.1-compliant shared memory objects	POSIX_SHM	Y
LAP		
Local name server for LAP binding	LAPBIND	Y
LAP validity-check option	LAPSAFE	Y
Tools support		

Feature Description	Feature Name	Availability
Message logging	LOG	Y
Profiling and benchmark support	PERF	Y
System Monitoring	MON	Y
System debugging	DEBUG_SYSTEM	Y
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	Y
Driver for IDE disk	IDE_DISK	Y
`/dev/mem`, `/dev/kmem`, `/dev/null`, `/dev/zero`	DEV_MEM	Y
Support for RAM disk	RAM_DISK	Y
Support for FLASH media [2]	FLASH	Y
Virtual TTY	VTTY	Y
Driver for SCSI disk	SCSI_DISK	Y
Support for IPC	IOM_IPC	Y
Support for OSI	IOM_OSI	Y
Networking		
Serial link IP	SLIP	Y
POSIX 1003.1g-compliant sockets	POSIX_SOCKETS	Y
Point-to-point protocols	PPP	Y
Local sockets and pipes	AF_LOCAL	Y
Administration		

Feature Description	Feature Name	Availability
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	Y
`netstat` administration command	ADMIN_NETSTAT	Y
JVM		
Java Virtual Machine	JVM	Y

1. RTMUTEX is not supported for Intel 386. It is available for the other supported processors.
2. Logical-to-Physical block mapping for flash file system support.

Libraries

The ChorusOS operating system provides the elementary libraries indicated in the following list:

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`
The ‘‘embedded’’ C library [2]	`stdc.a`
The microkernel ‘‘visu’’ library [3]	`visu.a`

1. The `libebd.a`, `libcx.a`, `libm.a` and `libC.a` libraries have been made thread-safe in order to support multithreaded actors.
2. Included in `libebd.a`.
3. This library is provided for the sake of backwards compatibility only. It is not documented. Its use is strongly discouraged.

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)

java(1CC)

ls(1CC)

mkdir(1CC)

mkfifo(1CC)

mv(1CC)

netstat(1CC)

nfsstat(1CC)

pax(1CC)

PROF(1CC)

profctl(1CC)

rdbc(1CC)

rm(1CC)

rmdir(1CC)

startjvm(1CC)

touch(1CC)

uname(1CC)

ypcat(1CC)

ypmatch(1CC)

ypwhich(1CC)

arp(1M)

chat(1M)

chorusNS(1M)

chorusNSinet(1M)

chorusNSsite(1M)

dhclient(1M)

disklabel(1M)

flashdefrag(1M)

format(1M)

fsck(1M)

fsck_dos(1M)

ftpd(1M)

inetNS(1M)

inetNSdns(1M)

inetNShost(1M)

inetNSien116(1M)

inetNSnis(1M)

mkfd(1M)

mkfs(1M)

mount(1M)

mount_msdos(1M)

mount_nfs(1M)

mountd(1M)

newfs(1M)

newfs_dos(1M)

nfsd(1M)

portmap(1M)

route(1M)

shutdown(1M)

slattach(1M)

syncd(1M)

sysctl(1M)

telnetd(1M)

umount(1M)

ypbind(1M)

Host Utilities

The following utilities may be run on the host machine:

chadmin(1CC)

chconsole(1CC)

chlog(1CC)

chls(1CC)

ChorusOSMkMf(1CC)

chserver(1CC)

configurator(1CC)

configure(1CC)

ews(1CC)

mkmerge(1CC)

rdbs(1CC)

profrpg(1CC)

Reference Hardware

ChorusOS targets are described in this section from three different points of view:

Reference Processors and BSPs:

This subsection describes the processors on which the ChorusOS product can run, as well as the details of the BSPs included in the delivery.

Reference Target Platforms:

This section describes all the target platforms which can be used as references in the context of Sun support contracts.

Validated Reference Targets:

This section describes the precise platforms used to run the Sun QA tests; this may be useful, in case of bugs, as a hint or guide to help in identifying issues which are closely hardware related.

Reference Processors and BSPs

The ChorusOS system for x86/Pentium supports the following processors:

- Intel 386 (no software FPU emulation).
- Intel 486.
- Intel Pentiums.
- Intel Celeron.

