8.13.6 Platform node

Name	Category	Required subordinates	Optional subordinate
platform	core		

8.13.6.1 Description

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

8.13.6.2 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	*no*	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	*no*	A 64-bit unsigned integer giving the frequency in Hertz of the system (%stick) clock for the virtual machine.
watchdog-resolution	PROP-VAL	*no*	The resolution, in milliseconds, of the watchdog API service. This property is present if the watchdog timer is service is available, but is otherwise not required.
watchdog-max-timeout	PROP-VAL	*no*	The largest number of milliseconds that is valid as a parameter to the watchdog timer service API. This property is present if the watchdog timer is service is available, but is otherwise not required.

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 S UNW,Sun-Fire-T100.

10.1.6 mach_watchdog

```
trap#       FAST_TRAP
function#   MACH_WATCHDOG
arg0        timeout

ret0        status
ret1        time_remaining
```

This API service provides a basic watchdog timer service.

A guest uses this API to set a watchdog timer. Once the guest has set the timer, it must call the timer service again either to disable or re-set the expiration. If the timer expires before being re-set or disabled, then the hypervisor takes a platform specific action leading to guest termination within a bounded time period. The platform action may include recovery actions such as reporting the expiration to a Service Processor, and/or automatically restarting the guest.

If the *timeout* argument is zero, the watchdog timer is disabled. If the implementation cannot disable the watchdog timer, ENOTSUPPORTED shall be returned.

If the *timeout* argument is not zero, the watchdog timer is set to expire after a minimum of *timeout* milli-seconds. The implemented *timeout* granularity is given by the *watchdog-resolution* property in the platform node of the guest's machine description (see Â§8.13.6); the *timeout* specified is rounded up to the nearest integer multiple of watchdog-resolution milliseconds.

The largest allowed *timeout* value is specified by the *watchdog-max-timeout* property of the platform node. If the *timeout* value exceeds the value of the *watchdog-max-timeout* property, the hypervisor leaves the watchdog timer state unchanged, and returns a status of EINVAL.

The *time_remaining* return value is valid regardless of whether the return status is EOK or EINVAL. A non-zero return value indicates the number of milli-seconds that were remaining until the timer was to expire. The time remaining will be rounded up to the nearest millisecond of *watchdog-resolution* available. If the implementation cannot support the *time_remaining* feature, minus one (-1) shall be returned.

Programming note: If the hypervisor cannot support the exact timeout value requested, but can support a larger timeout value, the hypervisor may round the actual timeout to a value larger than the requested timeout, consequently the time_remaining return value may be larger than the previously requested timeout value.

Programming note: Any guest OS debugger should be aware that the watchdog service may be in use. Consequently, it is recommended that the watchdog service is disabled upon debugger entry (e.g. reaching a breakpoint), and then re-enabled upon returning to normal execution. The API has been designed with this in mind, and the time_remaining result of the disable call may be used directly as the timeout argument of the re-enable call.