



# Sun Fire High-End Systems Dynamic Reconfiguration User Guide

---

Sun Microsystems, Inc.  
www.sun.com

Part No. 817-7166-10  
January 2005, Revision A

Submit comments about this document at: <http://www.sun.com/hwdocs/feedback>

Copyright 2003, 2004 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries.

This document and the product to which it pertains are distributed under licenses restricting their use, copying, distribution, and decompilation. No part of the product or of this 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 in other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, AnswerBook2, docs.sun.com, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and in other countries.

All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and in 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.

U.S. Government Rights—Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

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 2003, 2004 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, Etats-Unis. Tous droits réservés.

Sun Microsystems, Inc. a les droits de propriété intellectuels relatants à la technologie qui est décrit dans ce document. En particulier, et sans la limitation, ces droits de propriété intellectuels peuvent inclure un ou plus des brevets américains énumérés à <http://www.sun.com/patents> et un ou les brevets plus supplémentaires ou les applications de brevet en attente dans les Etats-Unis et dans les autres pays.

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 des systèmes 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, AnswerBook2, docs.sun.com, et Solaris sont des marques de fabrique ou des marques déposées 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 licenciées de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.



Please  
Recycle



Adobe PostScript

# Contents

---

**Preface** vii

**1. Introduction to DR on the Sun Fire High-End System** 1

What Is DR? 1

Where You Execute DR Commands 1

Command Line Interface (CLI) 2

Graphical User Interface (GUI) 2

Automatic DR 2

Enhanced System Availability 3

DR Concepts 3

Detachability 3

Quiescence 4

Suspend-Safe and Suspend-Unsafe Devices 4

Attachment Points 5

Conditions and States 6

DR Operations 6

Hot-Plug Hardware 7

Dynamic System Domains 7

Component Types 8

DR on I/O Boards 8

Solving a Problem With an I/O Device	8
Golden IOSRAM	9
DR on hsPCI+ I/O Boards	9
Permanent and Non-permanent Memory	10
Target Memory Constraints	10
Correctable Memory Errors	10
Capacity on Demand (COD)	11
DR on COD Boards	11
Enabling DR on Domains Running the Solaris 8 Operating System	12
An Illustration of DR Concepts	12
<b>2. DR State and Condition Models</b>	<b>15</b>
Board States and Conditions	15
Board Slot States	16
Board Occupant States	16
Board Conditions	17
Component States and Conditions	17
Component Receptacle States	17
Component Occupant States	17
Component Conditions	18
<b>3. DR Operations and Software Components on the Domain</b>	<b>19</b>
DR Operations	19
Before You Perform DR Operations	19
Before Performing DR Operations on I/O Boards	20
Connect Operation	20
Configure Operation	21
CPUs and Memory	22
I/O Boards	22

After the Configure Operation	23
Disconnect Operation	23
Unconfigure Operation	24
Non-permanent Memory	24
Permanent Memory	24
Software Components	26
Domain Configuration Server	26
DR Driver	27
Reconfiguration Coordination Manager	27
System Events Framework	27
<b>4. DR User Interfaces on the Domain</b>	<b>29</b>
DR Commands and Options on the Domain	29
State Change Functions	30
Availability Change Functions	30
Condition Change Functions	30
Options and Operands	31
<b>5. DR Domain Procedures</b>	<b>33</b>
Attachment Points	33
Displaying Board Status	34
Basic Status Display	34
Detailed Status Display	34
Removing a Board	35
▼ To Remove a CPU/Memory Board	35
▼ To Remove an I/O Board	35
Adding a Board	37
▼ To Install a Board	37
DR Using <code>cfgadm(1M)</code> - Examples	38

Displaying Help	38
Displaying Verbose Messages	39
Suppressing User Confirmation	39
Power Control When Disconnecting Boards	40
Power Control of Disconnected Boards	40
Connecting and Configuring Boards	40
Hot Plugging PCI Adapter Cards	41
Testing a Board	41
Displaying Attachment Point Information	42
Tracking Memory Unconfigure Operations	43
Finding the Board Containing Permanent Memory	43

**Index 45**

# Preface

---

This book describes the Dynamic Reconfiguration (DR) feature of Sun™ Fire high-end (E25K/E20K/15K/12K) servers. DR enables you to attach system boards to and detach them from system domains while the Solaris operating system continues to run.

---

## Before You Read This Book

This book is intended for the Sun Fire high-end system administrator who has a working knowledge of UNIX® systems, particularly those based on the Solaris™ operating system. If you do not have such knowledge, first read the Solaris user and system administrator books provided with this system and consider UNIX system administration training.

---

## How This Book Is Organized

This book contains the following chapters:

*Chapter 1 “Introduction to DR on the Sun Fire High-End System” on page 1*

*Chapter 2 “DR State and Condition Models” on page 15*

*Chapter 3 “DR Operations and Software Components on the Domain” on page 19*

*Chapter 4 “DR User Interfaces on the Domain” on page 29*

*Chapter 5 “DR Domain Procedures” on page 33*

---

# Using UNIX Commands

This document might not contain information on basic UNIX® commands and procedures such as shutting down the system, booting the system, and configuring devices. See the following for this information:

- Software documentation that you received with your system
- Solaris™ operating system documentation, which is at

<http://docs.sun.com>

---

## Shell Prompts

Shell	Prompt
C shell	<i>machine_name%</i>
C shell super user	<i>machine_name#</i>
Bourne shell and Korn shell	\$
Bourne shell and Korn shell super user	#

---

# Typographic Conventions

Typeface <sup>1</sup>	Meaning	Examples
AaBbCc123	The names of commands, files, and directories; on-screen computer output	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. <code>% You have mail.</code>
<b>AaBbCc123</b>	What you type, when contrasted with on-screen computer output	<code>% su</code> password:
<i>AaBbCc123</i>	Book titles, new words or terms, words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the <i>User's Guide</i> . These are called <i>class</i> options. You <i>must</i> be super user to do this. To delete a file, type <code>rm filename</code> .

<sup>1</sup> The settings on your browser might differ from these settings.

---

## Related Documentation

Application	Title
SMS-related DR User information	<i>System Management Services (SMS) Dynamic Reconfiguration User Guide</i>
SMS Administration Guide	<i>System Management Services (SMS) Administrator Guide</i>
Platform-specific release notes	<i>Solaris Release Notes Supplement for Sun Hardware</i> (through Solaris 9 releases) and <i>Solaris Release Notes</i> (beginning with Solaris 10 releases)
SMS Release Notes	<i>System Management Services (SMS) Release Notes</i>
DR Webpage	<a href="http://www.sun.com/servers/highend/dr_sunfire">http://www.sun.com/servers/highend/dr_sunfire</a>

---

## Accessing Sun Documentation Online

You can view, print, or purchase a broad selection of Sun documentation, including localized versions, at:

<http://www.sun.com/documentation>

---

## Contacting Sun Technical Support

If you have technical questions about this product that are not answered in this document, go to:

<http://www.sun.com/service/contacting>

---

## Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

<http://www.sun.com/hwdocs/feedback>

Please include the title and part number of your document with your feedback:

*Sun Fire High-End Systems Dynamic Reconfiguration User Guide*, part number 817-7166-10

# Introduction to DR on the Sun Fire High-End System

---

This chapter contains descriptions about general concepts that pertain to the dynamic reconfiguration (DR) feature on Sun Fire high-end servers.

---

**Note** – Sun Fire E25K and Sun Fire 15K systems support up to 18 system boards and 18 I/O boards at a time, numbered 0 through 17. Sun Fire E20K and Sun Fire 12K systems support up to nine system boards and nine I/O boards at a time, numbered 0 through 8.

---

---

## What Is DR?

DR on the Sun Fire high-end system enables you to perform hardware configuration changes to a live domain that is running the Solaris operating system without requiring machine downtime. You can also use DR in conjunction with hot-swap to physically add boards to or remove them from the server.