The ChorusOS system for x86/Pentium supports the following reference BSPs:

- i386at Reference BSP.

i386at Reference BSP

Systems

The i386at reference BSP supports standard PC-AT boards.

Devices

The i386at reference BSP supports the following on board devices:

Device Id	ChorusOS Driver
/pci (PCI bridge)	sun:x86-generic-(bus,pci)
/pci/pci-isa (ISA bridge)	sun:pci-generic-(bus,isa)
/pci/pci-isa/i8254 (TIMER)	sun:bus-i8254-timer
/pci/pci-isa/mc146818 (RTC, TIMER)	sun:bus-mc146818-(rtc,timer)
/pci/pci-isa/ns16550-1 (UART)	sun:bus-ns16550-uart
/pci/pci-isa/ns16550-2 (UART)	sun:bus-ns16550-uart
/pci/pci-isa/generic-ide: (IDE disk)	sun:bus-generic-ide
/pci/pci-isa/generic-ide: (IDE other)	not supported
/pci/pci-isa/fdd (floppy)	not supported
/pci/pci-isa/kbd (keyboard)	not supported
/pci/pci-isa/lpt (parallel)	not supported
/pci/pci-isa/mouse (mouse)	not supported

The i386at reference BSP supports the following expansion devices:

Device Id	ChorusOS Driver	Reference Device
/pci/epic100 (ETHER)	sun:pci-epic100-ether	SMC EtherPowerII 10/100 TP
/pci/dec21140 (ETHER)	sun:pci-dec21x4x-ether	ZNYX ZX345 Fast Ethernet

Device Id	ChorusOS Driver	Reference Device
/pci/ne2000 (ETHER)	sun:bus-ne2000-ether	Kingston KNE2000TLC Novell NE2000 plus
/pci/ncr53c825 (SCSI HBA)	sun:pci-ncr53c8xx-scsi_hba sun:scsi_hba-generic-scsi	Symbios Logic SYM8750SP
/pci/ncr53c825/disk@t,l (SCSI disks) where t is the SCSI TARGET number where l is the LUN number	sun:scsi-disk-BSD	
/pci/ncr53c825/xxx (SCSI other)	not supported	
/pci/pci-isa/smc1660 (ETHER)	sun:isa-smc1660-ether	SMC EtherCard Elite Ultra 16 bits SMC EtherCard EZ
/pci/pci-isa/el3 (ETHER)	sun:bus-el3-ether	3Com EtherlinkIII 3C509B
/pci/pci-isa/ne2000 (ETHER)	sun:bus-ne2000-ether	
/flash (FLASH memory)	not supported	

Reference Target Platforms

This section describes all the target platforms which can be used as references in the context of Sun support contracts.

PC Compatible

Type: Generic Computer

Processors: i486/Pentiums/Celeron (33-400 Mhz)

Main memory: 16-256 MB

L2 cache: 256-512 KB

Bus bridges:	Processor to PCI, PCI to ISA
Devices:	Asynchronous serial ports (38.4 Kbaud), 10/100BaseT Ethernet (PCI: SMC EtherPowerII, compatible NE2000, ZNYX ZX345; ISA: SMC EtherCard Elite Ultra 16bits/EZ, compatible NE2000, 3Com EtherLinkIII), IDE disk, Ultra-Wide SCSI (PCI: Symbios logic SYM8750SP), Real-time clock, Timers
Firmware:	PC BIOS

Validated Reference Targets

This section describes the precise platforms used to run the Sun QA tests:

- Pentium II 300 Mhz/ 64 MB / 512 KB L2 cache / EtherPower II.
- Celeron 400 Mhz / 128 MB / 128 KB L2 cache / EtherPowerII.

How to Build and Boot a System Image on the Target

▼ Building a ChorusOS System Image

The following procedure assumes that the ChorusOS 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-x86

3. **Make sure your PATH has been set correctly to include the directory** *install_dir*`/4.0.1/chorus-x86/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:**

   If you are building from a binary distribution:

```
$ configure -b $DIR/kernel \
$DIR/os \
$DIR/tools \
-s $DIR/src/nucleus/bsp/drv \
$DIR/src/nucleus/bsp/x86 \
$DIR/src/nucleus/bsp/x86/i386at \
$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 or code examples, 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/drv \
$DIR/src/nucleus/bsp/x86 \
$DIR/src/nucleus/bsp/x86/i386at \
$DIR/src/iom \
$DIR/src/opt/examples
```

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

As a result of configuration, *build_dir* now 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. Configure the system image:

If you have only PCI Ethernet adapters, you can skip this step.

