Power management techniques, policies, and problems for embedded Linux

Mark Gross CELF PMWG chair
mark.gross@intel.com

Power Management means many things to many people

- Basic – on/off support
- Suspend / Resume
- Critical event handling
- throttling
- Idle behavior
- policy and control
- measurement

PM in embedded Linux is a Grab-Bag of stuff

- This presentation is a discussion of some PM topics, partitioned in the following categories:
  - Techniques
  - Policies
  - Problems
- My goal is to provide the audience with a overview of the state of Linux power management today, with emphasis on embedded interests.

Techniques

- Suspend to Disk
- Suspend to Ram
- Dynamic PM
  - Power Op
- Device PM
- CPU-IDLE
- CPUFREQ
- PM-Memory
- Custom platform PM drivers
- Clock Framework
- voltage Framework
- New PM Frameworks

Suspend to Disk

- Works with try_to_freeze yield loop trap's sprinkled around the kernel to stop processing safely.
  - limited in amount of memory it can "snapshot" to ½ of RAM.
- Main entry is pm_suspend_disk, to attempt making a snapshot of the memory and write it to swap partition.
- Wake-up is software_resume

Suspend to RAM

- Entry at enter_state, suspend_prepare and suspend_enter.
- reuses STD's process freeze design, frees up memory and caches
- walks driver model device tree calling suspend_device
  - suspend failures are typically some device not suspending as expected.
- On wake-up execution picks up after pm_finish.

Dynamic PM (aka DPM)

- Monte Vista / IBM joint activity, initially pushed to LKML in Nov, 2002.
  - was put in direct competition, by its authors, against the simpler CPUFREQ, and lost.
  - Is maintained by MV as a source forge project, and is included in its some of its products (PE and ME)
  - Is the origin of the term "operating point"
  - defines a N-dimensional phase space of system performance settings that can be set / unset in a somewhat atomic manner.
  - Is used with favor by a number of MV customers and is the source of efforts to get an operating point concept into the kernel.
- Includes a number of hooks in process creation and scheduler execution paths, as well as an interface for a custom power policy manager.

Device PM

- Based on driver model device tree (/sys)
- Is tied to the bus topology of the device tree in /sys.
- It was created for suspend.
- Not useful if the topology of the power domain doesn't match the device tree.

CPU-IDLE

- coming out in 2.6.21
- provides a framework for implementing various levels of CPU idle / sleep states and the policies for selecting the best sleep level given latency constraints.
- Developed to support multiple and new C-States on Intel processors

Low power Idle

- save as much power when idle as you can.
  - Tic-less idle
  - CPU-IDLE
  - self refresh memory
  - self refresh display
  - sleep selected devices in the device tree
  - deferrable timers

CPUFREQ

- Provides a framework for governors and platform drivers to provide CPU throttling based on controlling core frequency as a function of workload (typically kstats)
- Frequency centric.
- Works well with systems with platform firmware handling the voltage and frequency coordination underneath operating system.

custom platform drivers

- provide a way to set power state by pushing values into MSR’s or device registers external to any infrastructure.
- Not portable and result in maintenance problems if reusing software across multiple product versions and architectures.

Power Managed Memory

- If the memory isn't in use put it in self refresh.
- Some workloads lend themselves to PM memory.
- Memory affinity can be used to squeeze some savings
  - by delaying the on-lining some memory
  - by implementing allocation or access policies.
- there exists some NUMA approaches to this concept.

Power Opp

- Power Opp is the operating point subset of DPM, and was strongly pushed last year.
  - It almost got into the MM tree.
  - Got side lined and fell into a common trap of confusing the more vocal Linux-PM personalities.
    - OpPoint posting added to the confusion
  - Once again things went bad shortly after discussions referencing CPUFREQ.

Clock Framework

- basically a header file (clk.h) defining in C and abstract base class for representing a dependency relationship between clock devices.
- Started as an ARM only thing, but was moved to include/linux after multiple architectures started to use it.

Voltage Framework

- a new ARM patch put up by Nokia.
- Attempts to provide a framework, with implementation for omap, that is somewhat analogues to the clock framework.
- Patches where posted about a month ago.

New PM Framework

- Partially funded by CELF.
- Attempts to provide a unification of the clock, voltage frameworks along with operating points.
  - Will tie into existing work.
  - Will not compete with CPUFREQ.
- Design is trying to adapt to the recent voltage framework posting.

Policies

- On-demand (CPUFREQ)
- Low Power IDLE
- Modal Policies
- Race to Idle
- Dynamic use
- Graceful shutdown

On-demand

- is a CPUFREQ policy
- Loaded as a driver plug-in to CPUFREQ
- attempts to control the kstat idle to ~20%
- used to use timer events to compute idle.
- with tic-less idle its getting modified to not create events for ~2.6.23.
- deferrable timers will help.

Low power idle

- A degenerate policy
- Try to save the maximum power when idle while providing some specified level of latency in getting to a non-idle state.
- It’s harder than it sounds.
  - tic-less idle, CPU-IDLE
  - Platform hinting of acceptable wake up latencies
  - shutting down un-needed user mode processes with timer’s to extend idle periods.

Modal policies

- common with DPM / operating point based solutions.
- driven from user space.
- doesn’t need kernel infrastructure for dynamic processing or determination of power / performance settings.

Race to Idle

- Works well when CPU is faster than workload needs.
  - On-Demand was initially based on this high level policy.
- For CPU's that are sized for workloads, it may not be the best approach for optimal power savings.
  - NXP presented a good analysis of this for one of its products last fall.

Graceful shutdown

- When battery is low we need to save state and shut down the system.
- When there is a thermal critical event we also need to shutdown the system.

Problems

- Bad luck on LKML and Linux-PM
  - having the wrong discussions with key maintainers
  - Putting embedded PM interests in competition with non-embedded implementations, and loosing.
- Lack of interfaces for policy mangers
- Missing frameworks handling device dependencies, notifications and constraints
- User mode programs are not good at being idle
- Not enough people are focusing on PM

Bad Luck on Linux-pm and LKML

- Saying the wrong words on linux-pm
  - having the string “CPUFREQ” in any post is the kiss-of-death.
  - DPM died because it was put into competition with CPUFREQ.
  - Power Op was put down because of confusion on how it related to and could be used by CPUFREQ.
- Having the wrong discussions with the PC centric kernel developers.
  - Stick to “existing infrastructure doesn’t work because…”
  - New code needed for our requirements…
  - new code coexists with existing infrastructure…
  - Embedded power management IS different and needs different infrastructure.

Missing constraint, dependency and notification infrastructure

- infrastructure that supports topologies that do not map onto the driver model device tree.
- Whomever posts this code needs to be ready to address challenges that this infrastructure exists in the driver model.
  - The driver model doesn’t work for dependency topologies that don’t fit the bus / driver tree model.

User mode sucks

- UI code loves to set up timers.
- UI code is not good at allowing deep idle states.
- More people need to look at this to accelerate the clean up.
- /proc/timer_stats is your friend.

Not enough code is getting posted!

- There is a lot of talk
- There could be more code
- We need more people looking at power management and posting code.

Summary

- there are a fair number of PM techniques available in Linux today.
- There are only a handful of policies, and a limited number of interfaces for policy managers.
- Linux-PM has been hard for embedded interests to get mind share.
- Don’t put embedded PM interests in competition with PC Centric implementations!

The end

- Thank you.
- Questions?