## Where You Execute DR Commands

You can execute DR operations from the Sun Fire high-end server system controller (SC) by using the system management services (SMS) commands: `addboard(1M)`, `moveboard(1M)`, `deleteboard(1M)`, and `rcfgadm(1M)`; or from the domain by using the `cfgadm(1M)` command. DR operations using SMS commands are described in [Chapter 5, “DR Domain Procedures” on page 33](#).

---

**Note** – If the `addboard(1M)`, `moveboard(1M)`, `deleteboard(1M)`, `rcfgadm(1M)`, or `cfgadm(1M)` command fails during a DR operation, the board does not return to its original state. If the error is recoverable, you can retry the command. If the error is unrecoverable, you must reboot the domain to use the board.

---

## Command Line Interface (CLI)

The DR software has a command line interface through the `cfgadm(1M)` command, which is the configuration administration program. The DR agent also provides a remote interface to the Sun Management Center software.

## Graphical User Interface (GUI)

The optional Sun Management Center software provides features such as domain management, as well as a graphical user interface (GUI) where you perform DR operations. If you prefer to use a graphical user interface instead of a command line interface, use the Sun Management Center software.

To use the Sun Management Center Platform software, you must attach the system controller board to a network. With a network connection, you can view both the command line interface and the graphical user interface. For instructions on how to use the Sun Management Center software, refer to the *Sun Management Center User's Guide*, shipped with the Sun Management Center software. For instructions on how to connect the system controller to a network connection on the system controller board, see your systems installation documentation.

## Automatic DR

Automatic DR enables an application to execute DR operations without requiring user interaction. This ability is provided by an enhanced DR framework that includes the reconfiguration coordination manager (RCM) and the system event facility, called `sysevent`. The RCM enables application-specific loadable modules to register callbacks. The callbacks perform preparatory tasks before a DR operation, error recovery during a DR operation, or clean-up after a DR operation. The `sysevent` facility enables applications to register for system events and receive notifications of those events. The automatic DR framework interfaces with the RCM and with the `sysevent` facility to enable applications to automatically give up resources prior to unconfiguring them and to capture new resources as they are configured into the domain.

# Enhanced System Availability

The DR feature enables you to hot-swap system boards without bringing the server down. It is used to unconfigure the resources on a faulty system board from a domain so that the system board can be removed from the server. The repaired or replacement board can be inserted into the domain while the Solaris operating system continues to run. DR then configures the resources on the board into the domain. If you use the DR feature to add or remove a system board or component, DR always leaves the board or component in a known configuration state. See [Chapter 2, “DR State and Condition Models” on page 15](#), for more information about configuration states for system board and components.

---

## DR Concepts

This section contains descriptions of general DR concepts that pertain to Sun Fire high-end system domains. For more information about DR concepts on the SC, refer to the *System Management Services (SMS) Dynamic Reconfiguration User Guide*.

## Detachability

A detachable device is one that conforms to the following rules:

- The device driver must support `DDI_DETACH`.
- Critical resources must be redundant or accessible through an alternate pathway. CPUs and memory banks can be redundant critical resources. Disk drives are examples of critical resources that can be accessible through an alternate pathway.

Some boards cannot be detached because their resources cannot be moved. For example, if a domain has only one CPU board, that CPU board cannot be detached. An I/O board is not detachable if it controls the boot drive.

If there is no alternate pathway for an I/O board, you can:

- Put the disk chain on a separate I/O board. The secondary I/O board can then be detached.
- Add a second path to the device through a second I/O board so that the I/O board can be detached without losing access to the secondary disk chain.

---

**Note** – If you are unsure whether a device is detachable, consult your Sun service representative.

---

# Quiescence

During the unconfigure operation on a system board with permanent memory (OpenBoot™ PROM or kernel memory), the operating system is briefly paused, which is known as operating system *quiescence*. All operating system and device activity on the domain must cease during this critical phase of the operation.

Before it can achieve quiescence, the operating system must temporarily suspend all processes, CPUs, and device activities. If the operating system cannot achieve quiescence, it displays the reasons, which may include the following:

- An execution thread did not suspend.
- A device exists that cannot be paused by the operating system.

---

**Note** – Real-time processes do not prevent quiescence.

---

The conditions that cause processes to fail to suspend are generally temporary. Examine the reasons for any failure, and if the operating system encountered a failure to suspend a process, simply try the operation again.

## Suspend-Safe and Suspend-Unsafe Devices

When DR suspends the operating system, all of the device drivers that are attached to the operating system must also be suspended. If a driver cannot be suspended (or subsequently resumed), the DR operation fails.

A *suspend-safe* device does not access memory or interrupt the system while the operating system is in quiescence. A driver is suspend-safe if it supports operating system quiescence (if it can be suspended and then resumed). A suspend-safe driver also guarantees that when a suspend request is successfully completed, the device that the driver manages will not attempt to access memory, even if the device is open when the suspend request is made.

A *suspend-unsafe* device allows a memory access or a system interruption to occur while the operating system is in quiescence.

DR uses an *unsafe driver list* in the `dr.conf` file to prevent unsafe devices from accessing memory or interrupting the operating system during a DR operation. The `dr.conf` file resides in the following directory: `/platform/SUNW,Sun-Fire-model_number/kernel/drv/`, where `model_number` is the machine name, such as 15000. The unsafe driver list is a property in the `dr.conf` file with the following format:

```
unsupported-io-drivers="driver1" , "driver2" , "driver3" ;
```

DR reads this list when it prepares to suspend the operating system so that it can unconfigure a memory component. If DR finds an active driver in the unsafe driver list, it aborts the DR operation and returns an error message. The message includes the identity of the active, unsafe driver. You must manually remove the usage of the device by performing one, or more, of the following tasks.

- Kill the processes using the device.
- Unload the driver by using the `modunload(1M)` command.
- Disconnect the cables (depending on the type of device).

You can retry the DR operation after you have stopped usage of the device.

---

**Note** – If you are unsure whether a device is suspend-safe, contact your Sun service representative.

---

## Attachment Points

An *attachment point* is a collective term that refers to a board slot, a system board installed in the slot, and any devices connected to the board. DR can display the status of an attachment point. The term *occupant* refers to the combination of a board and its attached devices.

- A board slot (sometimes referred to as a *receptacle*) has the ability to electrically isolate the occupant from the host machine. The software can put a board slot into low-power mode.
- Board slots can be named according to slot numbers, or can be anonymous (for example, a SCSI chain).
- An occupant I/O board includes any external storage devices connected by interface cables.

There are two types of names for attachment points:

- A *physical* attachment point describes the software driver and location of the slot. Examples of physical attachment point names are:

```
/devices/pseudo/dr@0:SBx (for a CPU/memory board in slot 0)
```

```
/devices/pseudo/dr@0:IOx (for an I/O board or Max CPU board in slot 1)
```

Where  $x$  represents the expander number for a particular board.

---

**Note** – CPU/memory boards are installed only in slot 0. I/O boards and Max CPU boards are installed only in slot 1.

---

- A *logical* attachment point is an abbreviated name created by the system to refer to the physical attachment point. Logical attachment points take one of the following two forms:

SBx (for CPU/memory boards in slot 0)

**-OR-**

IOx (for I/O boards or Max CPU boards in slot 1)

To obtain a list of all available logical attachment points, use the `cfgadm(1M)` command with its `-l` option.

## Conditions and States

A *state* is the operational status of either a board slot or its occupant. A *condition* is the operational status of an attachment point. The `cfgadm(1M)` command can display nine types of states and conditions. See [Chapter 2, “DR State and Condition Models” on page 15](#), for descriptions of the conditions and states for system boards and components.

## DR Operations

There are four main types of operations related to boards: connect, configure, unconfigure, and disconnect. A board that is brought into a domain is first connected and then configured. A board that is removed from a domain is first unconfigured and then disconnected.

During the connect operation, the system provides power to the slot, and the operating system begins monitoring the board’s temperature.

During the configure operation, the operating system assigns functional roles to the board, and loads device drivers for the board and for devices attached to it.