ISA adapters require that you provide configuration information by setting an ETHER_*id* environment variable in the system image configuration with a command of the form:

`configurator -setenv ETHER_`*id*`=`*name*`,`*irq*`,`*io_base*`[,`*mem_base*`]`

The parameters are as follows:

id	A decimal value ranging from `0` to `9`. Up to ten ISA Ethernet adapters can be declared using ETHER_0, ETHER_1, …, ETHER_9.
name	The type of the device: `EL3`, `NE2000` or `SMC`.

irq	A decimal value representing the interrupt request level on the ISA bus.
io_base	A hexadecimal value representing the base for I/O ports, such as `0x300`.
mem_base	A hexadecimal value representing the base for shared memory, such as `0xD0000`. The memory base parameter is required for SMC Ethernet adapters only.

For example, the following command configures the build environment for one SMC Ethernet adapter with IRQ `9`, I/O base at `0x240` and memory base at `0xD0000`:

```
$ configurator -setenv ETHER_0=SMC,9,0x240,0xD0000
```

The following command configures the build environment for a system with both an NE2000 adapter and an EL3 adapter:

```
$ configurator -setenv ETHER_0=NE2000,5,0x300
$ configurator -setenv ETHER_1=EL3,7,0x300
```

7. Build a system image:

```
$ make chorus
```

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

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

```
$ make kernonly
```

▼ Placing the System Image on the Boot Server

See the *ChorusOS 4.0 Installation Guide for Solaris Hosts* for instructions on how to configure the boot server.

1. **Copy the system image to the boot server.**

   For example, on a Solaris host workstation:

```
$ rcp chorus.bmon boot_server:/tftpboot
```

2. **Verify that everyone has at least read access to the system image on the boot server.**

   For example:

```
$ rlogin boot_server
Password: password_for_user
$ ls -l /tftpboot/chorus.bmon
-rwxr-xr-x   1 user    group     1613824 Dec 15 17:33 chorus.bmon*
```

3. **While logged in to the boot server, create a configuration file for the target.**

   For a target system with IP address `129.157.197.88` using a boot server with IP address `129.157.197.144`, the configuration file contains the following:

```
AUTOBOOT=YES
BOOTFILE=chorus.bmon
BOOTSERVER=129.157.197.144
```

   The configuration file is named `/tftpboot/819DC558.ChorusOS.4.0`, which is constructed from the target system IP address `129.157.197.88` as a concatenation of the following:

   - `129` in decimal translates to `81` in hexadecimal.
   - `157` in decimal translates to `9D` in hexadecimal.
   - `197` in decimal translates to `C5` in hexadecimal.
   - `88` in decimal translates to `58` in hexadecimal.

- (optional) `.ChorusOS.4.0` identifies the release, and is appended to the concatenation of the IP address expressed in hexadecimal.

Note - The system first attempts to find the configuration file with the `.ChorusOS.4.0` extension. If it fails to find one, however, it attempts to find a configuration file without the `.ChorusOS.4.0` extension.

▼ Creating a bootMonitor Diskette

See `bootMonitor`(1CC) for details about how bootMonitor works.

1. **Create a build directory where you will build a** bootMonitor **image:**

```
$ mkdir bootmon
$ cd bootmon
```

Note that this build directory is different from the directory where you build system images.

2. **Configure the** bootMonitor **build directory based on the binary distribution:**

```
$ configure -b $DIR/kernel \
$DIR/os \
$DIR/tools \
-s $DIR/src/nucleus/bsp/drv \
$DIR/src/nucleus/bsp/x86 \
$DIR/src/nucleus/bsp/x86/i386at \
$DIR/src/iom
```

3. **Generate the build environment:**

```
$ make
```

4. **Edit the special** *bootmon*`/conf/mini` **profile so that it reads:**

```
#
# Mini Profile
#

#
# Kernel features
#
-set USER_MODE=false
-set VIRTUAL_ADDRESS_SPACE=false
-set SEM=false
-set EVENT=false
-set MONITOR=false
-set TIMER=false
-set DATE=false
-set RTC=false
-set PERF=false
-set IPC=false
-set MIPC=false
-set LAPBIND=true # Change this from 'false' to 'true'
-set LAPSAFE=true # Change this from 'false' to 'true'
-set MON=false
-set LOG=false
```

5. **Configure the build environment for** bootMonitor**:**

```
$ configurator -p conf/mini
$ configurator -set LOADER=lilo
```

6. **Configure the build environment for any ISA Ethernet adapters on the target system.**

   See Step 6 on page 21 for examples on how to configure the build environment for different adapters.

