ECE5461:
Low Power SoC Design

Tae Hee Han: than@skku.edu

Semiconductor Systems Engineering

Sungkyunkwan University

System-level Power Optimization: Introduction

Low Power Design Issues Impact Profitability

- For different drivers in different verticals

Mobile/Hand-held

Battery Life
Unit Cost (chip package)

Consumer/Digital Home

Unit Cost (chip package)
Unit Cost (fans etc.)
Reliability

Network/Data Center

Power Efficiency
Total Cost of Ownership
Reliability
Green

Low power requirements drive different design decisions:

- Product design architecture and integration decisions
- IP make versus reuse versus buy decisions
- Manufacturing process decisions

Need for Energy Management

- Today's mobile consumers want:
  - longer battery life *and*
  - smaller, lighter products
- Manufacturers are adding new features and applications to add product appeal:
  - Media players (audio, video)
  - Gaming
  - Video capture

- Increasing processing power requirements and longer battery life are conflicting requirements
- Battery technology alone offers only incremental improvement over the next several years

Developing Low Power Strategies

- A system definition is an important step in every embedded system design

- Three elements of a power efficient system definition are:
  - Making a power budget – Determine the maximum amount of power that can be supported by the power source
  - Identifying and analyzing required system tasks – Determine the tasks that need to be performed by the system and estimate the power required to perform these tasks
  - Developing a low-power strategy – Having a strategy for low power helps the power needed to perform the required system tasks to fit within the power budget

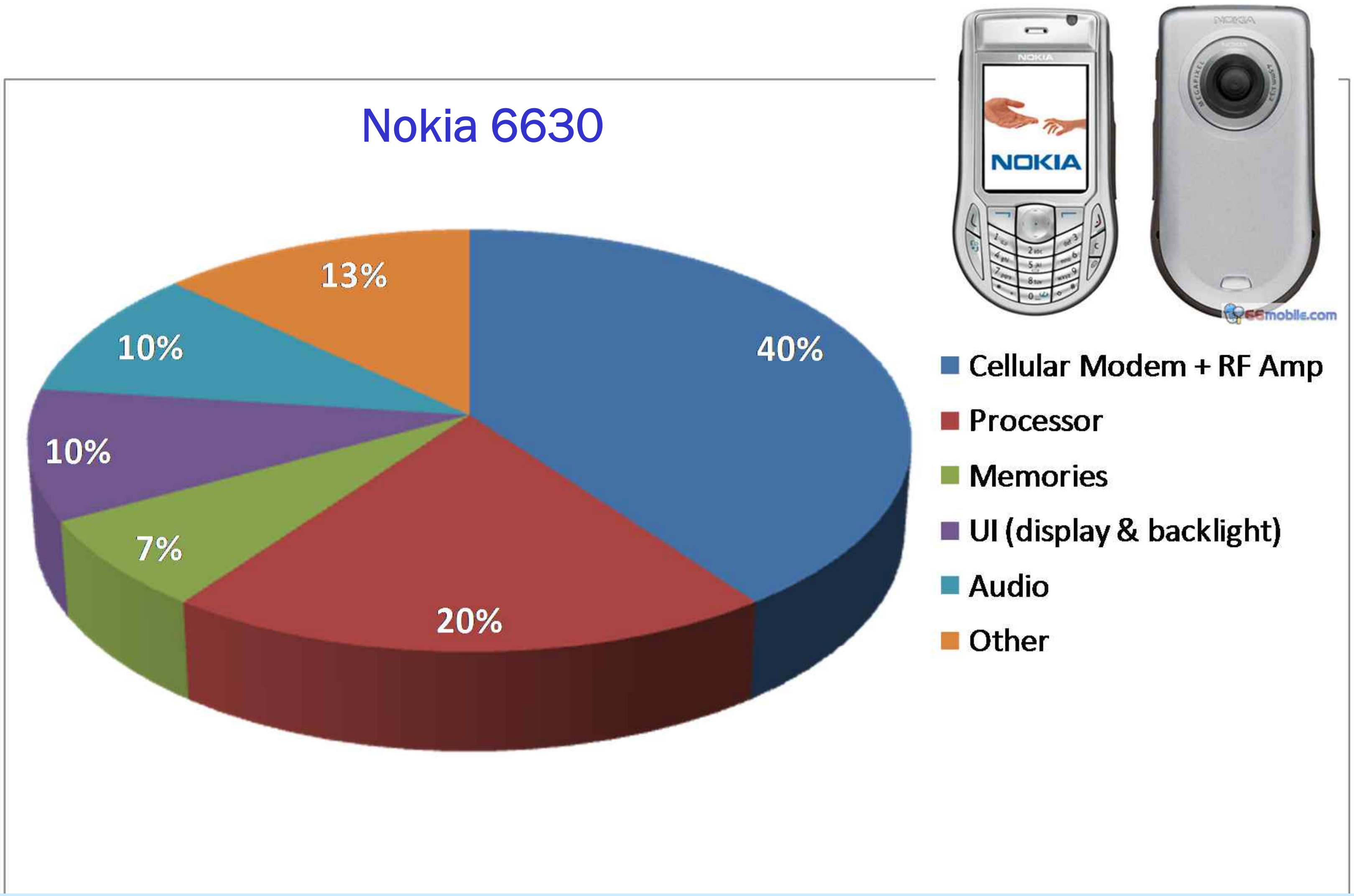
Making a Power Budget

- What is a Power Budget?
  - A quick first order calculation that gives you a ballpark figure of the “total average current” supported by your power source
- Required information for making a power budget:
  - How long must the system operate without replacing batteries?
    - Obtain from System Specification
  - How much capacity can I expect from my battery?
    - Obtain from battery data sheet
  - Power budget calculation:

$$\text{Maximum Average Current [mA]} = \frac{\text{Battery Capacity [mAH]}}{\text{Required Battery Life [H]}}$$

Power Budget in Mobile Device (Source: iSuppli, 2006)

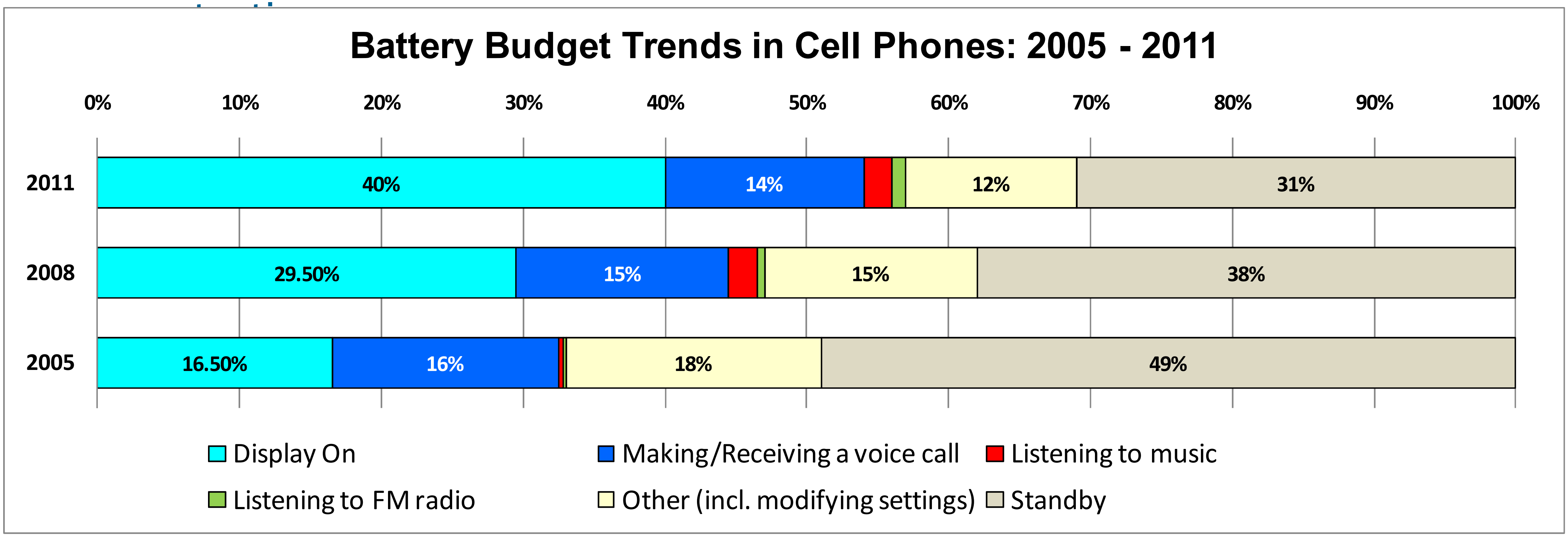
Nokia 6630



Power consumption for video conferencing in the Nokia 6630.
Constant average 3W can be tolerated without uncomfortable heating, but smaller products may have power budgets closer to 2W.

Power Budget in Mobile Device (Source: Strategy Analytics, 2007)

The Battery Budget Model continues to face challenges with growing power consumption, handset usage and feature

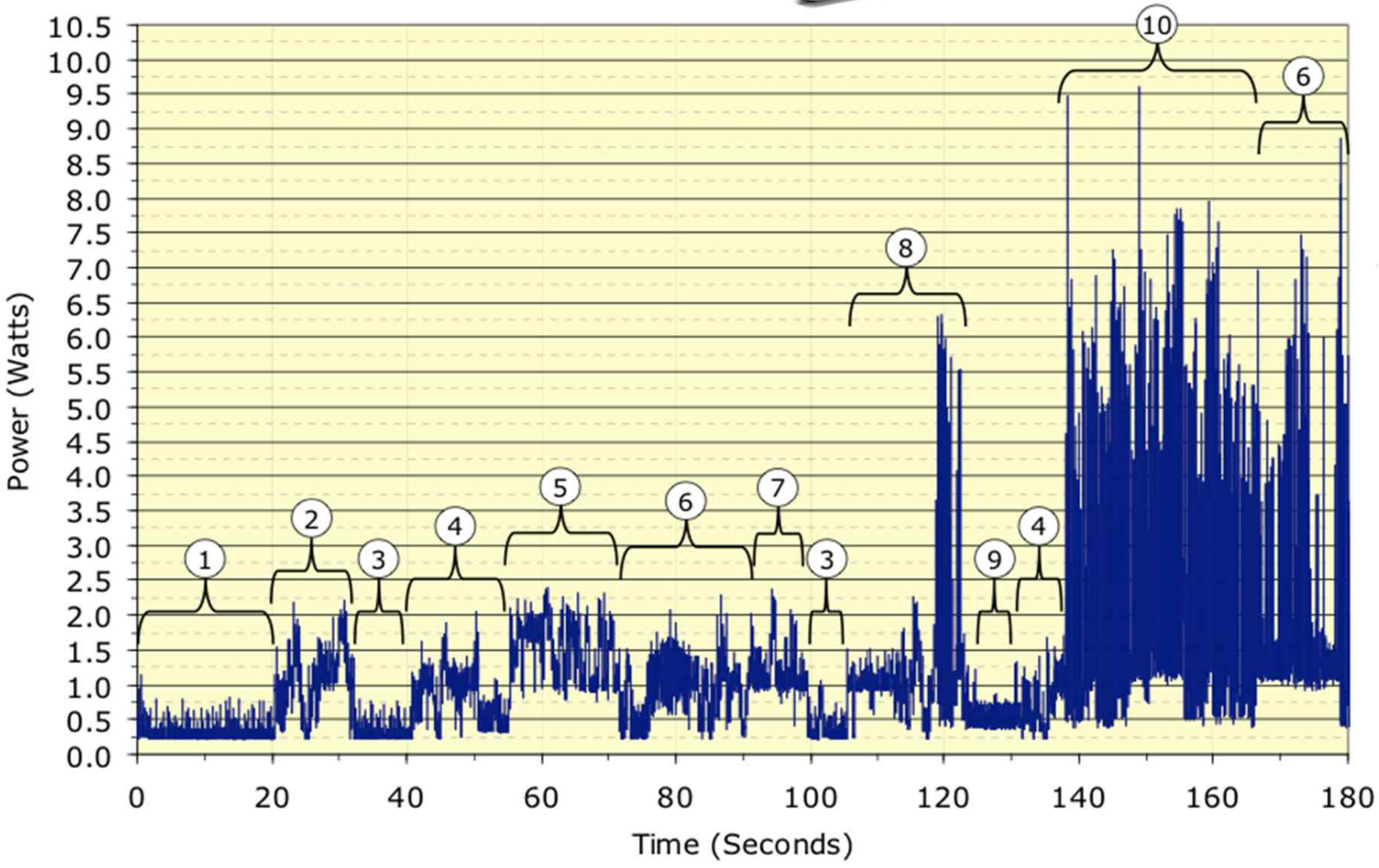


□ "Display On" Activities

- Watching broadcast TV
- Browsing the web with 3G+
- Sending/receiving SMS
- Making/receiving a video call
- Reading/sending Emails
- Taking/viewing a photo
- Taking/viewing a video
- Playing games

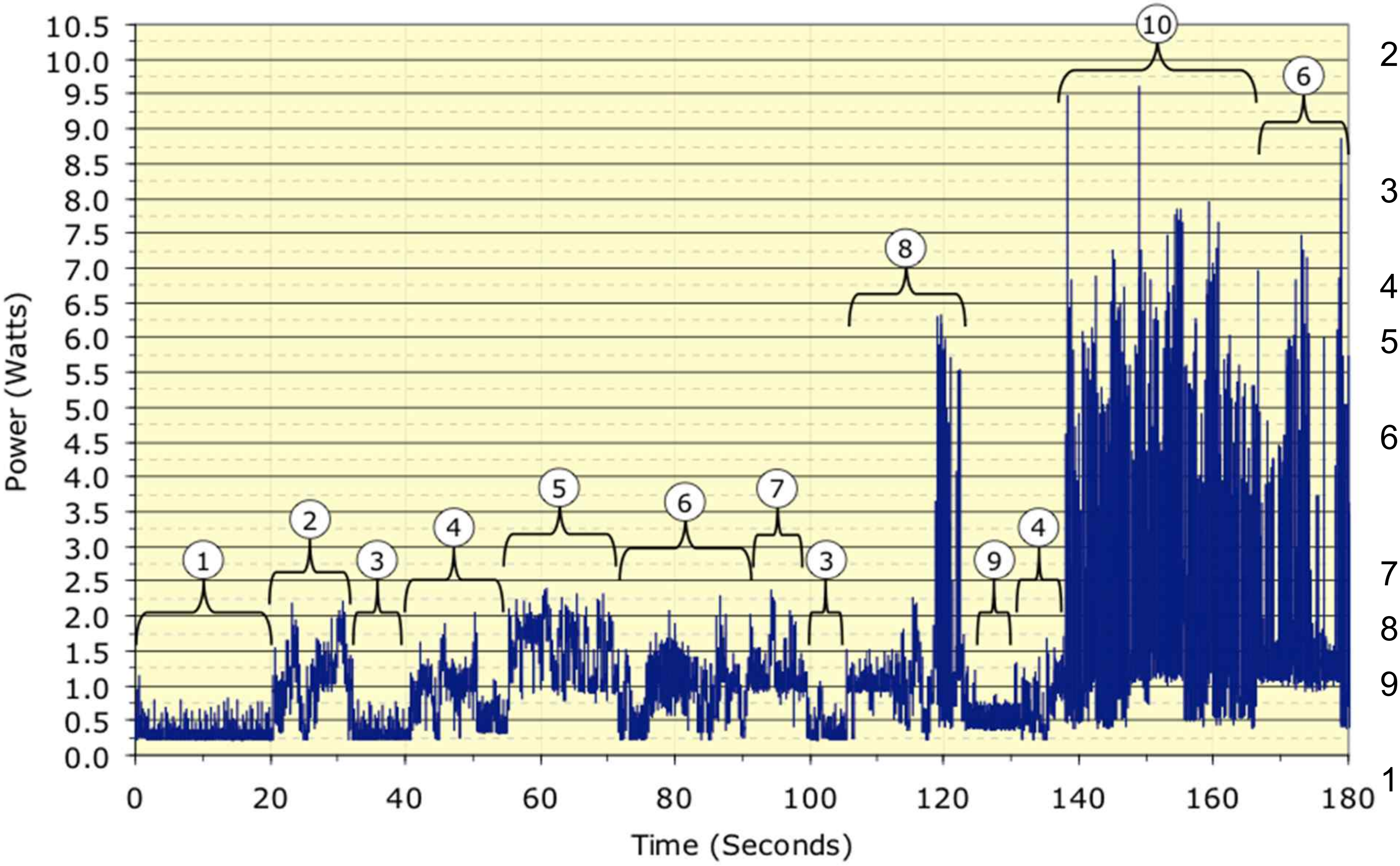
Power Budget in Mobile Device (Source: Nokia, 2010)

N900 power usage



1. Sitting idle at the home screen. (**0.29 W**)
2. Turning on WiFi
3. Sitting idle at the home screen with WiFi on. (**0.29 W**)
4. Navigating to the web browser.
5. Surfing Internet via WiFi.
   (Avg. = **1.48 W**)
   (Peak = **2.39 W**)
6. Zooming in and out.
7. Navigating back to home screen.
8. Turning off WiFi and turning on Cellular Data. Tower search occurs. (Peak = **6.35 W**)
9. Sitting idle at the home screen with Cellular Data on. (**0.51 W**)
10. Surfing Internet via Cellular Data.
   (Avg. = **1.60 W**)
   (Peak = **9.61 W**)

Power Budget in Mobile Device (Source: Nokia, 2010)