During the unconfigure operation, the system detaches the board logically from the operating system and takes the associated device drivers offline. Environmental monitoring continues, but devices on the board are not available for system use.

During the disconnect operation, the system stops monitoring the board and power to the slot is turned off.

To power-off a board that is in use (configured), first stop its use (unconfigure it), and then disconnect it from the domain. After a new or upgraded system board is inserted into the slot, connect the board and configure it.

The `cfgadm(1M)` command can connect and configure (or unconfigure and disconnect) in a single command. To connect and configure a board using a single command, see the section “[Adding a Board](#)” on page 37. To unconfigure and disconnect a board using a single command, see the section “[Removing a Board](#)” on page 35.

If necessary, each operation (connect, configure, unconfigure, or disconnect) can be performed separately using the `cfgadm(1M)` command.

## Hot-Plug Hardware

Hot-plug boards and modules have special connectors that supply electrical power to the board or module before the data pins make contact. Boards and devices that do not have hot-plug connectors cannot be inserted or removed while the system is running.

I/O boards and CPU/memory boards used in the Sun Fire high-end server are hot-plug devices. Some devices, such as the peripheral power supply, are not hot-plug modules and cannot be removed while the system is running.

---

## Dynamic System Domains

The Sun Fire high-end server can be divided into dynamic system domains, which are comprised of logical and physical groupings of system board slots. Each domain is electrically isolated into hardware partitions, which ensures that a problem encountered in one domain cannot affect other domains.

Domain configuration is determined by the domain configuration table in the platform configuration database (PCD), which resides on the SC. The domain table controls how system board slots are logically partitioned into domains. The domain configuration represents the intended domain configuration. Thus, the configuration can include empty slots and occupied slots.

The number of slots available to a given domain is controlled by an available component list that is maintained on the system controller. (Refer to the *System Management Services (SMS) Administrator Guide* for more information about the available component list.) After a slot has been assigned to a domain, it becomes visible to that domain and unavailable and invisible to any other domain. Conversely, you must disconnect and unassign a slot from its domain before you can assign and connect it to another domain.

The logical domain is the set of slots that belong to the domain. The physical domain is the set of boards that are physically interconnected. A slot can be a member of a logical domain and not be part of a physical domain.

After a domain is booted, the system boards and empty slots can be assigned to (or unassigned from) a logical domain; however, they cannot become a part of the physical domain until the operating system requests it.

System boards or slots that are not assigned to a domain are available to all domains in whose available component lists they appear. These boards can be assigned to a domain by the platform administrator. Or, an available component list can be set up on the system controller to allow users with appropriate privileges to assign available boards to a domain.

---

## Component Types

You can use DR to configure or to unconfigure several types of components:

Component Type	Description
cpu	An individual CPU
memory	All of the memory on the board
pci	Any I/O device, controller, or bus

---

## DR on I/O Boards

You must use caution when you add or remove I/O boards to which devices are attached. Before you can remove a board with I/O devices, all of its devices must be closed and all of its file systems must be unmounted.

If you need to remove an I/O board with attached devices from a domain temporarily and then re-add it before any other boards with I/O devices are added, reconfiguration is not necessary. In this case, device paths to the board devices remain unchanged.

## Solving a Problem With an I/O Device

- Run `showdevices(1M)` on the SC to determine the state and usage of the device.

- If disk mirroring is being used to access a device connected to the board, reconfigure the device so that it is accessible by way of controllers on other system boards.
- Unmount file systems.
- Remove multipathing databases from board-resident partitions. The location of multipathing databases is explicitly chosen by the user and can be changed.

Refer to the *System Management Services (SMS) Dynamic Reconfiguration User Guide* for special instructions for I/O devices.

- Remove any private regions used by volume managers. By default, volume managers use a private region on each device that they controls. Such devices must be removed from volume manager control before they can be detached.
- Take any RSM 2000 controllers offline by using the `rm6` or `rdacutil` commands.
- Remove disk partitions from the swap configuration.
- If a detach-unsafe device is present on the board, close all instances of the device and use `modunload(1M)` to unload the driver.
- Unmounting file systems may affect NFS client systems.

---

**Note** – Either kill any process that directly opens a device or raw partition, or direct it to close the open device on the board. If you use the `ndd(1M)` command to set the configuration parameters for network drivers, the parameters may not persist after a DR operation. Use the `/etc/system` file or the `driver.conf` file for a specific driver to set the parameters permanently.

---

## Golden IOSRAM

Each I/O board in a domain contains an IOSRAM device. However, only one IOSRAM device, called *golden IOSRAM*, is used for SC-to-domain communications at a time. The golden IOSRAM contains the “tunnel” that is used for SC-to-domain communications. Because DR can remove I/O boards, it is sometimes necessary to stop using the current golden IOSRAM and make another IOSRAM device the golden IOSRAM. This process is called a “tunnel switch,” and takes place whenever DR unconfigures the current golden IOSRAM.

When a domain is booted, the lowest-numbered I/O board in the domain is typically selected to be the initial golden IOSRAM.

## DR on hsPCI+ I/O Boards

DR supports dynamic reconfiguration of hsPCI+ I/O boards. Each hsPCI+ I/O board includes two XMITS ASICs and four hot-pluggable hsPCI slots.

---

# Permanent and Non-permanent Memory

Before you can delete a board, the operating system must vacate the memory on that board. Vacating a board entails flushing the contents of its non-permanent memory to swap space; and copying the contents of its permanent memory (that is, the kernel and OpenBoot™ PROM software) to another memory board.

To relocate permanent memory, the operating system on a domain must be temporarily quiesced. The length of the quiescence depends on the domain I/O configuration and the running workloads.

Detaching a board with permanent memory is the only time when the operating system is quiesced; therefore, you should know where permanent memory resides so that you can avoid impacting the operation of the domain significantly. To display the size of permanent memory, use the `cfgadm(1M)` command with its `-av` option. To vacate a board that has permanent memory, the operating system must find a sufficiently large block of available memory, called target memory, on which to copy the current contents of permanent memory, which is referred to as source memory.

## Target Memory Constraints

A DR memory operation can be disallowed if the target domain does not have enough memory to hold the contents currently stored in permanent memory.

## Correctable Memory Errors

Correctable memory errors indicate that the memory on a system board (that is, one or more of its Dual Inline Memory Modules (DIMMs), or portions of the hardware interconnect) may be faulty and need replacement. When the SC detects correctable memory errors, it initiates a record-stop dump to save the diagnostic data, which can interfere with a DR operation.

When a record-stop occurs from a correctable memory error, allow the record-stop dump to complete before you initiate a DR operation.

If the faulty component causes repeated reporting of correctable memory errors, the SC performs multiple record-stop dumps. If this happens, you should temporarily disable the dump-detection mechanism on the SC; allow the current dump to finish; then initiate the DR operation. After the DR operation finishes, re-enable the dump detection.

---

## Capacity on Demand (COD)

The COD option provides additional CPU resources on COD CPU/Memory boards that you install in your Sun Fire high-end system. Although your system comes configured with a minimum number of standard (active) CPU/Memory boards, it can have a mix of both standard and COD CPU/Memory boards installed, up to a maximum 18 boards on, for example, the Sun Fire E25K server. At least one active CPU is required for each domain in the system.

### DR on COD Boards

You can use DR to move COD boards into and out of domains in the same way you use DR to move standard CPU/Memory boards.

You can use the CPUs on a COD board only after you purchase right-to-use (RTU) licenses for them. Each COD RTU license entitles you to receive a COD RTU license key that enables a specified number of CPUs on COD boards in a single system. Whenever you use DR to configure a COD board into a domain, make sure that enough RTU licenses are available to the target domain to enable each active CPU on the COD board. If there are not enough RTU licenses available to a target domain when you add a COD board, a status message is displayed for each CPU that cannot be enabled in the domain.

For more information about the COD option, see the *System Management Services (SMS) Administrator Guide*.