7. **Build a** bootMonitor **image:**

```
$ make bootMonitor
```

   The resulting system image file is located in the build directory, *bootmon* and is called `bootMonitor.image`.

8. **Copy the** bootMonitor **image to a diskette:**

```
$ su
Password: root_password
# /etc/init.d/volmgt stop
# fdformat -v /dev/fd0
Formatting 1.44 MB in /dev/rfd0
Press return to start formatting floppy.
................................................................................
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
# cp bootMonitor.image /dev/fd0
# /etc/init.d/volmgt start
volume management starting.
# eject
/vol/dev/rdiskette0/unlabeled can now be manually ejected
# exit
```

▼ How to Boot the Target System Using a bootMonitor Diskette

1. **Shut down the system.**

2. **Connect a serial line from the first serial line port on the target system to the host workstation in order to view the console output.**

   You can use a terminal-type serial cable, where wires 2 and 3 are crossed and 7 (the ground) is straight. However, a serial cable in which all wires are parallel will not work.

3. **Add a line to the host workstation** `/etc/remote` **file to make it possible to connect through the serial line using the** `tip`**(1) utility. For example:**

```
target_hostname:dv=/dev/cua/a:br#9600
```

4. **Connect to the target system using the** `tip`**(1) utility:**

```
$ tip target_hostname
connected
```

5. **Place the** bootMonitor **diskette in the drive.**

6. **Reboot, making sure the BIOS causes the system to boot first from the diskette.**

   If the system boots correctly, messages similar to the following are displayed on the console in the terminal where you issued the `tip`(1) command:

```
DebugAgent: trying to sync with DebugServer...
RAM size: 0x8000000 bytes

ChorusOS r4.0.1 for Intel x86 - Intel x86 PC/AT
Copyright (c) 2000 Sun Microsystems, Inc. All rights reserved.

Kernel modules : CORE SCHED_FIFO MEM_FLM KDB TICK ENV LAPSAFE MUTEX PERF \
TIMEOUT LAPBIND DKI
/pci/i8259: sun:pci-i8259-pic driver started
/pci: sun:x86-bios-(bus,pci) driver started
/pci/pci8086,7190@0,0: device node is created by sun:pci-enumerator-
/pci/pci8086,7191@1,0: device node is created by sun:pci-enumerator-
/pci/pci8086,7110@4,0: device node is created by sun:pci-enumerator-
/pci/pci8086,7111@4,1: device node is created by sun:pci-enumerator-
/pci/pci8086,7112@4,2: device node is created by sun:pci-enumerator-
/pci/pci8086,7113@4,3: device node is created by sun:pci-enumerator-
/pci/pci10b8,5@a,0: device node is created by sun:pci-enumerator-
/pci/pci-isa: sun:pci-bios-(bus,isa) driver started
/pci/pci-isa/i8254: sun:bus-i8254-timer driver started
/pci/pci10b8,5@a,0: sun:pci-epic100-ether device started
/pci/pci10b8,5@a,0: Ethernet Address 00:e0:29:3c:6c:7f

Boot Monitor Loader (v1.0)

Searching for adapters...
Unit: 0  device name: pci10b8,5@a,0

Using unit 0

My IP 129.157.197.88, RARP Server IP 129.157.197.144

Loading file 819DC558.ChorusOS.4.0 on server 129.157.197.144: loaded!

Loading file chorus.bmon on server 129.157.197.144: loaded!

Booting downloaded file.

Boot new image ...
DebugAgent: trying to sync with DebugServer...
RAM size: 0x8000000 bytes

ChorusOS r4.0.1 for Intel x86 - Intel x86 PC/AT
Copyright (c) 2000 Sun Microsystems, Inc. All rights reserved.

Kernel modules : CORE SCHED_FIFO SEM MIPC IPC_L MEM_PRM KDB TICK MON ENV \
ETIMER LOG LAPSAFE MUTEX EVENT UI DATE PERF TIMEOUT LAPBIND DKI
MEM: memory device 'sys_bank' vaddr 0x7bc43000 size 0x189000
/pci/i8259: sun:pci-i8259-pic driver started
/pci: sun:x86-bios-(bus,pci) driver started
/pci/pci8086,7190@0,0: device node is created by sun:pci-enumerator-
/pci/pci8086,7191@1,0: device node is created by sun:pci-enumerator-
/pci/pci8086,7110@4,0: device node is created by sun:pci-enumerator-
```