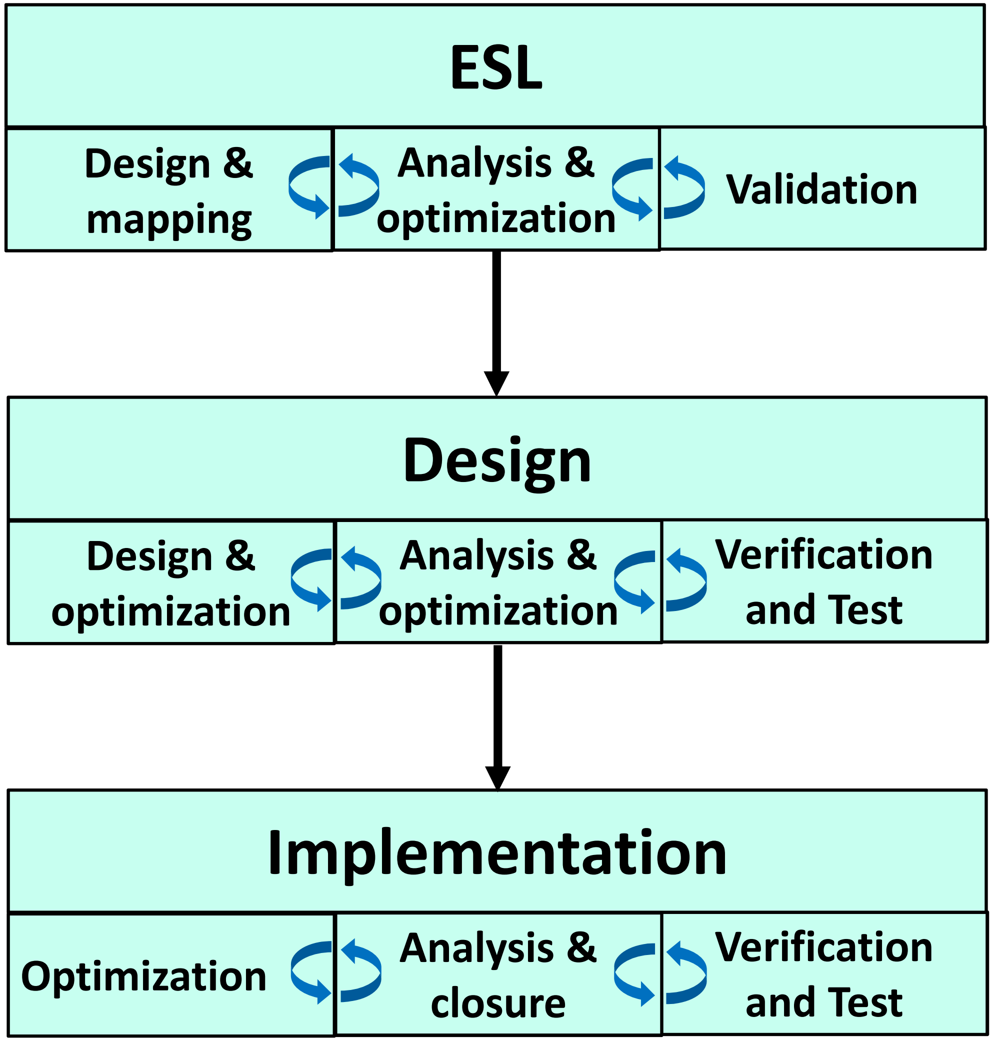
1. Sitting idle at the home screen. (**0.35 W**)
2. Pressing the camera key to get into camera menu. (Peak = **2.16 W**)
3. In video mode with an active viewfinder. (**1.07 W**)
4. Recording video. (**1.62 W**)
5. Zooming in while recording video. (**1.83 W**)
6. Covering light sensor, causing screen to dim while recording. (**1.58 W**)
7. Stopping video.
8. Saving video. (**1.04 W**)
9. Navigating to play back previously recorded video.
10. Playing back video. (Avg. = **0.99 W**) (Peak = **2.35 W**)

Power Efficiency: Three Scopes of Activity

- 3 levels addressed concurrently:
  - System: System-level issues, e.g., system architecture, software, power supplies, power distribution, data compression, density/real-estate, power monitoring and control
  - SoC Design: SoC - level issues, e.g., architecture, systems/applications to circuits, design methodologies, design & simulation tools
  - Silicon & Technology: Device-level issues, e.g., process technologies, libraries, memories, design IP blocks, modeling tools, design flows, packaging

Three Phases of the Power Aware Flow

- ESL (algorithm & system models)
  - HW / SW (and/or FW) partitioning
  - Architecture
  - TLM (Transaction-level Modeling)
- Design
  - RTL module design/selection
  - IP selection and chip integration
- Implementation
  - Synthesis
  - Physical design


ESL

Design & mapping

Analysis & optimization

Validation

Design

Design & optimization

Analysis & optimization

Verification and Test

Implementation

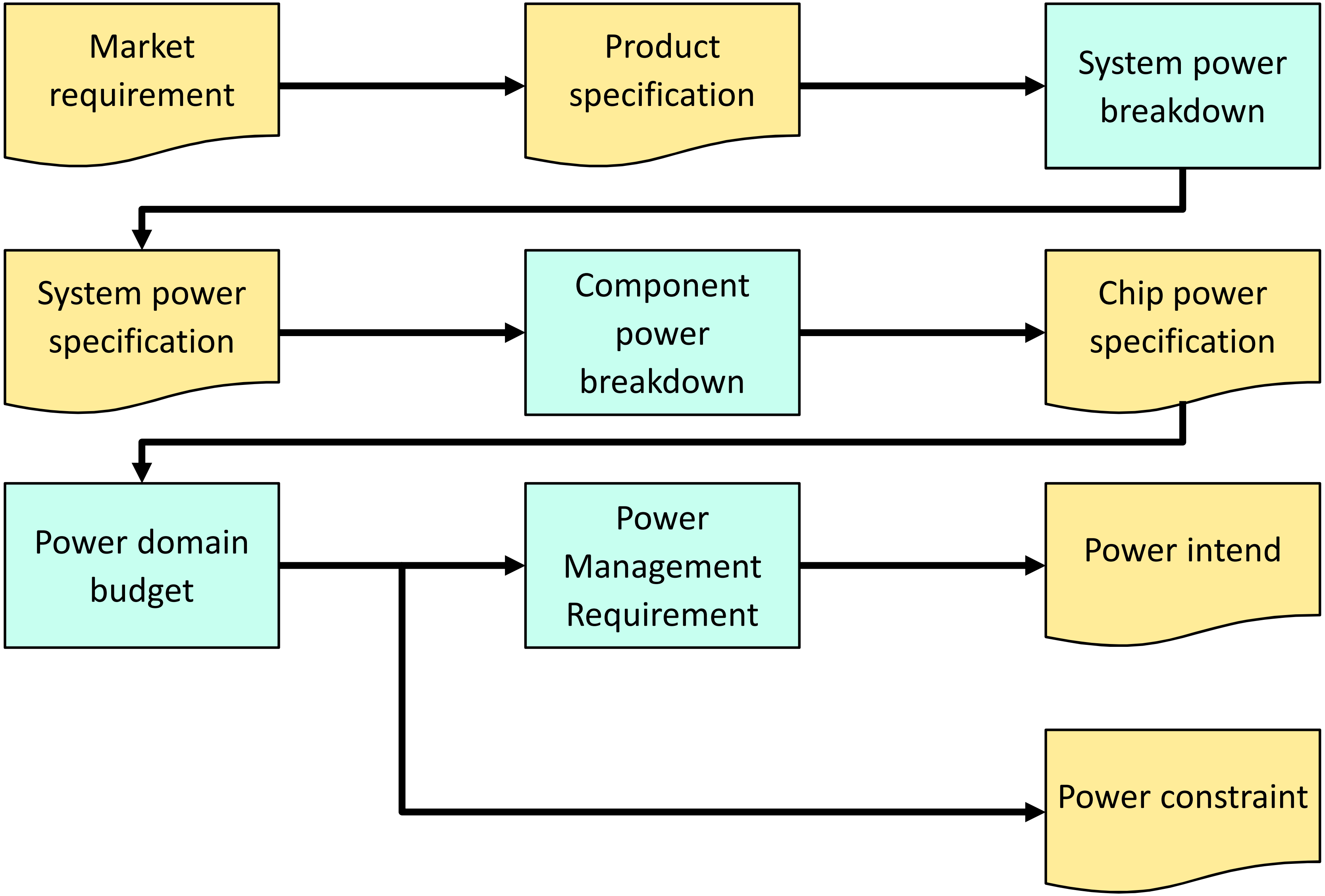
Optimization

Analysis & closure

Verification and Test

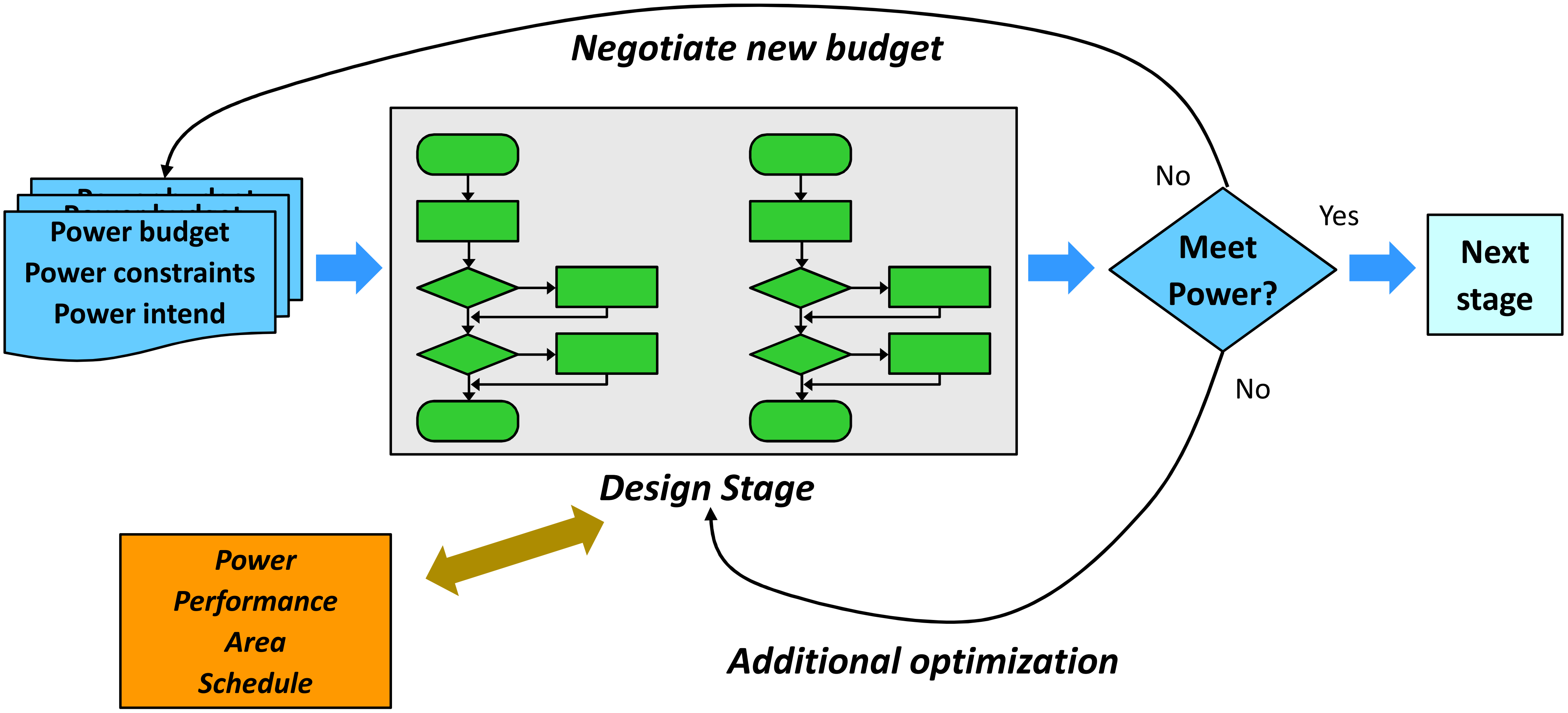
Source: Si2 (2008)

Power Budgeting

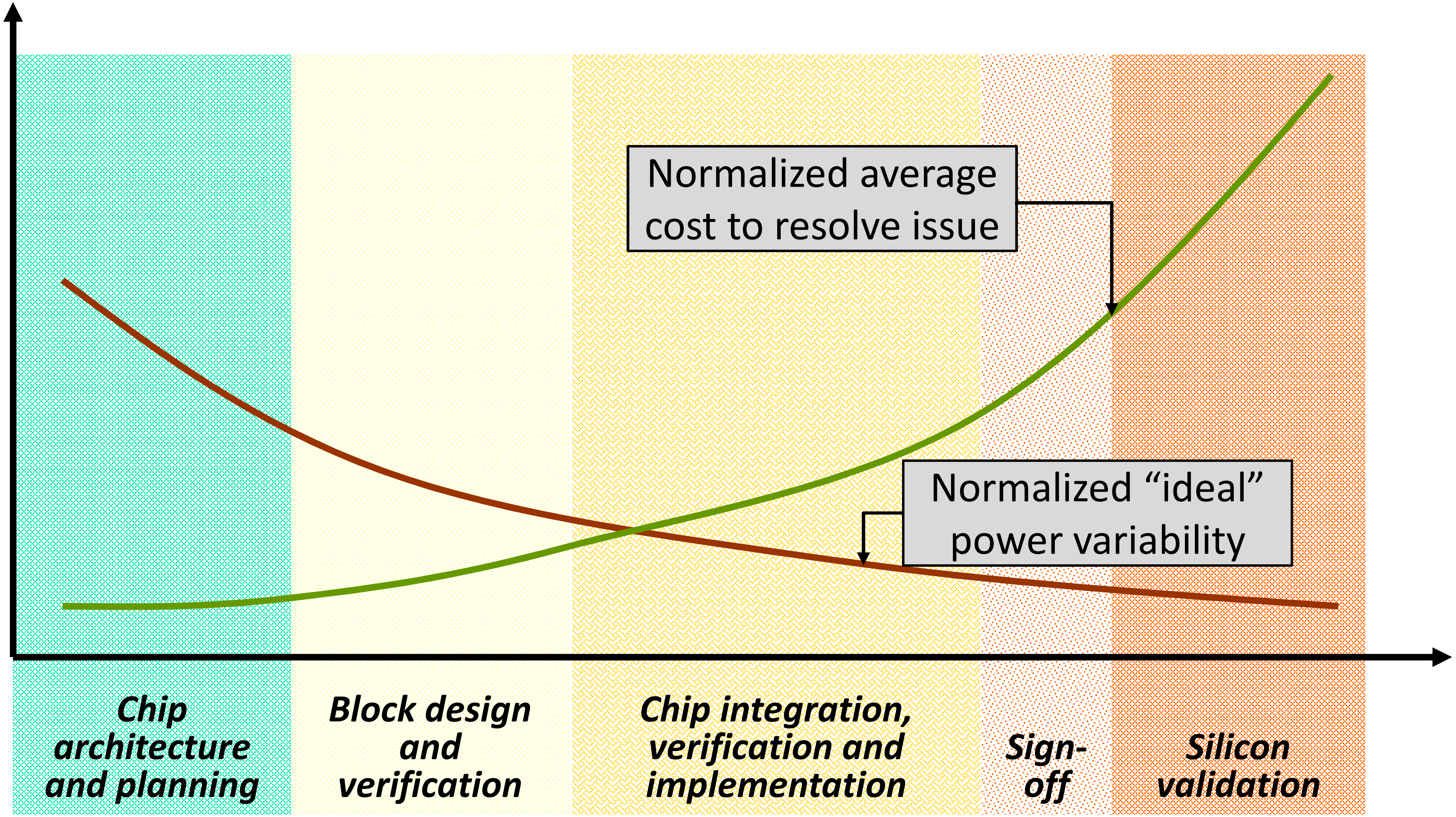


Power Closure

Develop power budget and intend.
Predict power accurately.
Signoff power at each design flow stages.



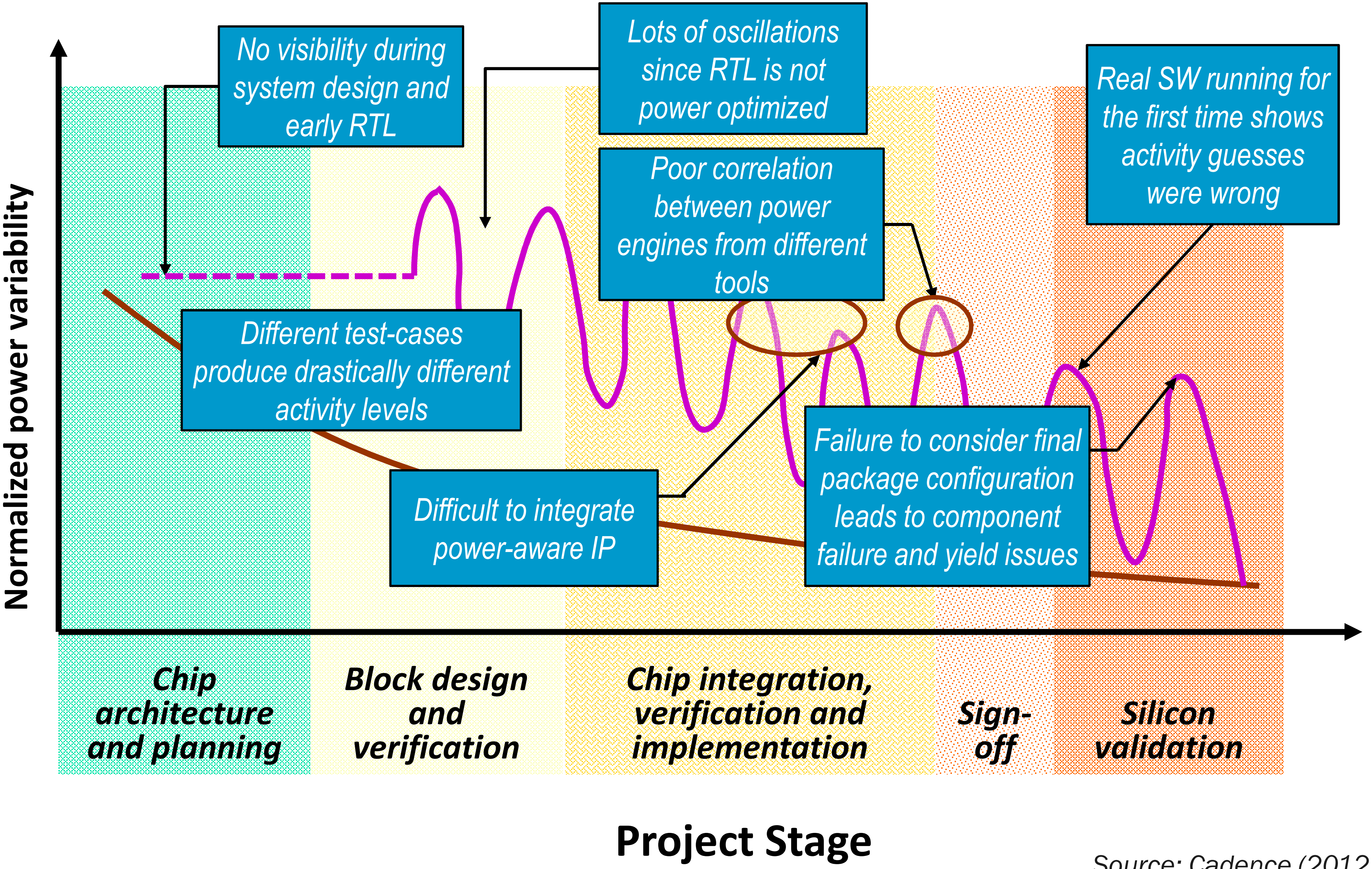
Power Closure in Design Flow



Project Stage

Source: Cadence (2012)

