LCA14-206: Scheduler tooling and benchmarking

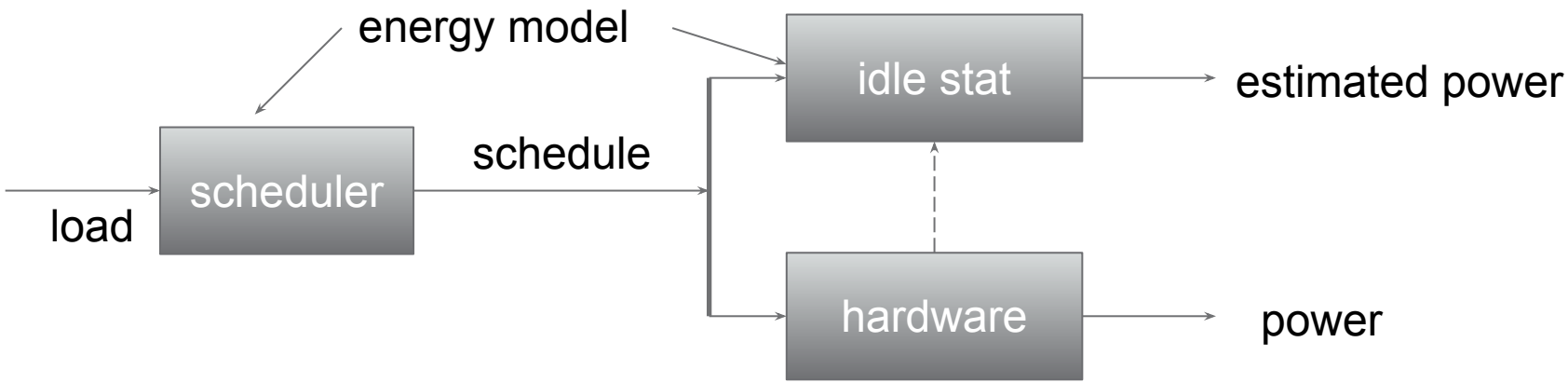
Tue-4-Mar, 11:15am, Zoran Markovic, Vincent Guittot

Scheduler Tools and Benchmarking

- From Energy Aware mini-summit @ Ksummit 2013
  - extract from [1]: “Ingo Molnar came in with a complaint: none of the power-management work starts with measurements of the system's power behavior. Without a coherent approach to measuring the effects of a patch, there is no real way to judge these patches to decide which ones should go in. We cannot, he said, merge scheduler patches on faith, hoping that they somehow make things better.”

[1] https://lwn.net/Articles/571414/

- Tools need to:
  - Generate repetitive, deterministic load patterns
  - Evaluate performance and/or power consumption
  - Check for violation of scheduling constraints



Load Generation

Load Generation: Cyclictest

- Maintained by Clark Williams of RedHat
- Devised to measure real-time performance
- Measures latencies in response times
  - Performs timer sleep, followed by clock_gettime()
  - Compares requested sleep time to measured time
  - Difference is the latency
- Starts a number of threads whose sleep can be staggered in time
- Linaro version also busy-loops a specified number of iterations after wakeup
  - Together with sleep time, this represents the periodic load pattern
  - Auto Load calibration

Load Generation: rt-app

- Maintained by Giacomo Bagnoli and Juri Lelli
- Used to test deadline scheduler
- Use json file to describe scenario
- Locking scenario for thread dependency
- Task priority setting
- Runtime, Period and deadline
- Generate stats and trace events for debugging and analyses

Load Generation: New Development

- Ability to generate custom load sequences
  - Period, Load & Deadline
  - non periodic load
- Auto Load calibration
- Task priority setting
- Task dependency with lock scenario
- Load configuration file
- Generate ftrace event and statistic
- Shared object library for linking into other scheduler tools
- Other?

