CE Linux Forum

Korea Tech Conference

2005년 5월 14일, 서울

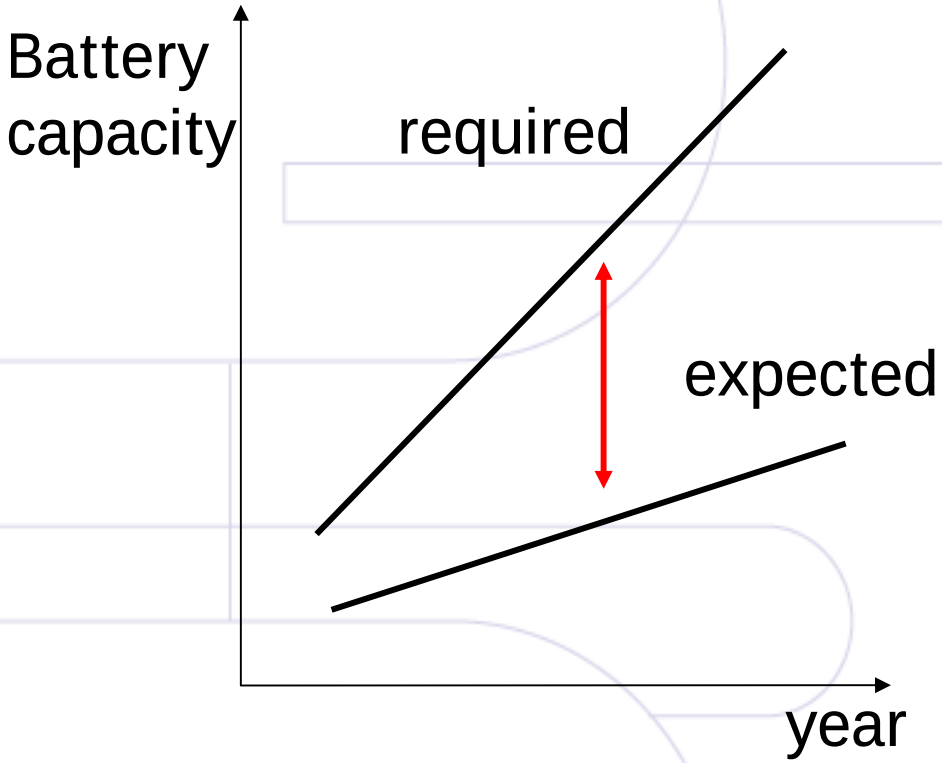
CELF Power Management Spec. 기반 저전력 Linux Platform 개발

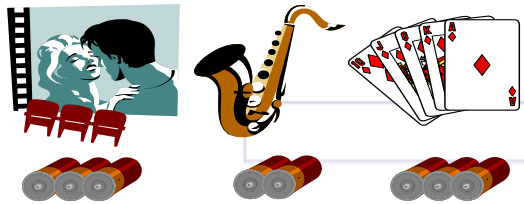
장민성
삼성전자

차례

- Introduction
- Power Management on Linux
  - CELF Spec. Based PM
    - Dynamic PM
    - Static PM
  - Variable Scheduling Time Out
- Conclusion

Introduction – Why Low Power ?





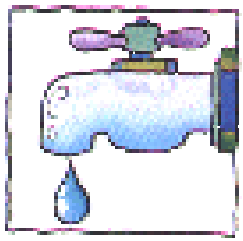
Introduction – Software PM

- Software PM에 대한 2가지 오해
  - 구현만 하면 어떤 Hardware Platform에서도 큰 효과
  - 하나의 기술만 잘 구현해도 큰 효과 기대 가능

- Software PM의 진실

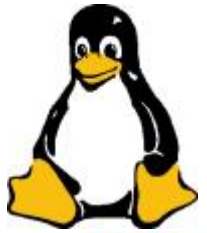
Software PM의 Magic이 아니라
Chip 이나 Hardware Platform의
전력 관리기능을 "상황"에 맞게 최적화

단위 Hardware Component에
PM기능에 의존적



Software PM은 "티끌 모아 태산"

