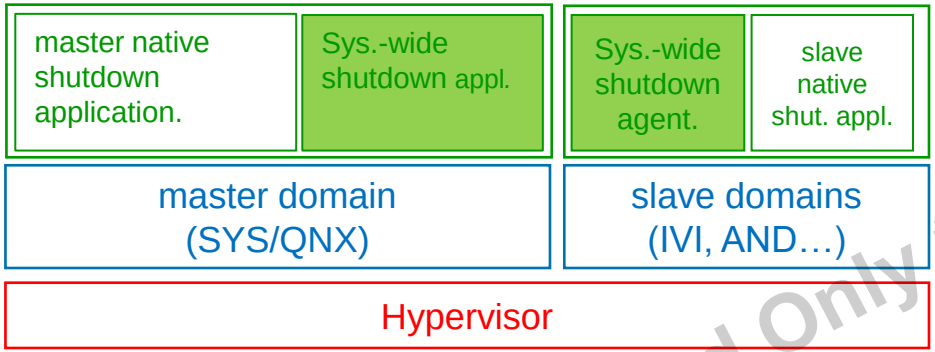
HARMAN
A SAMSUNG COMPANY

SYSTEM-WIDE SHUTDOWN DOCUMENTATION

V2.0

28.01.2021

SYSTEM-WIDE SHUTDOWN OVERVIEW

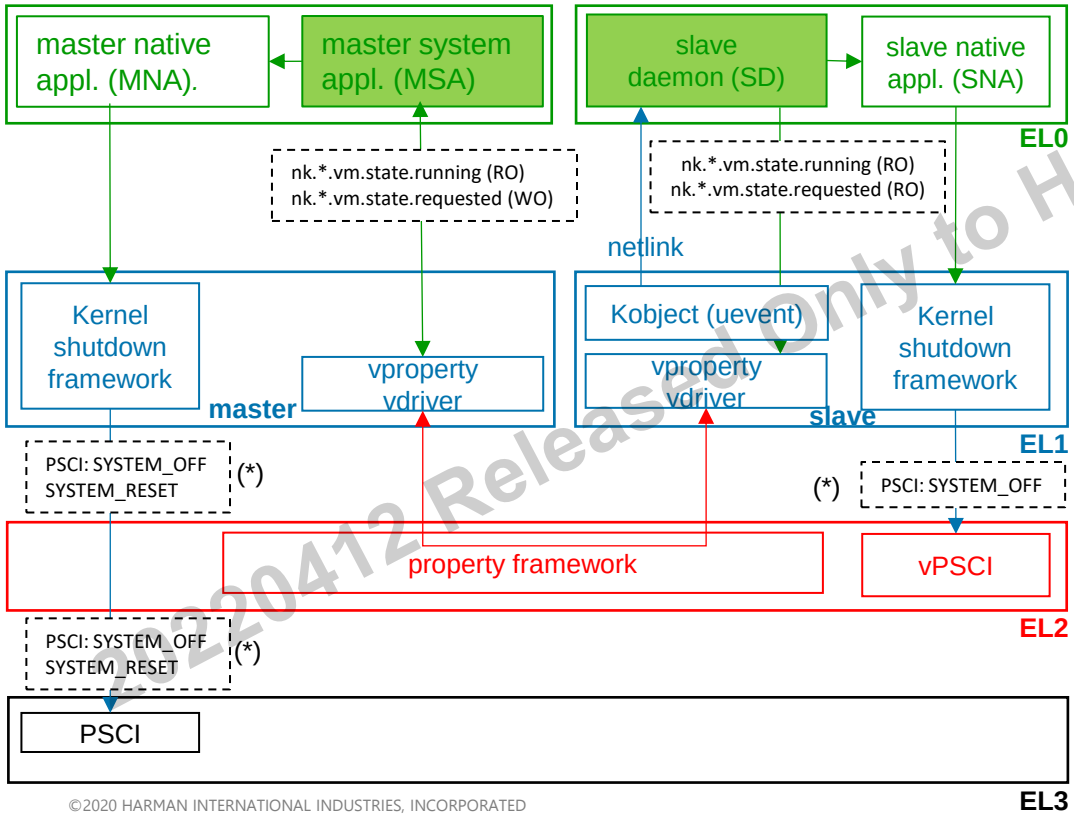


Preliminary definitions:

- Slave shutdown: takes place on a slave domain, and only affects a slave virtual machine.
- Master shutdown: takes place on the master domain, and affects the physical machine as a whole. No particular caution is taken with regards to the others virtual machines, more than likely to be brutal.
- System-wide shutdown: spread over all domains, and affects the physical machine as a whole in a clean and coordinated way. It relies on both the slave and master shutdown functionalities.