---

# Enabling DR on Domains Running the Solaris 8 Operating System

While the Solaris 9 4/03 operating system supports the full functionality of DR, some previous versions of the Solaris operating system did not support reconfiguration of I/O boards.

Solaris 8 2/02 software is the first release of the Solaris 8 operating system to support the full functionality of DR on domains. Requirements include appropriate patches and a new kernel update on the domain, and SMS software no earlier than SMS 1.3 on the SC.

For complete information and instructions for enabling DR on a domain that is running Solaris 8 software, visit:

[http://www.sun.com/servers/highend/dr\\_sunfire](http://www.sun.com/servers/highend/dr_sunfire)

---

## An Illustration of DR Concepts

DR lets you disconnect, then reconnect system circuit boards without bringing the system down. You can use DR to add or remove system resources while the system continues to operate.

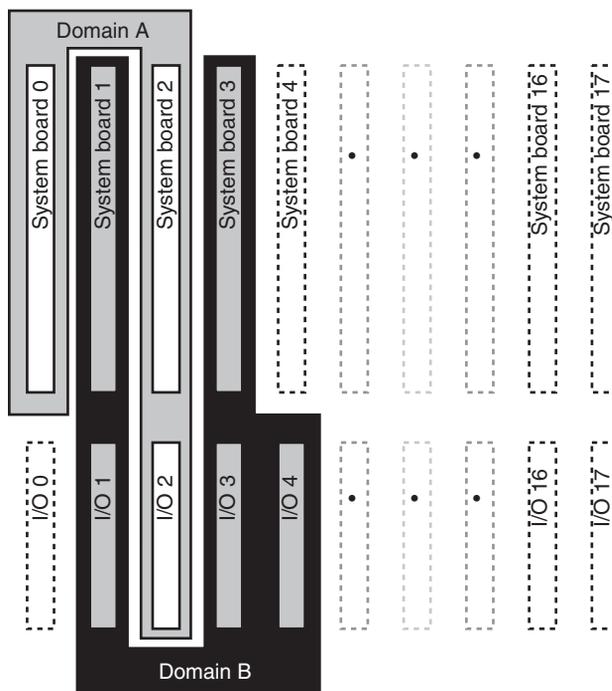
To illustrate reconfiguration of system resources consider the following Sun Fire E25K system configuration, as depicted in the diagram that follows.

---

**Note** – Sun Fire E25K and Sun Fire 15K systems support up to 18 system boards and 18 I/O boards at a time, numbered 0 through 17. Sun Fire E20K and Sun Fire 12K systems support up to nine system boards and nine I/O boards at a time, numbered 0 through 8.

---

Domain A contains system boards 0 and 2, and I/O board 2. Domain B contains system boards 1 and 3, and I/O boards 1, 3, and 4.



**FIGURE 1-1** Domains A & B before Reconfiguration

To assign system board 4 and I/O board 0 to Domain A, and to move I/O board 4 from Domain B to Domain A, you can use the Sun Management Center software's GUI. Or you can perform the following steps manually on the CLI in each domain as follows:

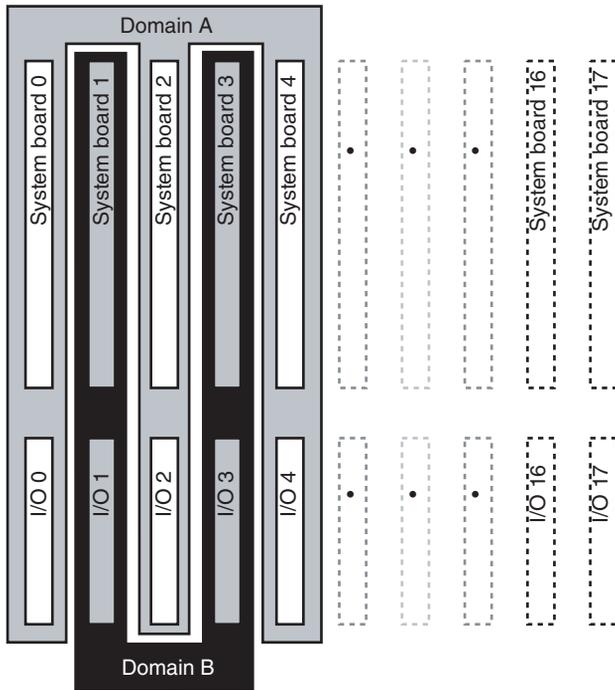
1. Enter the following configuration command on the command line in Domain B to disconnect I/O board 4 from Domain B:

```
# cfgadm -c disconnect -o nopoweroff,unassign IO4
```

2. Then, enter the following single command on the command line in Domain A, which assigns, connects, and configures system board 4 and I/O boards 0 and 4 into Domain A:

```
# cfgadm -c configure SB4 IO0 IO4
```

The following system configuration is the result. Only the way in which the boards are connected has changed, not the physical layout of the boards within the cabinet.



**FIGURE 1-2** Domains A & B after Reconfiguration

## DR State and Condition Models

---

This chapter contains descriptions of the state and condition models for boards and components. The state models are divided into two categories: receptacle and occupant.

Before you attempt to perform any DR operation on a board or component from the domain, determine its state and condition. To display the type, state, and condition of each component and the state and condition of each board slot in the domain, use the `cfgadm(1M)` command with the `-la` options. See the section “[Component Types](#)” on page 8 for a list of the component types.

You can use the `prtdiag(1M)` command to display information about board slots and components. The `prtdiag(1M)` command displays board numbers in the format `SBxx` or `IOxx`, where `xx` is the board number that includes leading zeroes.

---

## Board States and Conditions

This section contains descriptions of the states and conditions of system boards and board slots (also known as receptacles).

## Board Slot States

A slot can have one of three states: empty, disconnected, or connected.

Name	Description
empty	A board is not present.
disconnected	The board is disconnected from the system bus. A board can be in the disconnected state without being powered off. However, a board must be powered off and in the disconnected state before you remove it from the slot.
connected	The board is powered on and connected to the system bus. You can view the components on a board only after it is in the connected state.

Whenever you insert a board into a slot, the slot's state changes from empty to disconnected. Whenever you remove a board, the slot's state changes from disconnected to empty.



**Caution** – Physically removing a board that is in the connected state, or that is powered on and in the disconnected state, crashes the operating system and can result in permanent damage to that system board.

## Board Occupant States

A board can have one of two occupant states: configured or unconfigured. The occupant state of a disconnected board is always unconfigured.

Name	Description
configured	At least one component on the board is configured.
unconfigured	All of the components on the board are unconfigured.

## Board Conditions

A board can be in one of four conditions: unknown, ok, failed, or unusable.

<b>Name</b>	<b>Description</b>
unknown	The board has not been tested.
ok	The board is operational.
failed	The board failed testing.
unusable	The board slot is unusable.

---

## Component States and Conditions

This section contains descriptions of the states and conditions for components.

### Component Receptacle States

A component cannot be individually connected or disconnected. Thus, components can have only one state: connected.

### Component Occupant States

A component can have one of two occupant states: configured or unconfigured. The following table contains the name and description of the occupant states for components.

<b>Name</b>	<b>Description</b>
configured	The component is available for use by the Solaris operating system.
unconfigured	The component is not available for use by the Solaris operating system.

# Component Conditions

A component can have one of three conditions: unknown, ok, and failed. The following table contains the name and description of the conditions for components.

<b>Name</b>	<b>Description</b>
unknown	The component has not been tested.
ok	The component is operational.
failed	The component failed testing.

## DR Operations and Software Components on the Domain

---

This chapter contains descriptions of the four general DR operations: connect, configure, disconnect, and unconfigure. For more information on how to perform these operations, see [Chapter 5, “DR Domain Procedures” on page 33](#).

This chapter also contains information about the various software components that work together to accomplish DR operations. The components that are used during a DR operation depend entirely on the point of initiation of the DR operation. For example, if you initiate the DR operation from the SC, the system uses several more software components to accomplish the DR operation than when you initiate the DR operation from the domain.

