

# Implementing Root Domains with Oracle VM Server for SPARC

ORACLE TECHNICAL WHITE PAPER | NOVEMBER 2015







Table of Contents	
Table of Contents	1
Introduction	4
Overview of Virtualization Technologies	4
Type 1 Hypervisors	4
Type 2 Hypervisors	5
OS-Based Virtualization	5
Overview of Oracle VM Server for SPARC	5
Traditional Deployment Model	8
Introducing Root Domains	8
I/O Domain Models	9
Root Domains	10
Best Practices for Availability	11
Operational Model with Oracle Solaris Zones	12
Zone Workloads	12
Using Guest Domains for Test and Development	12
Migrating Zones to the Production Environment	13
Operational Use Cases	13
General Domain Considerations	13
PCIe Root Complex Mapping	14
SPARC M7 Processor–Based Server Boot Options: NVMe PCIe Switch Cards, Disks, and Network Boot	14
SPARC T7-1 Server Root Complexes	15



SPARC T7-1 Server Domaining	16
Two Root Domains	17
Three Root Domains	17
Four Root Domains	17
SPARC T7-2 Server Root Complexes	17
SPARC T7-2 Server Domaining	18
Two Root Domains	18
Three Root Domains	18
Four Root Domains	18
More than Four Root Domains	18
SPARC T7-4 Server Root Complexes	19
SPARC T7-4 Server Domaining	20
Two Root Domains	20
More than Two Root Domains	20
Even More than Ten Root Domains	20
SPARC M7-8 and SPARC M7-16 Server Root Complexes	20
SPARC M7-8 and SPARC M7-16 CMIOU Board Root Complexes	21
SPARC M7-8 Server Two-PDom Root Complex	21
SPARC M7-8 Server Single-PDom Root Complex	22
SPARC M7-16 Server Example Root Complex	22
SPARC M7-8 and SPARC M7-16 Server Domaining	23
Two Root Domains	23
More than Two Root Domains	24



Populating PCIe Slots	24
How to Build Root Domains	24
Preparing the Control Domain	25
Building Root Domains	25
Building Root Domains with Virtual I/O for Boot and Networking	26



## Introduction

This paper describes how to implement an increasingly useful type of virtualization known as root domains. It also describes an operational model where root domains can be used in conjunction with Oracle Solaris Zones for maximum performance and a high degree of flexibility.

Root domains are a type of logical domain and are characterized by the fact that they own one or more PCIe root complexes and, therefore, do not share I/O components with other logical domains. System resources are merely divided among the domains (rather than being virtualized and shared). This type of virtualization is unique to the SPARC platform and offers a number of benefits, specifically bare-metal performance (that is, without virtualization overhead) and lower interdependence among domains.

Oracle Solaris Zones technology is a feature of Oracle Solaris, and this paper describes how to use them in a complementary, layered fashion with root domains. In this model, root domains form the lower layer, and zones form the second layer above the domains. The combination allows for highly flexible and effective virtualization, resource management, workload isolation and mobility.

The main objective of this paper is to introduce the root domain model as one of the architectures that the Oracle VM Server for SPARC product supports. It has a number of benefits, as well as a number of restrictions. The architectural decisions for how to best deploy Oracle VM Server for SPARC technology is driven by business needs and requirements, and might involve a combination or hybrid approach.


This document should always be read in conjunction with the Oracle VM Server for SPARC best practices document, the latest version of which can always be found here:

[oracle.com/technetwork/server-storage/vm/documentation/logical-domains-articles-160640.html](http://oracle.com/technetwork/server-storage/vm/documentation/logical-domains-articles-160640.html)

## Overview of Virtualization Technologies

### Type 1 Hypervisors

A hypervisor layer sits between the hardware and general-purpose operating system guests. It provides some level of platform virtualization and abstraction to these guests. It is generally said that the guests run on top of the hypervisor. This type of compute virtualization is primarily designed for servers.



The way the hypervisor is implemented has a measurable impact on performance. This is discussed in more detail in the next section.

Oracle VM Server for SPARC is a Type 1 hypervisor.

### Type 2 Hypervisors

The hypervisor runs within a conventional operating system. Guest operating systems then run within this hypervisor. This type of compute virtualization is primarily designed for desktop users and is outside the scope of this paper.

Oracle VM VirtualBox product is a Type 2 hypervisor.

### OS-Based Virtualization

In this type of virtualization, the general-purpose operating system itself provides a form of workload isolation within its own kernel. Only one operating system is actually running even though it might appear that multiple instances exist simultaneously. A hypervisor is not required to implement OS-based virtualization.

Oracle Solaris Zones technology is a mature, market-leading form of OS-based virtualization.

In this paper, we explain how to utilize Oracle Solaris Zones in conjunction with Oracle VM Server for SPARC in order to provide a high-performance and flexible virtualization environment.

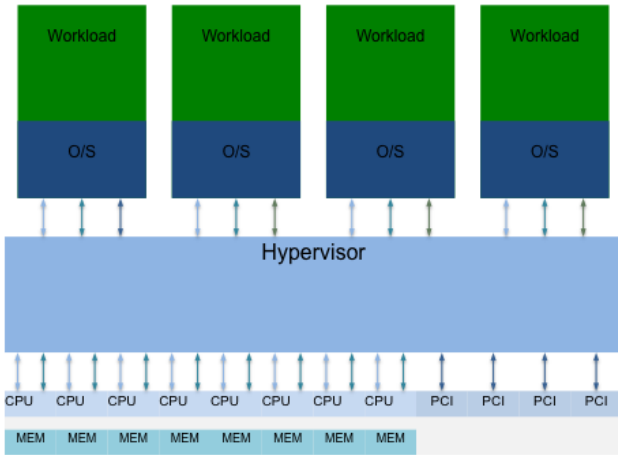
## Overview of Oracle VM Server for SPARC

A traditional hypervisor virtualizes all system resources (CPU, memory, and I/O devices). By design, the hypervisor abstracts the underlying hardware platform from the guest operating systems.

Most virtual machine (VM) systems run guests in a constrained user mode, so every time the guest OS wants to perform a privileged instruction it has to perform additional actions. The hypervisor then executes or emulates the privileged instruction in the context of that virtual machine alone. This is necessary because a VM guest OS cannot be allowed to perform certain “privileged” operations, such as changing memory mapping or masking hardware interrupts, directly on the real hardware. To permit virtual machines to do such operations would violate the separation of virtual machines from one another and compromise integrity.

This level of virtualization often incurs measurable performance overhead. Operating systems frequently perform privileged instructions of the types mentioned above, and in a virtualized environment, this can cause tens of thousands of context switches per second to emulate privileged operations, especially for I/O-intensive workloads. This penalty manifests itself in two ways.

1. A portion of the resources must be dedicated to running the hypervisor (a “virtualization tax”).
2. Overall throughput is lower and latency is higher due to the overhead inherent in this type of virtualization.



The hypervisor sits directly on bare metal, and owns all the hardware resources: CPU, memory, and I/O devices.

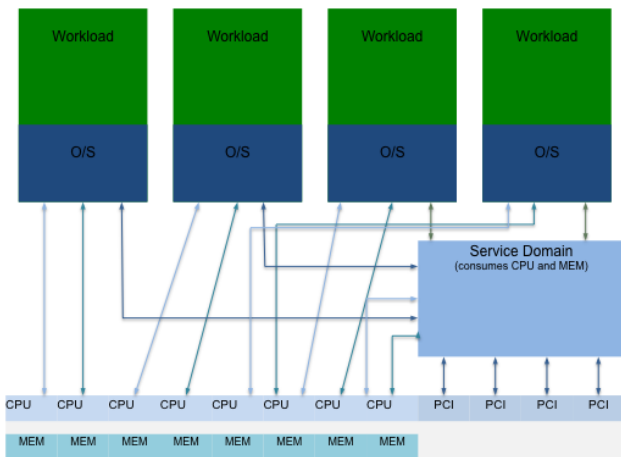
The guest domains running the workloads must involve the hypervisor for access to ALL resources.

Figure 1. Traditional hypervisor.

The architecture of Oracle VM Server for SPARC, on the other hand, is modular and consists of two main components:

- » A firmware-based hypervisor that allocates resources (CPU, memory and, optionally, I/O devices) directly to logical domains.
- » Software-based I/O virtualization, which runs in a service domain and provides virtual I/O services to the guest domains. When there is a single service domain, it is usually also the control domain. This is an optional component.

The hypervisor is a feature of the SPARC architecture. It is controlled from the control domain.



The single service/control domain owns all the I/O devices and is allocated CPU and memory for its own purposes.

The guest domains running the workloads have direct access to the CPU and memory allocated to them, and I/O services are provided as virtualized I/O services via the service/control domain.

Figure 2. Oracle VM for SPARC single Service Domain Model.