Power Closure Challenges



Normalized power variability
Chip architecture and planning
Block design and verification
Chip integration, verification and implementation
Sign-off
Silicon validation
Project Stage

No visibility during system design and early RTL

Lots of oscillations since RTL is not power optimized

Poor correlation between power engines from different tools

Real SW running for the first time shows activity guesses were wrong

Different test-cases produce drastically different activity levels

Difficult to integrate power-aware IP

Failure to consider final package configuration leads to component failure and yield issues

Source: Cadence (2012)

Power Closure Objective

- ESL phase
  - Meeting marketing objectives for system level power and develop power budget and power intend for subsequence design phases
    - IP component and Process selection
    - Power management strategies
    - Hardware and firmware algorithm partition
- Design phase
  - Meeting chip level power budgets and develop power constraints
    - RTL power optimization with targeted mode based budgets
    - Power intent and power constraints
    - Develop power coverage requirement
- Implementation phase
  - Power and IR signoff based on budgeted constraints
    - Power intent physical design
    - Power optimization
    - Power signoff

Power Closure Challenges

- Power metrics –what is important?
- Power analysis accuracy and consistency across all phases of design flow
- Power analysis needs combinations of spatial and temporal information
- Power models are complex
  - Complex operating modes
  - Temperature dependency of leakage models
- Good power vectors are difficult to generate and simulation time can be lengthy because it is operational data dependent

Power Models used for System-level Analysis

- Processor and Peripherals
  - Mode or state based
  - Function-level Macro-models
  - Analytical, Instruction-class based
  - Instruction-level
  - Pipeline state aware
  - Cycle-accurate functional level
  - Register-transfer level (structural)
  - Gate-level (structural)

- Interconnect
  - Transaction based analytical (temporal)
  - Lumped capacitance (structural)
  - Distributed RC models (structural)

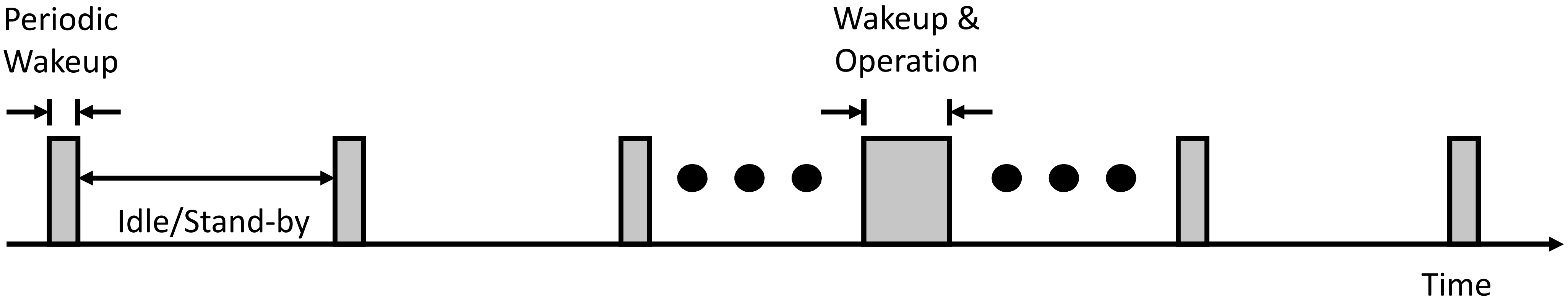
- Custom Special Components
  - Analytical, access-statistics based (temporal) [Caches]
  - Structure-aware, access level [Caches]

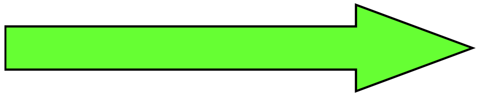
System-Phase Low Power Design

- Primary objectives: minimize f_{eff} and V_{DD}
- Modes
  - Modes enable power to track workload
  - Software programmable; set / controlled by OS
    - Hardware component needed to facilitate control
    - Software timers and protocols needed to determine when to change modes and how long to stay in a mode
- Parallelism and Pipelining
  - V_{DD} can be reduced, since equivalent throughput can be achieved with slower speeds
- Challenges
  - Evaluating different alternatives

System Level Consideration for Low Power Design

- Mobile Device's Behavior according to Time (Operation Time is less than 10%)





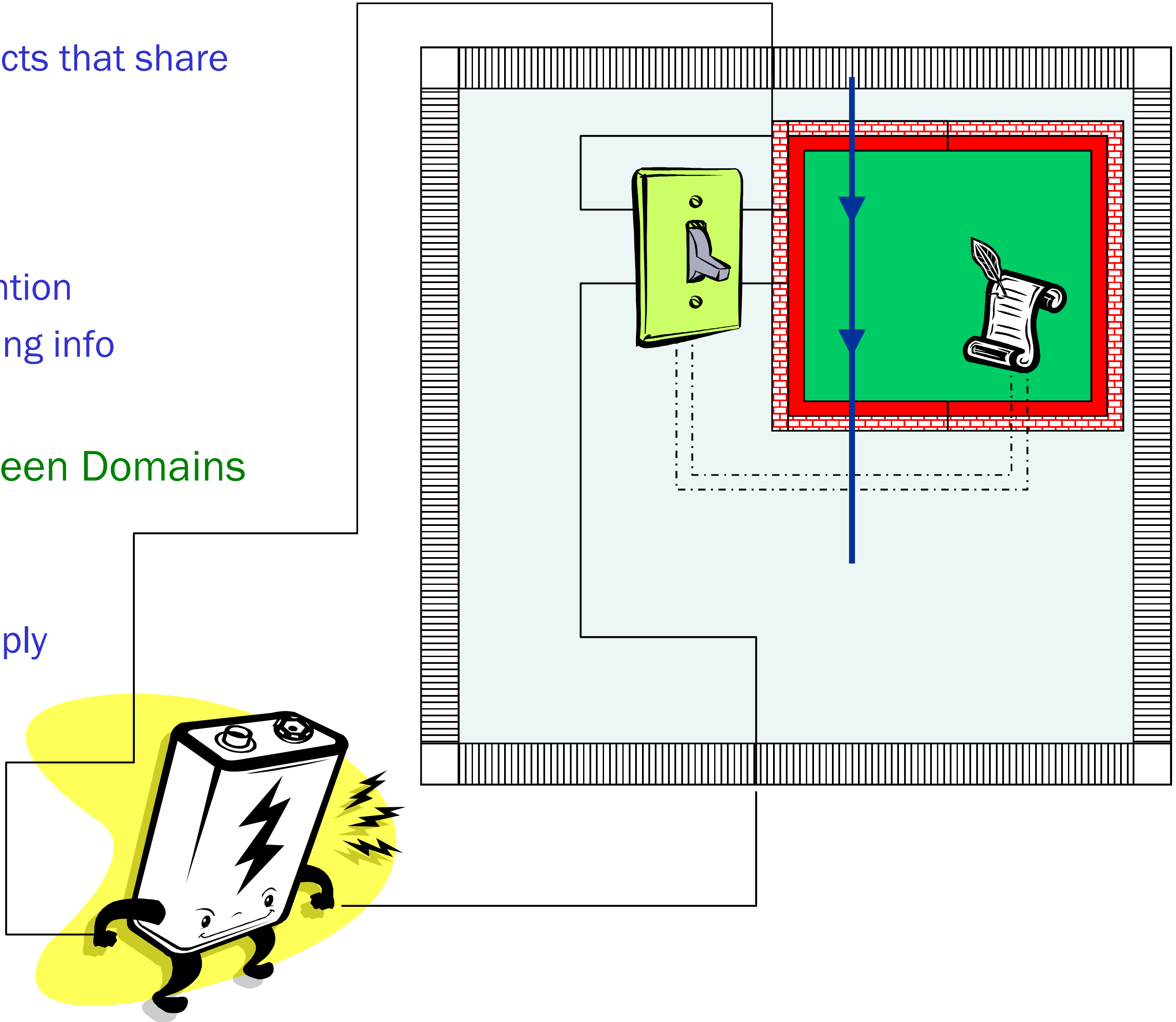
"Need Various Power Modes In System"

Power Down Modes - Example

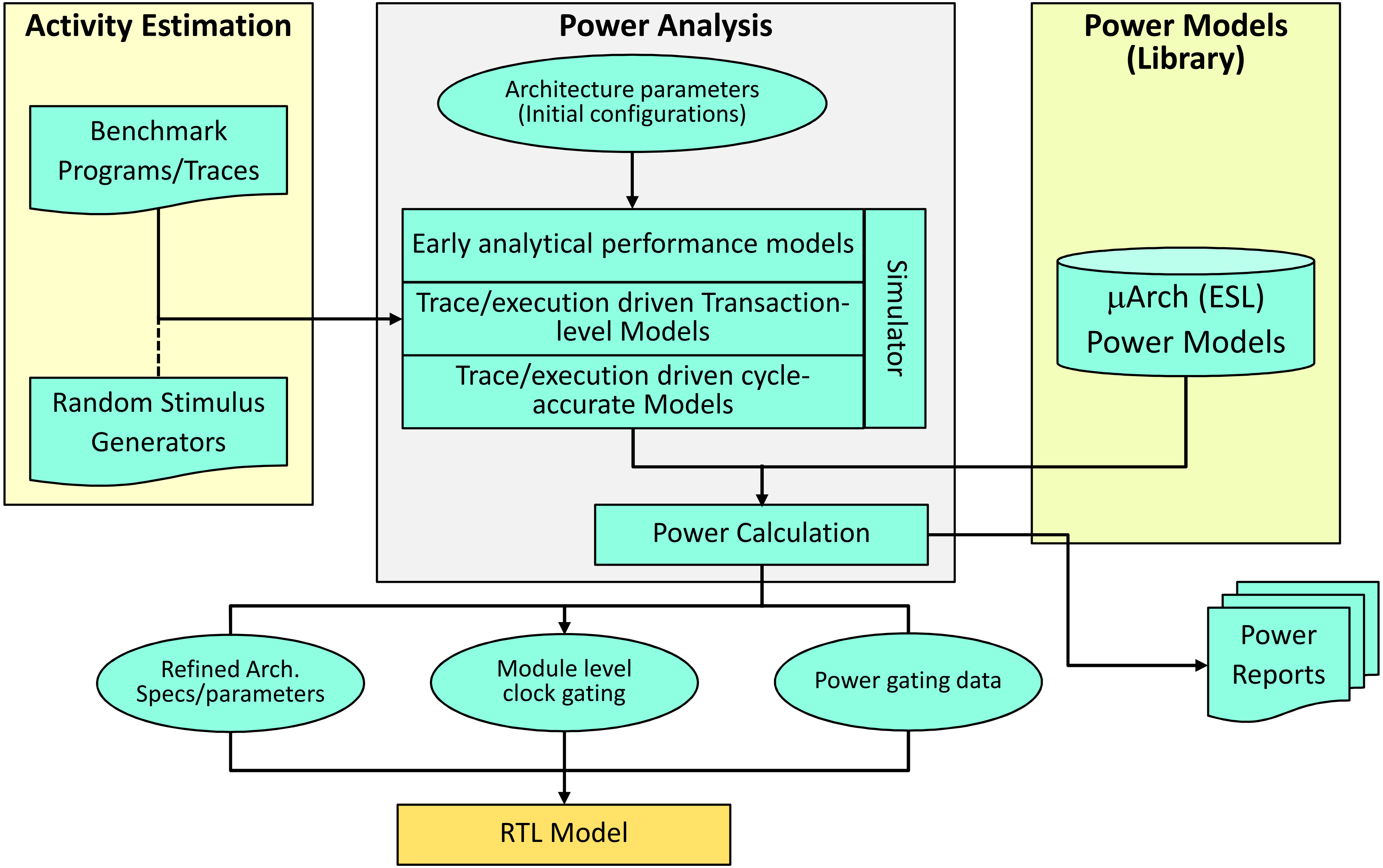
- Modes control clock frequency, V_{DD}, or both
  - Active mode: maximum power consumption
    - Full clock frequency at max V_{DD}
  - Doze mode: ~10X power reduction from active mode
    - Core clock stopped
  - Nap mode: ~ 50% power reduction from doze mode
    - V_{DD} reduced, PLL & bus snooping stopped
  - Sleep mode: ~10X power reduction from nap mode
    - All clocks stopped, core V_{DD} shut-off
- Issues and Tradeoffs
  - Determining appropriate modes and appropriate controls
  - Trading-off power reduction to wake-up time

Power Management Structures: The Data Objects

- Power Domain
  - The collection of design objects that share common power attributes
- Power States
  - Controlled by Switches
  - Memories may require Retention
  - States may require sequencing info
  - States will effect simulation
- Relations & Connections between Domains
  - Level shifters
  - Isolation logic
  - “Gas Stations” alternate supply
- Identify elements
- Manage
- Implement
- Analyze
- Reuse



Generic Power Modeling and Analysis in ESL Flows



Activity Estimation

Benchmark Programs/Traces

Random Stimulus Generators

Power Analysis

Architecture parameters (Initial configurations)

Early analytical performance models

Trace/execution driven Transaction-level Models

Trace/execution driven cycle-accurate Models

Simulator

Power Calculation

Power Models (Library)

μArch (ESL) Power Models

Power Reports

Refined Arch. Specs/parameters

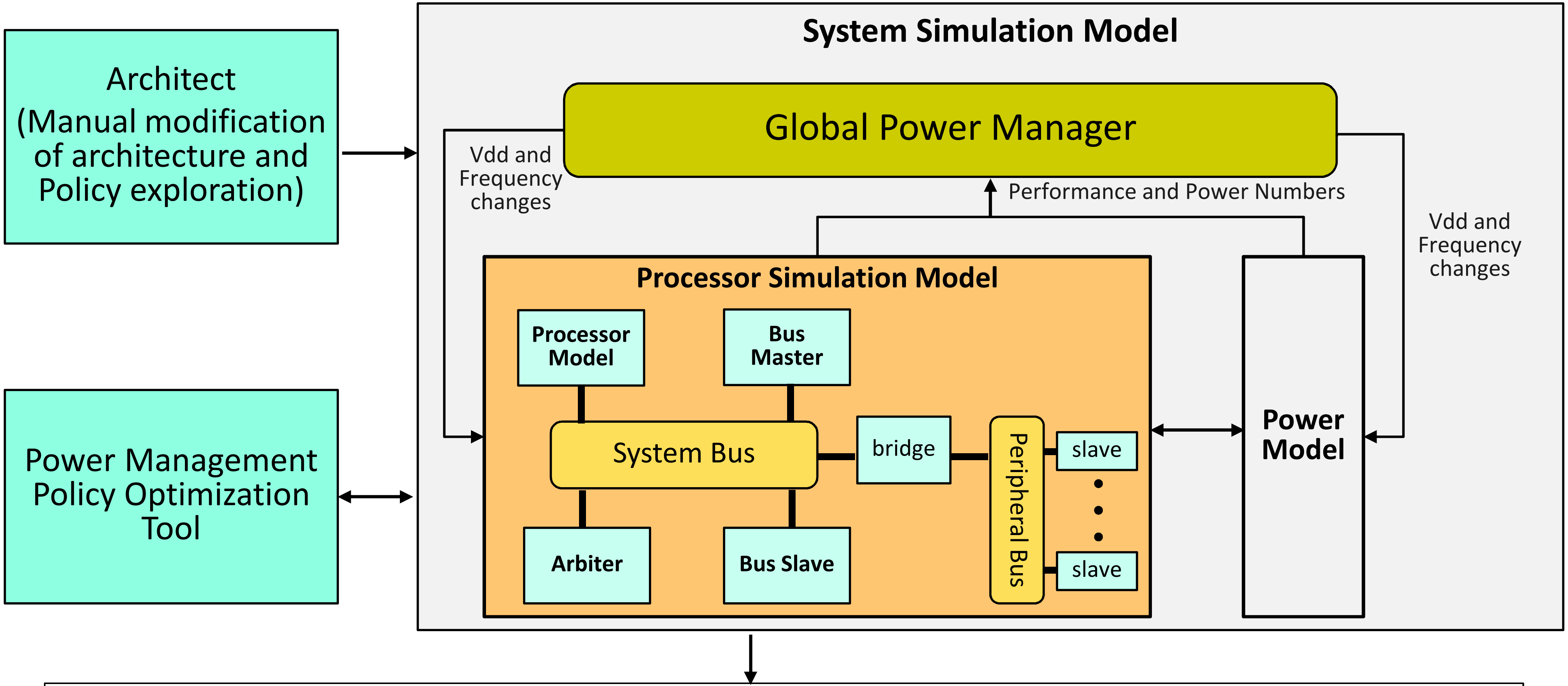
Module level clock gating

Power gating data

RTL Model

Power Management Design

- Use Scenario for Power Analysis at Architecture-level



Architect (Manual modification of architecture and Policy exploration)