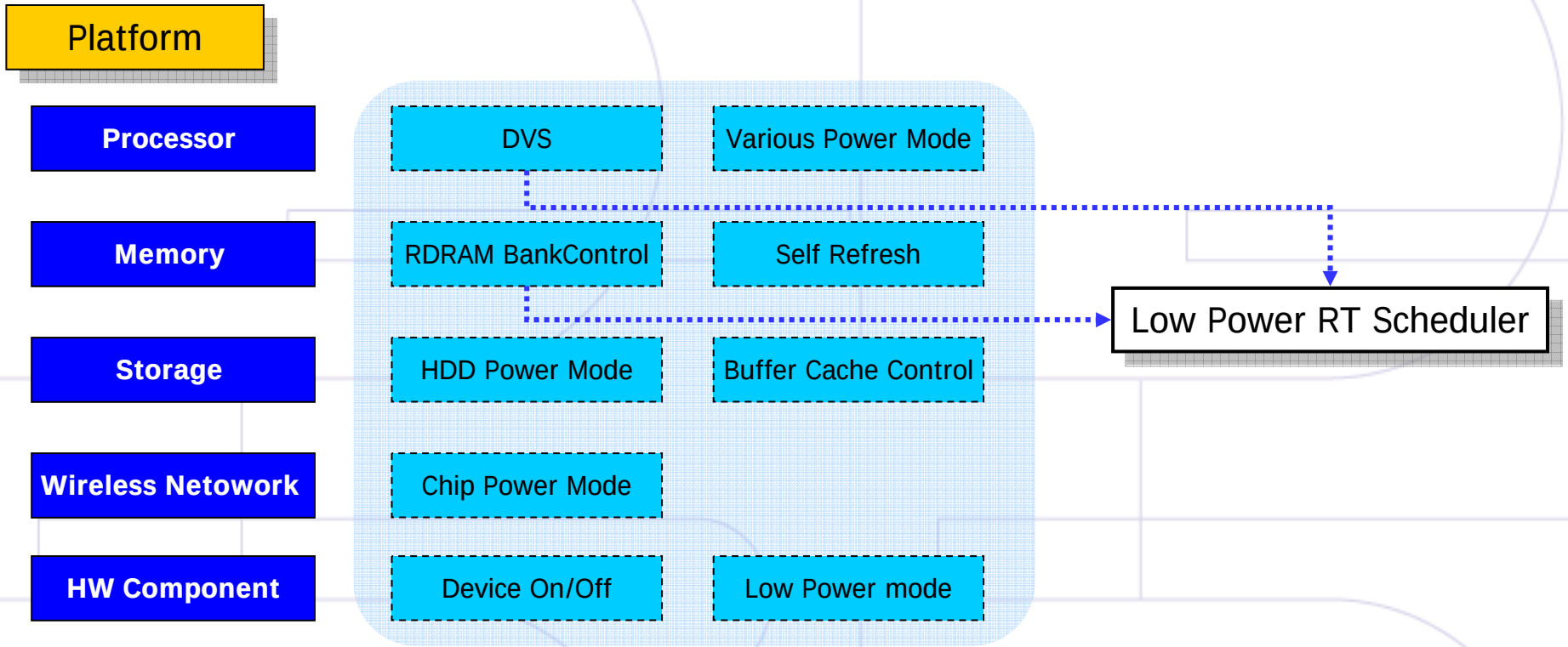
Introduction – Why OS Support for PM?



- OS time base may be affected by Frequency scaling
  - *clock APIs, Clock functions, periodic functions, timeouts*
- Some drivers is affected by power events
  - *e.g., serial port affected by frequency scaling*
- Some drivers should know of power events
  - *e.g.,, to put peripheral in low power mode when system is entering a deep sleep state.*
- Scheduler effectiveness impacted by Frequency scaling
- Central coordinator for processing power envents
- Central Voltage/Frequency control to avoid unsupported V/F setpoints and over clocking
- Appropriate idling to avoid unnecessary blocking

(from esc class #209)

Introduction – OS Level Power Management

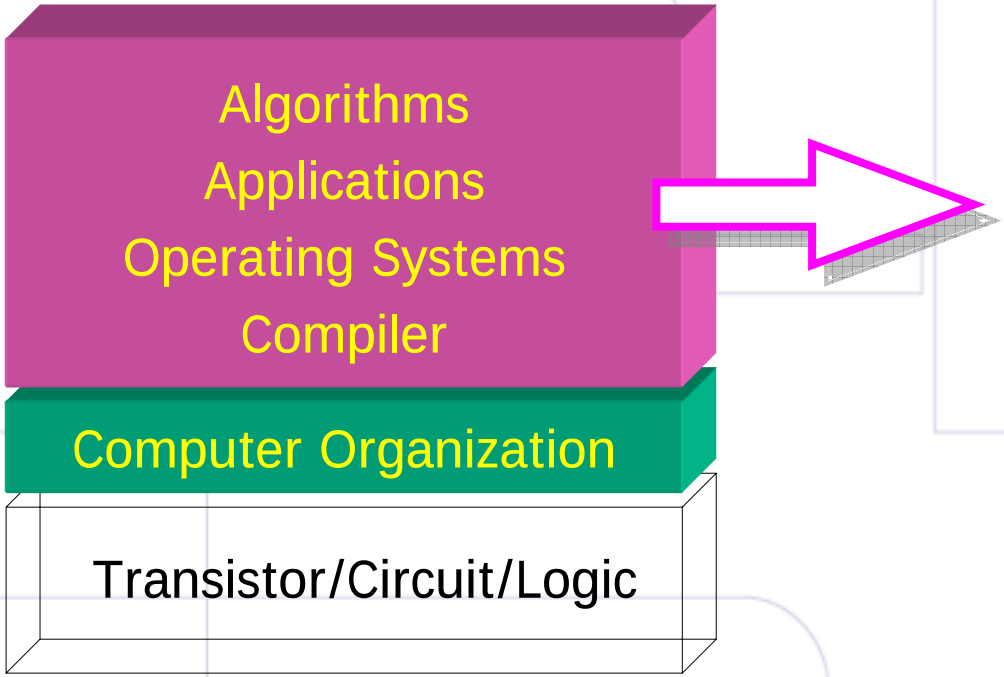


*** Hardware의 Low Power특성을 OS의 Subsystem별 동작특성과 조합**

Introduction – OS PM Techniques

- Idle clocks when not needed
- Activate built-in peripheral low power modes
- Leverage activity detectors for auto savings
- Adjust F/V based upon activity
- Dynamically schedule F/V to accommodate predicted work load
- Use low-power code sequences / data patterns
- Optimize speed to max. idle time
- …