(continued)

(Continuation)

```
/pci/pci8086,7111@4,1: device node is created by sun:pci-enumerator-
/pci/pci8086,7112@4,2: device node is created by sun:pci-enumerator-
/pci/pci8086,7113@4,3: device node is created by sun:pci-enumerator-
/pci/pci10b8,5@a,0: device node is created by sun:pci-enumerator-
/pci/pci-isa: sun:pci-bios-(bus,isa) driver started
/pci/pci-isa/i8254: sun:bus-i8254-timer driver started
/pci/pci-isa/mc146818: sun:bus-mc146818-(rtc,timer) driver started
/pci/pci-isa/ns16550-2: sun:bus-ns16550-uart driver started
/pci/pci-isa/generic-ide: sun:bus-generic-ide driver started
/pci/pci10b8,5@a,0: sun:pci-epic100-ether device started
/pci/pci10b8,5@a,0: Ethernet Address 00:e0:29:3c:6c:7f
IOM: SOFTINTR DISABLED (-31). Using an Interrupt thread
IOM Init cluster space from: 0x7bc1f000 to: 0x7bc3f800 [65 items of size: 2048]
IOM Init io-buf pool from: 0x7bc3f850 to: 0x7bc3fd70 [8 items of size: 164]
IOM Init raw io-buffer pool from: 0x7bc3fd70 to: 0x7bc411f0 [32 items of size: 164]
Copyright (c) 1992-1998 FreeBSD Inc.
Copyright (c) 1982, 1986, 1989, 1991, 1993
        The Regents of the University of California.  All rights reserved.

max disk buffer space = 0x10000
/rd: sun:ram--disk driver started
C_INIT: started
C_INIT: /image/sys_bank mounted on /dev/bd00
C_INIT: found /image/sys_bank/sysadm.ini
C_INIT: executing start-up file /image/sys_bank/sysadm.ini
bpf: ifeth0 attached
IOM: ifnet ifeth0 bound to device /pci/pci10b8,5@a,0
bpf: lo0 attached
C_INIT: Internet Address: 129.157.197.88
C_INIT: RARP Network Initialization OK
ifeth0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet 129.157.197.88 netmask 0xffff0000 broadcast 129.157.255.255
        ether 00:e0:29:3c:6c:7f
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet 127.0.0.1 netmask 0xff000000
C_INIT: rshd started
```

7. **Issue a command to the target system to make sure things are working properly. For example:**

```
$ rsh target_hostname mount
root_device on / (pdevfs)
devfs on /dev (pdevfs)
devfs on /image (pdevfs)
/dev/bd00 on /image/sys_bank (msdos)
```

APPENDIX **A**

ChorusOS 4.0.1 for x86/Pentium 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 release of the product.

section 1CC: Host and Target Utilities

java(1CC)
startjvm(1CC)

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

JVM(5FEA)
MONITOR(5FEA)

NAME

java – Java interpreter

SYNOPSIS

rsh *target* **arun $JVM_ROOT/bin/java** [–quit] [–rehash [*ENVAR=VALUE*]] [–viewclasses] [–viewthreads] *classname* [*args*]

FEATURES

JVM

DESCRIPTION

`java` is a target utility.

The `java` command executes Java bytecodes created by the Java compiler, javac, on the host system.

The *classname* argument is the name of the class to be executed and must be fully qualified by including the package in the name, for example:

```
example% rsh target arun JVM_ROOT/bin/java java.lang.String
```

Note that any arguments that appear after *classname* on the command line are passed to the **main()** method of the class.