Unlike a traditional hypervisor, CPU threads and memory are allocated directly to the logical domains. The guest OS (Oracle Solaris) running within that logical domain runs on “bare metal” and can perform the privileged operations

directly, without the need to context-switch into the hypervisor. This key differentiator illustrates why this logical domain model does not carry a “virtualization tax” from a CPU and memory overhead perspective as the conventional model does.

Furthermore, in Oracle VM Server for SPARC, using virtualized I/O services is optional. It is possible to assign the I/O services directly to the logical domains themselves, creating what we call the root domain model.

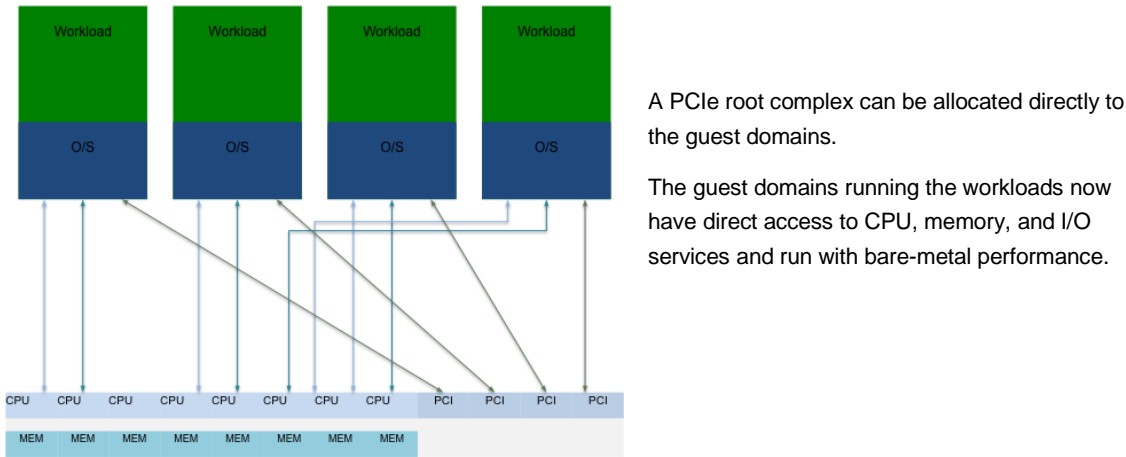


Figure 3. Oracle VM for SPARC Root Domain Model.

In this model, once these domains have been configured from the control domain, each domain has direct access to CPU, memory, and I/O devices. The hypervisor is not involved in the execution of instructions, in the allocation and access of memory, or in performing I/O operations. This further eliminates the “virtualization tax” from an I/O perspective.

In summary, Oracle VM Server for SPARC differs from traditional “thick” hypervisor designs in the following ways:

- » Memory and CPU resources are assigned directly to logical domains and are neither virtualized nor oversubscribed (and, therefore, do not suffer the traditional inefficiencies of having to pass through a virtualization layer).
- » Many I/O virtualization functions are offloaded to domains (known as service domains).
- » I/O resources can also be assigned directly to guest domains, thus avoiding the need to virtualize I/O services and incur the associated performance overhead.

This permits a simpler hypervisor design, which enhances reliability and security. It also reduces single points of failure by assigning responsibilities to multiple system components, which further improves reliability and security. In this architecture, management and I/O functionality are provided within domains.

Oracle VM Server for SPARC does this by defining the following domain roles:

- » **Control domain:** This domain is the management control point for virtualization of the server, which is used to configure domains and manage resources. It is always an I/O domain, and is usually a service domain as well. There is only one control domain per server or per SPARC M-Series physical domain.



- » **I/O domain:** This domain is assigned physical I/O devices: a PCIe root complex or an SR-IOV (Single-Root I/O Virtualization) function. It has native performance and functionality for the devices it owns, unmediated by any virtualization layer. There can be multiple I/O domains.
- » **Service domain:** This domain provides virtual network and disk devices to guest domains. There can be multiple service domains, although in practice there is usually one or sometimes two or more for redundancy. A service domain is always an I/O domain, because it must own physical I/O resources in order to virtualize them for guest domains<sup>1</sup>.
- » **Guest domain:** This is a domain whose devices are all virtual rather than physical: for example, virtual network and disk devices provided by one or more service domains. The intent of creating the guest domain is to host one or more applications.

Domain roles may be combined; for example, a control domain can also be a service domain.

### Traditional Deployment Model

The typical deployment pattern has focused primarily on using a single service domain, which owns all physical I/O devices and provides virtual devices to multiple guest domains where the applications are run. This single service domain is both an I/O domain and the control domain.

A variant of this model uses a second service domain to provide redundant access to disk and network services to the guest domains. This allows guests to survive an outage (planned or unplanned) of one of the service domains, and can be used for nondisruptive rolling upgrades of service domains.

This model offers many benefits: a high degree of flexibility, the ability to run an arbitrary number of guest domains (limited by CPU and memory resources), the ability to easily clone domains (when their virtual boot disks are stored on ZFS), and—when appropriately configured—the ability to live-migrate domains to another system.


These benefits come at the expense of having to allocate CPU and memory resources to service domains, thus reducing the amount of CPU and memory available to applications, plus incurring the performance overhead associated with virtualized I/O. This expense is justified by the operational flexibility of using service domains. This configuration is considered the “default” or baseline model due to its operational flexibility. The achieved near-native performance is usually acceptable and remains appropriate for many deployments. This model is well documented in numerous Oracle publications.

Hybrid models also exist where individual PCI root complexes or virtual functions on a PCIe card (via SR-IOV technology) can be assigned to a guest domain, and other guest devices are virtual devices provided by a service domain. These are beyond the scope of this paper other than to say that guidance for the creation of root domains is equally applicable to the provisioning of service domains.

## Introducing Root Domains

The guest domain model has been widely and successfully used, but more configuration options are available now. Servers have become larger and much more capable since the original Sun Fire T2000 class of machines with two I/O busses, so there is often much more I/O capacity that can be used for individual applications. Increased SPARC

<sup>1</sup> There are exceptions: A service domain with no physical I/O devices could be used to provide a virtual switch for internal networking purposes, or it could be configured to run the virtual console service.



server capacity, particularly single-thread performance, has made it attractive to run more vertical applications, such as databases, with higher resource requirements than the “light” applications originally deployed on earlier generations of SPARC T-Series servers from Oracle. This has made it more attractive to run applications directly in I/O domains so they can get native, nonvirtualized I/O performance.

This model is leveraged by the Oracle SuperCluster engineered system, first [introduced in late 2011 at Oracle OpenWorld](#). In Oracle SuperCluster systems, root domains are used for high-performance applications, with native I/O performance for disk and network and optimized access to the InfiniBand fabric.

### I/O Domain Models

Oracle's SPARC M7 processor-based servers (the SPARC T7-1, T7-2, T7-4, M7-8, and M7-16 servers) share the same basic design for the I/O subsystem. Each SPARC M7 processor in these servers is connected to one or two I/O controller ASICs via the interprocessor I/O links. There are two I/O link ports on the SPARC M7 processor and the I/O controller ASIC.

There are two different implementations of the connection between the SPARC M7 processor and the I/O controller ASIC, depending on the specific platform design. In Oracle's SPARC T7-1, M7-8, and M7-16 servers, each SPARC M7 processor is connected to a single I/O controller ASIC using both I/O links. Oracle's SPARC T7-2 and T7-4 servers utilize a cross-over connection scheme in order to provide connectivity to two I/O controller ASICs (and PCIe devices) in case one of the processors or I/O links is not available. In the cross-over connection, one I/O link from the processor is connected to one I/O controller ASIC, and the other I/O link is connected to another I/O controller ASIC.


The PCIe infrastructure is provided by the I/O controller ASIC. Each ASIC provides five PCIe 3.0 root complexes. Each PCIe root complex may be further subdivided from its native x16 or x8 width port into x4 or x8 ports, and on each platform, these ports service specific PCIe slots or internal devices. The mapping of I/O domains to root complexes is flexible and allows granular allocation of one or more individual PCIe devices to each I/O domain. Figures 6 – 11 on the following pages show the mapping of the PCIe root complexes to the I/O devices.

Using Oracle VM Server for SPARC, I/O devices can be assigned to an I/O domain using two technologies with different granularity:

- » Whole PCIe root complex. In this model, the I/O domain owns one or more PCIe root complexes and the associated PCIe slots or internal devices. This model is most commonly implemented for a service domain. It is also the model discussed in this paper for root domains, the key difference being that there are multiple root domains and no service domain and, therefore, no shared I/O resources<sup>2</sup>. The latest generation of SPARC M7 processor-based servers has significantly better PCIe root complex granularity. In most cases, each PCIe slot has its own dedicated root complex, which provides a large amount of granularity.
  