Conclusion



SW Techniques

- 많은 Power Management 기술은 HW, SW의 Hybrid Approach 필요

Linux Power Management based on CELF Spec. 1.0

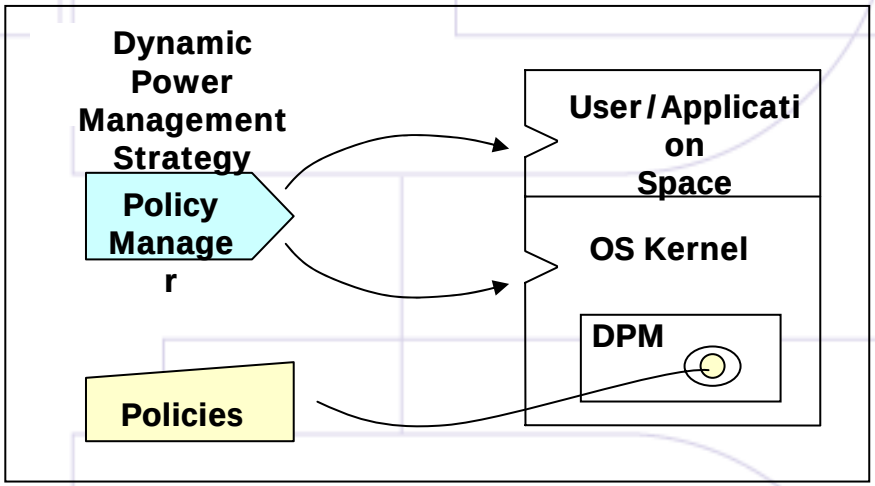
CELF PM Spec. 기반 PM

- CPU 동작 상태에 따른 분류
  - Static Power Management
    - Goal : Leakage 전류에 의한 Power Consumption 관리
    - $P_{static} \sim V \times I_q$
  - Dynamic Power Management
    - Goal : Workload에 따라 Dynamic Power Consumption 관리
    - $P_{Dynamic} \sim C \times f \times V^2$

- OS PM을 위해 요구되는 CPU 기능
  - Static Power Management
    - Deep Sleep mode 지원 - System stand-by時 Static Power 또는 leakage current최소화
    - Deep Sleep mode時 External voltage supply 최소화
  - Dynamic Power Management
    - CPU 내부 Block의 Automatic clock gating
    - Unused Block의 Manual Clock Gating
    - Clock frequency를 동적으로 변환 가능 (based on MIPS or Bandwidth)
    - Max. Clock Frequency 불 필요시 외부전압공급 최소화 (Dynamic voltage scaling)

DPM – Architecture Overview

- Dynamic Power Management on Linux
  - Linux Framework Developed by MontaVista, IBM, CELF
  - Running State 일때, processor의 voltage와 frequency를 변경하여 power consumption을 조절
  - *Function:*
    - Scaling of CPU frequency and voltage
    - Suspend and resume of CPU
    - Suspend and resume of devices
    - Power policy manager application



- 구성요소
  - operating point
  - operating state
  - policy
  - policy manager

DPM – Policy Architecture (1/4)

- Operating Points
  - 특정 시간에 시스템이 동작하는 상태
  - Core voltage, CPU와 bus의 frequency, device의 상태등