For more information about the software components that reside on the SC, refer to the *System Management Services (SMS) Dynamic Reconfiguration User Guide*.

---

## DR Operations

This section contains descriptions of the four general DR operations: connect, configure, disconnect, and unconfigure. These operations are described from the point of view of the domain. They do not contain information that is specific to the SC.

### Before You Perform DR Operations

Before you perform DR operations for the first time on a domain after it has been booted, make sure the board is available to the domain. To display a list of boards that are available to the domain, use the `cEgadm(1M)` command with its `-l` option.

An error may occur if you attempt to perform DR operations on a board that:

- Is not listed in the domain's available component list and is not assigned to the domain; or
- Is listed in the domain's available component list, but is assigned to another domain.

In either of these cases, the board is not available to the domain. For more information about the available component list refer to the *System Management Services (SMS) Administrator Guide*.

## Before Performing DR Operations on I/O Boards

Before you attempt to perform DR operations on an I/O board in a domain, make sure there are at least two CPUs available to the domain. Further, make sure that at least one of those CPUs is located on a CPU/memory board; and that no processes are bound to it. See the `pbind(1M)` man page for more information about bound processes.

When you use DR to configure an I/O board into a domain (or to test an I/O board explicitly using the `cfgadm(1M)` command with its `-t` option), one CPU that is an occupant on a CPU/Memory board in the same domain is selected to test the board. Further, no process can be bound to the CPU, and at least one additional CPU must remain in the domain. If no such CPU is available to perform the test, a message such as the following is displayed:

```
WARNING: No CPU available for I/O cage test
```

The CPU is unconfigured from the domain and the I/O board tested. After the test is complete, the CPU is configured back into the domain. After the CPU is successfully reconfigured, its timestamp as displayed by the `psrinfo(1M)` command will differ from timestamps for other CPUs in the domain.

## Connect Operation

During the connect operation, DR attempts to assign the slot to the domain if a system board is available and if it is not part of any logical domain. After the slot has been assigned, DR requests that the SC power on and test the board. After the board has been tested, DR requests the SC to connect the board electronically to the system, which makes the board part of the physical domain. The operating system then probes the components on the board.

To connect a system board through the domain rather than the SC, use the `cfgadm(1M)` command as follows:

```
# cfgadm -c connect SBx
```

where *x* represents the number of a particular board.

---

**Note** – If the `cfgadm(1M)` command fails during a DR operation, the board does not return to its original state. If the error is recoverable, you can retry the command. If the error is unrecoverable, you will need to reboot the domain to use the board.

---

The syntax of the `cfgadm(1M)` command to connect an I/O board is as follows:

```
# cfgadm -c connect IOx
```

where *x* represents the number of a particular board.

The states and conditions for the attachment point before a board is inserted are:

- Receptacle state—Empty
- Occupant state—Unconfigured
- Condition—Unknown

After a board is physically inserted, the states and conditions are:

- Receptacle state—Disconnected
- Occupant state—Unconfigured
- Condition—Unknown

After the attachment point is logically connected, the states and conditions are:

- Receptacle state—Connected
- Occupant state—Unconfigured
- Condition—OK

## Configure Operation

During the configure operation, DR attempts to connect the board slot if its state is disconnected. It then traverses the tree of devices that was created during the connect operation. (DR creates Solaris device tree nodes and attaches device drivers if necessary.)

The CPUs are added to the CPU list; and memory is initialized and added to the system memory pool. After the configure function has completed successfully, the CPUs and memory are ready for use.

For I/O devices, use the `mount(1M)` and the `ifconfig(1M)` commands before the devices can be used.

When you configure a board into a domain using `cfgadm`, the board is automatically connected and configured

## CPUs and Memory

To configure a CPU on a system board through the domain rather than the SC, use the `cfgadm(1M)` command as follows:

```
# cfgadm -c configure SBx:cpu $y$ 
```

where  $x$  represents the board number and  $y$  represents the CPU number, which is 0 through 3 for all Sun Fire high-end systems.

The syntax of the `cfgadm(1M)` command to configure memory is as follows:

```
# cfgadm -c configure SBx:memory
```

where  $x$  represents the board number. For memory, the command applies to all the memory on the system board.

To configure all the CPUs and memory on a system board, use the following command:

```
# cfgadm -c configure SBx
```

## I/O Boards

To configure one of the PCI slots that holds the PCI adapter with hotplug capability, the syntax of the `cfgadm(1M)` command is as follows:

```
# cfgadm -c configure pci_ap_id
```

For more information, see [“Hot Plugging PCI Adapter Cards” on page 41](#).

To configure an I/O board, use the following command:

```
# cfgadm -c configure IOx
```

## After the Configure Operation

The states and conditions for a configured attachment point are:

- Receptacle state—Connected
- Occupant state—Configured
- Condition—OK

Now the system is aware of the usable devices that reside on the board, and all devices can be mounted or configured for use.

## Disconnect Operation

During a disconnect operation, the DR framework communicates with the SC to program the interconnect so that the system board is removed from the physical domain. It then attempts to perform the tasks related to the unconfigure operation.

A board can be in the disconnected state without being powered off. However, the board must be powered off and in the disconnected state before you can remove it from the slot.

The syntax of the `cfgadm(1M)` command to disconnect the board is as follows:

```
# cfgadm -c disconnect SBx
```

where *x* represents the number of a particular board.

Before the board is disconnected, the states and conditions are:

- Receptacle state—Connected
- Occupant state—Configured
- Condition—OK

After the board is disconnected, the states and conditions are:

- Receptacle state—Disconnected
- Occupant state—Unconfigured
- Condition—Unknown

# Unconfigure Operation

The unconfigure operation can consist of a single operation or two separate operations, depending on the presence of permanent memory. If the system board hosts permanent memory, before the unconfigure operation DR moves the memory contents from the specified board to available memory on a target board in the domain. See the section “[Permanent and Non-permanent Memory](#)” on page 10 for more information about boards that host permanent memory.

## Non-permanent Memory

If the reconfiguration coordination manager (RCM) is present, then DR informs the RCM about the DR operation. The RCM informs client applications, and the client applications perform preparatory tasks such as stopping the usage of devices. The clients communicate their readiness to the RCM, and the RCM communicates its readiness to DR. Depending on the responses, DR either continues, or aborts the operation and reports an error to the user.

During the unconfigure operation, DR unconfigures the board resources from the Solaris operating system and leaves the board in the disconnected state.

If the board hosts CPUs and/or memory, DR removes them from the Solaris operating system, making them unusable to the operating system. If the board is an I/O board, DR detaches the device drivers.

## Permanent Memory

The following paragraphs and examples specifically illustrate the unconfigure operation for permanent memory.

In the following code examples, the permanent memory on board 0 must be moved to another board in the domain, board 1. Board 0 is the *source* board, and board 1 is the *target* board.

For brevity, the CPU information has been removed from the code examples. On the domain, the unconfigure operation is started with the `cfgadm(1M)` command:

```
# cfgadm -c unconfigure -y SB0::memory &
```

First, a block of memory on the target board that resides in the same address range as the permanent memory on the source board must be deleted. During this phase, the source board, the target board, and the memory attachment points are marked as busy. You can display the status with the following command:

```
# cfigadm -a -s cols=ap_id:type:r_state_o_state=busy SB0 SB1
```

Ap_Id	Type	Receptacle	Occupant	Busy
SB0	CPU	connected	configured	y
SB0::memory	memory	connected	configured	y
SB1	CPU	connected	configured	y
SB1::memory	memory	connected	configured	y

After the memory has been deleted on board 1, it is marked as unconfigured. The memory the source board remains configured, but it is still marked as busy, as in the following example.

Ap_Id	Type	Receptacle	Occupant	Busy
SB0	CPU	connected	configured	y
SB0::memory	memory	connected	configured	y
SB1	CPU	connected	configured	y
SB1::memory	memory	connected	unconfigured	n

The memory from the source board is then copied to the target board. After it has been copied, the occupant state for the memory is switched. The memory on the source board becomes unconfigured, and the memory on the target board becomes configured. At this point in the process, only the source board remains busy, as in the following example.

