Dynamic Power Management Framework for Multi-Core Portable Embedded System

Presenter: Yan Like

Work of Chen Tianzhou, Huang Jiangwei, Xiang Lingxiang and Shi Qingsong

Intel-Zhejiang Univ. Technology Center

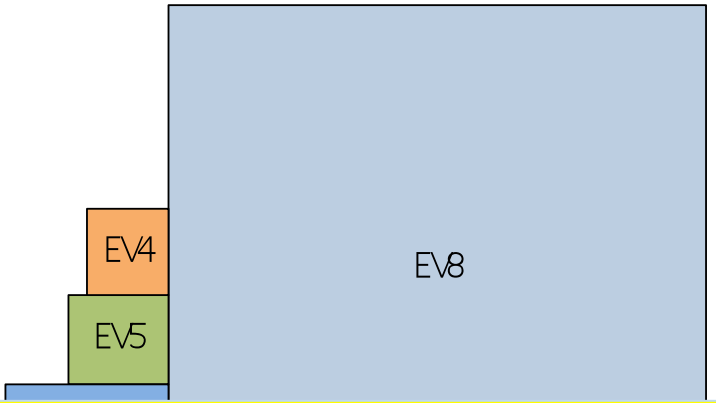
Agenda

1. INTRODUCTION
   - Power and Multi-core
   - Multi-core and Embedded system
2. DPM FRAMEWORK
   - Profiler Model
   - Global Policy Layer
3. EXPERIMENT and RESULT
4. FUTURE WORKS

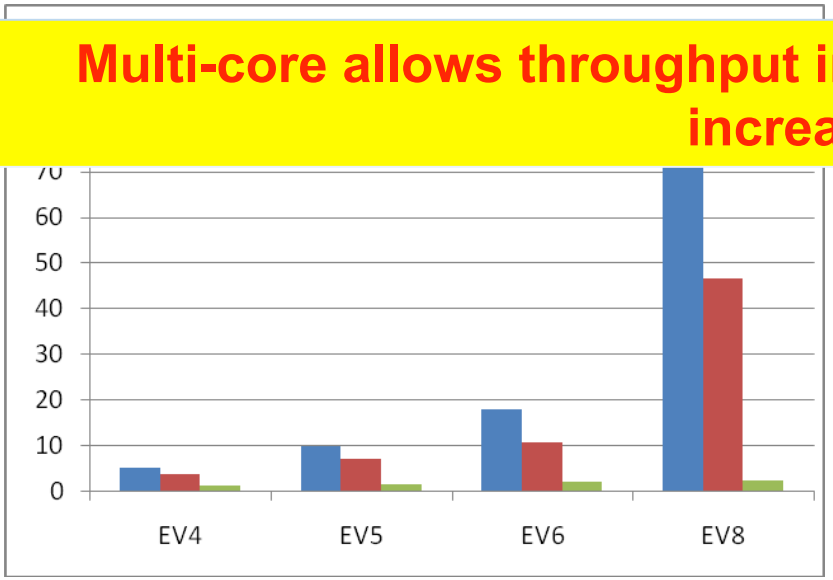
Power and Multi-core

Number of transistors on a chip doubles about every two years------ Moore's Law

EV8 is 80 times bigger but provides only two to three times more single-threaded performance



Multi-core allows throughput increase while avoiding a significant increase in power



The power dissipation and performance of four generations of Alpha microprocessor cores scaled to run at the same frequency

Multi-core for Embedded system

- Power continues to be the key factor whcih limits the performance in multi-core regime, especially in portable multi-core embedded system
- Driven by battery.
  - Mobile Phone, PDA, UMPC, etc..
- The limited capacity of Battery.
  - The power resource is limited, so we have to pinch pennies