※ The Operating Points for Samsung S3C2440A

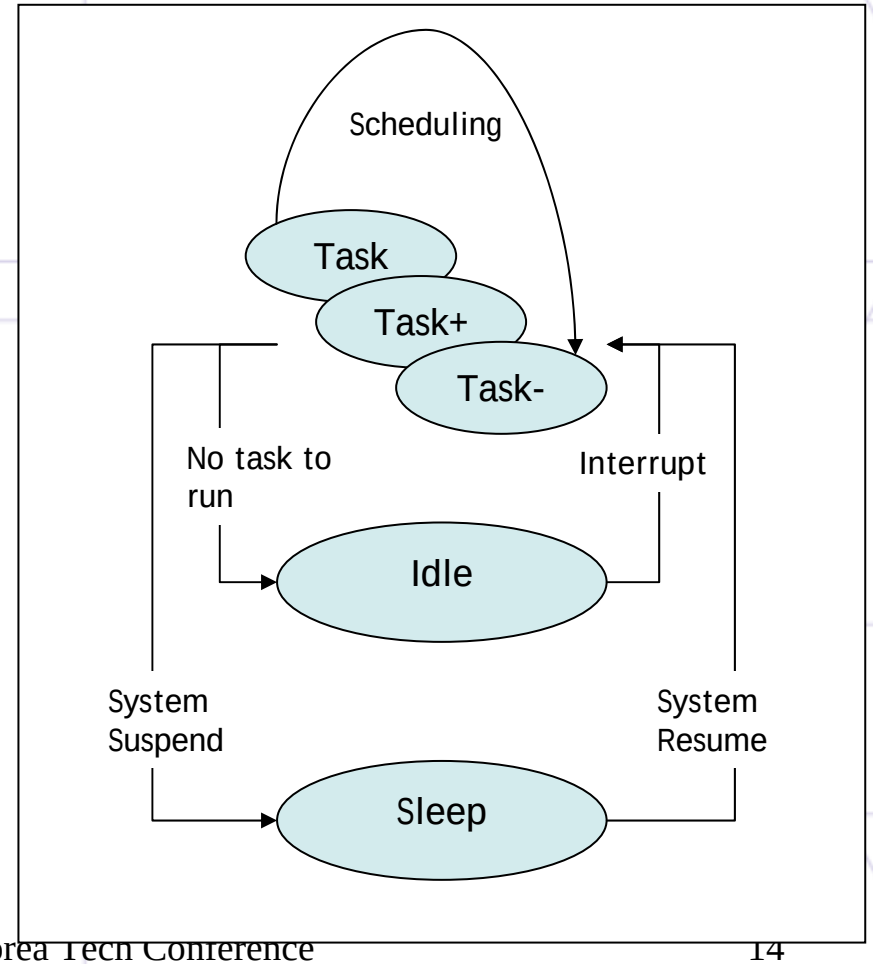
Operating point	406	266
Core Voltage	1.2V	1.2V
CPU frequency	406MHz	266MHz
Bus frequency (HCLK/PCLK)	133MHz/66MHz (1:3:6)	133MHz/66MHz (1:2:4)

DPM – Policy Architecture (2/4)

- Operating States
  - OS가 동작하는 system state
  - Running State :
    - task+, task, task- 로 세분화

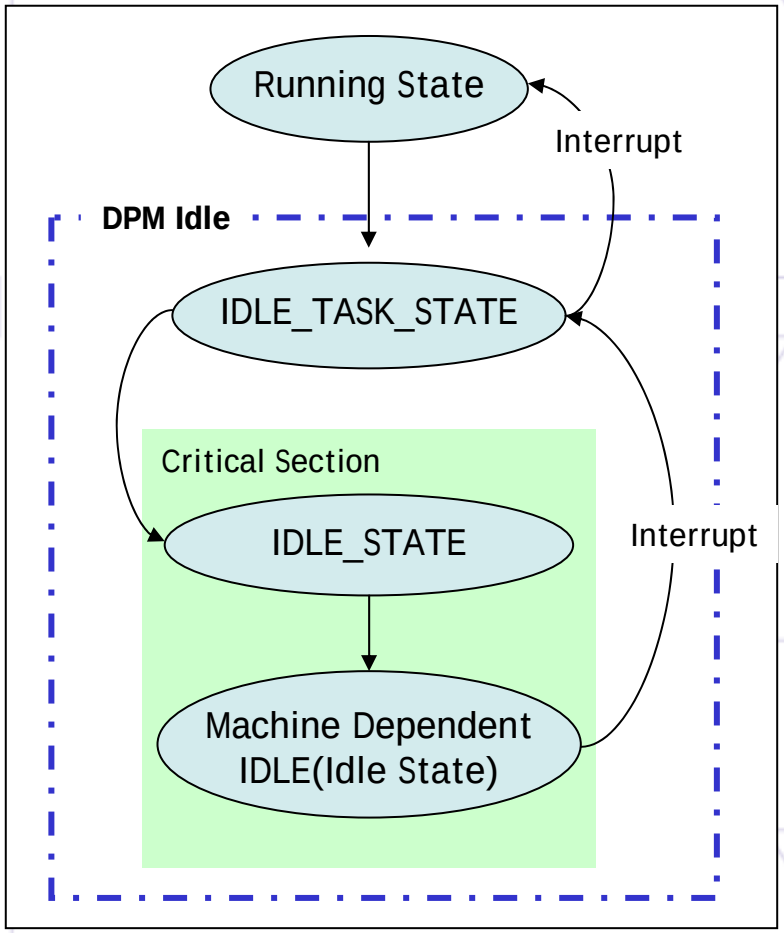
s3c2440_dpm.h

```
#define DPM_STATE_NAMES
{ "relock", "idle-task", "idle", "sleep",
  "task-4", "task-3", "task-2", "task-1",
  "task",
  "task+1", "task+2", "task+3", "task+4"
}
```