Ap_Id	Type	Receptacle	Occupant	Busy
SB0	CPU	connected	configured	y
SB0::memory	memory	connected	unconfigured	n
SB1	CPU	connected	configured	n
SB1::memory	memory	connected	configured	n

After the entire process has been completed, the memory on the source board remains unconfigured, and the attachment points are not busy, as in the following example.

Ap_Id	Type	Receptacle	Occupant	Busy
SB0	CPU	connected	configured	n
SB0::memory	memory	connected	unconfigured	n
SB1	CPU	connected	configured	n
SB1::memory	memory	connected	configured	n

The permanent memory has been moved, and the memory on the source board has been unconfigured. At this point, you can initiate a new state change operation on either board.

---

## Software Components

This section describes the software components that reside on the domain and make DR operations possible. However, it does not contain descriptions of all of the DR components on the system platform. Refer to the *System Management Services (SMS) Dynamic Reconfiguration User Guide* for descriptions of the software components that reside on the SC.

### Domain Configuration Server

The domain configuration server (DCS) is a daemon process that runs on a domain and is started by `inetd(1M)` when the first remote DR request is received. A single instance of the DCS runs in each domain. The DCS accepts DR requests from the domain configuration agent (DCA) that runs on the SC. After the DCS accepts a DR operation, it performs the request and returns the results to the DCA. Refer to the *System Management Services (SMS) Dynamic Reconfiguration User Guide* for more information about the DCA.

---

**Note** – In domains that run Solaris 10, the DCS has no entries in the `inetd.conf` file. In domains running earlier versions of the Solaris software, DCS does have an entry in `inetd.conf`. In this latter case, if you alter or remove the `sun-dr` entry in `inetd.conf`, make the same change to the `sun-dr` entry in the `ipsecinit.conf` file.

---

## DR Driver

The DR driver consists of a platform independent driver, named `dr`, and a platform specific module, named `drmach`. The DR driver uses standard features of the Solaris operating system whenever possible to control DR operations, and it calls the platform specific module as needed. The DR driver is responsible for creating minor nodes in the file system that are used as attachment points for DR operations.

## Reconfiguration Coordination Manager

The reconfiguration coordination manager (RCM) is a daemon process that coordinates DR operations on resources that are present in the domain. The RCM daemon uses generic application program interfaces (APIs) to coordinate DR operations between DR initiators and RCM clients.

The RCM consumers consist of DR initiators, which request DR operations, and DR clients, which react to DR requests. Normally, the DR initiator is the configuration administration command, `cfgadm(1M)`. However, it can also be a GUI such as Sun<sup>TM</sup> Management Center.

The DR clients can be:

- Software layers that export high-level resources comprised of one or more hardware devices (for example, multipathing applications)
- Applications that monitor DR operations (for example, Sun Management Center)
- Entities on a remote system, such as the system controller on a server

## System Events Framework

DR uses the Solaris system events framework to notify other software entities of the occurrence of changes that result from a DR operation. DR accomplishes this by sending DR events to the system event daemon, `syseventd`, which, in turn, sends the events to the subscribers of DR events. For more information about the system events daemon, refer to the `syseventd(1M)` man page.



## DR User Interfaces on the Domain

---

This chapter describes the user interfaces on the domain, which include the commands and options that are available to the user, and important files.

---

### DR Commands and Options on the Domain

The `cfgadm(1M)` command is used to perform DR operations on the domain. DR operations are passed to the `libcfgadm(3LIB)` library interface, which dynamically loads a hardware-specific library plugin that actually performs the DR operations.

---

**Note** – If the `cfgadm(1M)` command fails during a DR operation, the board does not return to its original state. If the error is recoverable, you can retry the command. If the error is unrecoverable, you must reboot the domain to use the board.

---

The `sbd.so.1` hardware-specific plug-in provides DR functionality: connect, configure, unconfigure, and disconnect system boards, which enables you to connect or disconnect a system board from a running system without having to reboot the system.

The `cfgadm(1M)` command resides in the `/usr/sbin` directory. (See the `cfgadm(1M)` man page for more information.)

Each board slot appears as a single attachment point in the device tree. You can view the type, state, and condition of each component, and the state and condition of each board slot, by using the `cfgadm(1M)` command with its `-a` option.

## State Change Functions

State change functions, which change the state of a board slot or a component on the board, can be issued against any attachment point.

The following are state change functions:

- configure
- unconfigure
- connect
- disconnect

## Availability Change Functions

Availability change functions, which change the availability of a board, can be issued against any attachment point.

The following are availability change functions:

- assign
- unassign

## Condition Change Functions

Condition change functions, which change the condition of a board slot or a component on the board, can be issued against any attachment point.

The following are condition change functions:

- poweron
- poweroff
- test

## Options and Operands

The following options and operands are supported for the functions shown, where *ap\_id* specifies the attachment point of the system board or component.:

Options and Operands	Specifies
-c connect <i>ap_id</i>	Changes the receptacle state to connected.
-c disconnect <i>ap_id</i>	Changes the receptacle state to disconnected.
-c configure <i>ap_id</i>	Changes the occupant state to configured.
-c unconfigure <i>ap_id</i>	Changes the occupant state to unconfigured.
-x assign <i>ap_id</i>	Changes the occupant state to assigned.
-x unassign <i>ap_id</i>	Changes the occupant state to unassigned.
-x poweron <i>ap_id</i>	Changes the occupant state to powered-on.
-x poweroff <i>ap_id</i>	Changes the occupant state to powered-off.
-l <i>ap_id</i>	Displays the state, status, and condition of system boards and components.
-h [ <i>ap_id</i> ]	Prints out a help message text.If <i>ap_id</i> is specified, the help routine of the hardware-specific library for the attachment point indicated by the argument is called.
-v	Executes in verbose mode.
-n	Suppresses command prompts for user confirmation, and automatically answers "no."
-y	Suppresses command prompts for user confirmation, and automatically answers "yes."

Options and Operands	Specifies
-s <i>listing_options</i>	The state of attachment points to be displayed according to <i>listing_options</i> . Supplies listing options to the list (-l) command. The <i>listing_options</i> argument conforms to the syntax conventions of the <code>getsubopt(3C)</code> man page, and specifies: the attachment point selection criteria (i.e., <code>select=select_string</code> ); the type of matching desired (i.e., <code>match=match_type</code> ); the order of listing (i.e., <code>sort=field_spec</code> ); the data that is displayed (i.e., <code>cols=field_spec</code> and <code>cols2=field_spec</code> ); the column delimiter (i.e., <code>delim=string</code> ); and whether to suppress column headings (i.e., <code>noheadings</code> ).
-o <i>hardware_options</i>	Supplies hardware-specific options to the main command option. The format and content of the <i>hardware_options</i> string is completely hardware-specific; and the string conforms to the syntax conventions of the <code>getsubopt(3C)</code> man page.
-t <i>ap_id</i>	Performs a test of one or more attachment points. The test function is used to re-evaluate the condition of the attachment point. Without a test-level specifier in <i>hardware_options</i> , the fastest test that identifies hard faults is used.

## DR Domain Procedures

---

This chapter describes how you use the `cfgadm(1M)` command on the domain to perform DR operations. It also describes attachment points and procedures for displaying the status of system boards.

---

### Attachment Points

Before you use the `cfgadm(1M)` command, make sure you understand the syntax for attachment points on the Sun Fire high-end system platform. There are physical and logical attachment points. In addition, single attachment points are used for board slots, and dynamic attachment points are used for components. Attachment points created by the DR driver have a physical and logical path.

Physical attachment points for system boards take the following form:

```
/devices/pseudo/dr@0:SBx (for CPU/memory boards)
-OR-
/devices/pseudo/dr@0:IOx (for I/O boards)
```

where  $x$  represents the number of an expander board.