Power Management Policy Optimization Tool

System Simulation Model

Global Power Manager

Vdd and Frequency changes

Performance and Power Numbers

Vdd and Frequency changes

Processor Simulation Model

Processor Model

Bus Master

System Bus

bridge

Peripheral Bus

slave

slave

Arbiter

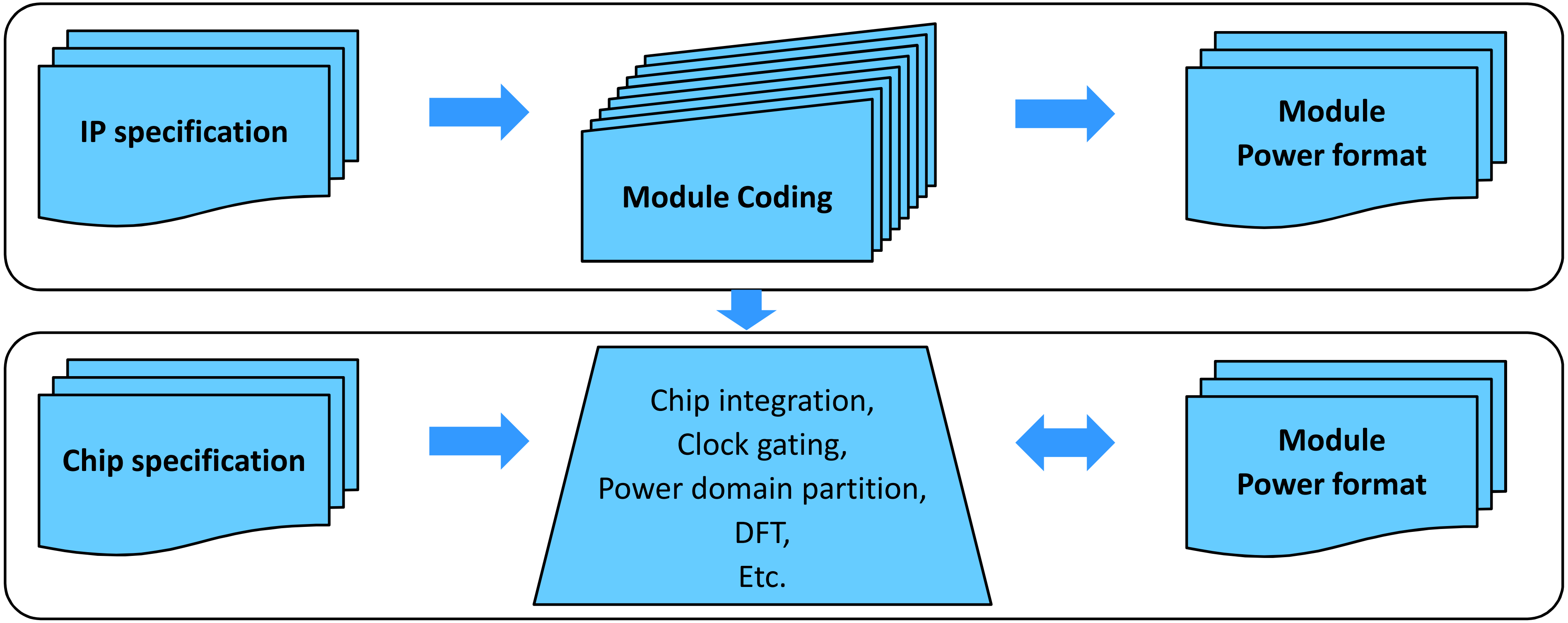
Bus Slave

Power Model

1. Architecture configuration (cache sizes, bus sizes, etc.)
2. Power management specification (power state definition, functionality of the power state, power consumption for that state, transition latency between states)
3. Power and clock domain definitions

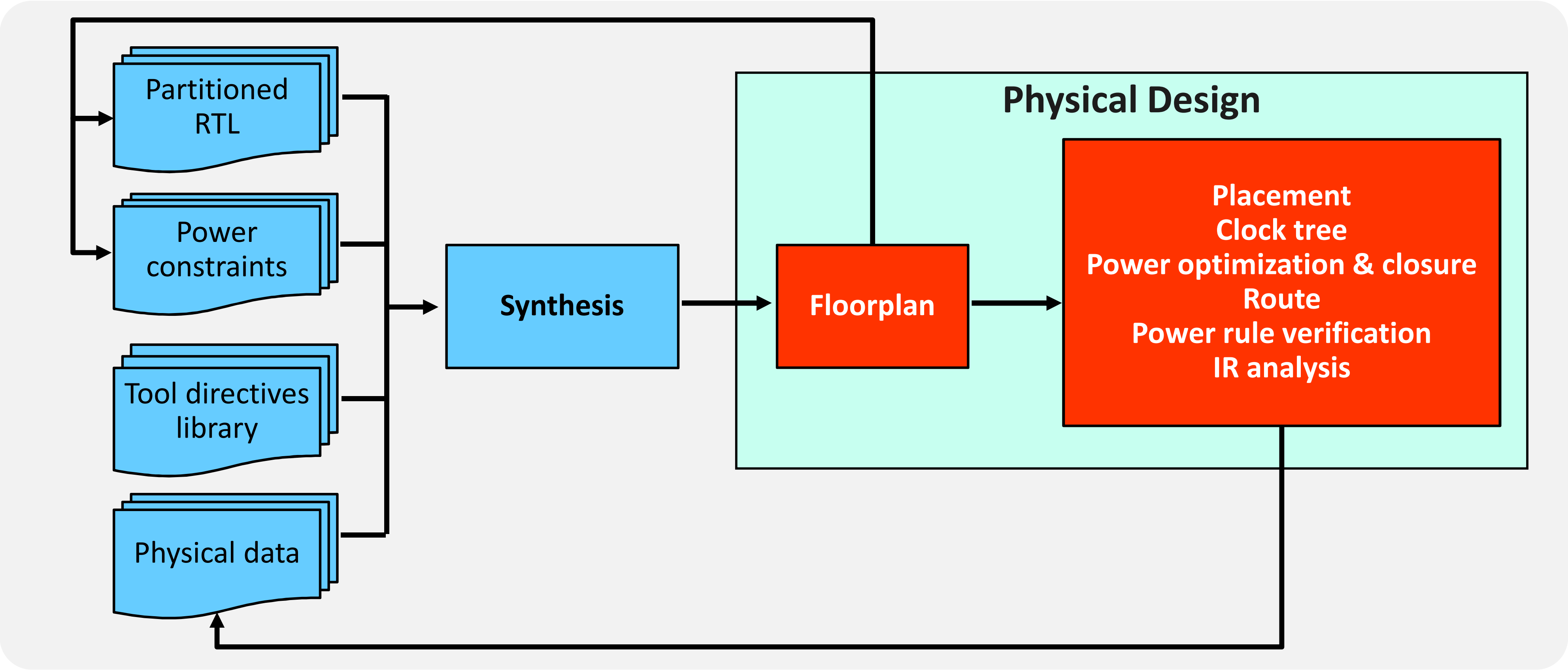
Design Phase

- The Design Phase is the detailed selection, modification, coding and interconnection of RTL descriptions of the major subsystems within the architecture
- Each sub-system is mapped into one or more power domains and the modes of operation for the design are refined into specific nominal operation conditions for each domain
- Domains that must preserve state (or some portion thereof) are identified
- Rules for the treatment of inter-domain signals are defined

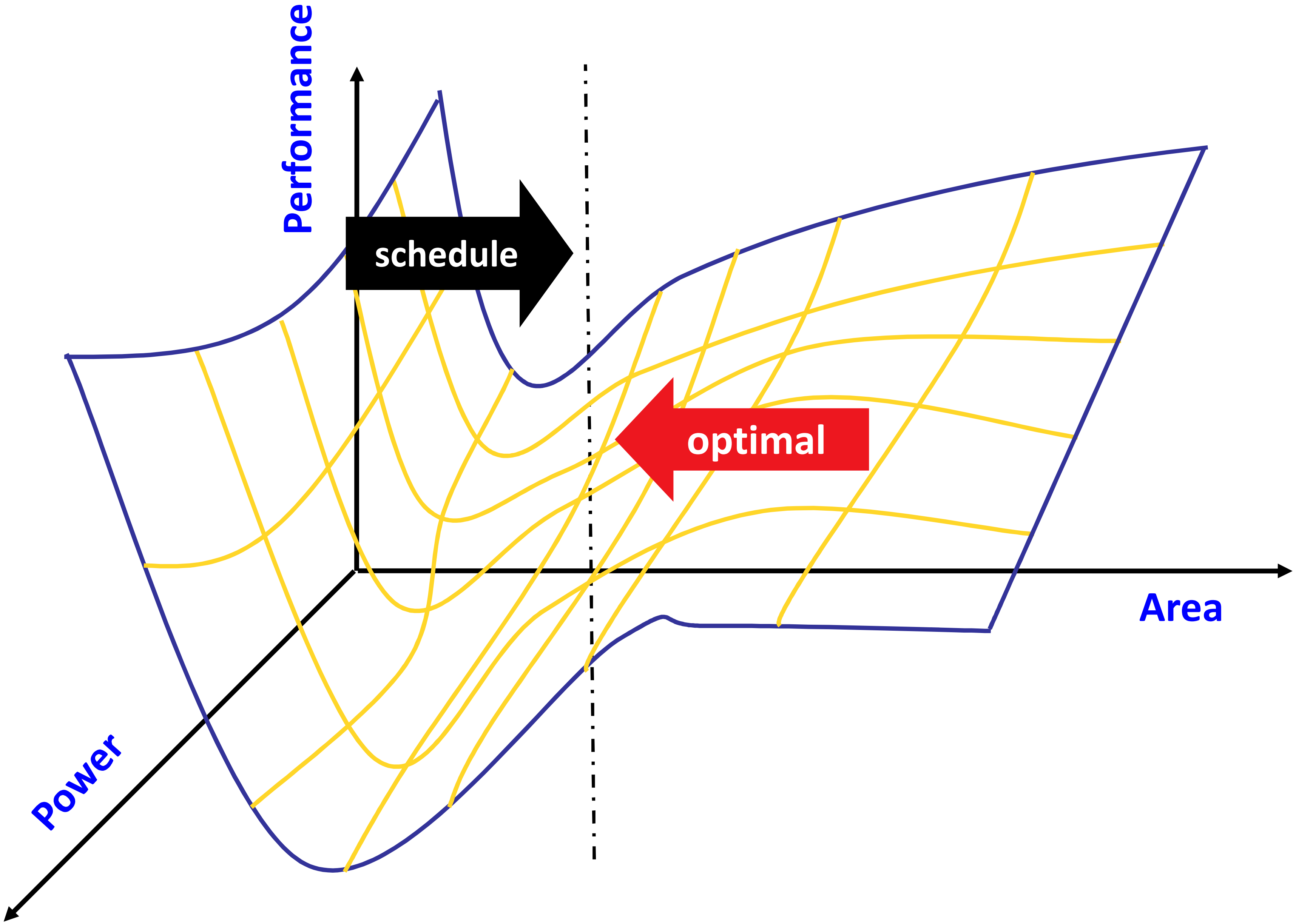


Implementation Phase

- The mapping of the RTL and power description file into physical devices that can realize the design intent
- Implementation tools must ensure that all constraints are satisfied
- In particular, implementation tools may generate additional constraints that must be satisfied by subsequent tools



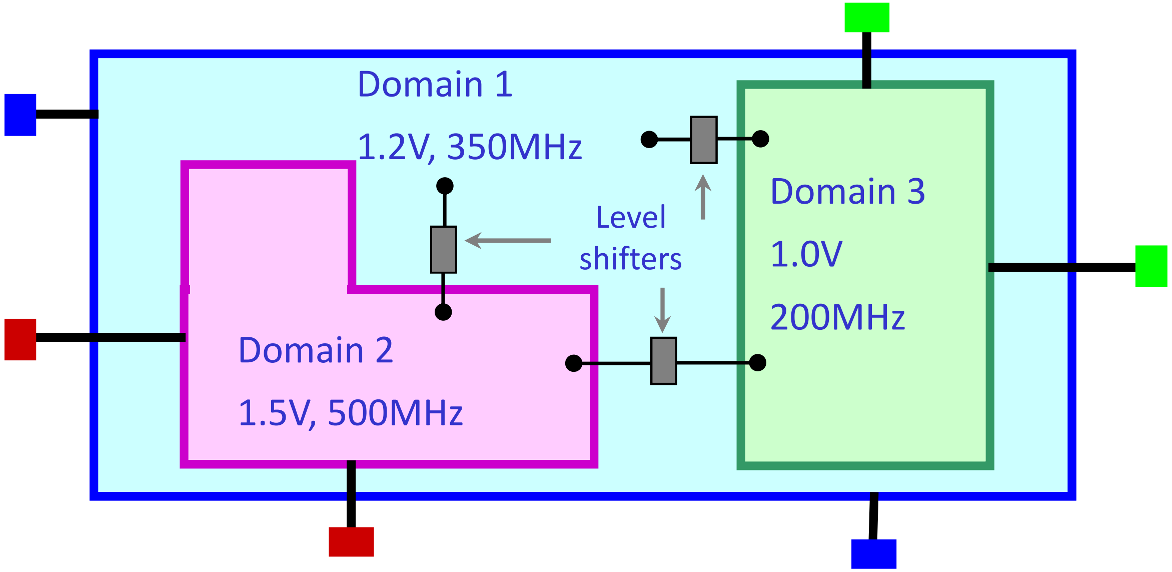
A Three Dimensional View of Design Closure



System-level Power Optimization Techniques: Multi VDD / Voltage Island

Multiple V_{DD} (Voltage Island)

- The concept of MVDD (Multi Vdd, Multiple voltage islands)
  - The control of supply voltage is the best effective method to reduce the total power
  - Voltage island
    - Region supplied through separated and dedicated power source
  - Multiple Voltage island
    - Define groupings of circuit of macros within an SoC which can be powered by lower supply while maintaining the required frequency and offering lower power consumption

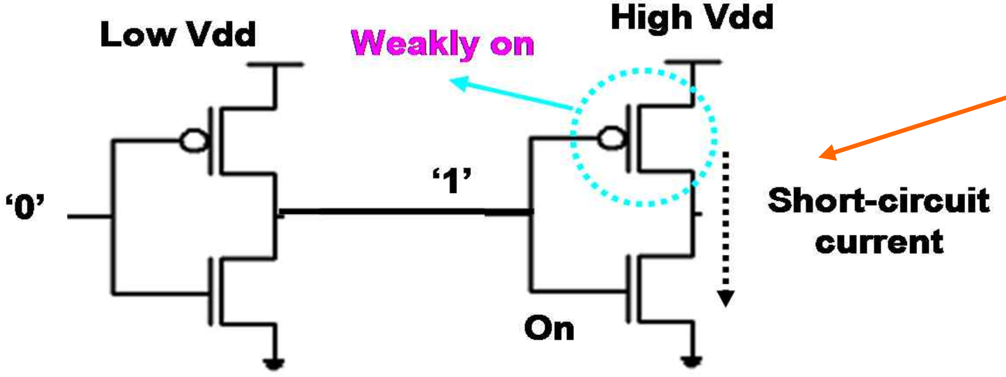


Multiple V_{DD} (Voltage Island) Issues

- Partitioning
  - Which blocks and modules should use with voltages?
  - Physical and logical hierarchies should match as much as possible
- Voltages
  - Voltages should be as low as possible to minimize CV_{DD}^2f
  - Voltages must be high enough to meet timing specs
- Level shifters
  - Needed (generally) to buffer signals crossing islands
    - May be omitted if voltage differences are small, ~ 100mV
  - Added delays must be considered
- Physical design
  - Multiple V_{DD} rails must be considered during floorplanning
- Timing verification
  - Signoff timing verification must be performed for all corner cases across voltage islands
  - For example, for 2 voltage islands V_{hi}, V_{lo}
    - Number of timing verification corners doubles

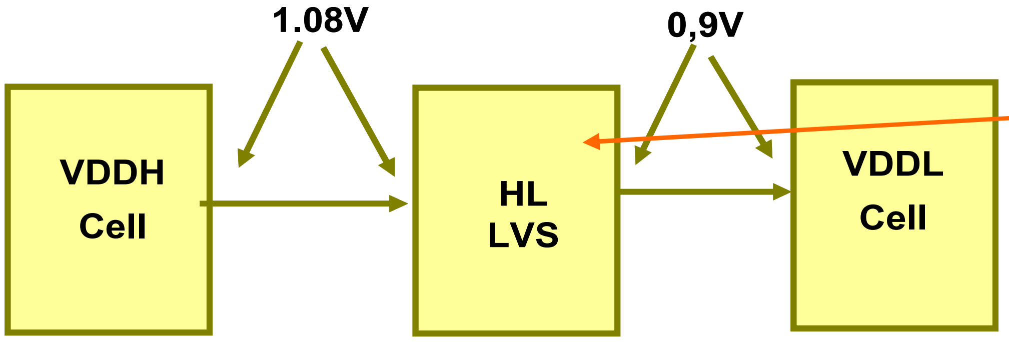
Multiple V_{dd} (Voltage Island): Level shifters

- Why level shifters are needed for multiple voltage islands?