Estimating Power Consumption

Energy Model

- In reality, very complex
  - Focusing on CPU power consumption

- Approximated with a simple linear energy model:
  - $f = \sum_{cpu} PCi * (TCi - TCCi) + \sum_{cpu} PPi * TPi + \sum_{cluster} PCCi * TCCi$
  - $PCi,$ – CPU power consumption in state Ci
  - TCi – Total time during which CPU was in state Ci
  - PPi – CPU power consumption in state Pi
  - TPi - Total time during which CPU was in state Pi
  - $PCCi$ – Cluster power consumption in state Ci
  - $TCCi$ – Total time during which cluster was in state Ci

- Cluster C-states are defined as in ACPI spec

Energy Model

- Each platform has different power consumption parameters

- As a consequence, schedules are platform-specific, i.e.
  - Each platform may have its own view of what's most efficient
  - There is no apples-to-apples comparison of efficiency across platforms, but
  - We can compare multiple scheduler solutions on a single platform
  - It would be prudent to characterize scheduler implementation on a multitude of platforms

Benchmarking

Benchmarking

- Given an energy model, evaluate a schedule
  - Capture the time spent in each C-state and P-state
  - Run it through the energy model to assess power consumption

- Also, verify constraints
  - How long did it take to complete processing?
  - Were any of the deadlines missed?
  - Were tasks properly prioritized?
  - Was it done within thermal budget?

Idlestat

- Helps to assess how much energy was spent for a particular schedule
  - Documentation RFC pending
- Makes no assumption about the energy model
- Uses kernel FTRACE function to capture:
  - Entry and exit times for each C-state
  - Entry and exit times for each P-state
  - Raised IRQs
- idlestat is non-intrusive to C-state and P-state transitions:
  - Sleeps while traces are captured
  - Parses/analyzes traces after the acquisition is complete

Idlestat

```
clusterA@state  hits           total(us)          avg(us) min(us) max(us)
         C1     10821        5879554.00          543.35 0.00    23163.00
         C2     0                  0.00            0.00 0.00    0.00
         C3     78           2929290.00        37555.00 0.00    101441.00
  cpu0@state    hits           total(us)          avg(us) min(us) max(us)
       C1       6744         6407808.00          950.15 0.00    23194.00
       C2       3               8819.00         2939.67 549.00  5310.00
       C3       75           2960110.00        39468.13 213.00  101441.00
       350      1047          204490.00          195.31 0.00    4578.00
       700      5628          396247.00           70.41 0.00    1465.00
       920      0                  0.00            0.00 0.00    0.00
  cpu0 wakeups  name             count
       irq109   ehci_hcd:usb1    1727
       irq029   twd              4524
  cpu1@state    hits           total(us)          avg(us) min(us) max(us)
       C1       6544         6398931.00          977.83 0.00    36255.00
       C2       1               1129.00         1129.00 1129.00 1129.00
       C3       77           2955293.00        38380.43 122.00  101471.00
       350      1124          212428.00          188.99 0.00    18677.00
       700      5366          408782.00           76.18 0.00    946.00
       920      0                  0.00            0.00 0.00    0.00
  cpu1 wakeups  name             count
       irq029   twd              4737
```

Idlestat

- How do we get power parameters?
  - Device tree/manufacturer's data?
  - Linear fitting?
- Method evaluated on TC2
  - Work in progress
  - Has registers to measure per-cluster power consumption
  - Has 2 clusters, 5 cores, 2 C-states and 8 P-states each
  - Large solution space (~50 power parameters)
  - Linear fitting simplified to solving 6x6 equation system for A7s only, single P-state (350MHz)
  - ~2.6% error* (cyclictest -t 10 -L c0p15 --latency 100000 -q)

Benchmarking: Enhancements

- Related to verification of scheduling constraints
- Assessment of performance:
  - Time execution: launch executable and wait to complete
  - Add synchronization points to load generation tool
  - Time between synchronization points represents a measure of performance
- Assessment of processing latencies
  - How do we do this in a non-intrusive manner?
  - Inside load generation utility?
- Thermal assessment
  - Energy consumed over time
  - Overshooting/undershooting target

More about Linaro Connect: http://connect.linaro.org
More about Linaro: http://www.linaro.org/about/
More about Linaro engineering: http://www.linaro.org/engineering/
Linaro members: www.linaro.org/members