NVDIMM Runtime Firmware Activation

Some persistent memory devices run a firmware locally on the device / "DIMM" to perform tasks like media management, capacity provisioning, and health monitoring. The process of updating that firmware typically involves a reboot because it has implications for in-flight memory transactions. However, reboots are disruptive and at least the Intel persistent memory platform implementation, described by the Intel ACPI DSM specification [1], has added support for activating firmware at runtime.

A native sysfs interface is implemented in libnvdimm to allow platform to advertise and control their local runtime firmware activation capability.

The libnvdimm bus object, ndbusX, implements an ndbusX/firmware/activate attribute that shows the state of the firmware activation as one of 'idle', 'armed', 'overflow', and 'busy'.

- idle: No devices are set / armed to activate firmware
- armed: At least one device is armed
- busy: In the busy state armed devices are in the process of transitioning back to idle and completing an activation cycle.
- overflow: If the platform has a concept of incremental work needed to perform the activation it could be the case that too many DIMMs are armed for activation. In that scenario the potential for firmware activation to timeout is indicated by the 'overflow' state.

The 'ndbusX/firmware/activate' property can be written with a value of either 'live', or 'quiesce'. A value of 'quiesce' triggers the kernel to run firmware activation from within the equivalent of the hibernation 'freeze' state where drivers and applications are notified to stop their modifications of system memory. A value of 'live' attempts firmware activation without this hibernation cycle. The 'ndbusX/firmware/activate' property will be elided completely if no firmware activation capability is detected.

Another property 'ndbusX/firmware/capability' indicates a value of 'live' or 'quiesce', where 'live' indicates that the firmware does not require or inflict any quiesce period on the system to update firmware. A capability value of 'quiesce' indicates that firmware does expect and injects a quiet period for the memory controller, but 'live' may still be written to 'ndbusX/firmware/activate' as an override to assume the risk of racing firmware update with in-flight device and application activity. The 'ndbusX/firmware/capability' property will be elided completely if no firmware activation capability is detected.

The libnvdimm memory-device / DIMM object, nmemX, implements 'nmemX/firmware/activate' and 'nmemX/firmware/result' attributes to communicate the per-device firmware activation state. Similar to the 'ndbusX/firmware/activate' attribute, the 'nmemX/firmware/activate' attribute indicates 'idle', 'armed', or 'busy'. The state transitions from 'armed' to 'idle' when the system is prepared to activate firmware, firmware staged + state set to armed, and 'ndbusX/firmware/activate' is triggered. After that activation event the nmemX/firmware/result attribute reflects the state of the last activation as one of:

- none: No runtime activation triggered since the last time the device was reset
- success: The last runtime activation completed successfully.
- fail: The last runtime activation failed for device-specific reasons.
- not_staged: The last runtime activation failed due to a sequencing error of the firmware image not being staged.
- need_reset: Runtime firmware activation failed, but the firmware can still be activated via the legacy method of power-cycling the system.

[1]: https://docs.pmem.io/persistent-memory/