5 Node definitions

The type of a node is defined by the name string associated with the NODE element designating the start of the node in the machine description node block. Nodes can be found by linear search matching on type or by following the PROP_ARCs of a DAG.

5.1 Node categories

Nodes in a machine description serve one or two purposes; to provide information about a virtual machine resource they represent and, optionally to function as a construction node within a DAG formed within the machine description. A construction node may contain properties about certain resources, however its primary function is as a container for the arc links (PROP_ARC properties) that connect to other descriptive nodes.

Nodes belong to one of four categories that determine what walkers must handle within the MD. A node's category determines whether nodes of that type can be expected to found within the MD, or whether nodes of that type are optional. The categories are defined below:

core	Nodes of this type are always required to be present in the MD.
resource required	If the resource described by the node is available within the virtual machine, an associated node of this type is required to be present in the MD in order to describe the resource.
required by X	If a node of type X is present in the MD, then one (or more) nodes of this type will be present in the MD and associated with X.
optional	A node of this type need not appear as part of the MD, it is entirely optional, and guest OS code should have a default policy to continue functioning despite this absence.

5.2 Content versions

The “root” node (section 5.5.1) is unique in the entire machine description. It is; the one node from which all other nodes can be reached, guaranteed to be the first node defined in the node block, and is required to be present in a properly formed machine description.

The root node is primarily a construction node, with arc properties connecting to other nodes in the description. The root node carries a string property “content-version” that defines the version number of the content of the machine description”.

Content versioning is defined independently of the machine description transport version. The content version identifies the rules surrounding construction of the DAG describing the machine.

This specification is for content version “1”.

Minor changes such as the addition of new node types, properties or arc names, or the removal of optional nodes or properties, do not require a content version number change.

Incompatible changes to the node definitions such that any possible earlier machine description consumer will encounter problems with the newer content cause a version change.

5.3 Summary of node definitions

The list of currently defined nodes is as follows:

Node Name	Defined in section	Brief description
cache	5.6.1	Definition of a cache in the memory system hierarchy
cpu	5.5.3	Definition node for a single CPU
cpus	5.5.2	Construction node pointing to all cpu nodes
exec_unit	5.6.2	Node describing an execution unit of processor
mblock	5.5.5	Definition of single block of available memory
memory	5.5.4	Construction node pointing to all available mblock nodes
platform	5.5.6	Node describing intrinsic platform properties
root	5.5.1	The primary node
tlb	5.6.3	Definition of a TLB in the memory system heirarchy

Each of these nodes is defined in more detail in the following sections.

5.4 Common data definitions

As defined by the machine description transport, data values for string and data property elements (PROP_STR and PROP_DATA) are placed in the data block of the machine description. This section defines commonly used formats of data placed in the data block of a machine description and referred to using elements with the PROP_DATA tag.

Additional data formats may also be defined explicitly with a specific node definition.

5.4.1 String array

A string array is a commonly used data property that defines a concatenated list of nul character terminated strings. The PROP_DATA element that refers to this structure carries an offset (within the MD data block) to the start of the first string. The size field corresponds to a count of all the string bytes comprising the compound string list.

In this format strings are concatenated one immediately after the next. Thus if p is a pointer to the first string, then p+strlen(p)+1 is a pointer to the second. The overall size of this data field is used to determine the last string in the list. Every string in the list must terminate with the nul character. The string pointed to by p is the last string in the array if p+strlen(p)+1 equals the address of the property data plus its length. A string array of zero elements is not possible since the data length of a PROP_DATA element cannot be zero. Consumers should interpret the absence of the property as indicating an array of zero elements.

For example; the string list { “data”,”load”,”store” } would be encoded as a PROP_DATA pointing to a 16byte block of the data section of the MD with the byte values: 0x64 0x61 0x74 0x61 0x00 0x6c 0x6f 0x61 0x64 0x00 0x73 0x74 0x6f 0x72 0x65 0x00.

5.5 Generic nodes

5.5.1 Root node

Name	Category	Required subordinates	Optional subordinate
root	core	cpus (§5.5.2) memory (§5.5.4) platform (§5.5.6)	

Description

A node of this type must always be the first node in a machine description.

Only one node in the machine description may be named “root”.

This root node must be the first node defined in the node block of the machine description.

All other nodes in the forward graph can be reached starting at the root node.

Properties

Name	Tag	Required	Description
content-version	PROP_STR	yes	Version string for the content of this machine description. Currently defined version is “1”

5.5.2 Cpus node

Name	Category	Required subordinates	Optional subordinate
cpus	required by root		cpu(§5.5.3)

Description

This construction node leads directly to all the virtual CPUs supported within this virtual machine. The number of cpus is expected to be derived by counting the number of subordinate cpu nodes.