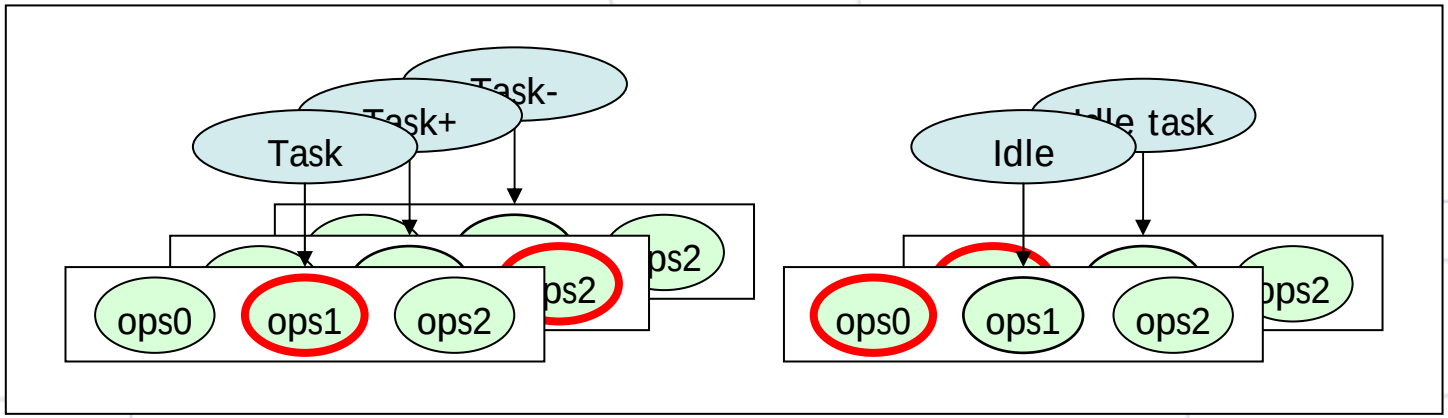
DPM – Policy Architecture (3/4)

- DPM Idle
  - 일반적인 CPU Idle을 사용할 경우 Interrupt Latency가 길어지는 문제를 개선
  - Idle Task State: Running state와 Idle state의 중간단계의 frequency로 동작
  - Idle State: Idle Task State에서 진입 Ultra-low-power Operating Point
  - Interrupt Latency는 Idle Task State가 Idle State보다 짧다



DPM – Policy Architecture (4/4)

- Policy
  - DPM 구조의 최상위 단계
  - 각 operating state를 operating point의 class에 mapping