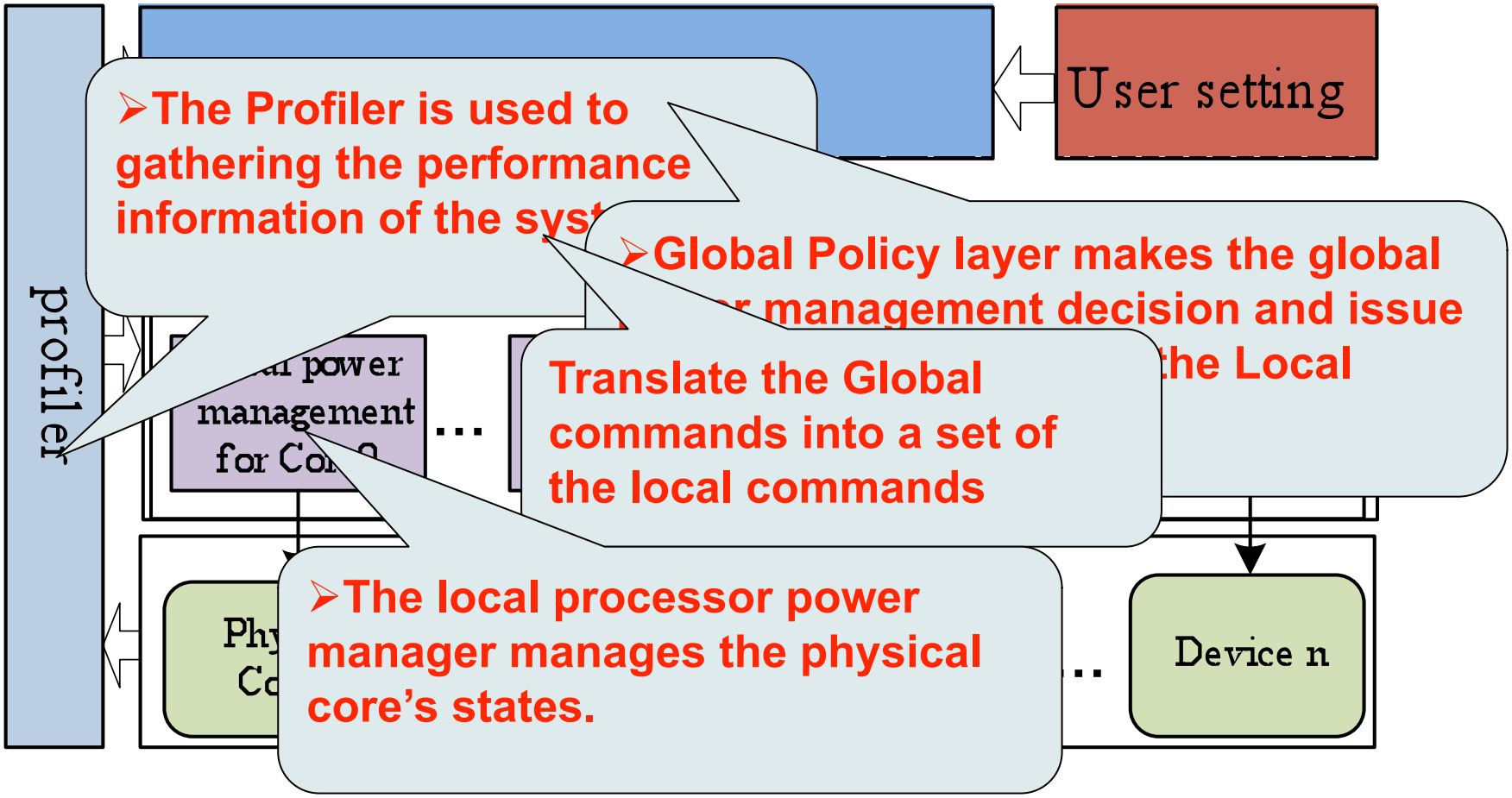
power dissipation problem still exists in the new multi-core regime for Embedded system

Our Work

A dynamic power management framework

- Combining with core-level and global scheduling heuristic management.
- Tries to assign applications to the proper cores, at the same time the frequency and voltage of core could be adjusted according to the application behavior.
- Provide a global policy, and offer a single core DVS algorithm.