1. Low to High

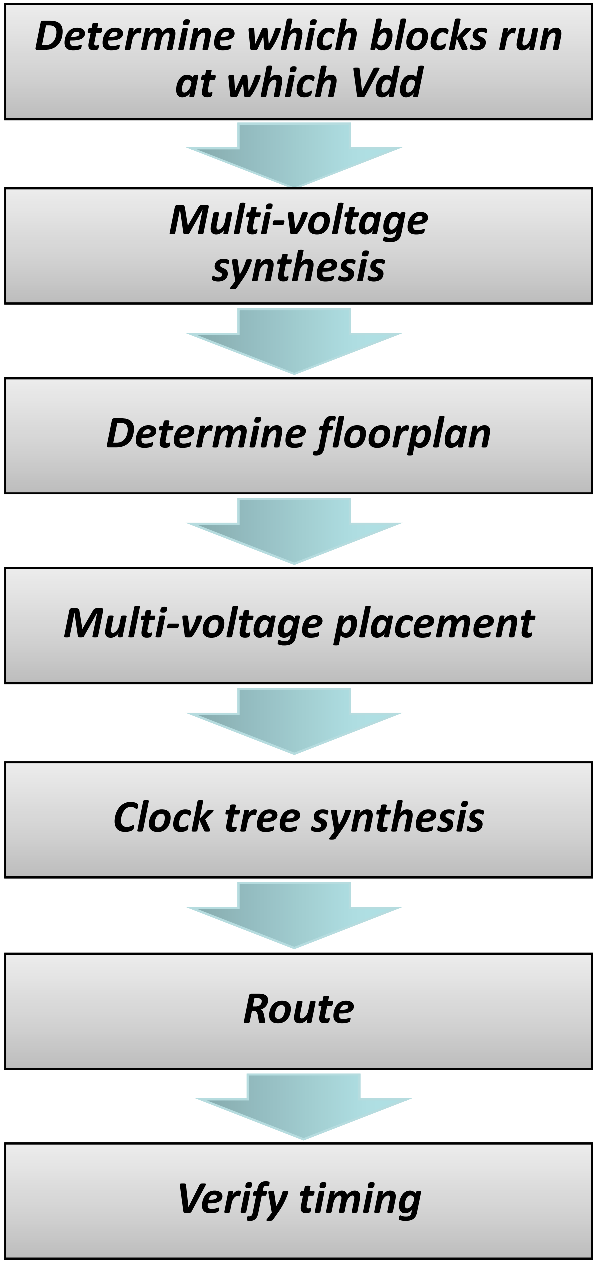
Low-to-high level shifter is needed to prevent the short-circuit current



2. High to Low

For the accurate STA, need a cell characterized with the different voltage for input and output pin like high-to-low level shifter

Multi-V_{DD} Flow



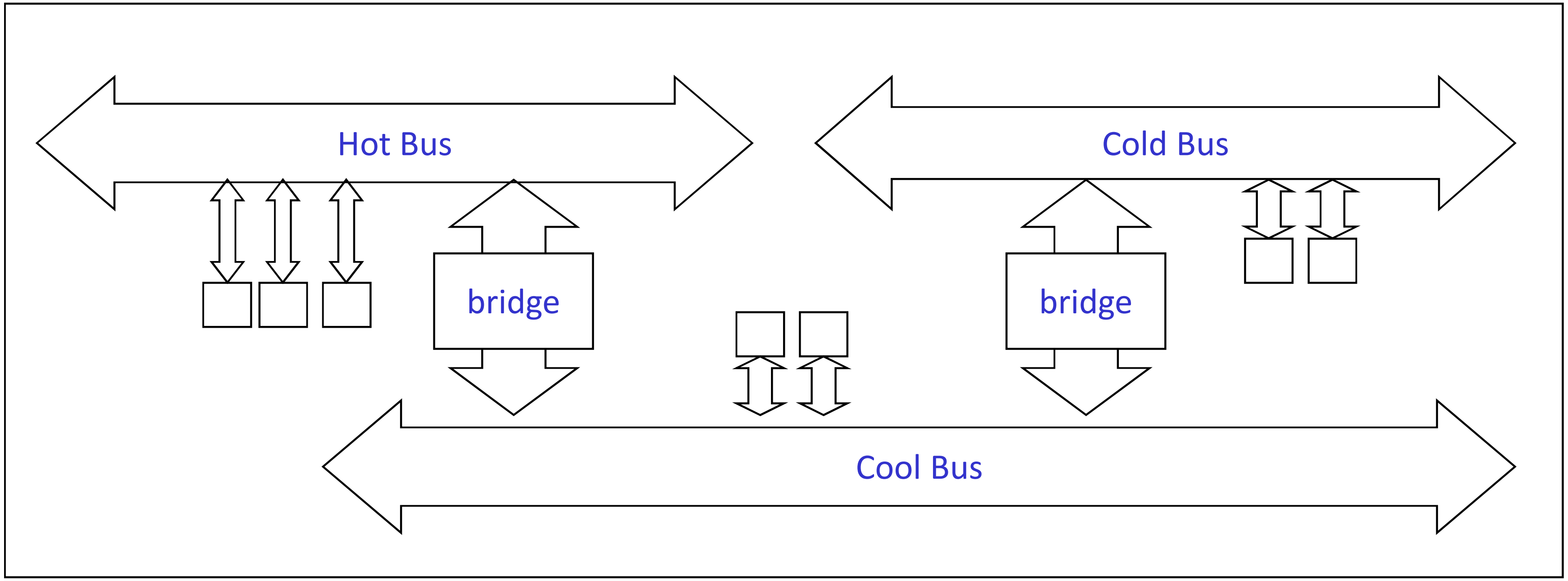
Determine which blocks run at which Vdd
Multi-voltage synthesis
Determine floorplan
Multi-voltage placement
Clock tree synthesis
Route
Verify timing

Functional Partitioning

- Identifying functional components with similar inactive periods
- Assigning functional components to possible chip-level power sources capable of providing required voltage level
- Identifying the optimal grouping of components, based upon power sequencing (affects static power) and operating voltage (affects active power) that minimizes chip power within the limits (such as peak power) of the SoC
- Identifying or creating, and connecting, logic signals that will be used to control power-sequencing circuitry or control clock gates
- Connecting alternate voltage sources to latches or arrays used to save state across power sequencing

Busses with Different Voltages

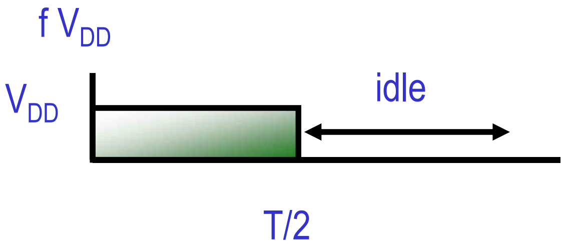
- One clock & One signaling voltage
- Some approaches:
  - Temporarily scaling V & F for comm.
  - Separate different voltages with bridges

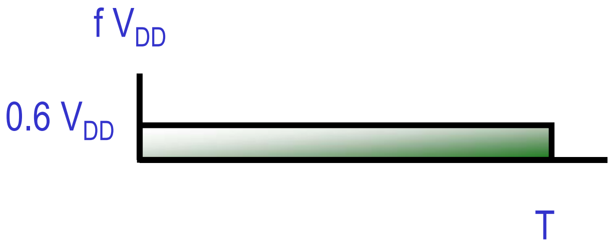


System-level Power Optimization Techniques: Dynamic Power Management (DVS, DVFS, AVFS)

Introduction to DPM

- Dynamic Power Management (DPM)
  - DPM controls power consumption of components based on its usage
  - Prediction of component usage is essential
  - Methods
    - Shutdown (clock gating, power gating)
    - Slowdown (frequency scaling, voltage scaling, V_{TH} scaling)





$$P = \alpha \cdot C_L \cdot f \cdot V_{DD}^2$$

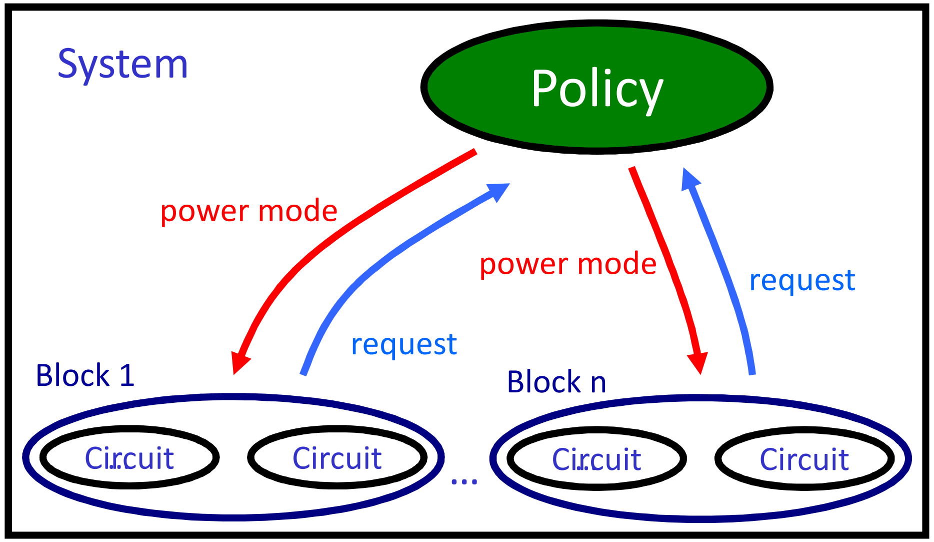
$$E = P \cdot T / 2$$

$$P' = \alpha \cdot C_L \cdot f / 2 \cdot (0.6 \cdot V_{DD})^2$$

$$E = P' \cdot T = 0.36 \cdot E$$

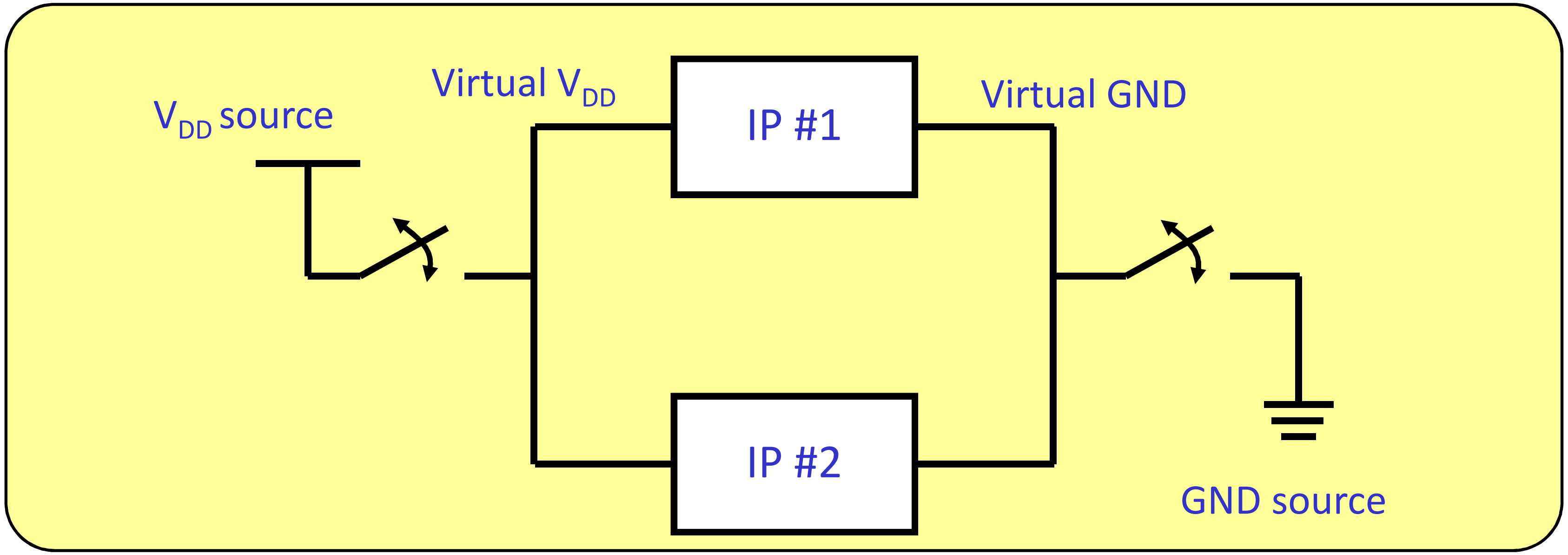
Structure of DPM

- Levels of embodiments of DPM
  - Component level
    - Circuit, Block
    - Power mode
  - System level
    - Policy
      - The procedure which controls the power level of each module in a system



Component Level DPM Scheme

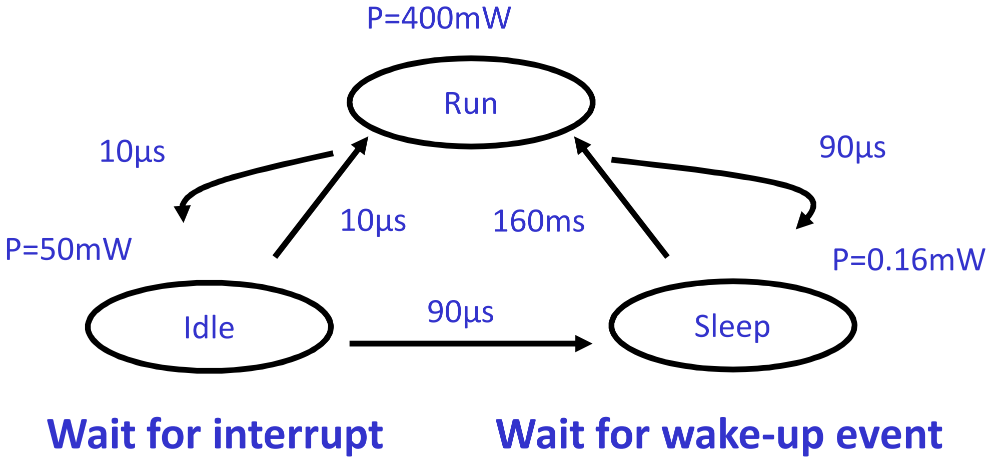
- Circuit level
  - Clock off by clock gating
  - Power off by footer/header of MTCMOS
  - Multiple voltage supply
- Block level
  - Power off by shutdown of power supply to IPs
  - When power off pattern of two block are similar, shutdown together



Component Level DPM Scheme

- Power mode
  - Each state has combination of enabled DPM technique.
    - e.g.) The case that system uses clock gating and block shutdown
  - Transitions between modes of operation have a cost

Power mode	Clock gating	Block shutdown
Run	**disabled**	**disabled**
Idle	**enabled**	**disabled**
Sleep	**enabled**	**enabled**

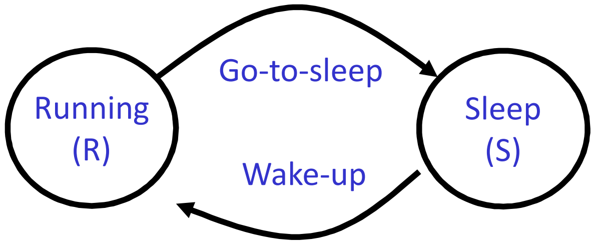


Power state machine for the StrongARM processor

SA-100 Microprocessor Technical Reference Manual, Intel, 1998

DPM Policy

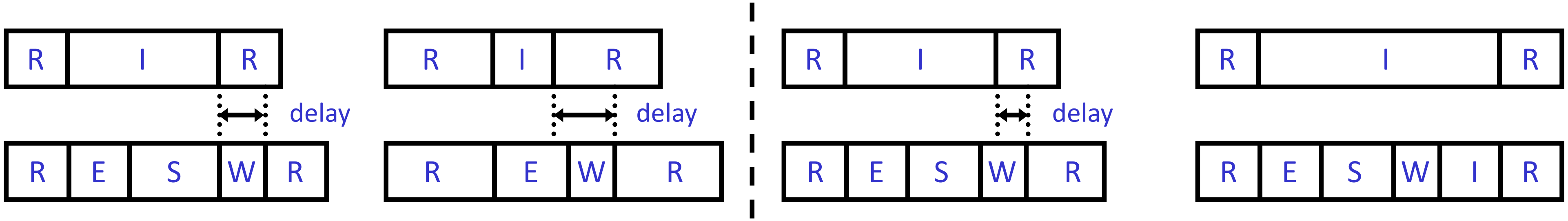
- Predictive technique
  - Uses a regression equation based on previous “On” and “Off” times of the component to estimate the next “turn on” time
  - Limitation
    - It cannot handle components with more than two power modes



R: Running period
I: Idle time period
S: Sleeping time period
E: delay overhead of entering the sleep state
W: delay overhead for resuming the running state from the sleep state

Predictive power management scheme

Pre-wakeup scheme



M. Srivastava et al, “*Predictive system shutdown and other architectural techniques for energy efficient programmable computation*”, IEEE TVLSI, Vol. 4, No.1 ,1996

C.H. Hwang et al, “*A predictive system shutdown method for energy saving of event-driven computation*”, Proc. Int. Conf. on Computer Aided Design, pages 28-32, Nov. 1997

I: Idle state   E: Entering state   W: Waking up state

DPM Policy

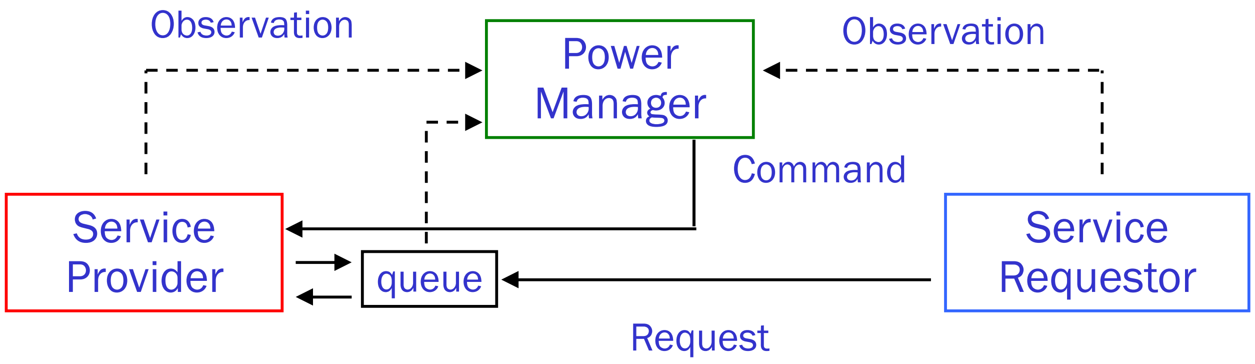
- Markov process
  - Markov process is a process which uses a previous state and pre-characterized probability to choose next state
  - Power management optimization has been studied within the framework of Markov process

G.A. Paleologo et al, “*Policy optimization for dynamic power management*”, Proc. DAC, 1998

- When system is modeled as Markov chains
  - It can model the uncertainty in system power consumption and response times
  - It can model complex systems with many power states, buffers, queues
  - It can compute power management policies that are globally optimum