- Extends the Linux shutdown functionality, which usually operates at the OS level, to the whole virtualized system.
- Behavioural variants of the system-wide shutdown allow to
  - 1) selectively shutdown/reboot a particular slave domain when a vmid is provided (so-called selective shutdown),
  - 2) shutdown the whole system when no vmid is provided (so-called system-wide shutdown),
  - 3) reboot after shutdown (-r option),
  - 4) cancel in case of unresponsive slave (the default behavior),
  - 5) forces completion in case of unresponsive slave (-f option),
  - 6) wait for the specified amount of time before completing the command (-t ms
  - option),
- One domain has a privileged role (master) and controls the other (slaves) domains.
- Is implemented as a “system-wide shutdown application” running on the master domain, and system-wide shutdown agents running on each of the slave domains (green boxes in the diagram).
- Relies on pre-existing native shutdown applications residing in each domain.
- Shutdown master can be hosted by a Linux or a QNX guest OS.
- Shutdown slave can be hosted by a Linux or Android guest OS.

SYSTEM-WIDE SHUTDOWN DESIGN - STATIC VIEW



Guest OS agnostic elements of design

- The MSA detects the present slave domains with the existence of their associated nk.*.vm.state.running vproperty
- nk.*.vm.state.running vproperties are maintained by the Hypervisor. The initial value is 0, it is incremented upon each VM start and each VM stop, hence, value is even for stopped VMs, odd for running ones
- The MSA requests an <id> slave domain to shutdown setting the value of its nk.<id>.vm.state.requested vproperty to the next even nk.*.vm.state.running value
- The MSA requests an <id> slave domain to reboot setting the value of its nk.<id>.vm.state.requested vproperty to the next odd nk.*.vm.state.running value
- When a SD receives a « new » shutdown/reboot request, it self-shutdown/reboot accordingly. The HV updates nk.*.vm.state.running
- A request is considered as new if nk.<id>.vm.state.requested value is higher than nk.*.vm.state.running value.

Linux specific elements of design

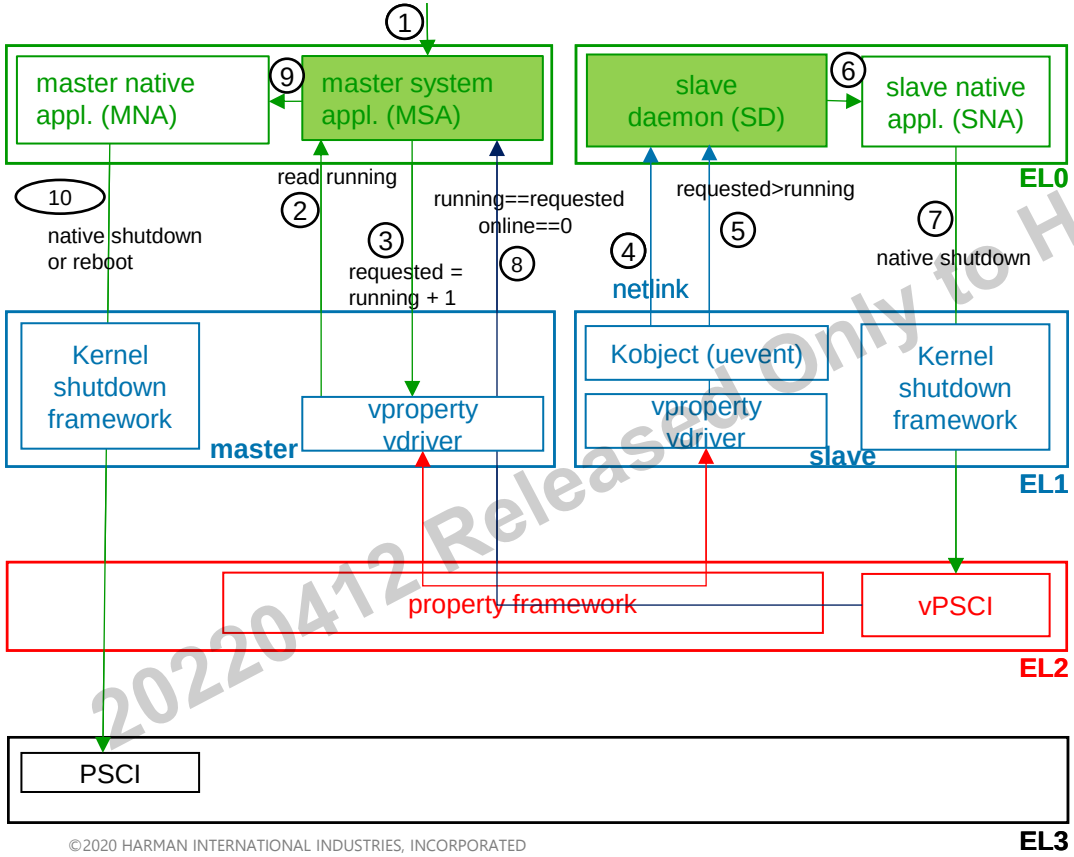
- The MSA access vproperties through the « /sys/nk/prop/ » filesystem exposed by the vproperty vdriver.
- the shutdown agent is a daemon (SD) listening on the netlink socket for vproperty change notifications.