Properties

None defined

5.5.3 Cpu node

Name	Category	Required subordinates	Optional subordinate
cpu	resource required		exec_unit (§5.6.2) cache (§5.6.1) tlb (§5.6.3)

Properties

Name	Tag	Required	Description
clock-frequency	PROP_VAL	yes	A 64-bit unsigned integer giving the frequency of the sun4v virtual CPU in Hertz and thereby the frequency of the processor's %tick register
compatible	PROP_DATA*	yes	String array of cpu types this virtual cpu is compatible with. The most specific cpu type must be placed first in the list, finishing with the least specific.
id	PROP_VAL	yes	A unique 64-bit unsigned integer identifier for the virtual CPU. This identifier is the one to use for all hypervisor CPU services for the CPU represented by this node.
isalist	PROP_DATA*	yes	List of the instruction set architectures supported by this virtual CPU.
mmu-#context-bits	PROP_VAL	no	A 64-bit unsigned integer giving the number of bits forming a valid context for use in a sun4v TTE and the MMU context registers for this virtual CPU. sun4v defines the minimum default value to be 13 if this property is not specified in a cpu node.
mmu-#shared-contexts	PROP_VAL	no	A 64-bit unsigned integer giving the number of primary and secondary shared context registers supported by this virtual CPU's MMU. If not present the default value is assumed to be 0
mmu-#va-bits	PROP_VAL	no	A 64-bit unsigned integer giving the number of virtual address bits supported by this virtual CPU. If not present a default value of 64 is assumed. Note: It is legal for there to be fewer VA bits than real address bits.
mmu-compatible	PROP_DATA*	no	String array listing alternate mmu-type values that this virtual CPU's MMU interface is also compatible with
mmu-max-#tsbs	PROP_VAL	no	A 64-bit unsigned integer giving the maximum number of TSBs this virtual CPU can simultaneously support. If not present the default value is assumed to be 1. *Note: sun4v Solaris assumes at least 2 are available.*
mmu-page-size-list	PROP_VAL	no	A 64-bit unsigned integer treated as a bit field describing the page sizes that may be used on this virtual CPU. Page size encodings are defined according to the sun4v Architecture Specification. A bit N in this field, if set , indicates that sun4v defined page size with encoding N is available for use. For example bit 0 corresponds to the availability of 8K pages. If not present, a default value of 0x9 is assumed, indicating the sun4v default availability of 8K and 4M pages.
mmu-type	PROP_STR	yes	Name for the kind of MMU in use by this cpu Currently defined names are: “sun4v”

Name	Tag	Required	Description
nwins	PROP_VAL	yes	A 64-bit unsigned integer giving the number of SPARCv9 register windows available on this virtual CPU
q-cpu-mondo-#bits	PROP_VAL	yes	A 64-bit unsigned integer the maximum size (in bits) of the cpu mondo queue head and tail registers
q-dev-mondo-#bits	PROP_VAL	yes	A 64-bit unsigned integer giving the maximum size (in bits) of the device mondo queue head and tail registers
q-resumable-#bits	PROP_VAL	yes	A 64-bit unsigned integer giving the maximum size (in bits) of the resumable error queue head and tail registers
q-nonresumable-#bits	PROP_VAL	yes	A 64-bit unsigned integer giving the maximum size (in bits) of the non-resumable queue head and tail registers

Note: The 'compatible' will have “SUNW,sun4v” as the last element for systems of the sun4v machine class.

Note: Currently defined ISAs for constructing an 'islist' are: "sparcv9", "sparcv8plus", "sparcv8", "sparcv8-fsmuld", "sparcv7", "sparc".

5.5.4 Memory node

Name	**Category**	**Required subordinates**	**Optional subordinate**
memory	required by root		mblock(§5.5.5)

Description

This construction node leads directly to all the blocks of real address space backed by memory within this virtual machine.

Properties

None defined

5.5.5 Mblock node

Name	Category	Required subordinates	Optional subordinate
mblock	resource required		

Description

This node represents a single contiguous range of a virtual machine's real address space that is associated with real memory.

Properties

Name	Tag	Required	Description
base	PROP_VAL	yes	A 64-bit unsigned integer giving the base real address of the memory block represented bythis node
size	PROP_VAL	yes	A 64-bit unsigned integer giving the size in bytes of the memory block represented by this node

5.5.6 Platform node

Name	Category	Required subordinates	Optional subordinate
platform	core		

Description

This node holds general properties describing the platform a guest operating system is running on.