DPM Policy

- Structure of stochastic DPM



Observation

Power Manager

Observation

Command

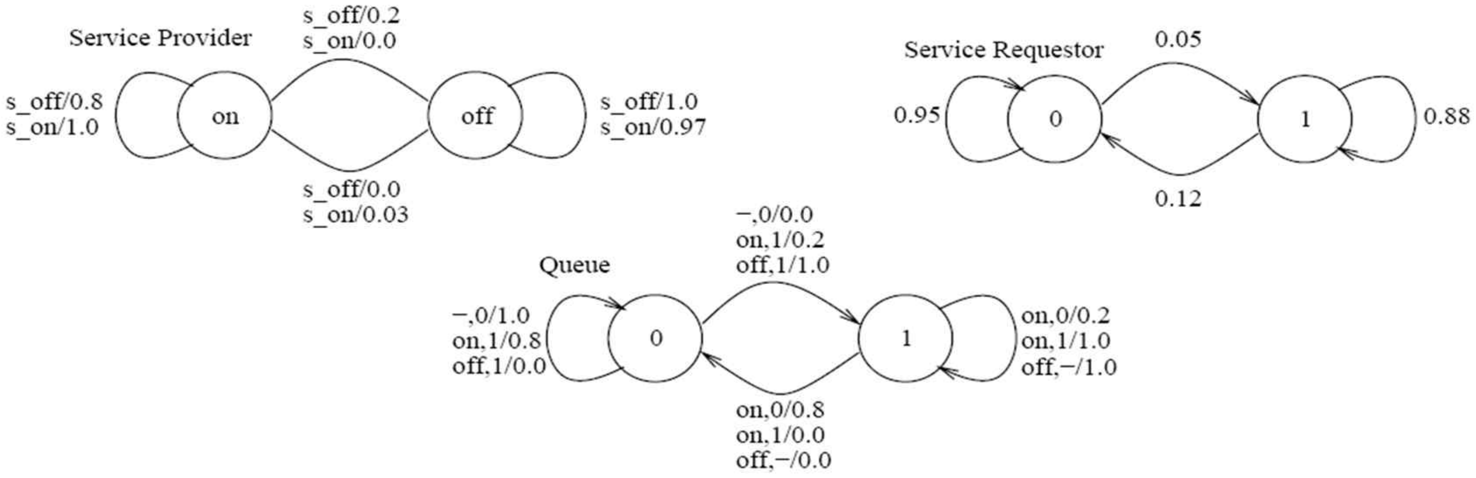
Service Provider

queue

Service Requestor

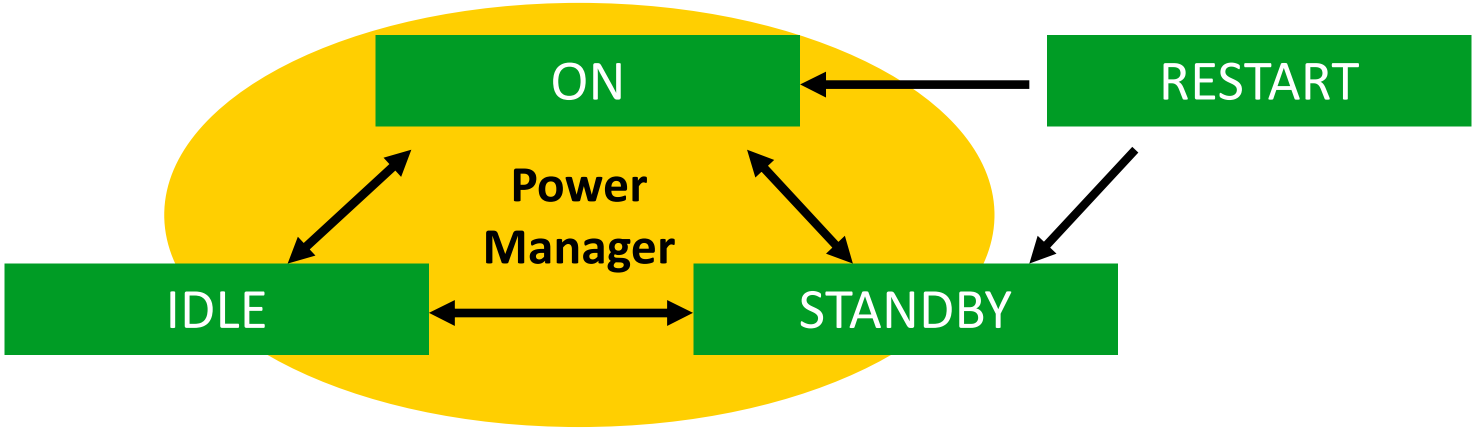
Request

- FSM of each module



Conventional Power Management

- Conventional power management schemes manage the transitions between defined power states

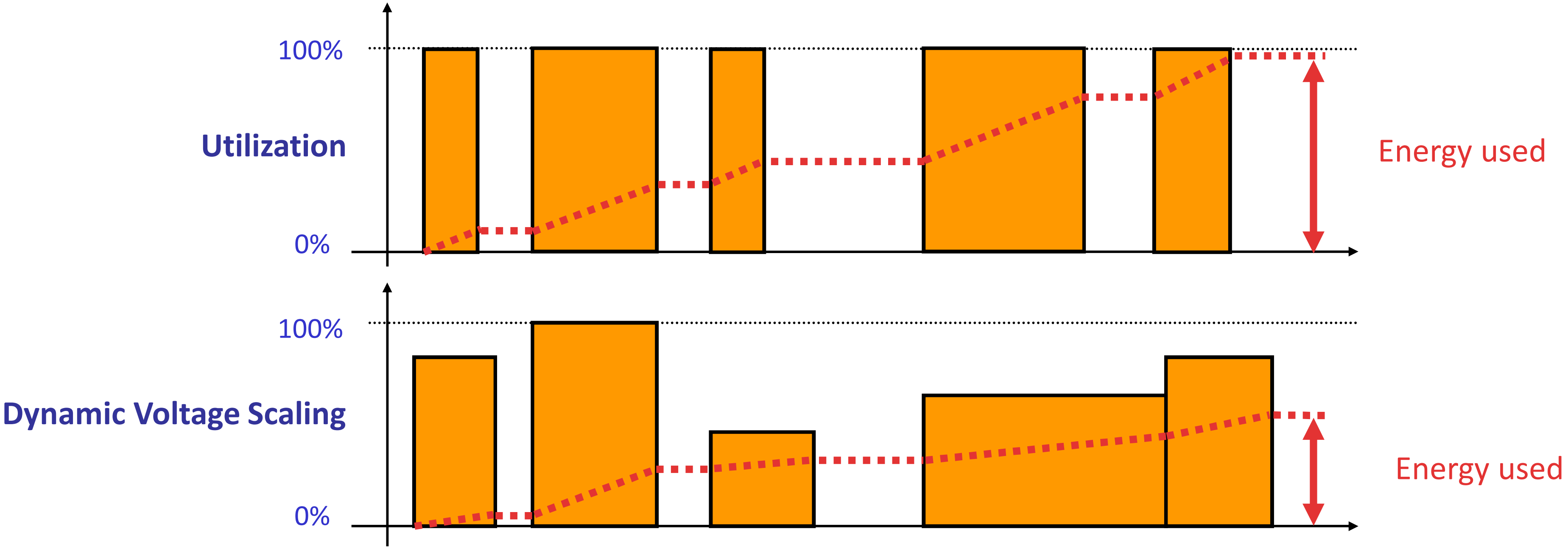


  - STANDBY is off but with state retained with clocks stopped
  - IDLE is a lower power mode with a slow clock running
  - ON state is fully powered up at maximum clock frequency

- Despite the changing software workload, system runs at maximum performance while there is any work to be done

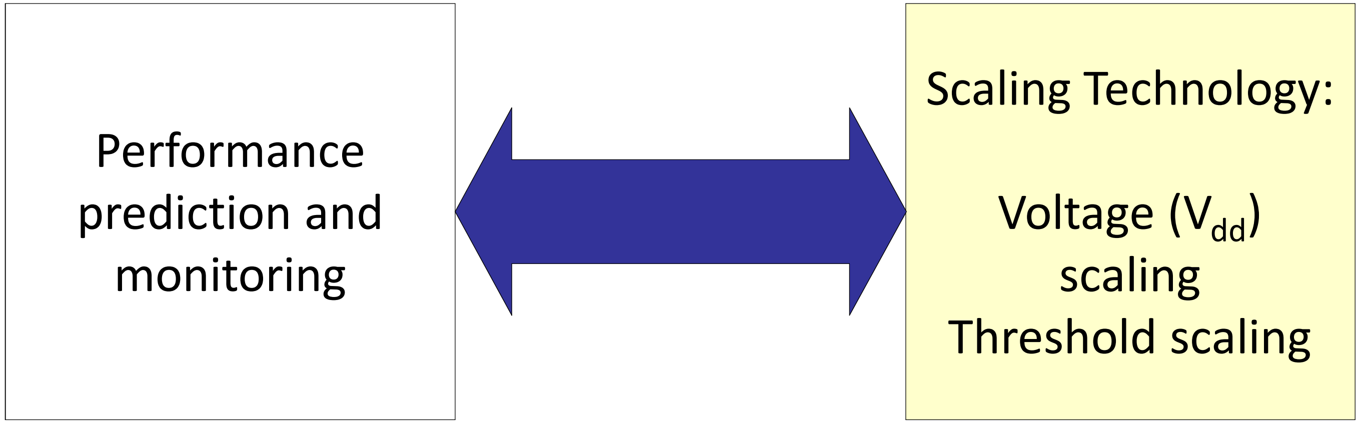
Optimizing for utilization Characteristics

- Conventional power management optimizes power consumption when there is nothing to do (sleep modes)
- IEM optimizes power when work is being done
  - Only run fast enough to meet deadlines! – Just-in-Time
    - Running fast and idling wastes power
- The active- and sleep-mode techniques are orthogonal



Meeting the Performance Requirement

- Effective Energy Management requires:
  - Automatic Performance Prediction technology
    - Determining the lowest performance level that will get the software workload done just in time
  - Performance Scaling technology
    - Delivering just enough performance to meet the current requirement
    - Responding rapidly to changing performance levels



Performance prediction and monitoring

Scaling Technology:

Voltage (V_{dd}) scaling

Threshold scaling

Energy Management Control Components

- Software component
  - To automatically predict future software workloads by interacting with instrumented Operating Systems and application software
  - To determine the software deadlines
  - To balance workload and deadlines with performance
- Hardware component
  - To accurately measure the actual system performance
  - To independently manage the transitions of hardware scaling blocks. e.g., clock generators and power controllers
- Together these components determine and manage the lowest performance level that gets the work done

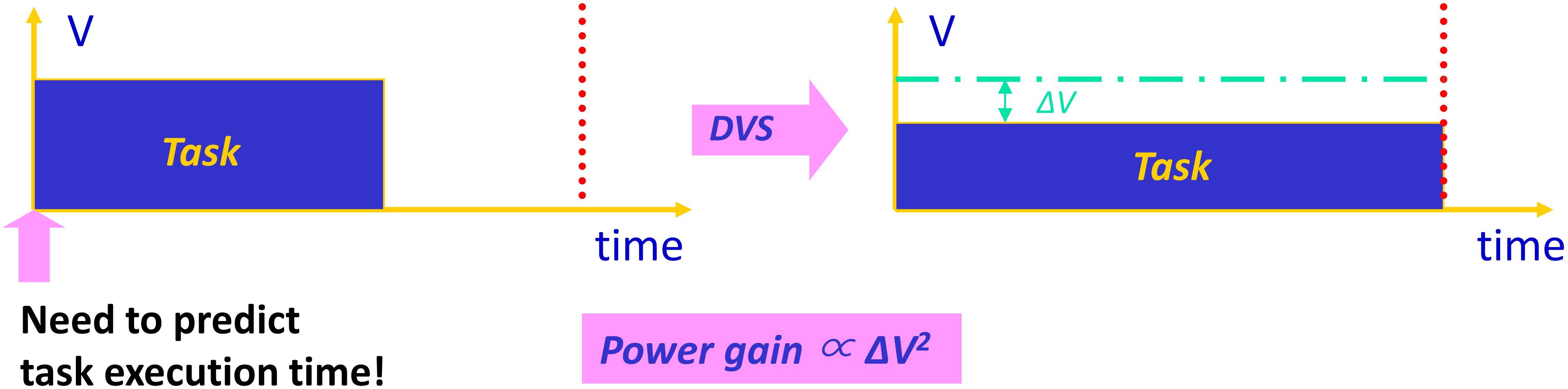
Terminology

Dynamic voltage scaling (DVS)	In this subset of DVFS, selected portions of the device are dynamically set to run at different voltages on the fly while the chip is running.
Dynamic voltage and frequency scaling (DVFS)	Selected portions of the device are dynamically set to run at different voltages and frequencies on the fly while the chip is running. Used for dynamic power reduction
Adaptive voltage and frequency scaling (AVFS)	In this variation of DVFS, a wider variety of voltages are set dynamically, based on adaptive feedback from a control loop; involves analog circuitry

- Adaptive voltage and frequency scaling is an extension of DVFS
- In DVFS, the voltage levels of the targeted power domains are scaled in fixed discrete voltage steps. Frequency-based voltage tables typically determine the voltage levels. It is an open-loop system with large margins built in, and therefore the power reduction is not optimal
- On the other hand, AVFS deploys closed-loop voltage scaling and is compensated for variations in temperature, process, and IR drop using dedicated circuitry (typically analog in nature) that constantly monitors performance and provides active feedback. Although the control is more complex, the payoff in terms of power reduction is higher.

Dynamic Voltage Scaling (DVS)

- Reduction of Stand-by Power in Leaky Process
  - By Monitoring Data Bus Congestion
  - By Monitoring/Guessing Performance Needed, for Specific Application



Need to predict task execution time!

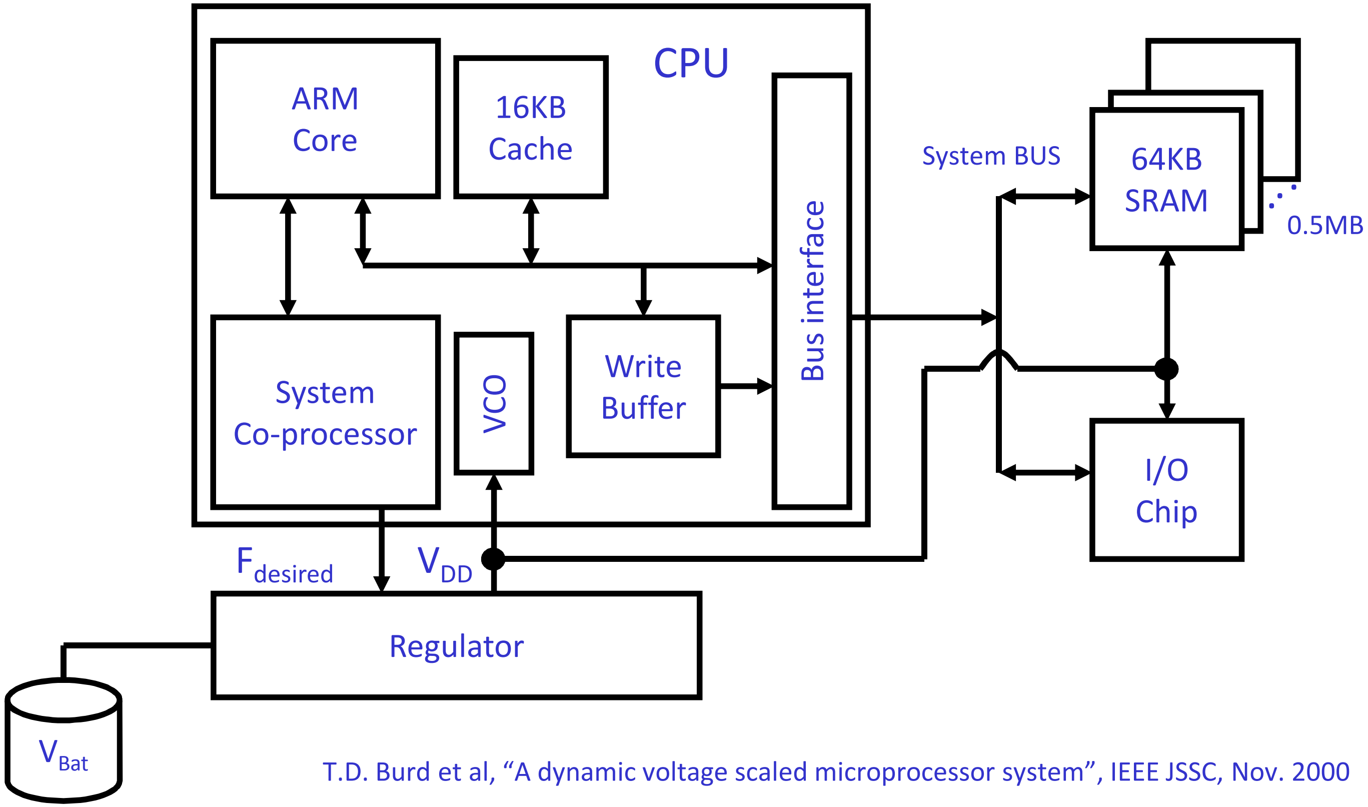
Power gain $\propto \Delta V^2$

Dynamic Voltage Scaling (DVS)

- Stretch the execution by lowering the supply voltage
  - Quadratic Power saving
  - No later than the deadline
- DVS Algorithms
  - Can be implemented as HW or SW
  - Optimal solution in continuous voltage domain, but not in discrete voltage domain

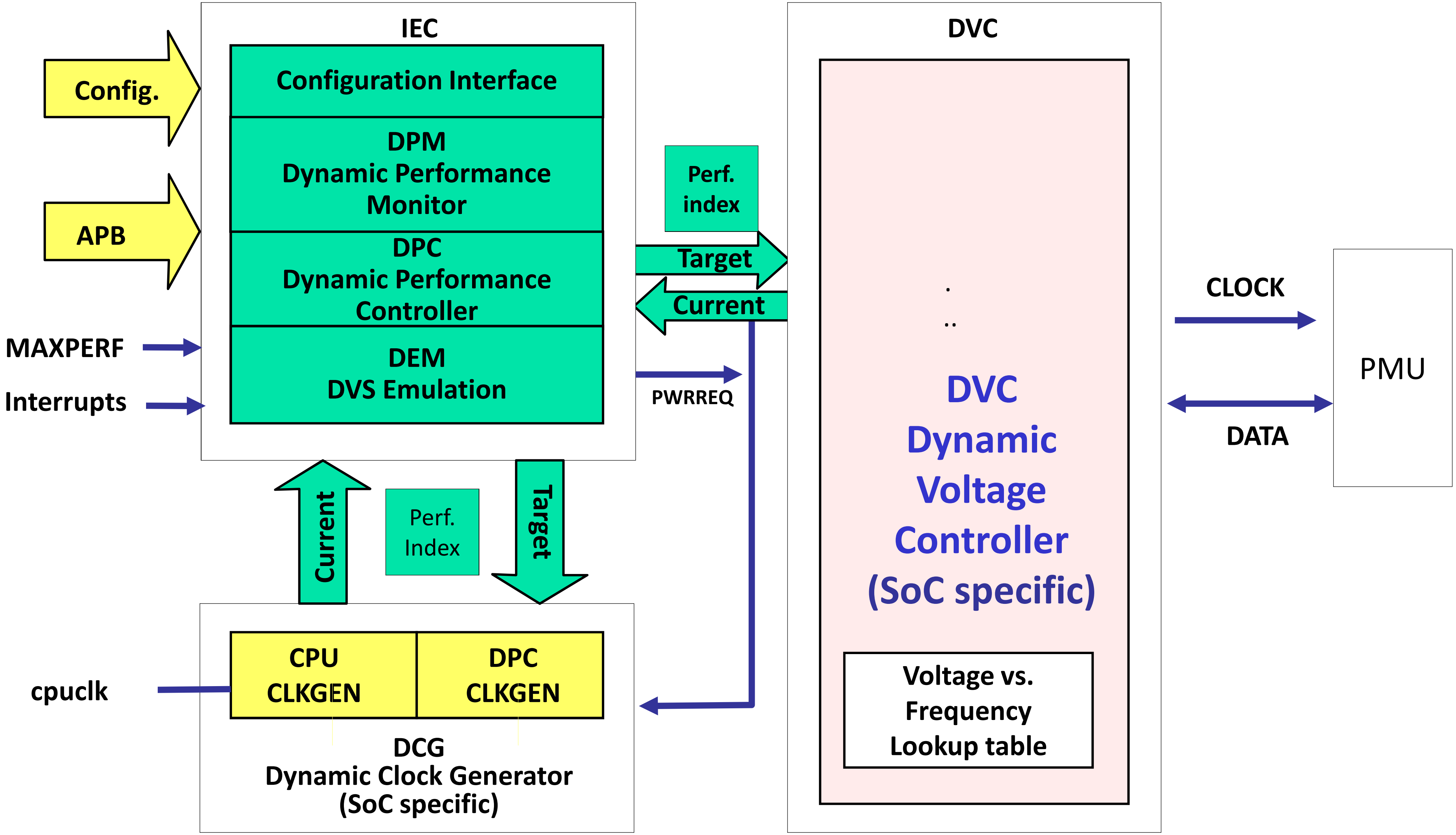
DVS Applied Processor

- Transition overhead
  - Max 70μs for 5~80MHz transition
  - Max 4μJ for 5~80MHz transition