Logical attachment points for system boards take the following form:

```
SBx (for CPU/memory boards)
-OR-
IOx (for I/O boards)
```

where  $x$  represents the number of a board.

Dynamic attachment points refer to components (CPUs and memory) on system boards and I/O devices on I/O boards. The attachment points are created by the DR driver. Refer to the `dr(7D)` man page for more details.

---

## Displaying Board Status

The `cfgadm(1M)` command displays information about boards and slots. Refer to the `cfgadm_sbd(1M)` man page for options to this command.

### Basic Status Display

Many operations require that you specify the system board names. To obtain these system board names, type:

```
# cfgadm -a -s "select=class(sbd)"
```

The `cfgadm(1M)` command displays information only about those boards that are assigned to the domain; or those boards that appear in the available component list for the domain and are not assigned to any other domain.

The following output is typical:

Ap_Id	Type	Receptacle	Occupant	Condition
SB0	CPU	connected	configured	ok
SB0::cpu0	cpu	connected	configured	ok
SB0::memory	memory	connected	configured	ok
IO1	PCI	connected	configured	ok
IO1::pci0	io	disconnected	unconfigured	failed

### Detailed Status Display

For a more detailed status report, use the `cfgadm(1M)` command with its `-v` (verbose) option, which turns on expanded descriptions. In addition to basic information such as the attachment point ID, receptacle and occupant states, and board status, the expanded status report also includes the date when the board was configured into the domain, the type of board, the activity state, and the physical attachment point.

---

# Removing a Board

This section describes how to remove a CPU/Memory and an I/O board.

## ▼ To Remove a CPU/Memory Board

To perform the following steps, you must have domain administrator privileges.

1. **Log in to the domain.**
2. **Use the `cfgadm(1M)` command with the `-l` option to determine the attachment point for the board.**
3. **Verify that the board does not have bound processes running.**

If a process is bound to a CPU, the board cannot be removed until the process is unbound. Refer to the `pbind(1M)` man page for more information.

---

**Note** – Beginning with Solaris 10 software, the `-f` (force) option attempts to unbind processes bound to a CPU being unconfigured. No attempt is made to rebind any automatically bound process.

---

4. **Unconfigure and disconnect the board using the following single command:**

```
# cfgadm -v -c disconnect SBx
```

where *x* represents the board number.



---

**Caution** – Do not physically remove a board until it is disconnected. Otherwise the board will be damaged.

---

## ▼ To Remove an I/O Board

To remove an I/O board, you must first stop all usage of the board. To complete the steps in this procedure, you must have domain administrator privileges.

1. **Log in to the domain.**

2. Check the status of the board.

```
# cfgadm -a -s "select=class(sbd)"
```

3. If the system is using multipathing software:

- a. Switch all board functions to the alternate board.
- b. Remove any multipathing databases and/or private regions.
- c. Wait until all of the alternate paths are functioning before proceeding.

4. Unmount file systems, including metadevices that have a board resident partition (for example: `umount /partition`).

---

**Caution** – Unmounting file systems may affect NFS client systems.

---

5. If the board contains Sun RSM Array™ 2000 controllers, take the controllers off line, using the `rm6` or `rdacutil` commands.

6. Remove disk partitions from the swap configuration.

7. If any process directly opens a device or raw partition, either kill the process or direct it to close the open device on the board.

8. If a detach-unsafe device is present on the board, close all instances of the device and use `modunload(1M)` to unload the driver.

9. Disconnect the board.

```
# cfgadm -v -c disconnect IOx
```

where *x* represents the board number.

---

**Note** – If the `cfgadm(1M)` command fails during a DR operation, the board does not return to its original state. If the error is recoverable, you can retry the command. If the error is unrecoverable, you will need to reboot the domain to use the board.

---

---

# Adding a Board

Before installing a board, consider the following points:

- Never use a board that is defective or suspected to be unreliable; it can crash the system.
- The board type and option cards must be supported by DR.

---

**Note** – Whenever you use DR to add a COD board into a domain, make sure that enough COD right-to-use (RTU) licenses are available to the target domain to enable each active CPU on the COD board. If there are not enough RTU licenses available to a target domain when you add a COD board to the domain, DR displays an error message for each CPU that cannot be enabled in the domain. For more information about the COD option, see the *System Management Services (SMS) Administrator Guide*.

---

## ▼ To Install a Board

To perform a board installation from the domain, the board must already be assigned to the domain, or must be in the available component list. Refer to the *System Management Services (SMS) Administrator Guide* for information on how to assign boards or to update the available component list.

### 1. Verify that the selected board slot can accept a board.

```
# cfgadm -a -s "select=class(sbd)"
```

The states and conditions should be:

- Receptacle state—Empty
- Occupant state—Unconfigured
- Condition—Unknown

-OR-

- Receptacle state—Disconnected
- Occupant state—Unconfigured
- Condition—Unknown

## 2. Connect and configure the board using a single command.

```
# cfgadm -v -c configure SBx (CPU/memory board)
-OR-
# cfgadm -v -c configure IOx (I/O board)
```

where  $x$  represents the board number.

After a short delay during which the system tests the board, a message appears in the domain console log indicating that the components have been configured. The states and conditions for a connected and configured attachment point should be:

- Receptacle state—Connected
- Occupant state—Configured
- Condition—OK

Now the system is aware of the usable devices on the board and the devices can be used.

---

**Note** – If the `cfgadm(1M)` command fails during a DR operation, the board does not return to its original state. If the error is recoverable, you can retry the command. If the error is unrecoverable, you will need to reboot the domain to use the board.

---

---

# DR Using `cfgadm(1M)` - Examples

## Displaying Help

To display help text for commands use the `-h` option. If the `-h` option is followed by an attachment point identifier, help messages (syntax) related to the hardware-specific library of the attachment point are shown.

**TABLE 5-1** Commands that Display Help

Command	Description
# <code>cfgadm -h</code>	Shows general syntax.
# <code>cfgadm -h pci</code>	Shows PCI hotplug-specific commands.
# <code>cfgadm -h SB2</code>	Shows help related to system board-specific commands and options.

## Displaying Verbose Messages

The `-v` option displays detailed messages as DR operations proceed. For example:

To configure the memory on system board 2 (SB2) with the verbose option use:

```
# cfgadm -v -c configure SB2::memory
```

To unconfigure CPU 3 (cpu3) on system board 0 (SB0) with the verbose option, use the following command:

```
# cfgadm -v -c unconfigure SB0::cpu3
```

## Suppressing User Confirmation

When certain `cfgadm` commands are entered (such as unconfigure permanent memory), the command prompts the user to confirm the operation, yes or no. For example, the following command unconfigures system board 6 (SB6), which holds permanent system memory, and prompts the user for confirmation:

```
# cfgadm -c unconfigure SB6::memory  
System may be temporarily suspended, proceed (yes/no)?
```

You can suppress the confirmation prompt by using the `-y` or `-n` option on the command line. The `-y` option automatically responds with “yes” and the `-n` option responds with “no.” The following example performs exactly the same operation as the previous command, but uses the `-y` option to bypass user confirmation:

```
# cfgadm -y -c unconfigure SB6::memory  
#
```

## Power Control When Disconnecting Boards

To unconfigure system board 6 (SB6), which holds the domain's permanent memory, use a command similar to the following. In the following example, `-y` answers yes to any prompts, `unassign` overrides the default behavior of `disconnect`, leaving the board assigned, and `nopoweroff` overrides the default behavior of `disconnect`, powering off the board.

```
# cfgadm -y -v -c disconnect -o unassign,nopoweroff SB6
```

To disconnect I/O board 12 (IO12), but leave it powered-off and assigned to the same domain, use:

```
# cfgadm -c disconnect IO12
```

## Power Control of Disconnected Boards

To power-on system board 2 (SB2), use the following command:

```
# cfgadm -x poweron SB2
```

To power-off system board 2 (SB2), use the following command:

```
# cfgadm -x poweroff SB2
```

## Connecting and Configuring Boards

When DR configures a board into a domain, it first connects the board electrically to the system, putting it into the *connected* state. DR then configures the system board so that it is fully available to all applications running in the domain, putting it into the *configured* state.