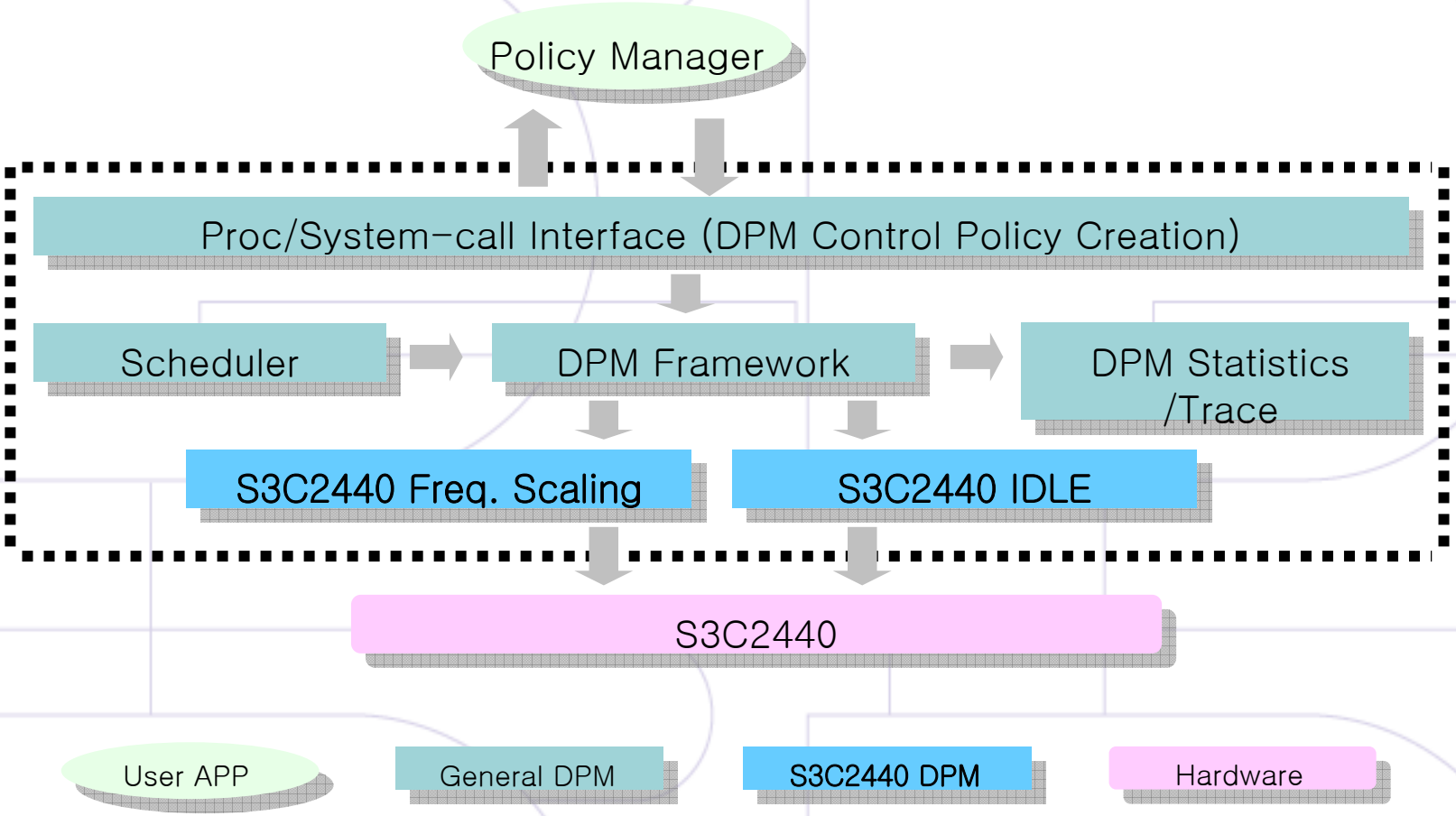


- Policy manager
  - 한 시스템에 여러 개의 policy가 있을때, 시스템의 상태에 따라서 적합한 policy를 선택하고 적용한다

※ A task-state specific, three-policy strategy

Policy	TASK+	TASK	TASK-	IDLE-TASK	IDLE
High	406Mhz	406Mhz	266Mhz	133Mhz	48Mhz
Middle	406Mhz	266Mhz	133Mhz	133Mhz	48Mhz
Low	266Mhz	266Mhz	133Mhz	133Mhz	48Mhz

Dynamic Power Management for S3C2440



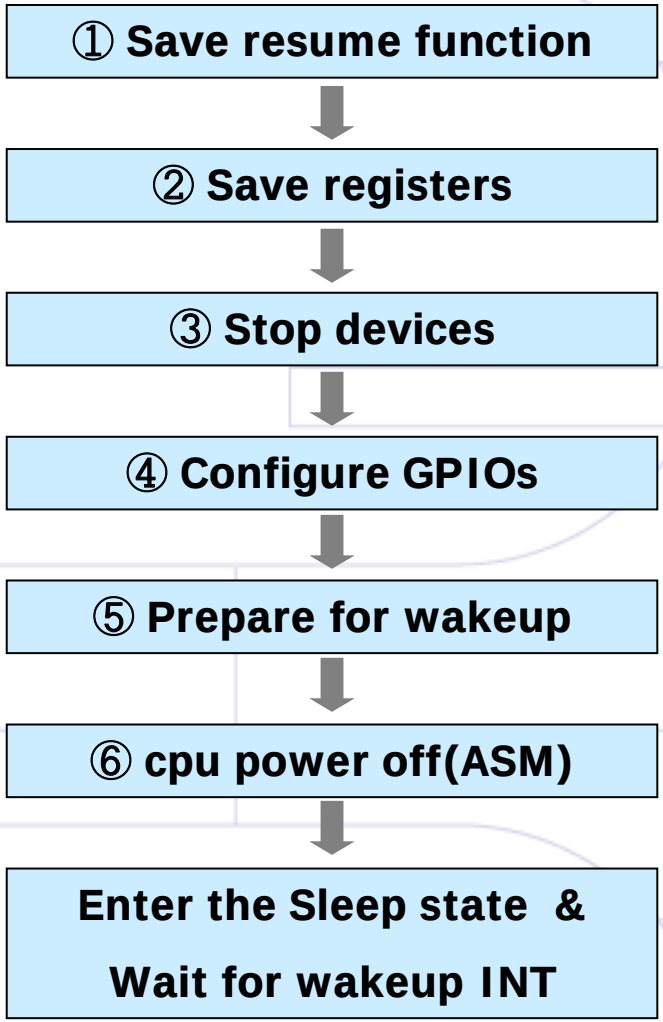
Static PM

- APM (Advanced Power Management)
  - by Intel, Microsoft
  - Intel Architecture를 사용하는 PC의 BIOS에 기능 내장
  - 일반적인 Embedded System에서는
    - i386 계열의 CPU를 사용하지 않는 경우가 많으므로 사용불가
    - APM과 같은 역할을 하는 Daemon을 사용해서 BIOS의 기능 처리

- CELF based Static PM
  - Define Spec. by CELF (MontaVista, Samsung Electronics)
  - Suspend / Resume

Suspend - Entering Sleep state

- sleep bit을 set 하고 , resume function address를 저장
- system 동작에 필요한 register 저장
  - Sleep state에서 빠져나왔을 때 system setup에 필요한 register
- stop 시킬 수 있는 device를 끈다
  - LCD / DMA / I2C bus controller etc.
- GPIO의 output을 설정한다
  - Sleep state때의 GPIO output을 설정
- wakeup INT를 setup한다
  - wakeup resource 설정
- CPU를 Sleep 시키는 레지스터에 값을 써서 cpu를 off