- » Single Root IO Virtualization (SR-IOV). In this model, a PCIe card owned by one root PCIe complex is virtually sliced to appear physically in one or more domains simultaneously. This capability is available for the Ethernet, Fibre Channel, and InfiniBand devices listed in the My Oracle Support note “Oracle VM Server for SPARC PCIe Direct I/O and SR-IOV Features” (Doc ID 1325454.1). The virtualization is performed by virtual functions in the PCIe card itself, not by the hypervisor or a service domain. Device bandwidth is shared across the virtual

---

<sup>2</sup> Certain resources, such as console interfaces or management network interfaces, can still be virtualized via a small service domain function. However, these are not on the data path for performance.



functions, which otherwise perform at native speed. SR-IOV allows physical cards to be shared, while the root complex model allocates whole cards to domains.

For the remainder of this paper we focus exclusively on a very specific implementation of the domain model we refer to as root domains.

## Root Domains

So far, we've discussed I/O domains primarily in the context of them also being service domains, where the sole purpose for owning I/O is to virtualize it for guest domains to provide abundant, redundant, or independent I/O for other guest domains.

It should be noted that the term *root domains* is also used in the context of Oracle SuperCluster to describe domains that are assigned PCIe root complexes, and used solely to provide SR-IOV services to guest domains. For the purposes of this paper, the wider definition of *root domain* is used, which is simply that they are assigned PCIe root complexes. The Oracle SuperCluster root domain definition should be considered a specific subset of this general definition.


The focus of this paper is the concept of an I/O domain hosting one or more applications directly, without relying on a service domain. Specifically, domain I/O boundaries are defined exactly by the scope of one or more root PCIe complexes. SR-IOV is not utilized in this model.

This offers a number of key advantages over SR-IOV and over all other hypervisors that use the traditional “thick” model of providing all services to guest VMs through software-based virtualization (at the expense of performance).

- » Performance: All I/O operations are native (that is, bare metal) with no virtualization overhead. Even when it is technically possible to achieve near-native performance with virtual I/O, there's a benefit to being able to factor out the potential effects of I/O virtualization during planning or troubleshooting.
- » Simplicity: The guest domain and associated guest operating system owns the entire PCIe root complex. There is no need to virtualize any I/O. Configuring this type of domain is significantly simpler than with the service domain model.
- » I/O fault isolation: A root domain does not share I/O with any other domain. Therefore, the failure of a PCIe card or internal PCIe device (for example, a NIC or HBA) impacts only that domain. This is in contrast to the service domain or SR-IOV models, where all domains that share those components are impacted by their failure. Note that Oracle VM Server for SPARC 3.2 with Oracle Solaris 11.2 SRU8 and later support I/O resiliency for SR-IOV devices, permitting continued operation and recovery for domains using SR-IOV virtual functions from a root domain that reboots. Still, root domains eliminate interdomain dependency risk altogether.
- » Generality: Root domains support any I/O device available for Oracle Solaris and the hardware platform, while SR-IOV supports only devices listed in My Oracle Support note “Oracle VM Server for SPARC PCIe Direct I/O and SR-IOV Features” (Doc ID 1325454.1).
- » Improved security: There are fewer shared components or management points.

The number of root domains is limited by the number of root PCIe complexes available in the platform. With the SPARC M7 processor and the I/O controller ASIC, there are up to five root PCIe complexes per I/O controller ASIC, which are subdivided depending on the specific architecture choices made by the platform architects. Natively, the I/O controller ASIC is equipped with five root PCIe complexes, four PCIe 3.0 x16 ports (which may be implemented natively as x16, four x4, or two x8 PCIe ports), and a further PCIe 3.0 x8 port (that may be implemented as x8 or two x4 PCIe ports). The implementation of the PCIe ports per platform is illustrated in figures 6 to 11 on the following pages.

The maximum practical number of root domains that can be created on any given platform will be less than the theoretical maximum, because more than one PCI root complex may be needed by a domain to satisfy its needs for devices to boot, network, and perform its other functions.



It is important to note that root domains are not the ideal solution for all workloads and use cases. In practice, a solution that comprises some domains configured as guest domains dependent on service domains in conjunction with some independent root domains might be appropriate. In fact, it is common that the same physical server comprises multiple root domains running applications and multiple service domains providing services to a number of fully virtualized guest domains. This is discussed in more detail in the “Operational Use Cases” section.

Root domains have a number of restrictions that should be carefully considered:

- » They are less flexible. With the guest domain model, the number of domains can be changed dynamically, usually without impacting running workloads. Changing I/O allocation of a root domain might require downtime, depending on whether the platform and firmware requirements for dynamic PCIe bus assignment have been met. These requirements are listed in the Oracle VM Server for SPARC administration guide. CPU and memory can be dynamically allocated.
- » The number of root domains possible is limited to some number less than the number of PCIe root complexes. Additional workload isolation can be performed using Oracle Solaris Zones within the root domains.
- » Root domains cannot be live-migrated. In general, domains with direct access to physical I/O cannot be live-migrated.
- » Root domains require more planning, particularly during the purchase phase to ensure there are enough NICs and HBAs, because these components would not be shared to other domains.
- » Multiple PCIe root complexes are required in most root domains to provide sufficient access to disks, networks, or required PCIe card slots.

### **Best Practices for Availability**

While root domains provide a level of fault isolation, they do not provide full isolation against all failure modes. Performance is the primary objective in creating root domains, with a secondary benefit being improved isolation. Application or service availability should always be architected at a higher level.

Best practices for high availability, such as clustering and remote replication for disaster recovery, should always be applied to the degree the business requirements warrant. For example, HA should not be implemented with both nodes of the cluster located within the same PDom or server. However, multiple tiers (for example, web, application, and database) can be placed in different domains and then replicated on other nodes using common clustering technology or horizontal redundancy.

The lack of live-migration capability with root domains should not be viewed as an impediment to availability. Live migration does not solve availability problems. A domain that has failed cannot be live-migrated. It's not alive anymore and so it contains no usable state to migrate. Many of the objectives seen as desirable features of live migration (for example, workload migration for service windows) can be obtained by architecting availability at a higher level so that taking one node down does not interrupt the actual service. Then cold workload migration can be implemented by booting domains from a SAN or NAS or by using technologies such as Oracle Solaris Cluster, Oracle WebLogic Server, or Oracle Real Application Clusters (Oracle RAC) that provide high availability. It should be noted that live migration does enhance the serviceability aspects of a system, by allowing workloads to be migrated away from a physical server for maintenance purposes. The availability architectures noted above also assist in the serviceability aspects, in the absence of the live-migration capability of root domains.

In the proposed operational model, Oracle Solaris Zones become the migration entities (boundaries). Zone shutdown and restart time is significantly faster than for the main Oracle Solaris instance. This further mitigates the requirement for live migration within this scenario.

Oracle has published a number of papers that discuss these concepts further and with respect to individual workloads. Refer to the Maximum Availability Architecture and Optimized Solutions sections of the Oracle Technology Network.

## Operational Model with Oracle Solaris Zones

Although the maximum number of root domains is constrained by the I/O architecture of each of the platforms, this does not necessarily imply a limit on the number of workloads possible.

Oracle Solaris Zones provide a very granular, convenient, mobile, and manageable workload containment mechanism. The addition of Kernel Zones in Oracle Solaris 11.2 and above allows even greater isolation by allowing each zone to run its own unique OS kernel and associated update levels.

### Zone Workloads

Using Oracle Solaris Zones as workload containers inside root domains provides a number of benefits:

- » Workload isolation. Applications can each have their own virtual Oracle Solaris environment.
- » Administrative isolation. Each zone can have a different administrator.
- » Security isolation. Each zone has its own security context, and compromising one zone does not imply that other zones are also compromised.
- » Resource control. Oracle Solaris has very robust resource management capabilities that fit well with zones. This includes CPU capping (including for Oracle license boundaries), Fair Share Scheduler, memory capping, and network bandwidth monitoring and management.
- » Workload mobility. Zones can be easily copied, cloned and moved, usually without the need to modify the hosted application. More importantly, zone startup and shutdown is significantly quicker than traditional VM migrations.

We refer to this as “zone workloads” (ZW).

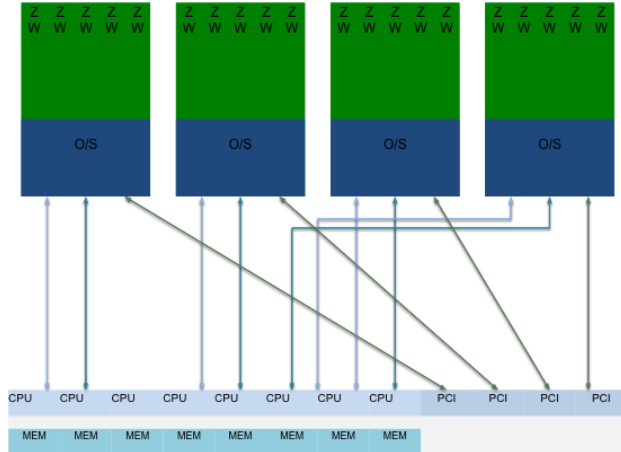



Figure 4. Oracle VM Server for SPARC with zone workloads.