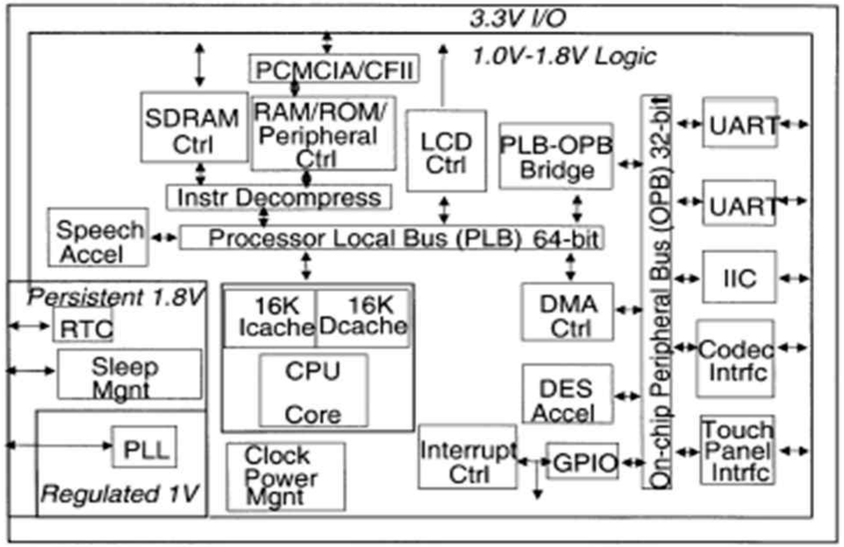
T.D. Burd et al, “A dynamic voltage scaled microprocessor system”, IEEE JSSC, Nov. 2000

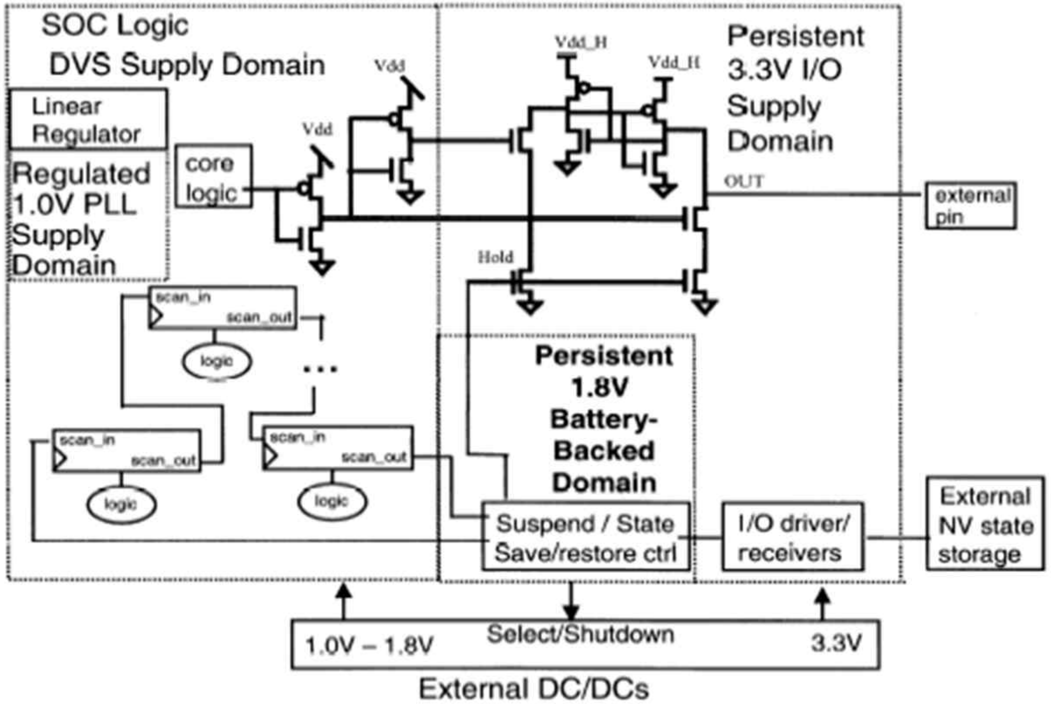
DVS Control Sub-system



DPM using DVS on SoC

- Divide SoC into 4 power domains
  - Persistent 3.3V : I/O drivers and receivers
  - Persistent 1.0V : PLL
  - Persistent 1.8V : RTC, sleep management
  - DVS : 1.0V ~ 1.8V (10mV/μs)

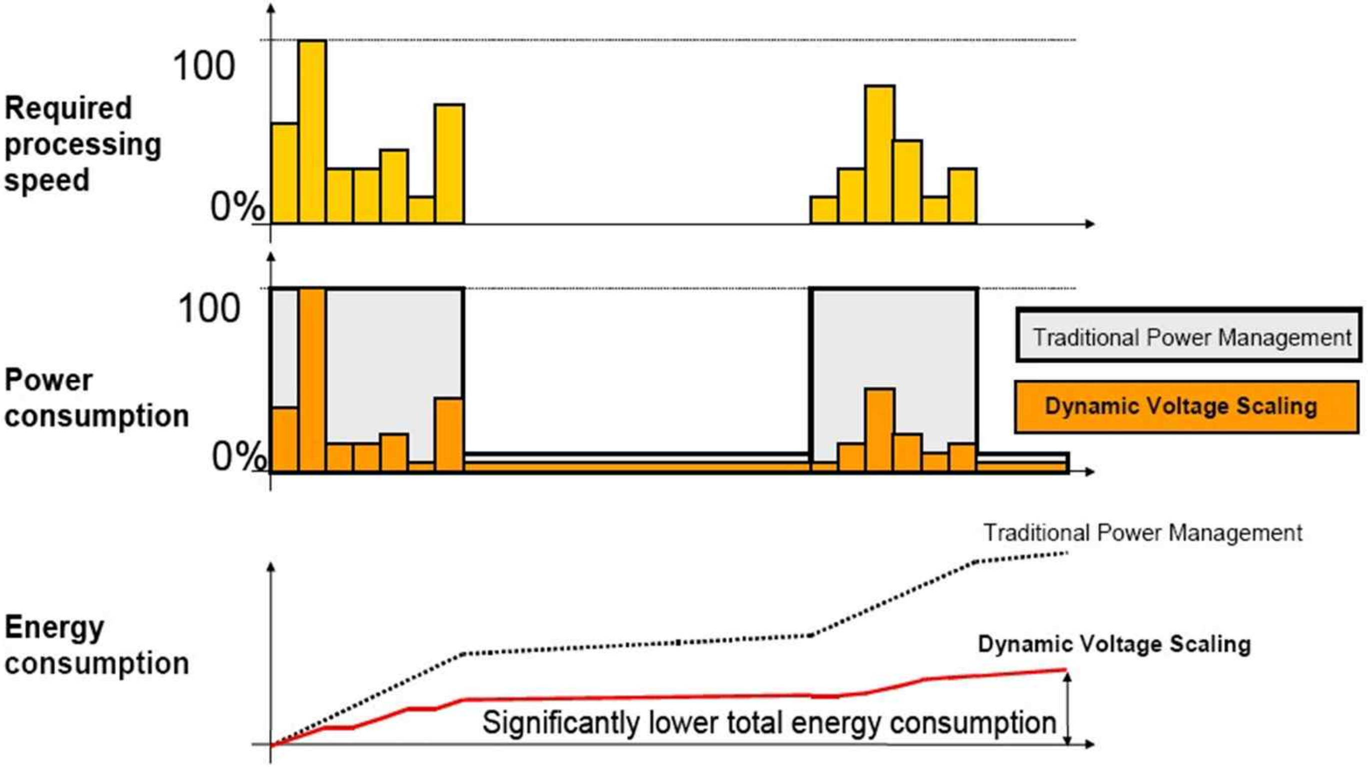




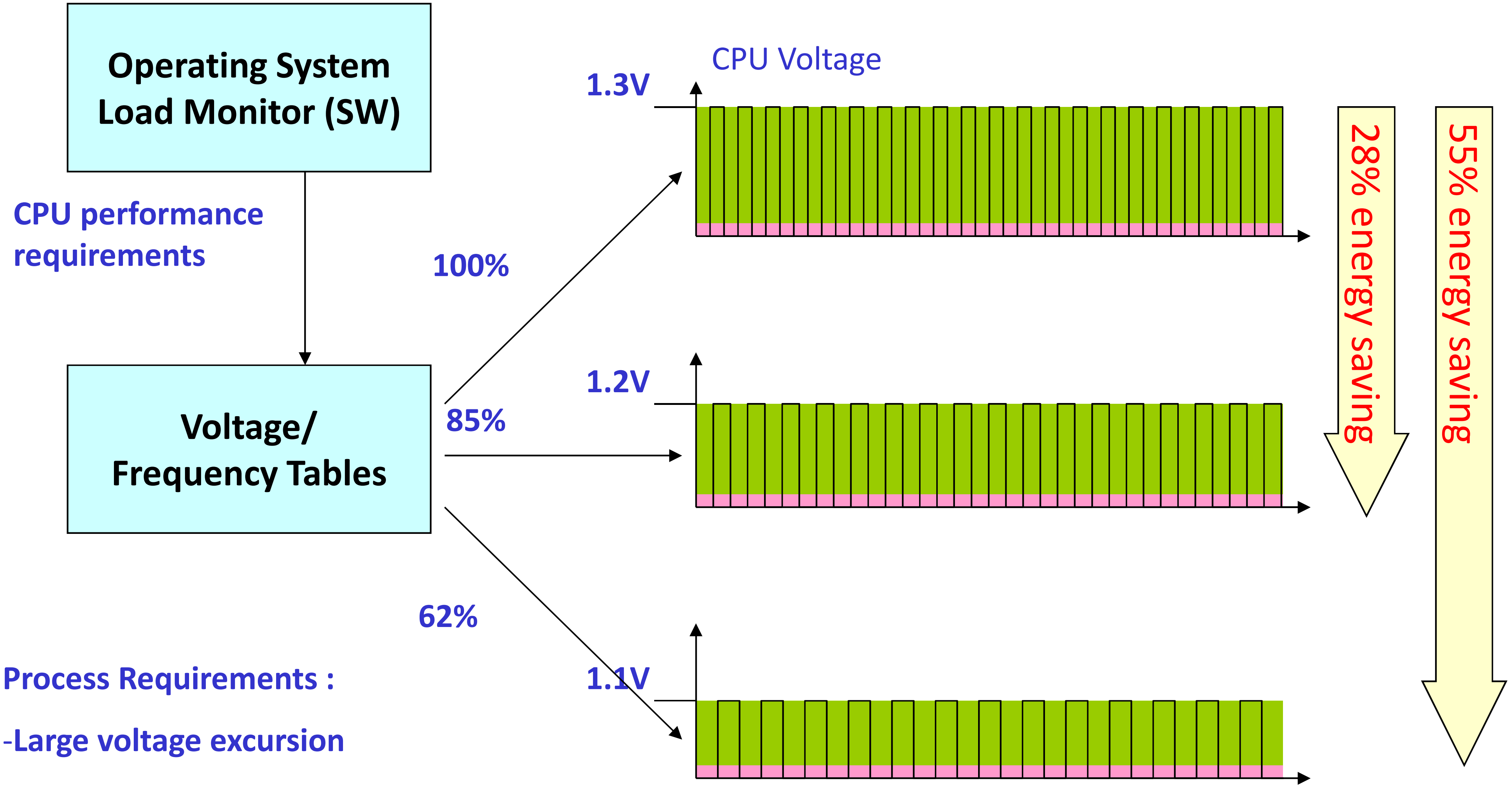
K.J. Nowka et al, “A 32-bit PowerPC System-on-a-Chip with support for dynamic voltage scaling and dynamic frequency scaling”, IEEE JSSC, Nov. 2002

DVFS

- The concept of DVFS (Dynamic Voltage Frequency Scaling)
  - Varying the voltage and frequency dynamically according to request of the processing load and environment
  - Fixed mode vs. dynamic control using 4~32 voltage steps and fixed voltage range
  - Fixed mode example : MPEG4 @ 1.0V/400MHz, MP3 @ 0.8V/50MHz
  - Dynamic control : Automatically derives required performance level



DVFS Principle

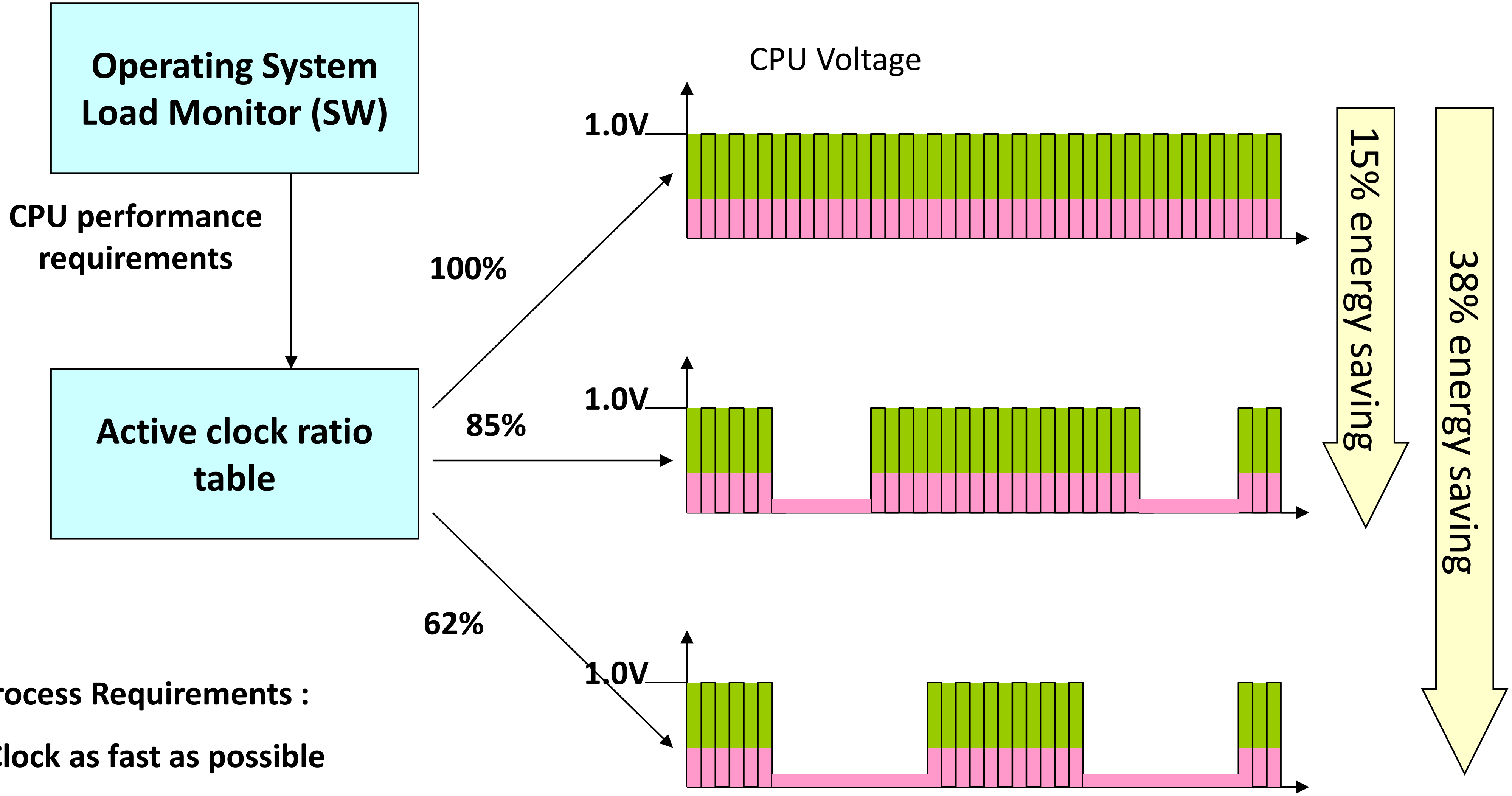

Operating System Load Monitor (SW)
CPU performance requirements
Voltage/ Frequency Tables
100%
85%
62%
CPU Voltage
1.3V
1.2V
1.1V
28% energy saving
55% energy saving

Process Requirements :