Properties

Name	Tag	Required	Description
banner-name	PROP_STR	yes	The banner name of the system.
hostid	PROP_VAL	no	A 64-bit unsigned integer in which the lower 32 bits hold the host id assigned to the virtual machine. The upper 32bits must be zero.
mac-address	PROP_VAL	no	A 64-bit unsigned integer in which the lower 48bits holds the mac address assigned to the virtual machine. The upper 16bits must be zero.
name	PROP_STR	yes	The platform binding name of the system. May not contain white space characters.
serial#	PROP_VAL	no	A 64-bit unsigned integer in which the lower 32 bits hold the serial number assigned to the virtual machine. The upper 32bits must be zero.
stick-frequency	PROP_VAL	yes	A 64-bit unsigned integer giving the frequency in Hertz of the system (%stick) clock for the virtual machine.

Note: A platform's banner-name is cosmetic only, typically of the form "Sun Fire T100", but the name is part of the platform binding, typically of the form "SUNW,Sun-Fire-T100".

5.6 Memory hierarchy nodes

The following nodes are used to convey information about the host memory system heirarchy to a guest.

5.6.1 Cache node

Name	Category	Required subordinates	Optional subordinate
cache	optional		cache (§5.6.1)

Description

This node describes a cache in the memory system hierarchy.

Properties

Name	Tag	Required	Description
associativity	PROP_VAL	yes	A 64-bit unsigned integer giving the associativity of the cache (number of ways in each set). A value of 0 indicates fully associative, a value of 1 indicates direct-mapped, a value of 2 indicates 2-way and so on.
compatible-type	PROP_DATA	no	Holds a string array of “type” field values. In the event that a precise type match cannot be made using the “type” property this property may be searched for compatible types.
level	PROP_VAL	yes	A 64-bit unsigned integer giving the notional level of this cache in the memory hierarchy.
line-size	PROP_VAL	yes	A 64-bit unsigned integer giving the number of bytes comprising a single cache line. This is the size of the caches allocation unit that is matched by a single cache tag
sub-block-size	PROP_VAL	no	A 64-bit unsigned integer giving the number of bytes comprising a single cache sub-block. This is the size of the cache's coherence unit size that is matched by a single state entry. This property may be omitted if it would have the same value as the line-size property.
size	PROP_VAL	yes	A 64-bit unsigned integer giving the capacity (size) in bytes of the cache.
type	PROP_DATA	yes	String array listing what may be held in this cache. Generic types are “instruction” and “data”.

5.6.2 Exec-unit node

Name	Category	Required subordinates	Optional subordinate
exec-unit	optional		cache (§5.6.1) tlb (§5.6.3)

Description

This node is describes an execution unit associated with a virtual CPU. Each execution unit may perform multiple functions / operations, and properties are defined appropriate not just to the whole execution unit, but also to individual function capabilities.

Properties

Name	Tag	Required	Description
compatible-type	PROP_DATA	no	If defined holds a string array of “type” field values. In the event that a precise type match cannot be made using the “type” property this property may be searched for compatible types.
type	PROP_DATA	yes	String array listing functional capabilities of this execution unit. Generic types are: “ifetch” - instruction fetcher “integer” - integer instruction execution “fp” - floating point instruction execution “vis” - vis instruction execution “integer-load” - integer load operations “integer-store” - integer store operations “fp-load” - floating point load operations “fp-store” - floating point store operations Niagara specific types are: “n1-crypto” - Niagara 1.0 crypto unit

5.6.3 TLB node

Name	Category	Required subordinates	Optional subordinate
tlb	optional		

Description

A TLB node describes a Translation Lookaside Buffer (MMU translation cache) in the memory system hierarchy.

Properties

Name	Tag	Required	Description
associativity	PROP_VAL	yes	A 64-bit unsigned integer giving the associativity of the TLB (number of ways in each set). A value of 0 indicates fully associative, a value of 1 indicates direct-mapped, a value of 2 indicates 2-way and so on.
compatible-type	PROP_DATA	no	If defined holds a string array of “type” field values. In the event that a precise type match cannot be made using the “type” property this property may be searched for compatible types.
entries	PROP_VAL	yes	A 64-bit unsigned integer giving the number of translation entries
level	PROP_VAL	yes	A 64-bit unsigned integer giving the notional level of this translation buffer in the overall page translation hierarchy
page-size-list	PROP_VAL	yes	A 64-bit unsigned integer treated as a bit field describing the page sizes that may be used in this TLB. Page size encodings are defined according to the sun4v Architecture Specification. A bit N in this field, if set , indicates that sun4v defined page size with encoding N is available for use. For example bit 0 corresponds to the availability of 8K pages.
type	PROP_DATA	yes	String array listing functional capabilities of this execution unit. Currently defined types are: “instruction” - translate instruction fetches “data” - translates data accesses