Qnx specific elements of design

- The MSA accesses vproperty through the ddi layer (not represented).

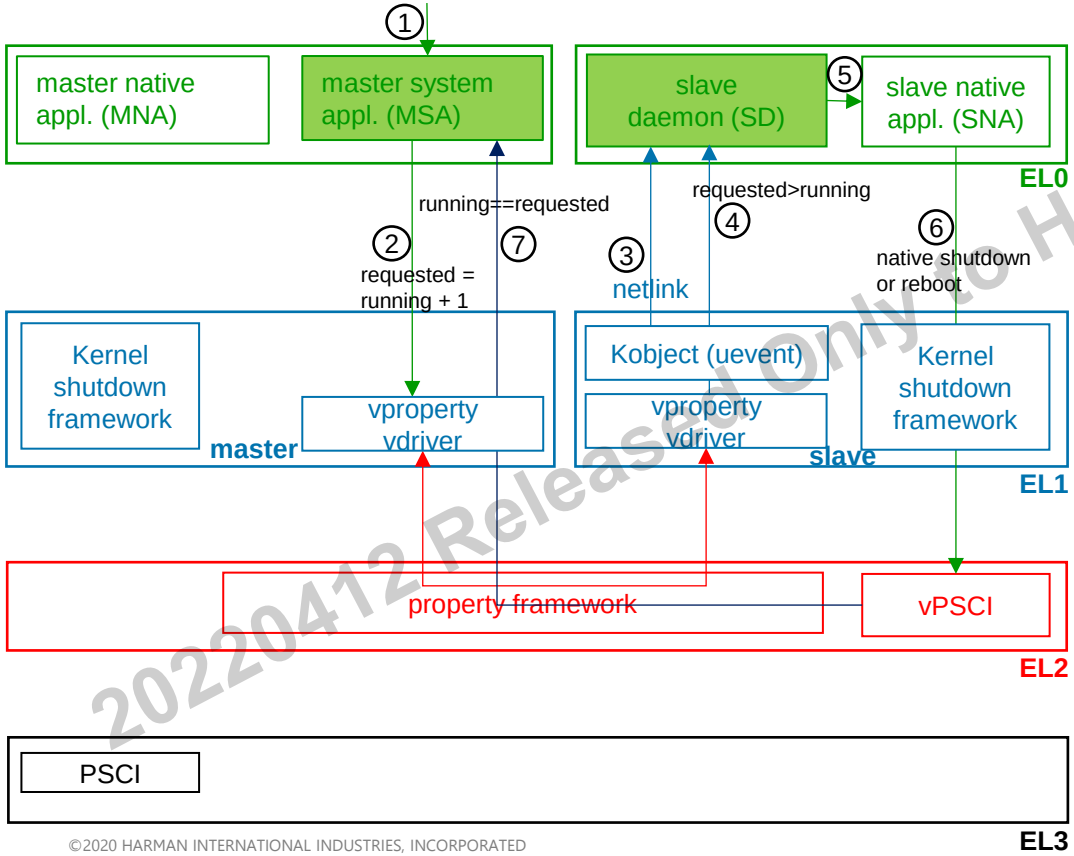
(*) this is a possible implementation. It might differ depending on the platform