The use of zone workloads on top of root domains is considered best practice, even if the intent is to run a single workload. There is no performance or financial cost to running zones in the domain, and it provides operational flexibility by providing the ability to dynamically resize or migrate to another platform. However, applications can be installed directly within the root domains if so desired.

### Using Guest Domains for Test and Development

For development systems, it is often more desirable to implement domains using the service domain model. This affords maximum flexibility when performance is generally not a concern. For maximum flexibility, place the



application or workload within a zone inside the domain, because this becomes the entity that is promoted through the development and test lifecycle.

Functional test can be implemented the same way. Zones can be copied, cloned, and/or moved from development to test. Zones can further be cloned multiple times to test them in parallel if desired.

### **Migrating Zones to the Production Environment**

When maximum performance is desired, production and pre-production can be implemented with root domains. Ideally, pre-production will be on hardware identical to the production system and with an identical root domain configuration. Workloads can be migrated from development or functional test guest domains to the root domains that exist in the production and pre-production environments.

## **Operational Use Cases**

When considering the use of root domains, the main limiting factor is the number of PCIe root complexes that are available in the physical server, and the devices that would be available to the domains once those PCIe root complexes are allocated. Each SPARC M7 processor socket is paired with an I/O controller ASIC that provides five PCIe root complexes; these root complexes may be unused, wired to an internal device (such as a network controller), or mapped to a PCIe slot. Because each root complex may be split into up to four smaller PCI busses, a mix of internal, external, and unused connections might be present on each PCIe root complex in each platform design. In order to clarify the options and availability of connections, the root complex layouts for each platform have been provided on the following pages.

One domain in a platform will always be designated the control domain, and in the case of a hybrid model, the control domain could also function as a service domain for the virtualized guest domains, if required. The control domain will need to retain a reasonable minimum set of root complexes and associated devices to allow it to function and administer the other domains. The minimum control domain root complex allocation will be highlighted for each of the platforms, with a discussion of which onboard/internal I/O devices may be used for additional root domains.

Each root complex owns a number of platform-specific combinations of I/O slots and/or onboard I/O devices. This is hardwired, and has a bearing on physical card placement and which domains are capable of accessing onboard devices when they own specific root complexes.

### **General Domain Considerations**

In most cases, it is expected that each root domain should be capable of booting independently and being able to communicate over the network natively. In other words, each root domain needs its own boot device and network ports.

A server might have a number of onboard disk controllers and network ports that could be shared among root domains, but it might also be necessary to provision additional network cards or HBAs in the allocated PCIe slots to a root domain.

In a hybrid scenario, it is also possible to boot the domain over virtual disk devices, while providing dedicated PCIe slots for native network and disk traffic. In this case, it is normal practice to provide at least two fully independent root domains that can be configured to provide redundant disk and network services to the guest or hybrid domains.

In the “PCIe Root Complex Mapping” section below, the two-domain configuration should also be considered useful guidance for the redundant service domains for both SR-IOV and virtualized guest domain services.

As can be seen from the diagrams, the minimum root complex allocations required to independently boot, and allow the network access to the control domain for each platform, have been identified. This allows the remaining root complexes to be de-assigned from the control domain and assigned to the new root domains being created.

PCIe cards might need to be installed in the other root domains to provide access to external disks for booting and for networking.

Given the use cases for these domains, it is expected that they will generally be provisioned with HBAs and additional network cards, because high-performance I/O is the main reason for creating these domains in the first place.

This discussion focuses solely on PCIe root complex allocation. CPU and memory can be freely and dynamically allocated to any domain, according to the usual Oracle VM Server for SPARC rules<sup>3</sup>.

### PCIe Root Complex Mapping

The mapping of the PCIe root complexes to I/O devices for the relevant SPARC M7 processor-based platforms is shown below. This will be used to define the best practices for creating the root domains for each server.

In all the diagrams below, the architecture has been simplified to show only components relevant to the root complexes. Color coding is done according to the information shown in Figure 5, indicating user accessible, internal, or optional devices or card slot locations.

PCIe SLOTx x16	- User accessible PCIe Slot number 'x', using a x16 width PCIe bus
SASy (disks a-b) x8	- Internal SAS disk controller number 'y' responsible for disks slots 'a' to 'b', using a x8 PCIe bus
NVMez (disks c-d) x8	- Location of an optional NVMe disk switch controller number 'z', enabling disk slots 'c' to 'd' using a x8 PCIe bus
NETn/NETm 10Gbps x4	- Internal 10GBase-T network controller, with two ports NET'm' and NET'n', using a x4 PCIe bus


Figure 5. PCIe root complex color coding.

Designations such as "pci\_0" indicate the device enumeration and identity of the root complex for administration purposes. The device-to-PCIe root complex mapping is hardwired and cannot be changed.

### SPARC M7 Processor-Based Server Boot Options: NVMe PCIe Switch Cards, Disks, and Network Boot

SPARC M7 processor-based servers have a number of boot options. The domains can boot from either SAS or NVMe internal disks, they can boot from external disks, and—with suitable cards in the PCIe slots—they can boot off Oracle Flash Accelerator F160 PCIe Cards. Finally, they can be configured to network-boot using either InfiniBand (IB) or Ethernet.

<sup>3</sup> It is recommended that CPUs be allocated in whole-core increments. For optimal performance, they should be on socket boundaries rather than being spanning sockets.



Any domain with access to a 10 GbE network card can be configured to iSCSI-boot from an external NAS device, and any domain with access to an IB card can be configured to boot over IB from an IB-connected Oracle ZFS Storage Appliance. In the IB case, the domain must also have access to the eUSB device, because this is required for the initial boot sequence.

When NVMe PCIe switch cards are installed, the disk bays in SPARC T7 servers become dual-purpose. If an NVMe disk is placed in the bay, it will connect to the NVMe switch. If a SAS disk is placed in the bay, it will connect to the SAS controller. This feature allows multiple independent boot media to be provided internally within each platform using a mixture of SAS and NVMe disks on different root complexes. The NVMe switch cards are optional additions to the platform.

The SPARC T7-1 server has eight disk bays, all eight of which are SAS-capable, and four of which are NVMe-capable. There is a single SAS controller, and a single NVMe switch. This allows two root domains to be bootable via the disk bays.

The SPARC T7-2 server has six disk bays. The first four disks are on the first SAS controller, and the remaining two are on the second. The NVMe-capable bays are the last four, which are either assigned to a single NVMe switch or split between two NVMe switches. This means that four root domains could be booted via the disk bays, but two of them would have only a single disk, and this would not be a recommended practice. Three root domains with mirrored disks could be supported with two SAS pairs and one NVMe pair using a single NVMe switch, or one SAS pair and two NVMe pairs using two NVMe switches.

The SPARC T7-4 server has eight disk bays, all of which are SAS- and NVMe-capable. With two NVMe switches, up to four root domains can be booted from the internal disk bays: two SAS pairs and two NVMe pairs.

In all of the examples below for the SPARC T7 servers, the control domain will be assigned one of the SAS controllers to boot locally off internal disks. The examples also show how to make maximum use of the internal drives for additional domains. This is expected to be the most common practice. However, the network boot options described above could also be applied to any of the domains, including the control domain. In effect, a root domain could consist of a single PCIe slot with a network card installed, used for both boot and network access.

### SPARC T7-1 Server Root Complexes

The five root complexes are defined by the top-level PCI path, as shown in Figure 6 and described in the list that follows.