The bytecodes for the class are put in a file called *classname*`.class` by compiling the corresponding source file with javac. All Java bytecode files end with the filename extension `.class`, which the compiler automatically adds when the class is compiled. The *classname* argument must contain a **main()** method defined as follows:

```
class Aclass {
     public static void main(String argv[]){
     . . .
     }
}
```

The `java` command returns control to the command interpreter as soon as it has succeeded in loading the class. It then executes the **main()** method and exits unless **main()** creates one or more threads. In this case, `java` does not exit until the last thread exits. Note that exiting a class *never* causes the Java Virtual Machine to exit in the context of ChorusOS.

When defining classes, specify their locations using the APP_CLASSPATH environment variable, which consists of a colon-separated list of directories that specifies the path.

OPTIONS

The following options are supported:

`–quit` — Kill all running Java threads and terminate the jvmd actor.

`–rehash` — Reload environment variables.

If the environment variables to reload are provided as a set of whitespace-separated

variable-value argument pairs to the –`rehash` option, the Java Virtual Machine will reload only those environment variables that are specified. Otherwise the –`rehash` option forces the Java Virtual Machine to reload all relevant environment variables. (See *ENVIRONMENT VARIABLES*.)

–`viewclasses` List all Java currently loaded classes in the jvmd actor.

–`viewthreads` List all Java threads currently running in the jvmd actor.

ENVIRONMENT VARIABLES

The following environment variables are supported:

JVM_ROOT Base directory where the Java Virtual Machine is installed.

Default value: `/opt/jvm` (as seen from the target system).

JVM_CLASSPATH Search path for non-verified bootstrap classes and resources.

Default value: `${`*JVM_ROOT*`}/classes`.

JVM_LIBPATH Search path for bootstrap native libraries.

Default value: `${`*JVM_ROOT*`}/lib`.

JVM_DEBUG Enables or diables tracing. Values assigned to this environment variable may be: `none` (no tracing), `all` (full tracing), `loading` (provide traces from internal primordial classloader), `verifying` (provide traces from the verifier), `loading`, `verifying` or `verifying, loading`.

Default value: `none`

JVM_GC Enables or disables garbage collection according to the mark and sweep method. Values assigned to this environment variable may be either `enable` or `disable`.

Default value: `enable`

APP_CLASSPATH Search path for application classes and resources.

Default value: None (user-defined).

APP_LIBPATH Search path for application native libraries.

Default value: None (user-defined).

ATTRIBUTES

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

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

startjvm(1CC), *JVM*(5FEA).

NOTES

The jvmd actor behaves somewhat differently from Java Virtual Machines designed for other general purpose operating systems. Refer to *startjvm*(1CC) for more information about the jvmd actor.

NAME

startjvm – run the Java Virtual Machine actor

SYNOPSIS

rsh *target* **arun /jvm/bin/startjvm**

FEATURES

JVM

DESCRIPTION

`startjvm` is a target utility.

The `startjvm` command starts the Java Virtual Machine for ChorusOS (the jvmd actor) initializing it according to the environment variables described in the *ENVIRONMENT VARIABLES* section of *java*(1CC). The jvmd actor is started only once, as all java applications executed on the target system run in the context of that actor.

Note that the `startjvm` command is not located under JVM_ROOT.

After the jvmd actor has been started using the `startjvm` command

```
Main JVM created
```

should appear on the ChorusOS console.

The corresponding command to terminate the jvmd actor is:

```
example% rsh target $JVM_ROOT/bin/java -quit
```

The jvmd actor

The jvmd actor runs on the target system.

The jvmd actor provides the Java Virtual Machine for ChorusOS. It may be terminated using the –`quit` option of the `java` command, but does not terminate simply because no Java applications are running.

The Java Virtual Machine component of ChorusOS is implemented as a single supervisor actor. That single actor holds all Java threads associated with all Java applications running on the target. In other words, all Java applications run in the supervisor space and all Java applications and associated threads belong to a single ChorusOS actor.

The following table indicates what is and is not supported.