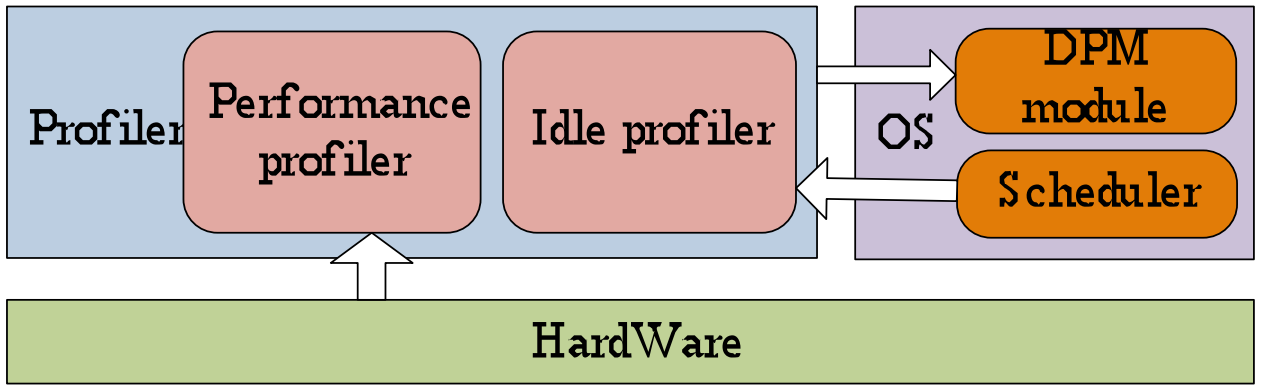
The DPM Framework



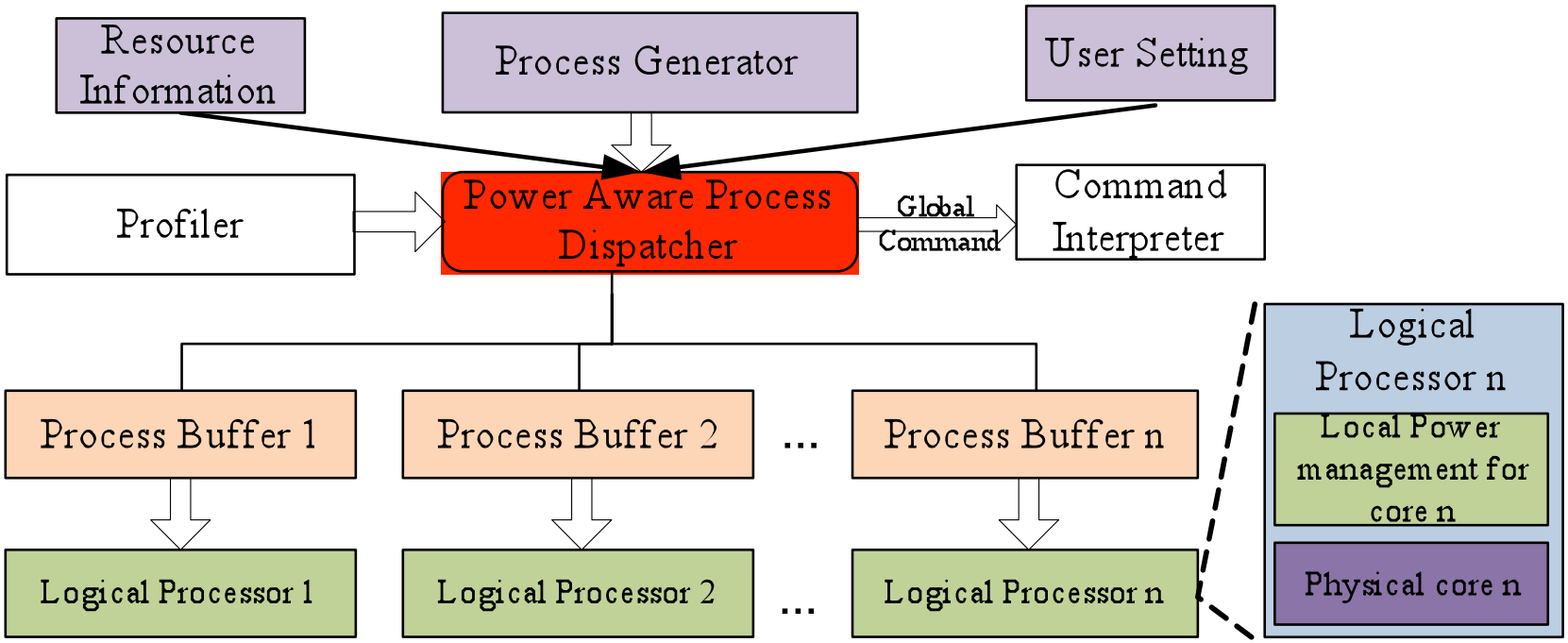
The Profiler

The Profiler is used to gathering the performance information of the system

- Performance profiler
  - monitors the hardware layer
  - gathers the performance information from the hardwares
- Idle profiler
  - gives the CPU usage and OS idle information to the DPM module by monitoring the idle thread of the OS



Global Policy Layer



Deadline test algorithm

DeadLineTest(P_{new}, CurrentBuffer)

Int DeadLine= 0;

for(all processes in the buffer)

DeadLine +=P_i.WT- P_i.RT;

if(P_{new}.DT-$T_{current}$-DeadLine- P_{new}.WT)>=0

return success;

else

return failed;

Processor throughput test algorithm

```
ThroughputTest(Pnew)
    Int BufferID=0;
SortingBuffer();
For(All the buffers)
  Int RunTime= 0;
  Int BiggestDeadLine=Pnew.DT;
  For( all processes in the buffer)
     RunTime +=Pi.WT- Pi.RT;
        If(BiggestDeadLine< Pi.DT)
              BiggestDeadLine= Pi.DT;
        If(RunTime+ Pnew.DT+ Tcurrent< BiggestDeadLine)
              BufferID=currentBuffer;
          If(DeadLineTest(Pnew, BufferID))
              Return;
```

Experiment Setup

Platform

- Use a cycle-accurate simulator on DELL INSPIRON 9400.
    - Modeling two cores and shared L2 cache based on Intel Duo Core Processor.
- The power statistics are acquired from a power modeling methodology using PMU.
- A DPM_DuoCore patch is added to Linux kernel 2.6.0.

Result

- 4 MPEG4 decoders on the platform.
- Two single core adjusting strategies: IS and AS
  - (From “Brock, B. and K. Rajamani, Dynamic Power Management for Embedded Systems, in IEEE International SOC Conference. 2003”)
- Result

Strategy	System Power (mw)	Normalized to Default
AS	4302.1	74.8%

AS strategy saves more power than IS and Default

Future works

- More experiments is on going
  - More kinds of workloads
  - More homogeneous cores
  - Heterogeneous cores
- Exploring an more power efficient scheduling algorithm for the processes in a same buffer under the deadline constraints
  - By grouping the different type of computing, so the hardware utilization could be improved…
  - By sequentially issue the same type of computing to burst the same operations of hardwares…

Thank you for your time!

tzchen@zju.edu.cn
hjw@zju.edu.cn