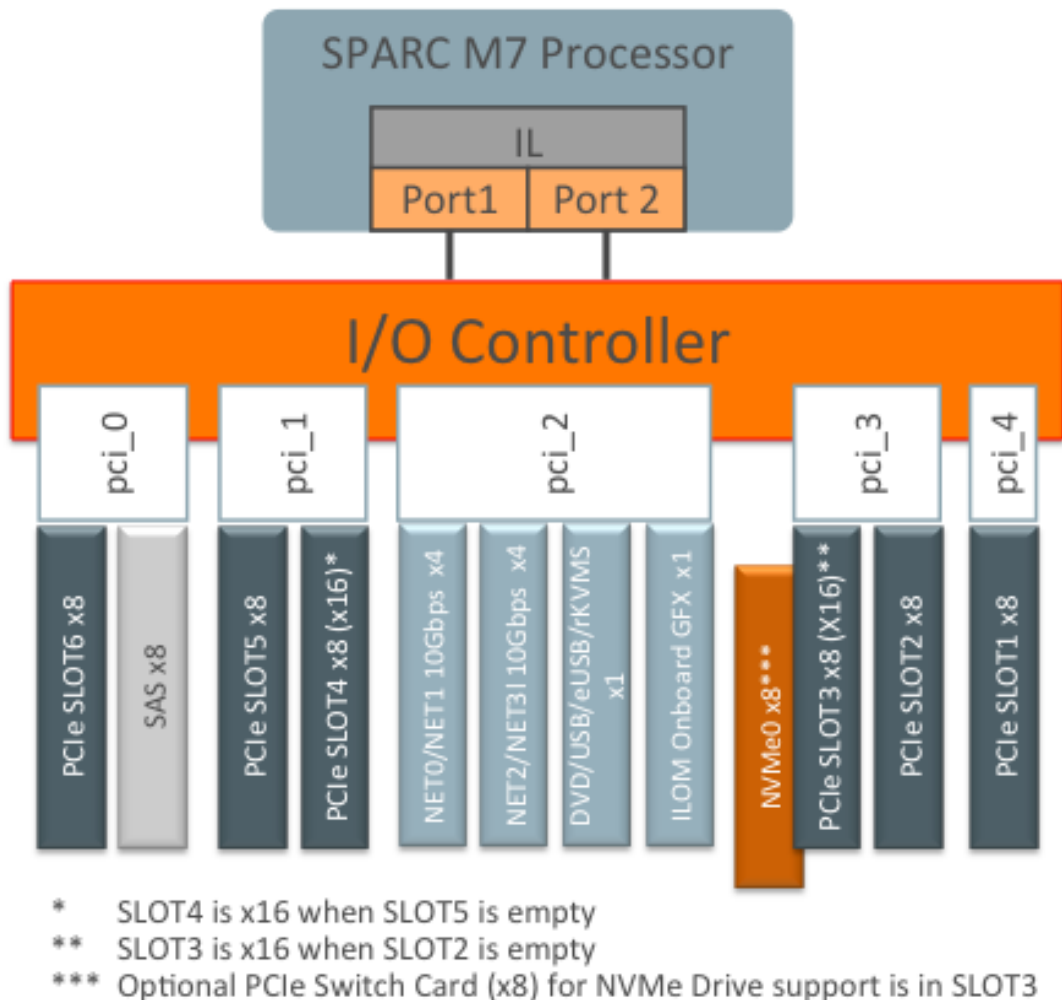


Figure 6. SPARC T7-1 server root complexes.

The minimum typical root complex allocation for the control domain is highlighted in bold in the list below.

- » **RC0: pci\_0 (pci@301), owns PCIe slot 6 and the onboard SAS disk controller**
- » RC1: pci\_1 (pci@302), owns PCIe slots 4 and 5
- » **RC2: pci\_2 (pci@300), owns the four onboard 10 GbE ports, the onboard graphics device, and all onboard USB devices including the DVD drive**
- » RC3: pci\_3 (pci@303), owns PCIe slots 2 and 3 (and the optional NVMe PCIe switch card, if it is installed)
- » RC4: pci\_4 (pci@304), owns PCIe slot 1

**SPARC T7-1 Server Domaining**

This platform has a single SAS controller for all internal disks and all onboard 10GBASE-T ports attached to a single root complex. Therefore, any additional root domains will need to make use of cards populated in the assigned PCIe slots.

The optional NVMe PCIe switch card in slot 3 allows the use of internal NVMe disks, if required.

The control domain requires RC0 and RC2, leaving the remaining RC1, RC3, and RC4 available for the creation of additional root domains.

### Two Root Domains

RC3 provides an ideal root complex to provide services for a second root domain. It provides access to PCIe slots 2 and 3. Slot 3 may be populated with a NVMe PCIe switch card—to make use of internal NVMe disks—or a flash storage card, or a Fibre Channel HBA for booting. Slot 2 should be populated with a networking card.

The remaining root complexes: RC1 (with slots 4 and 5) and RC4 (with slot 1) can be assigned to either domain. Optionally, RC4 could be assigned to this root domain, and the networking card placed there, allowing slot 2 to remain empty.

### Three Root Domains

If a third domain is required, RC1 can be assigned to it and populated as per the second domain (using slots 4 and 5), with the provision that the NVMe PCIe switch card option is not available.

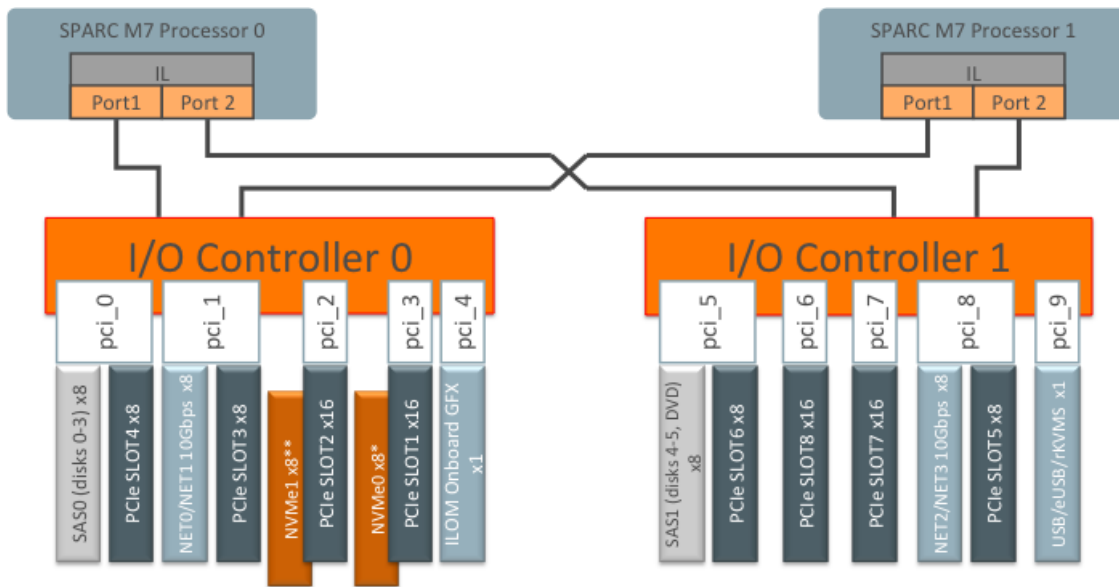
As before, the remaining RC4 can be assigned to any of the three domains.

### Four Root Domains

The remaining RC4 root complex can be assigned to this fourth domain. Because there is only a single PCIe slot available, this domain would need to either rely on virtual disk or virtual network services. Depending on the use case, slot 1 could be populated with a Fibre Channel HBA providing both boot and disk I/O services to the domain, with virtual network access, or with a network adapter. Boot capability could be provided either by network booting or via virtual disk services.

### SPARC T7-2 Server Root Complexes

A SPARC T7-2 server has 2 CPUs and 10 PCI root complexes. The 10 root complexes are provided by the two I/O controller ASICs, as shown in Figure 7 and described in the list that follows.



- \* First optional PCIe Switch Card (x8) for NVMe Drive support goes into SLOT1
- \*\* Second optional PCIe Switch Card (x8) for NVMe Drive support goes into SLOT2

Figure 7. SPARC T7-2 server root complexes.

The minimum typical root complex allocation for the control domain is highlighted in bold in the list below.

- » **RC0: pci\_0 (pci@301), owns the first onboard SAS disk controller for disks 0–3 and PCIe slot 4**
- » **RC1: pci\_1 (pci@300), owns network ports 0 and 1 and PCIe slot 3**
- » RC2: pci\_2 (pci@307), owns PCIe slot 2 (and the optional NVMe PCIe switch card if it is installed in this slot)
- » RC3: pci\_3 (pci@306), owns PCIe slot 12 (and the optional NVMe PCIe switch card if it is installed in this slot)
- » **RC4: pci\_4 (pci@309), owns the SP onboard graphics device**
- » RC5: pci\_5 (pci@303), owns the second onboard SAS disk controller for disks 4,5, and the DVD drive, and PCIe slot 6
- » RC6: pci\_6 (pci@305), owns PCIe slot 8
- » RC7: pci\_7 (pci@304), owns PCIe slot 7
- » RC8: pci\_8 (pci@302), owns network ports 2 and 3 and PCIe slot 5
- » **RC9: pci\_9 (pci@308), owns all onboard USB devices**

### SPARC T7-2 Server Domaining

The possible configurations of root domains are many, but to retain conventional boot and network capabilities in the control domain, it must retain at least RC0 (local disks 0–3), and RC1 (network ports 0 and 1) as well as the connections to the Oracle Integrated Lights Out Manager [Oracle ILOM] USB and graphics devices from RC4 and RC9.

