Linux Plumber Conference 2012

Scheduler micro-conf

Vincent Guittot <vincent.guittot@linaro.org>
Linaro Power Management Working Group

Topics

- How to keep CPU quiescent ?
  - Sharing information with other frameworks
  - Tasks placement for asymmetric system
  - RT scheduler and power consumption

Schedule

- 3 slots
- This morning: 45 min
  - Sharing information
- An extended slot this afternoon: 95min
  - Asymmetric system
- Friday: 45 min
  - RT scheduler and power management

Linux Plumber Conference 2012

Target CPU selection && Sharing scheduler Info

Vincent Guittot <vincent.guittot@linaro.org>
Linaro Power Management Working Group

Content

- Introduction
- Timer
- Workqueue
- Sharing information
- Next step

Content

- Introduction
- Timer
- Workqueue
- Sharing information
- Next step

Wake up CPU

- Everything comes from IRQ
  - IRQ balancing is not part of this discussion
- Main actions that generate activity on a core :
  - Task wake up
  - Timer action
  - Queue work

Selecting a CPU

- Task has got CPU load balance
- Timer has got partial CPU load balance
- Workqueue uses local CPU

Content

- Introduction
- Timer
- Workqueue
- Sharing information
- Next step

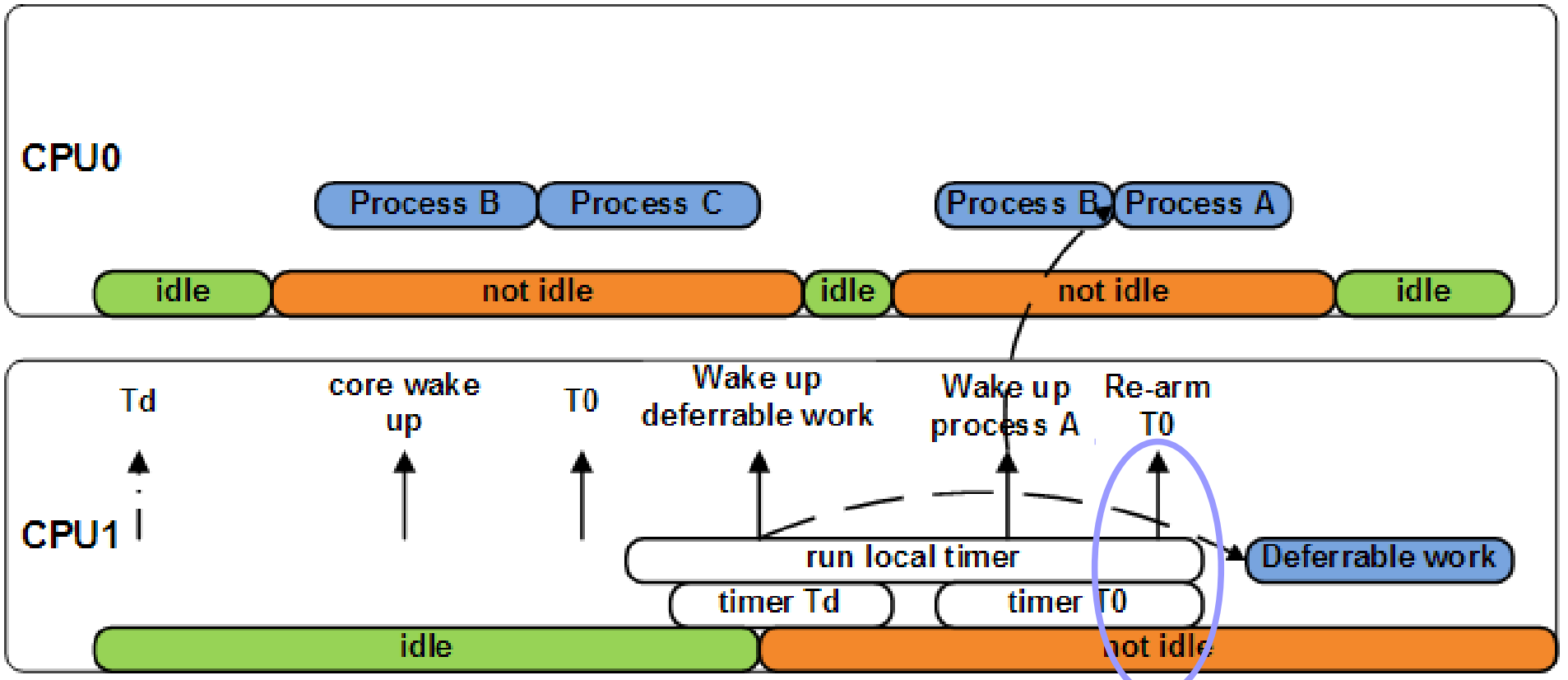
Current status

- We have an interface
  - get_nohz_timer_target()
  - scheduler looks for a non idle CPU
- Used by timer and hrtimer
  - Non pinned timer
  - Timer migration allowed
  - NO_HZ enable
  - Local CPU must be idle

Local CPU

- Local CPU must be idle
  - current task is idle task
  - nr_running == 0
  - wake_list is empty
- What about deferrable activity ?
  - Executed prior normal timer
  - Make a CPU no more idle for other timers

Deferrable



Vincent Guittot
Linaro Power Management Working Group

Proposal

- Remove check of Local CPU state
- Move this check into get_nohz_timer_target
- Let scheduler choose
  - An idle CPU could be the best choice
  - Pack short activity on few CPUs

Running timer

- Additional constraint for timer && hrtimer
  - Can't change a running timer → migration is canceled and lost

- What about timer that re-arms itself ?