Resume - Exiting Sleep states

① External INT가 들어오면 bootloader를 시작하여, 초기화 후에 저장된 resume function 으로 jump

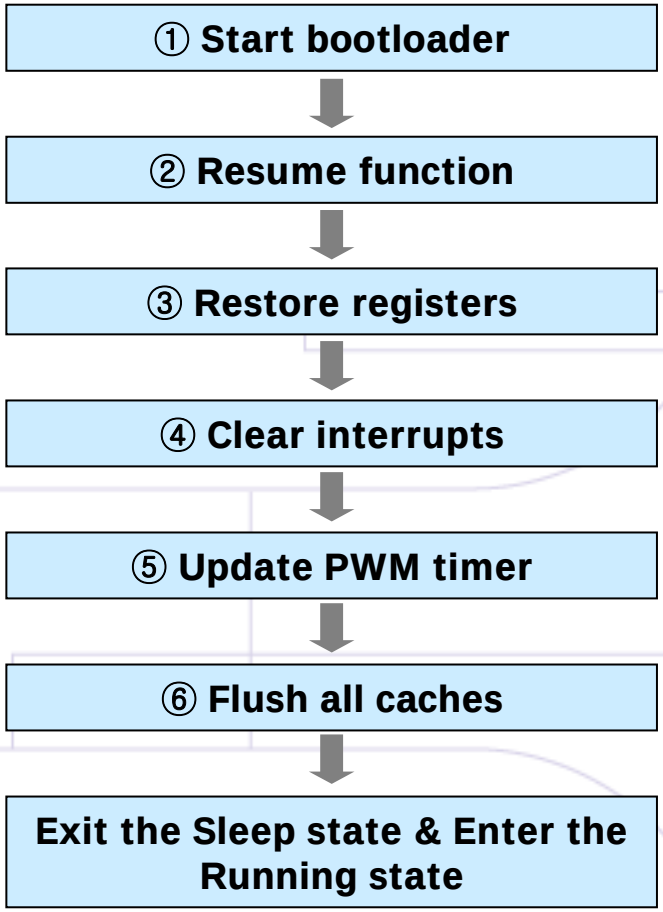
② ARM state register를 복원하고, Sleep state로 들어온 직후의 code로 돌아간다