Unlike the SPARC T7-1 server, the SPARC T7-2 server has the disk devices and onboard network ports split between root domains, so it is easy to create a second domain without additional components. Additional root domains will require the provision of additional network cards and boot devices.

NVMe PCIe switch cards can be installed in PCI slots 1 and 2 (controlled by RC3 and RC2, respectively).

The control domain requires RC0, RC1, RC4, and RC9, leaving the remaining RC2, RC3, RC5, RC6, RC7, and RC8 available for the creation of additional root domains.

#### Two Root Domains

RC5 and RC8 provide access to onboard disks and network ports for a second root domain. They also provide access to PCIe slots 5, and 6, which can be populated with additional cards as required.

The remaining root complexes—RC2, RC3, RC6, and RC7—each with access to one PCIe slot each, can be assigned to either domain.

#### Three Root Domains

If a third domain is required, RC2 and RC6 can be assigned to it, with a network card placed in slot 8 (RC6), and either an NVMe PCIe switch card, flash storage card, or Fibre Channel HBA in slot 2 (RC2).

As before, the remaining unused root complexes can be assigned to any of the three domains.

#### Four Root Domains

For a fourth domain, assign RC3 and RC7 and populate as per the third domain.

At this stage, all the root complexes are now fully consumed by the four root domains.

#### More than Four Root Domains

More than four domains require a configuration in which only a single PCIe slot can be assigned to a root domain. This domain would need to either rely on a virtual disk or the network for booting, or rely on virtual network services.

Depending on the use case, the slot could be populated with a Fibre Channel HBA providing both boot and disk I/O services to the domain, with virtual network access. Alternatively, it could be populated with a network adapter, and boot services could be provided via network or virtual disk services.

The third or fourth domains described above could each be converted to two domains of this type.

### SPARC T7-4 Server Root Complexes

The SPARC T7-4 server can have two or four processors. In either case, the server has 20 root complexes, as shown in Figure 8 and described in the list that follows.

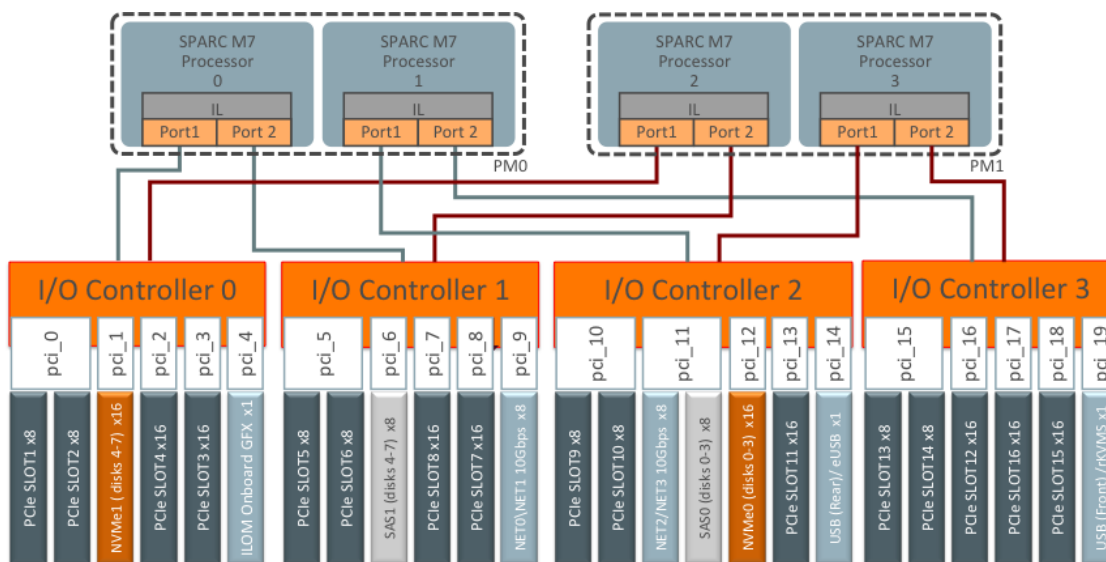


Figure 8. SPARC T7-4 server root complexes.

The minimum typical root complex allocation for the control domain is highlighted in bold in the list below.

- » RC0: pci\_0 (pci@305) owns PCIe slot 1 and PCIe slot 2
- » RC1: pci\_1 (pci@304) owns the second internal card slot that is dedicated for an optional NVMe PCIe switch card supporting NVMe disks 4–7
- » RC2: pci\_2 (pci@307) owns PCIe slot 4
- » RC3: pci\_3 (pci@306) owns PCIe slot 3
- » **RC4: pci\_4 (pci@313) owns the SP onboard graphics device**
- » RC5: pci\_5 (pci@308) owns PCIe slots 5 and PCIe slot 6
- » **RC6: pci\_6 (pci@301) owns the second onboard SAS disk controller for disks 4–7**
- » RC7: pci\_7 (pci@30a) owns PCIe slot 8
- » RC8: pci\_8 (pci@309) owns PCIe slot 7
- » **RC9: pci\_9 (pci@301) owns onboard 10GBASE-T network ports 0 and 1**
- » RC10: pci\_10 (pci@30b) owns PCIe slot 9 and PCIe slot 10
- » RC11: pci\_11 (pci@300) owns onboard 10GBASE-T ports 2 and 3 and the first onboard SAS controller for disks 0–3
- » RC12: pci\_12 (pci@303) owns the first internal card slot that is dedicated for an optional NVMe PCIe switch card supporting NVMe disks 0–3

- » RC13: pci\_13 (pci@30c) owns PCIe slot 11
- » **RC14: pci\_14 (pci@312) owns the rear USB and eUSB devices**
- » RC15: pci\_15 (pci@30e) owns PCIe slot 13 and PCIe slot 14
- » RC16: pci\_16 (pci@30d) owns PCIe slot 12
- » RC17: pci\_17 (pci@310) owns PCIe slot 6
- » RC18: pci\_18 (pci@30f) owns PCIe slot 15
- » **RC19: pci\_19 (pci@311) owns the front USB devices**

## SPARC T7-4 Server Domaining

The SPARC T7-4 server is similar to the SPARC T7-2 server configuration in the sense that the internal disks and internal network ports are also split between root complexes. It is slightly complicated by the fact that the first four boot disks and the second two network ports are on the same root complex, and some minor reconfiguration of the default device allocation in the control domain needs to be done to “free up” the root complexes to allow the second root domain to share the devices. The examples show this configuration.

In all other respects, the SPARC T7-4 server is similar to the SPARC T7-2 server, except that there are more root complexes and PCIe slots, allowing up to 10 root domains to be easily accommodated.

The control domain requires RC4, RC6, RC9, RC14, and RC19, leaving the remaining RC0, RC1, RC2, RC3, RC5, RC7, RC8, RC10, RC11, RC12, RC13, RC15, RC16, RC17, and RC18 available for the creation of additional root domains.

RC1 and RC12 provide access to the optional internal NVMe disk drives.

RC11 provides both internal network ports and access to internal SAS disks.

RC0, RC5, RC10, and RC15 each provide a pair of PCIe slots.

RC2, RC3, RC7, RC8, RC13, RC16, and RC17 each provide a single PCIe slot.

### Two Root Domains

RC11 can be assigned to this domain to provide boot and network devices using the onboard devices. If only two root domains are required, the remaining root devices can be distributed among these two domains as required.

### More than Two Root Domains

Using similar root complex assignment techniques as shown in the SPARC T7-2 section, additional root domains can be provisioned as required, using one of the slots for a networking card, and either another slot with a flash storage card or a Fibre Channel HBA, or assigning one of the internal root complexes with a NVMe PCIe switch card. Up to 10 root domains may be created in this fashion.

### Even More than Ten Root Domains

If more than 10 root domains are required, it will be necessary to adopt a similar approach to the SPARC T7-2 case with more than 4 domains, where the root domains are provided with a single PCIe slot.

## SPARC M7-8 and SPARC M7-16 Server Root Complexes

The SPARC M7 processor-based servers use a modular design, which simplifies the discussion regarding root complexes. Each SPARC M7 processor-based server is based on common building blocks, and for the purposes of the root complex discussion, the core building blocks are the CPU Memory and I/O Units (CMIOUs), which are connected together in a system containing at least two and up to sixteen CMIOUs in a physical domain (PDom).

The CMIU building block contains a single processor and IO controller ASIC. Of the five root complexes provided by the I/O controller ASIC, one is unused, three service PCIe hot-pluggable carriers, and one is dedicated to the onboard eUSB device and the Service Processor Module (SPM) connections.