Proposal

- Add a migration list
  - Add timer that failed to migrate in this list
- Check for possible migration
  - when core becomes idle

Expiration constraint

- Additional constraint for hrtimer
  - Must expire after next event of the target CPU
- Can be a potential issue

Content

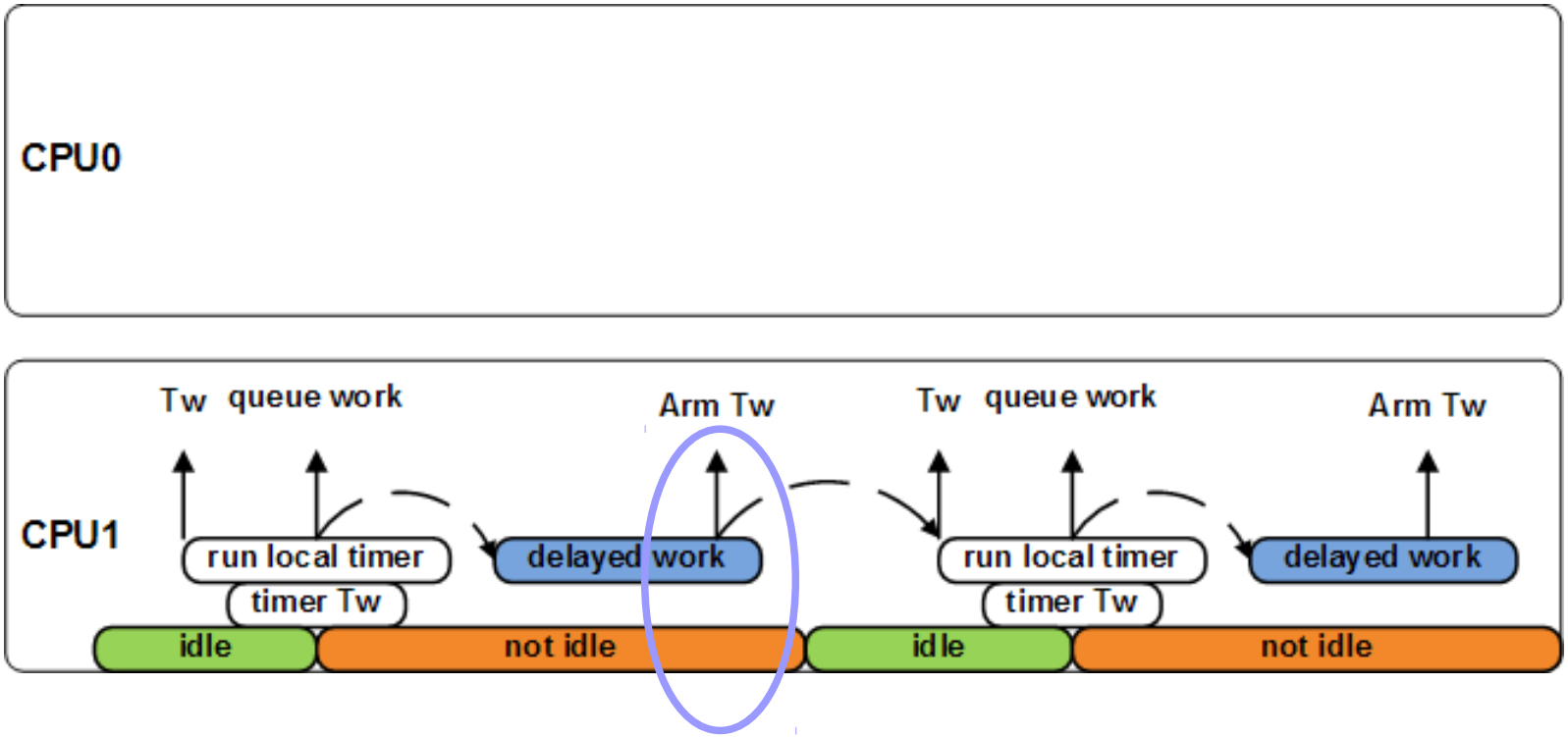
- Introduction
- Timer
- Workqueue
- Sharing information
- Next step

Workqueue

- __queue_work asks for a CPU
  - Unless unbounded → use Local CPU
  - Default system workqueue is bounded
- schedule/queue_work
  - Use get_cpu
- schedule/queue_work_on
  - Often use smp_processor_id

Delayed workqueue

- What about delayed workqueue that re-arms itself ?



- Stuck on local CPU

Choose a CPU

- How to choose the best CPU
  - Scheduler should help
  - Have a function to get preferred CPU

- int sched_select_cpu(struct cpu_mask* cpus)
  - Return the preferred CPU in the mask
  - Default behavior → return Local
  - To be used when no CPU is set

Content

- Introduction
- Timer
- Workqueue
- Sharing information
- Next step

Preferred CPUs

- Get Clock sharing topology from cpufreq
  - cpus and cpus_related masks
- Get coupled CPUs topology from cpuidle
- Get mask of CPUs in shallow C-state

Sharing information

- Scheduler performance is sensible
  - Avoid large X*Y matrix computation
- Keep it simple
  - Make it as simple a CPU mask manipulation

Accurate load

- Per task load tracking patch-set
  - From Paul Turner
- To be weighted by current cpu_power
  - DVFS
  - Asymetric system
- Can be use by other framework
  - Cpufreq
  - Cpuidle

Content

- Introduction
- Timer
- Workqueue
- Sharing information
- Next step

Next step

- RFC for timer and workqueue
  - preferred cpu
- RFC for gathering info from Power framework
  - cpumask
- RFC to provide info to Power framework
  - Accurate load

Thank you

Vincent Guittot
Linaro Power Management Working Group

Backup slide

- MP3 sequence