When DR removes a board from a domain, it first unconfigures the system board so that it is no longer available to all applications running in the domain, putting it into the *unconfigured* state. DR then disconnects the board electrically from the system, putting it into the *disconnected* state.

## Hot Plugging PCI Adapter Cards

Each hotplug slot on an I/O board can be individually connected, configured, unconfigured, and disconnected. Each attachment point for a hotplug slot, which identifies both the slot and the adapter card that is plugged into the slot, is created when the I/O board is configured into the domain.

To connect, but not configure, an adapter at slot 1 of I/O board 1 into a domain, use a command such as the following:

```
# cfgadm -c connect pcisch0:e01b1slot1
```

To configure the adapter at slot 1 of I/O board 1 into the domain, use a command such as the following:

```
# cfgadm -c configure pcisch0:e01b1slot1
```

To disconnect an adapter at slot 1 of I/O board 1 before unplugging the adapter, use a command such as the following:

```
# cfgadm -c disconnect pcisch13:e01b1slot1
```

To unconfigure the adapter at slot 1 of I/O board 1 out of the domain, use a command such as the following:

```
# cfgadm -c unconfigure pcisch0:e01b1slot1
```

For more information, see `cfgadm_pci(1M)`.

## Testing a Board

The `-t` option causes a board to be tested. Prior to running the following command, system board 2 (SB2) must be disconnected, assigned and powered-on. The following command includes the verbose option:

```
# cfgadm -vt SB2
```

The board is tested using the diagnostic level specified for the domain in the `.postrc` file; the default is 16.

## Displaying Attachment Point Information

This section includes several example of commands that you can use to display system information about attachment points. See the `cfgadm(1M)` man page for additional information.

To list the state, status, and condition of all attachment points with the verbose option use:

```
# cfgadm -val
```

To list the state and condition of an adapter at slot 1 of I/O board 3 use the following command:

```
# cfgadm -al pcisch13:e03b1slot1
```

The following command displays in columnar format the logical name of each attachment point; along with its condition; status time in both calendar and parsable formats; and other information:

```
# cfgadm -s "cols=ap_id:condition:status_time:status_time_p:info"
```

The following command displays in columnar format the logical name and physical ID of each attachment point:

```
# cfgadm -s "cols=ap_id:physid"
```

The following command displays in columnar format the logical name of each attachment point, along with its receptacle state; occupant state; occupant type; busy status; and class:

```
# cfgadm -s "cols=ap_id:r_state:o_state:type:busy:class"
```

## Tracking Memory Unconfigure Operations

When unconfiguring a system board that contains the domain's permanent memory, the following command tracks the memory delete process:

```
# cfgadm -a -s "select=type(memory),cols=ap_id:o_state:info"
```

## Finding the Board Containing Permanent Memory

To find the system board that contains the domain's permanent memory use the following command:

```
# cfgadm -val | grep permanent
```



# Index

---

## A

- adding a board, 37
- ADR on I/O boards, 8
- attachment points
  - as operands, 31
  - description of, 5
  - displaying information about, 42
  - states and conditions, 23, 26
  - syntax, 33
- Automatic DR, 2
- availability change functions, 30
- available component list, 7, 20

## B

- boards
  - conditions, 6, 15
  - hot plug, 7
  - receptacle states, 16
  - slots, 29
  - states, 15
  - testing, 41

## C

- Capacity on Demand (COD), 11, 37
- cfgadm(1M)
  - adding a board, 37
  - configuring a board via the domain, 22
  - configuring a bus on an I/O board via the domain, 22
  - configuring memory on a board via the domain, 22
  - connecting a board via the domain, 21

- displaying board status, 34
- displaying permanent memory size, 10
- examples, 38
- functions, 7
- removing a board, 35
- unconfiguring a board via the domain, 24

- COD, 11, 37
- component
  - conditions, 18
  - states, 17
  - types, 8
- condition change functions, 30
  - options and operands, 31
- condition models, 15
- configure operation, 6, 21
- configured state, 16, 17
- connect operation, 6, 20
- connected state, 16
- CPUs
  - as component types, 8
  - detachability of, 3
  - suspending, 4

## D

- DCA, 26
- DCS, 26
- DDI\_DETACH, 3
- detachability
  - device, 3
- disconnect operation, 6, 23
- disconnected boards

- power control of, 40
- disconnected state, 16
- disconnecting boards
  - power control when, 40
- disk
  - mirroring, 9
  - partitions, 9
- displaying board status, 34
- domain configuration agent, 26
- domain configuration server, 26
- domains
  - description of, 7
  - logical, 8
  - physical, 8
  - platform configuration database, 7
- DR
  - clients, 27
  - command line interface, 2
  - commands on the domain, 29
  - concepts, 3
  - configure operation, 6
  - connect operation, 6
  - disconnect operation, 6
  - driver, 27
  - hardware-specific plugin and, 29
  - illustration of DR operations, 12
  - on I/O boards
    - preparations for, 20
  - operations, 6
  - options on the domain, 29
  - unconfigure operation, 6
- dr(7D), 34
- dr.conf file, 4
- drivers
  - unsafe, 4
- drmach, 27
- DR-unsafe device, 4
- dual-inline-memory-modules, 10
- dynamic system domains, 7

## E

- empty
  - slots (receptacles), 7
  - state, 16

## F

- failed condition, 17, 18
- functions
  - availability change, 30
  - condition change, 30
  - state change, 30

## G

- golden IOSRAM, 9

## H

- help
  - displaying, 38
- hot swap, 3
- hot-plug boards, 7

## I

- I/O devices
  - as component types, 8
  - before performing DR operations on, 20
  - detachability, 3
  - suspending, 4
  - suspend-safe, 4
  - with ADR, 8
- ifconfig(1M), 22
- IOSRAM
  - golden, 9

## L

- logical attachment point, 6
- logical domain, 8

## M

- memory
  - as a component type, 8
  - correctable errors, 10
  - non-permanent, 10
  - permanent, 10
  - source, 10
  - target, 10
  - unconfiguring
    - tracking the progress of, 43
- mount(1M), 22
- multipathing databases, 9

## **N**

ndd(1M), 9  
non-permanent memory, 10, 24

## **O**

occupant  
    defined, 5  
occupant states, 16  
ok condition, 17, 18

## **P**

permanent memory, 10, 24  
    finding the board that contains, 43  
physical attachment point, 5  
physical domain, 8  
platform configuration database, 7  
populated slots (receptacles), 7  
power control  
    of disconnected boards, 40  
    when disconnecting boards, 40  
processes  
    real-time, 4  
prtdiag command, 15  
psrinfo(1M), 20

## **Q**

quiescence, 4, 10

## **R**

raw partitions, 9  
RCM consumers, 27  
real-time processes, 4  
receptacle (slot), 5  
receptacle state, 16  
reconfiguration coordination manager, 2, 27  
record-stop dumps, 11  
removing a board, 35  
RSM 2000 controllers, 9

## **S**

SC (system controller), 7  
showdevices(1M)  
    with I/O devices, 8  
slot numbers, 5

slots (receptacles), 7  
Solaris 8 2/02 operating environment  
    enabling full DR on domains running, 12  
state change functions, 30  
state models, 15  
status display  
    basic, 34  
    detailed, 34  
suspend-safe devices, 4  
sysevent, 2  
syseventd, 27  
system controller (SC), 7  
system events framework, 27

## **T**

testing boards, 41

## **U**

unconfigure operation, 6, 24  
unconfigured state, 16, 17  
unconfiguring memory  
    tracking the progress of, 43  
unknown condition, 17, 18  
unsafe devices, 4  
unusable condition, 17  
user confirmation  
    suppressing, 39, 40  
user interfaces, 29

## **V**

verbose messages  
    displaying, 39  
volume managers, 9

## **X**

XMITS boards, 9