### SPARC M7-8 and SPARC M7-16 CMIU Board Root Complexes

In a CMIU, pci\_0, pci\_1, and pci\_3 are brought out to hot-pluggable PCIe card carrier slots mounted at the rear of the CMIU, and pci\_2 is not used. If the CMIU is populated in the first or second CMIU slot in the PDom of DCU, both portions of pci\_4 are used for the SPM connection and the eUSB device; in other slots, only the eUSB portion is connected. There is a single eUSB device per CMIU, which allows the domain assigned to that device to network boot over InfiniBand, using the eUSB as the boot archive source.

In summary, each CMIU provides three PCIe x16 slots on different root complexes. It should also be noted that there are no onboard disk or network controllers on this platform, so boot and network devices must be provided via these PCIe slots for each root domain configured.

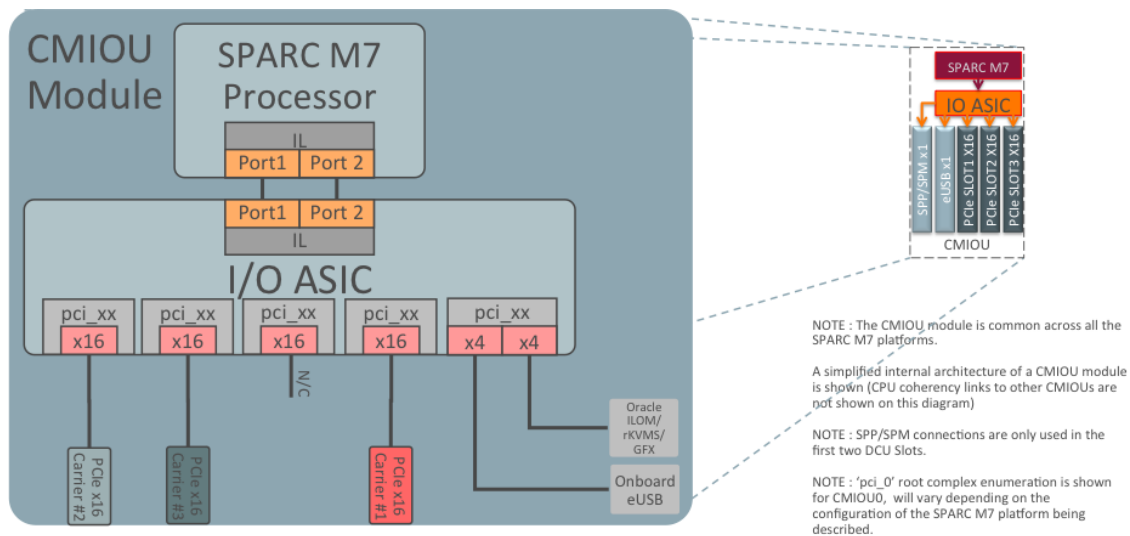


Figure 9. SPARC M7-8 and M7-16 CMIU board root complexes.

### SPARC M7-8 Server Two-PDom Root Complex

This SPARC M7-8 configuration is permanently arranged as a two PDom each consisting of four CMIU slots. At a minimum, two CMIUs must be installed in one PDom. The maximum configuration includes four CMIUs in both PDom.

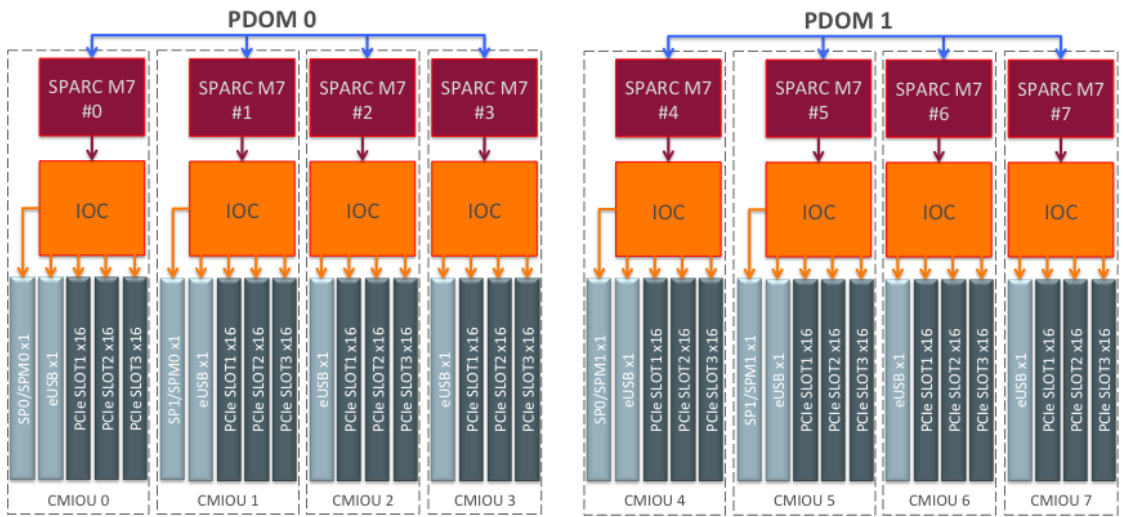


Figure 10. SPARC M7-8 server two-PDom root complex diagram.

### SPARC M7-8 Server Single-PDom Root Complex

This SPARC M7-8 server configuration is a single-PDom platform with at least two CMIOUTs and up to eight.

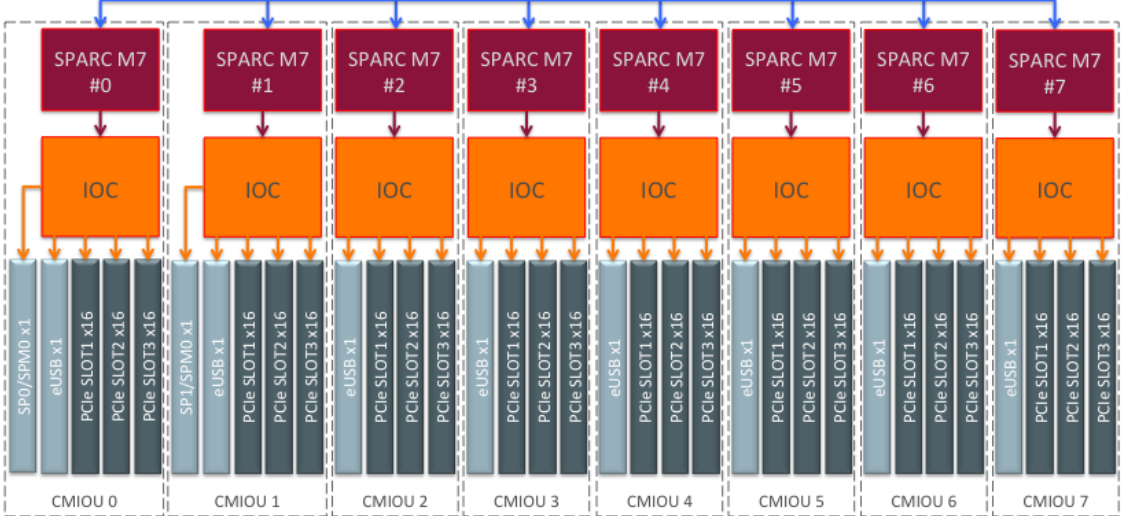


Figure 11. SPARC M7-8 server single-PDom root complex.

### SPARC M7-16 Server Example Root Complex

The SPARC M7-16 server is configurable into up to one, two, three, or four PDoms containing between 2 and 16 CMIOUTs. The PDoms are defined in the increments of domain configurable units (DCU). A DCU is a group of four adjacent CMIOUT slots: slots 0–3 and 4–7 in the CMIOUT chassis, respectively. Figure 12 illustrates a three-PDom configuration: two PDoms with four CMIOUTs each and a third PDom of eight CMIOUTs.