SYSTEM-WIDE SHUTDOWN DESIGN – DYNAMIC VIEW 1/2 (SYSTEM)



1. The MSA is invoked without any target slave identifier,
2. The MSA reads all possible nk.*.vm.state.running vproperties and selects slave domains involved in the system-wide shutdown (those who have a valid nk.*.vm.state.running property),
3. For each <id> involved slave, the MSA reads its nk.<id>.vm.state.running property, lets call the read value state[<id>]. The MSA requests each <id> involved running slave to shutdown, setting its nk.<id>.vm.state.requested to the (state[<id>] + 1) value. The new value shall be even. The MSA waits for the domains to acknowledge the request (i.e. for the nk.*.vm.state.running value to be updated), a timeout may be started,
4. Upon a nk.<id>.vm.state.requested change, a vproperty change notification is sent on the netlink socket, thus awaking the SD,
5. The SD detects a new request if nk.<id>.vm.state.requested > nk.<id>.vm.state.running,
6. The SD invokes the SNA native shutdown command,
7. The SNA shutdowns the slave domain, the HV updates the nk.*.vm.state.running vproperty,
8. The MSA detects the slave shutdown acknowledgement when nk.*.vm.state.running == nk.<id>.vm.state.requested. If an involved domain did not acknowledge the request, the MSA either exits with an error (default) without running the MNA, or forces to completion (-f option).
9. the MSA invokes the requested MNA native shutdown or reboot command,
10. the MNA shutdowns or reboots the system.

SYSTEM-WIDE SHUTDOWN DESIGN – DYNAMIC VIEW 2/2 (SELECTIVE)



1. The MSA is invoked with a target slave identifier: <id>,
2. the MSA reads the nk.<id>.vm.state.running property, lets call the read value: state. The MSA requests the target slave to shutdown or reboot, setting its nk.<id>.vm.state.requested to the value (state + 1). The new value shall be even for a shutdown, odd for a reboot, The MSA waits for the domains to acknowledge the request (i.e. for the nk.*.vm.state.running value to be updated), a timeout may be started,
3. Upon a nk.<id>.vm.state.requested change, a vproperty change notification is sent on the netlink socket, thus awaking the SD,
4. The SD detects a new request if nk.<id>.vm.state.requested > nk.<id>.vm.state.running,
5. The SD invokes the SNA native shutdown or reboot command depending on the nk.<id>.vm.state.requested value,
6. The SNA shutdowns/reboots the slave domain, the HV updates the nk.*.vm.state.running vproperty,
7. The MSA detects the slave shutdown acknowledgement when nk.*.vm.state.running == nk.<id>.vm.state.requested. If an involved domain did not acknowledge the request, the MSA either exits with an error (default), or forces to completion (-f option).

SYSTEM-WIDE SHUTDOWN USE-CASES – SELECTIVE

Selective shutdown/reboot:

- request domain **shutdown** with **infinite** wait: waits infinitely for the selected domain to cooperate and actually self-shutdown. CTRL-C is required if domain does not cooperate.

```
$ vshutdown 4                    # request domain 4 shutdown
```

- request domain **reboot** with infinite wait: waits infinitely for the selected domain to cooperate and actually self-shutdown. CTRL-C is required if domain does not cooperate.

```
$ vshutdown -r 4                 # request domain 4 reboot
```

SYSTEM-WIDE SHUTDOWN USE-CASES – SYSTEM

System shutdown/reboot:

- request system **shutdown** with **infinite** wait: waits infinitely for all domains to cooperate and actually self-shutdown. CTRL-C is required if a domain does not cooperate.

```
$ vshutdown                    # request system shutdown
```

- request system **reboot** with infinite wait: waits infinitely for all domains to cooperate and actually self-shutdown. CTRL-C is required if a domain does not cooperate.

```
$ vshutdown –r                 # request system reboot
```