③ suspend시 저장한 register를 복원

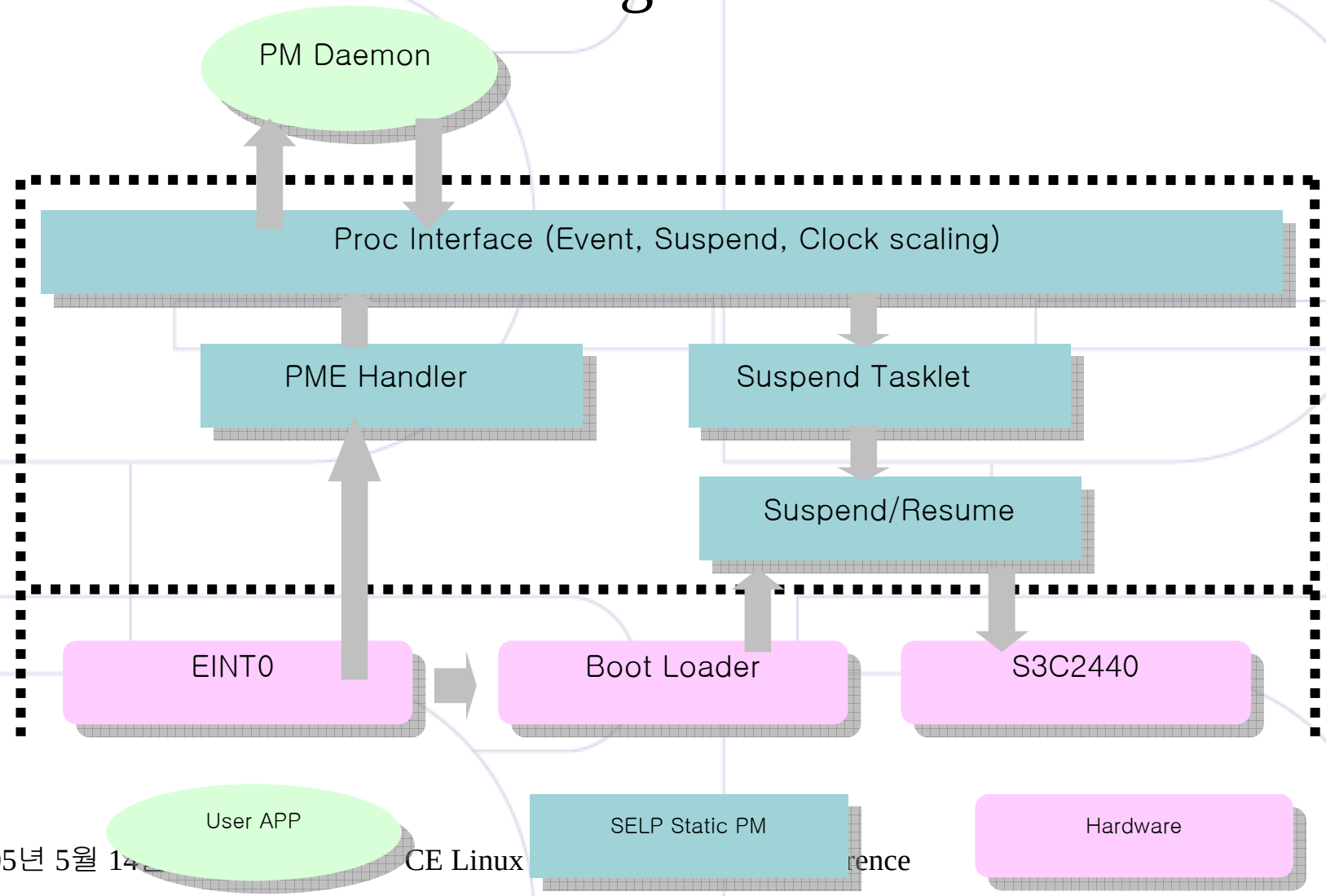
④ interrupt pending register를 clear

⑤ PWM timer 를 update

⑥ 모든 cache line을 clear 하고 invalidate 시킨다.

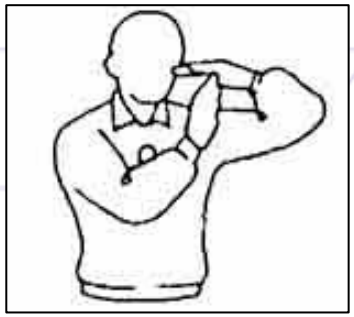


Static Power Management for S3C2440



Variable Scheduling Timeout (1)

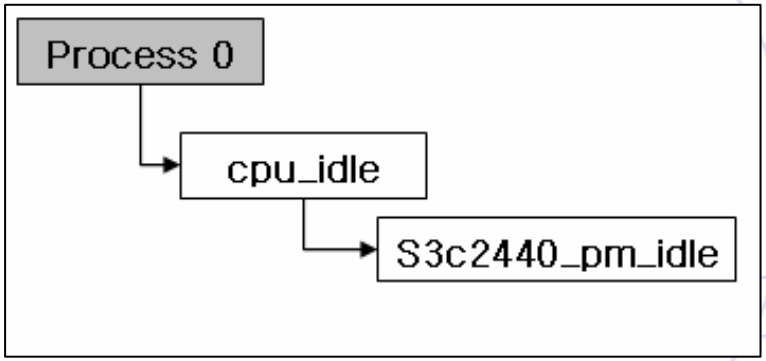
- Goal
  - CPU의 동작이 필요없는 경우, 저전력 모드에 잔류하는 시간 최대화
- 배경
  - CPU가 저전력 모드(IDLE mode) 체류 시간이 길수록 저전력에 유리
  - Linux 는 실제 CPU 사용이 없어도 10ms마다 Kernel Tick Interrupt가 발생
  - Kernel Tick Interrupt에 의한 불필요한 CPU wakeup를 막아, 저전력 mode상태유지

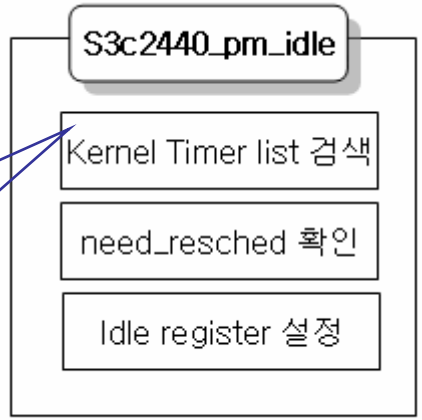


- Purpose
  - CPU IDLE mode 진입시 Tick Interrupt 금지
  - IRQ 발생시 CPU NORMAL mode 복귀
  - NORMAL mode 복귀시 Idle Time동안의 elaped Time을 update
  - Kernel timer List에 존재하는 timer들의 expire time준수
  - User mode에서 쉽게 VST 작동 상태를 파악할 수 있는 Proc Interface 제공

Variable Scheduling Timeout (2)

- Idle mode 진입
  - Linux idle process를 Processor의 idle mode진입으로 변경
  - idle mode 진입전 Kernel Timer등 확인
    - Timer List확인결과로 idle 진입여부 결정





400 <= *timer→expire*	NO_NEXT_TIMER
100 <= *timer→expire* < 400 jiffies	LONG_NEXT_TIMER
0 <= *timer→expire* < 100 jiffies	SHORT_NEXT_TIMER

결론

- 결론
  - 효과적인 PM은 Software 및 Hardware를 동시에 접근하는 Hybrid전략필요
  - Hardware상의 PM 기능의 OS Mapping이 중요

- 참고 Site
  - http://www.celinuxforum.org
  - http://sourceforge.net/projects/dynamicpower/

Thanks !