SUPPORTED	NOT SUPPORTED
All Java™ 2 Platform, Standard Edition, v1.2.2 application programming interfaces *except AWT*, including the following packages: java.beans, java.io, java.lang, java.math, java.net, java.rmi, java.security, java.sql, java.text, java.util, sun.beans, sun.dc, sun.io, sun.jdbc, sun.misc, sun.net, sun.rmi, sun.security, sun.tools, sunw.io, sunw.util	AWT and packages that depend on AWT
The Java™ Native Interface with native code running in supervisor space	The Java Native Interface with native code running in user space only
Loading of dynamic libraries whose symbols are known to the Java Virtual Machine actor	

The following particularities, limitations and restrictions apply:

Single Supervisor Actor

The Java Virtual Machine runs as a single actor.

All Java applications running on the target depend on this single actor, jvmd. All Java applications must be started after the jvmd actor has been launched, using the *java*(1CC) command.

System.exit() Stub

System.exit() takes no action, and simply returns control to the caller.

Java Virtual Machine implementations for other host platforms allow the developer to use **System.exit()** to terminate the Java Virtual Machine currently running. As the Java Virtual Machine actor for the ChorusOS operating system runs multiple Java applications, **System.exit()** is not designed to terminate the Java Virtual Machine itself.

Java Native Interface

Developers writing applications that use the Java Native Interface must use system calls that are available for use by supervisor actors.

The Java Virtual Machine is implemented as a supervisor actor. Some symbols that can be seen in a user space view of the system are not visible in supervisor space.

Messages

All messages from the Java Virtual Machine are directed to the ChorusOS console.

Developers who need to manage messages in some other way must implement their own mechanisms for doing so, for example, by using inter-process communication or log files written to a file system.

Thread Priority

By default, threads running in the Java Virtual Machine share the same priority scale than other system threads.

Before mapping Java thread priorities to the system priorities, developers may first tune the ChorusOS system to set the maximum priority for Java threads using the `jvm.thread.maxPriority` tunable. This value forces threads running in the Java Virtual Machine to be assigned a lower overall priority than other system threads.

ATTRIBUTES

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

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

java(1CC), *JVM*(5FEA)

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

JVM – Java Virtual Machine component

FEATURE SUMMARY

The JVM feature provides support for the Java Virtual Machine component. This feature requires the MONITOR feature to be set.

This feature allows the system to provide support for Java applications using the Java Virtual Machine actor, jvmd actor.

API

The JVM feature does not itself export an API.

ATTRIBUTES

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

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Evolving

SEE ALSO

java(1CC), *startjvm*(1CC), *MONITOR*(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 for x86/Pentium Product Packages and Part Numbers

The tables below list the Solaris packages available in this release and indicate the part number for each distinct product component.

Binary Product — for Solaris Host

Part Number	**CLX401-SBA0**
Package Name	**Description**
SUNWewbx	Sun Embedded Workshop for x86, Pentium 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
SUNWewdx	Sun Embedded Workshop for x86, Pentium XRAY Debugger
SUNWewgx	Sun Embedded Workshop for x86, Pentium GUI Tools
SUNWewix	Sun Embedded Workshop for x86, Pentium IOM source
SUNWewjx	Sun Embedded Workshop for x86, Pentium JVM

Part Number	CLX401-SBA0
Package Name	**Description**
SUNWewkx	Sun Embedded Workshop for x86, Pentium Kernel
SUNWewm	Sun Embedded Workshop On-Line Manual Pages
SUNWewox	Sun Embedded Workshop for x86, Pentium OS
SUNWewpx	Sun Embedded Workshop for x86, Pentium Examples
SUNWewsd	Sun Embedded Workshop PDF Format Specific Documentation
SUNWewsh	Sun Embedded Workshop HTML Format Specific Documentation
SUNWewsp	Sun Embedded Workshop PostScript Format Specific Documentation
SUNWewtx	Sun Embedded Workshop for x86, Pentium Build Tools
SUNWewux	Sun Embedded Workshop for x86, Pentium Debugger and Profiling Support
SUNWewxx	Sun Embedded Workshop for x86, Pentium X11 Library
SUNWewzx	Sun Embedded Workshop for x86, Pentium 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.

Flite Add-on for Solaris Host

Part Number	**FLT401-SBA0**
Package Name	**Description**
SUNWewfx	Sun Embedded Workshop for x86, Pentium Flite

Source Add-on for Solaris Host

Part Number	**CLX401-SBA0-S**
Package Name	**Description**
SUNWewhx	Sun Embedded Workshop for x86, Pentium OS source
SUNWewlx	Sun Embedded Workshop for x86, Pentium 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

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