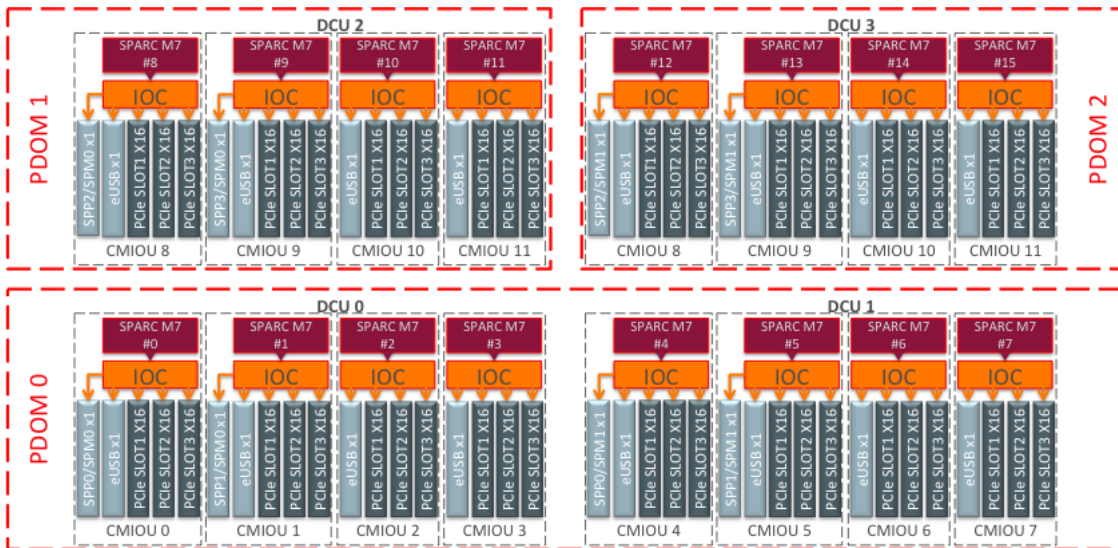


Figure 12. SPARC M7-16 server example root complex.

## SPARC M7-8 and SPARC M7-16 Server Domaining

As per figures 10 and 11 above, the M7-8 is factory-configured with either one or two PDoms; however, the SPARC M7-16 server is customer-configurable from one to four PDoms.

The root domaining layout of the SPARC M7-8 and SPARC M7-16 servers is simplified by the fact that their building blocks are composed of PDoms, and each PDom has a minimum of two CMIOUs, and each CMIOU provides three PCIe slots. The number of root domains possible is, therefore, limited simply by the number of CMIOUs present in the PDom and by the number of PCIe slots you wish to assign to each root domain.

Because the SPARC M7 processor-based servers don't have onboard disk or network cards, the PCIe slots allocated to a domain must always provide the boot and external network connectivity.

The minimum number of PCIe slots that can be assigned to a root domain is one. This configuration assumes that a network card is in this slot, and that the system has been configured to network-boot, or that it can access virtual disk services from existing root domains.

One root domain per CMIOU can make use of the eUSB device to allow network booting over InfiniBand.

The minimum control domain owns all the root complexes, except for two out of the three PCIe slots, using eUSB for the boot archive and a network card (IB or Ethernet) in the single assigned PCIe slot for network access and booting. Additional root complexes can be assigned to this domain as required, or if an alternative boot method is desired. The control domain will always get access to the eUSB devices on the first two CMIOUs.

The remaining root complexes can be assigned to any additional root domains.

### Two Root Domains

Given that the minimum number of CMIOUs per PDom is two, it will always be possible to create a second root domain similar to the control domain, but without access to the eUSB on the second CMIOU, and with one PCIe slot with an Ethernet card for network access and booting.

It is also equally possible to assign more root complexes to this domain as well.



## More than Two Root Domains

Additional root domains of the form defined above can be created per CMIOU, with a single PCIe slot. If InfiniBand network booting is required, the domain will also require access to the eUSB device.

Single root complex root domains can be created and rely solely on virtual disk services from the first two root domains, or they can network-boot over Ethernet. Alternatively, root domains can be created with more than one PCIe slot to accommodate traditional booting.

Theoretically, the SPARC M7-8 and SPARC M7-16 servers can support three root domains per populated socket. In practice, a root domain might require that more than one PCIe slot be assigned to it, and this ultimately limits the practical number of root domains.

## Populating PCIe Slots

The quantity and type of PCIe cards that will be used will vary depending on the requirements of the solution. However, there are some general rules that should be followed, if possible, to ensure that I/O operations are spread among the internal PCI switches to achieve both performance and redundancy.

Given the use cases that demand this type of configuration, it is expected that sufficient PCIe root complexes will be owned and populated with at least one HBA and one NIC card. This allows direct access from the domain to SAN-based disks and networking.

Figures 6 to 12 can be used to ensure that optimal card placements are made.

## How to Build Root Domains

Creating root domains is conceptually no different than creating standard I/O domains, except that you simply don't create virtual services in the I/O domains. Instead, we run them as pure bare-metal OS instances. When creating these domains, the initial control domain must be configured to use only the root complex that it will be left with after all the other domains are created. The remaining root complexes can then be allocated to the new root domains.

The new root domains will then need to have Oracle Solaris installed using its directly attached disk and networks or via virtual disks and networks provided by the control domain.

In the example below, it is assumed that the PCIe slots have been populated with HBA and NIC devices so that the newly created domains have access to boot disks and have sufficient network ports for connectivity to the data center infrastructure, for example, the production, backup, and management networks<sup>4</sup>.

It is, of course, possible to provide virtual devices to the domains using the usual methods, so that access to disks and networks without high-performance requirements can still be provided, but this removes some of the isolation and simplicity benefits that this model provides.

Full details are provided in the Oracle VM Server for SPARC documentation, which can be found at [www.oracle.com/technetwork/documentation/vm-sparc-194287.html](http://www.oracle.com/technetwork/documentation/vm-sparc-194287.html).

---

<sup>4</sup> Console access to the domains is provided via the control domain, which should be connected to the management network. It might also be worth attaching the root domains to the management network to allow `ssh` access and for OS installation and updates.

## Preparing the Control Domain

The first domain on a SPARC server is always the control domain. In the context of the root domain model, its main job is to host the LDom<sup>5</sup> manager software. Under the root domain model, the control domain does not typically provide any virtual services other than the virtual console service. For this reason, it can usually be treated as a domain that can be used to run applications, providing the security concerns for access to the `ldm` commands have been satisfied. The easiest way to accomplish this is to run applications within zones on the control domain.

The following instructions work on both Oracle Solaris 10 and Oracle Solaris 11 control domains; however, Oracle Solaris 10 control domains are not supported on SPARC M7 or SPARC T7 servers, and they are not available in Oracle VM Server for SPARC releases after 3.2. Oracle Solaris 11 is preferred on previous releases, too. Ensure that you are running the latest system firmware and the latest updates/patches for the version of Oracle Solaris you are running. If you are still using an Oracle Solaris 10 control domain, you will need to install the Oracle VM Server for SPARC software, which is pre-installed on Oracle Solaris 11.

The control domain will ALWAYS be left with some root complexes. For instance, a SPARC T7-1 server will retain RC0 and RC2. Remove the root complexes that will be used for the other root domains as part of the initial configuration. In the case of a SPARC T7-1 server, we will remove RC1, RC3, and RC4.

The following steps are examples. Values used for CPU, memory allocation, and root complexes might need to be changed to reflect the actual configuration.

Here are the steps to perform the initial configuration of the control domain:

```
# ldm add-vcc port-range=5000-5100 primary-vcc0 primary
# svcadm enable vntsd
# ldm set-core 8 primary
# ldm start-reconf primary
# ldm set-memory 256G primary
# ldm remove-io pci_1 primary
```

Repeat the commands above for `pci_3` and `pci_4`, as required. Then run the following command:

```
# ldm add-spconfig initial; reboot;
```

## Building Root Domains

Each root domain can now be created, as required. The standard domain creation steps are used, except you add the root complex to each domain, as required. The following example assumes that the root complex associated with the root domain contains disks suitable for installing Oracle Solaris and for network access. In this case, RC3 will be used as the second root domain on a SPARC T7-1 server.

Here are the steps to create a root domain:

```
# ldm create ldom1
# ldm set-vcpu 64 ldom1
```

---

<sup>5</sup> "LDoms" stands for "Logical Domains" and was the product name superseded by Oracle VM Server for SPARC. However, the naming convention of the commands and processes has not changed.



```
# ldm set-memory 256G ldom1
```

```
# ldm add-io pci_3 ldom1
```

Repeat the steps above, as necessary, for all other domains.

Save LDom configuration.

```
# ldm add-spconfig domains
```

(Yes, it really is that simple.)

### Building Root Domains with Virtual I/O for Boot and Networking

It is possible to configure the root domains so that they can use virtual disks and networks provided by other root domains with direct access to internal disks and networks.

In this case, it is simply a matter of configuring virtual disk and network services and configuring in the LDom configuration shown in the previous section.







**Oracle Corporation, World Headquarters**

500 Oracle Parkway  
Redwood Shores, CA 94065, USA

**Worldwide Inquiries**

Phone: +1.650.506.7000  
Fax: +1.650.506.7200

CONNECT WITH US

-  [blogs.oracle.com/oracle](http://blogs.oracle.com/oracle)
-  [facebook.com/oracle](http://facebook.com/oracle)
-  [twitter.com/oracle](http://twitter.com/oracle)
-  [oracle.com](http://oracle.com)

**Integrated Cloud Applications & Platform Services**

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0615

Implementing Root Domains with Oracle VM Server for SPARC  
November 2015  
Author: Peter Wilson, Mikel Manitiu, Michael Ramchand, Jeff Savit