-Large voltage excursion

-Low leakage

Power Management Principle



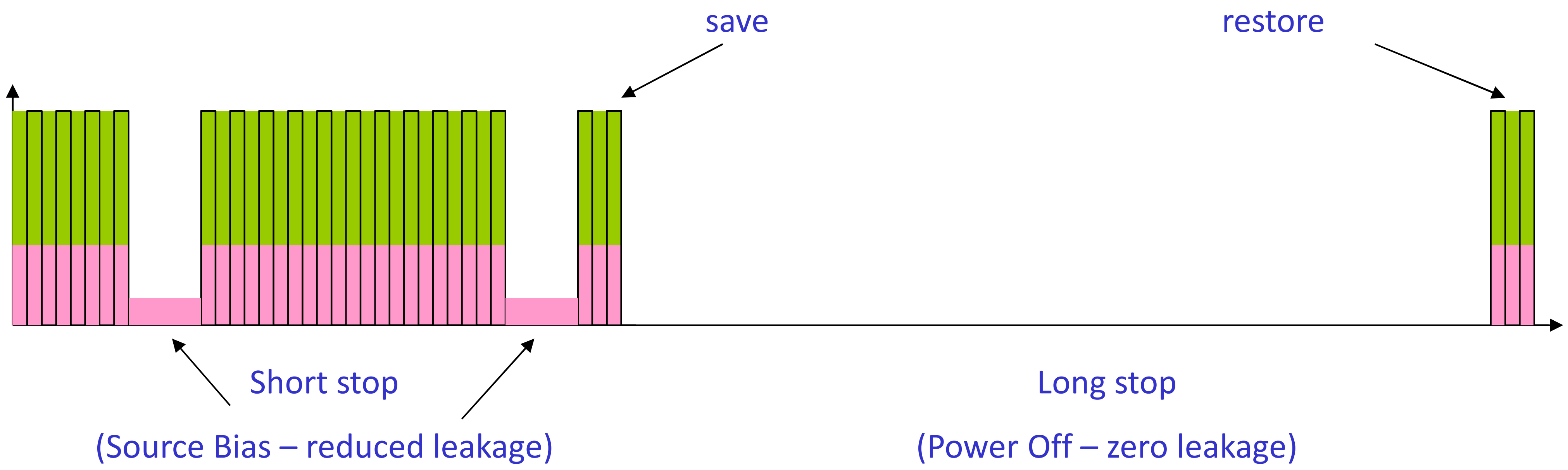
Process Requirements :

-Clock as fast as possible

-Source bias or switch off when clock is stopped

Multi-step Power Management

- Power management state machine under SW control
  - Source Bias for short clock stop period
  - Power off with context save/restore for long period



DVFS Power Management

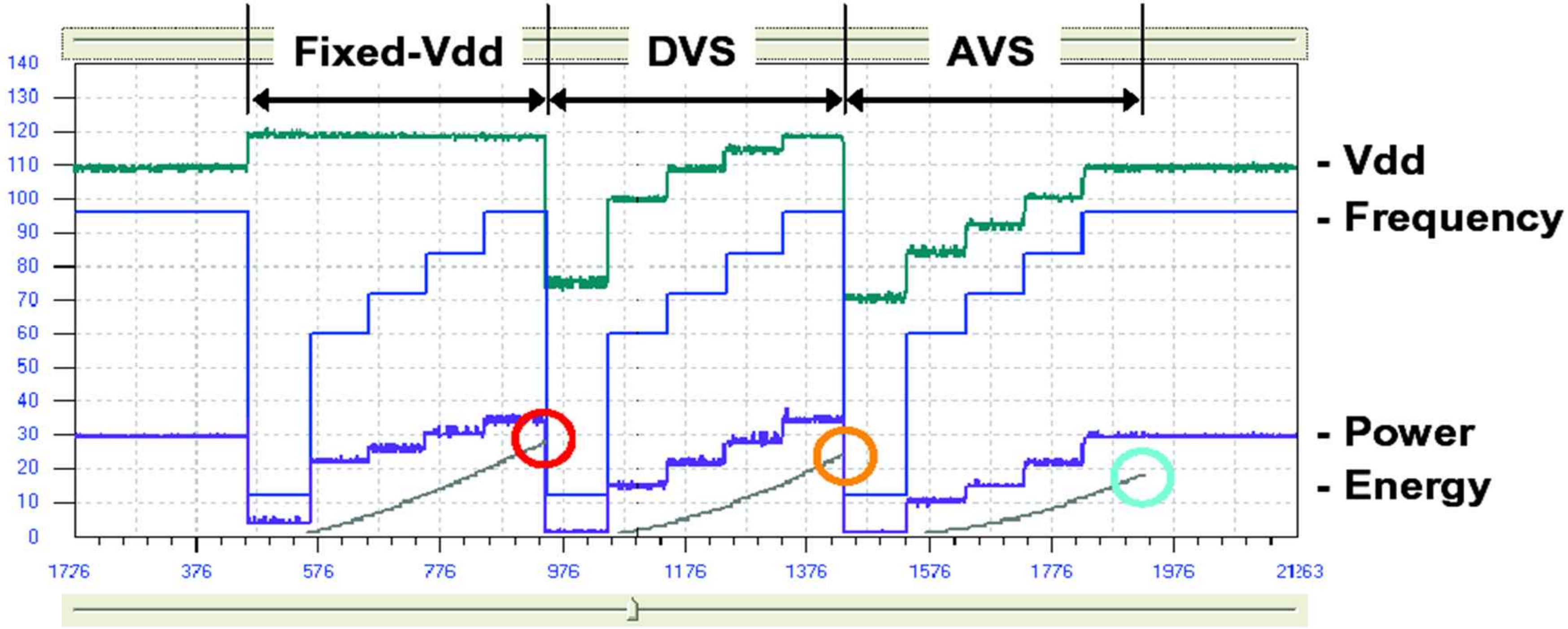
- Power mode changes are managed by software:
  - Constraints and impact must be known by software developer
  - Information initially needed only at design level is now flowing into the software space
- Power awareness in the software world is coming form the design world through better link between design tools and software development tools
- Need for a power view of the application accessible to software developers

Adaptive Voltage Scaling (AVS)

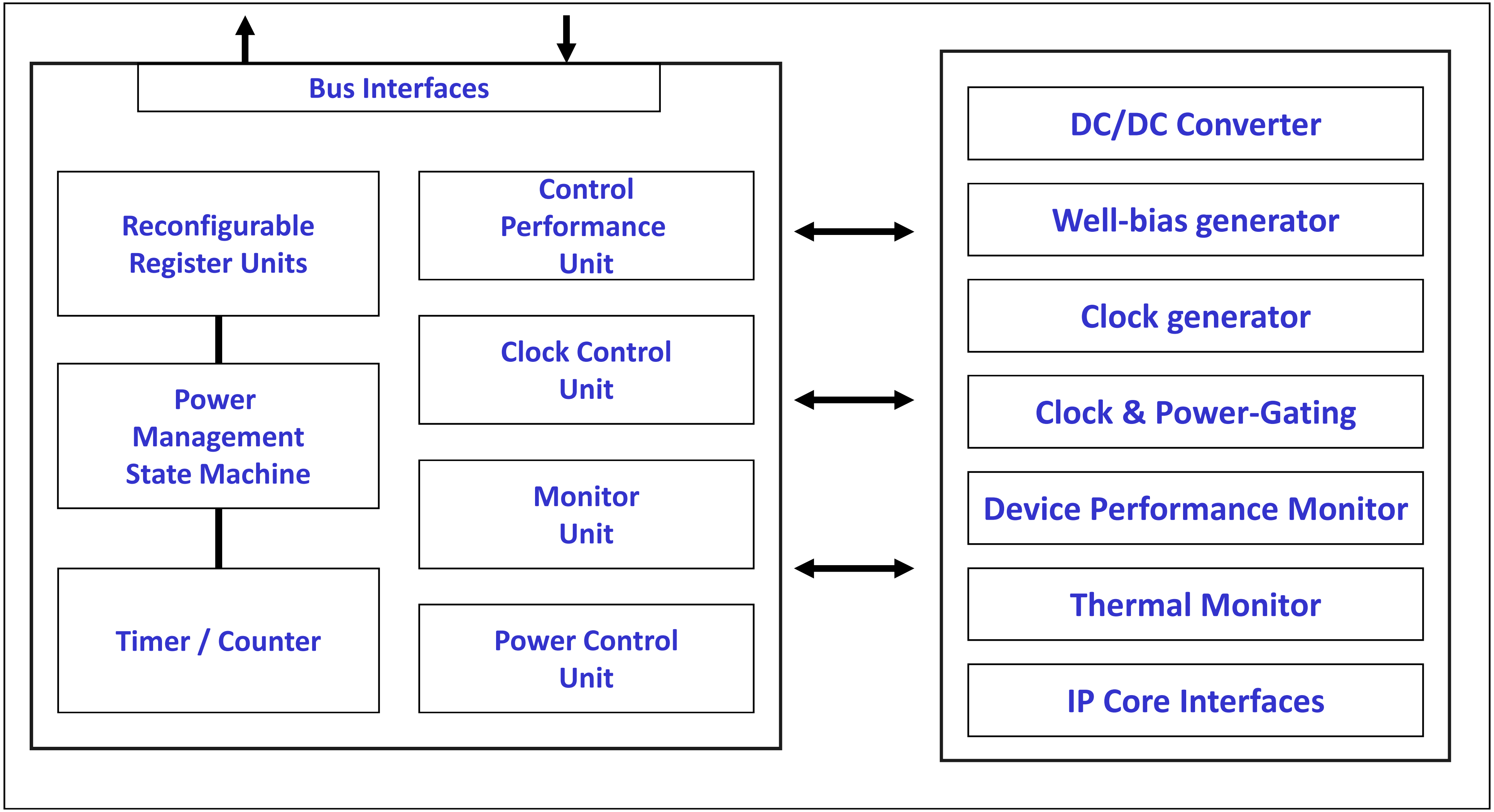
- AVS is a closed loop control mechanism
  - Feedback from the PMU (Power Management Unit) indicates the earliest opportunity to change processor frequency based on the voltage levels being output to the SoC
  - APC (Adaptive Power Controller) monitors the difference between the requested performance level and the actual level achieved
- Taking into account variations due to differences in process technology and ambient temperature the system dynamically changes the voltage applied
- The lowest energy consumption is achieved OR a specified performance level can be met

AVS

- The concept of AVS (Adaptive Voltage Scaling)
    - Power supply voltage is reduced to the absolute minimum required for a given operating mode
    - Automatic on-chip process and temperature compensation
    - Performance monitoring circuit based on process and temperature
    - Process ID (or Voltage ID) indicates process condition and start voltage level



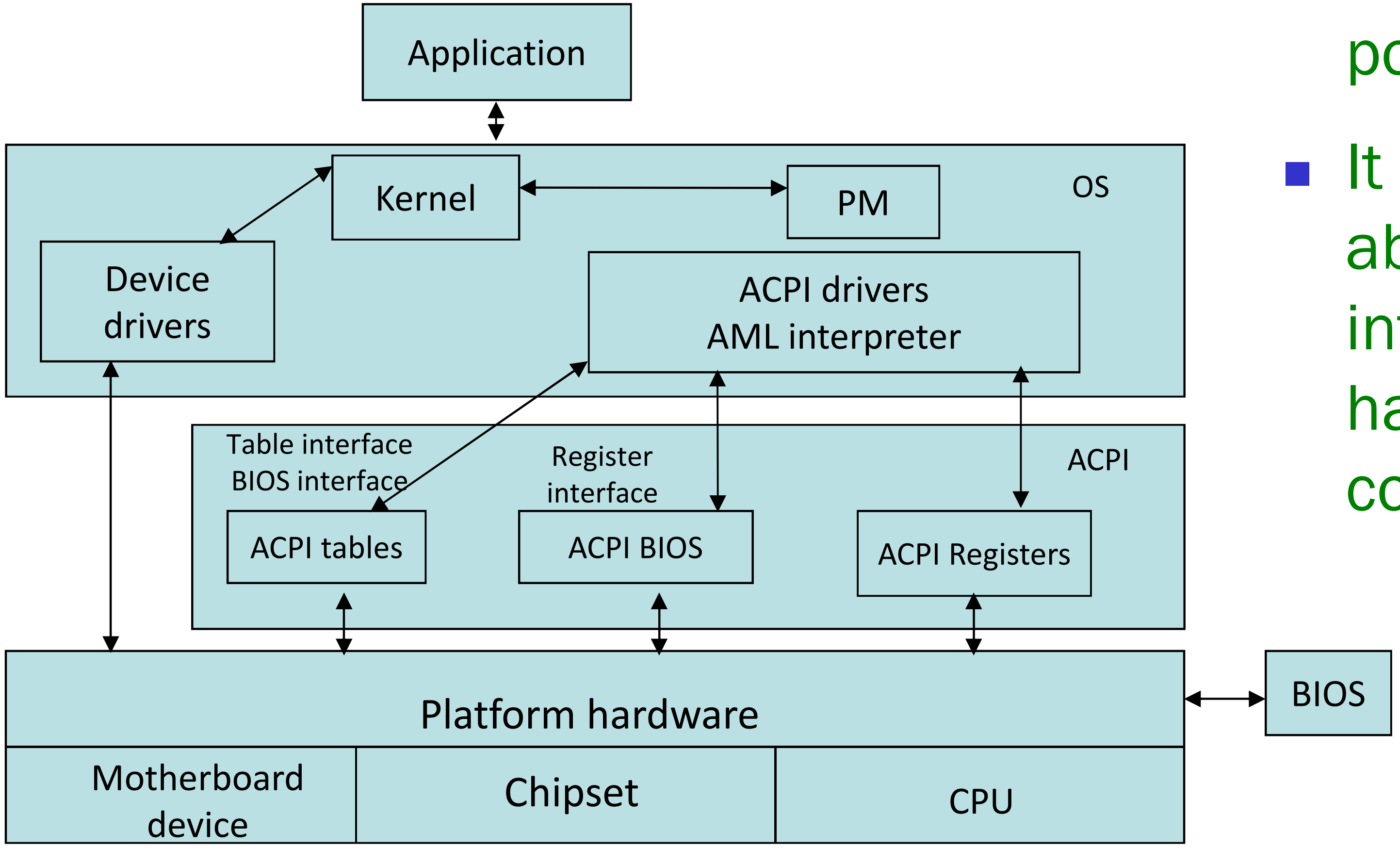
Power Management Unit



Advanced Configuration and Power Interface (ACPI)

- The Advanced Configuration and Power Interface (ACPI) specification is an open standard for unified operating system-centric device configuration and power management
- ACPI, first released in December 1996, defines platform-independent interfaces for hardware discovery, configuration, power management and monitoring
- The specification is central to Operating System-directed configuration and Power Management (OSPM); a term used to describe a system implementing ACPI and therefore removing device management from legacy firmware interfaces
- The standard was originally developed by Intel, Microsoft and Toshiba, and was later joined by HP and Phoenix
    - The latest version is "Revision 5.0", which was published on November 23, 2011.

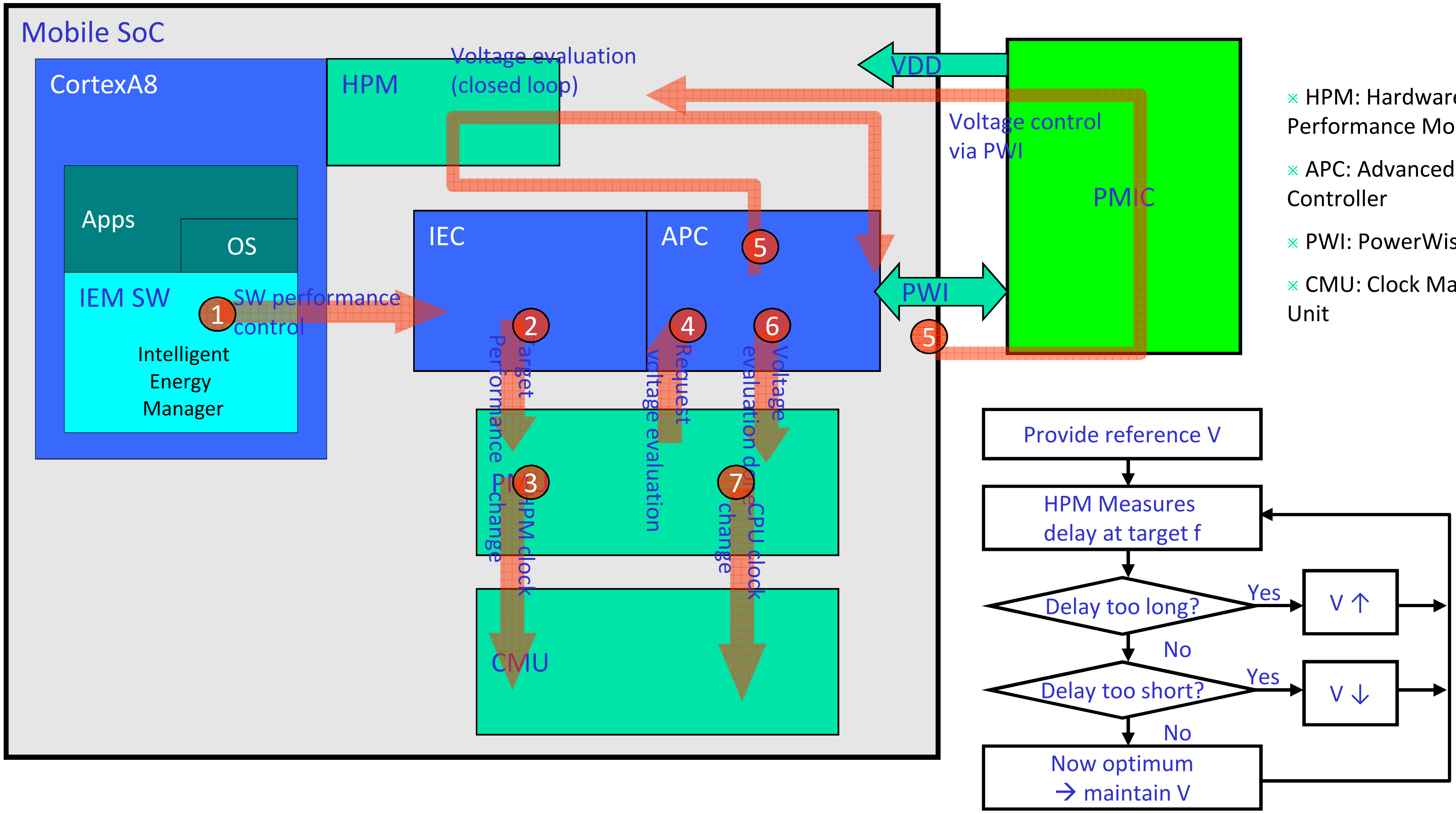
Advanced Configuration and Power Interface (ACPI)



- ACPI is a uniform HW/SW interface for power management
- It specifies an abstract and flexible interface between hardware components

System-level Low Power Design – ARM IEM^{TM}

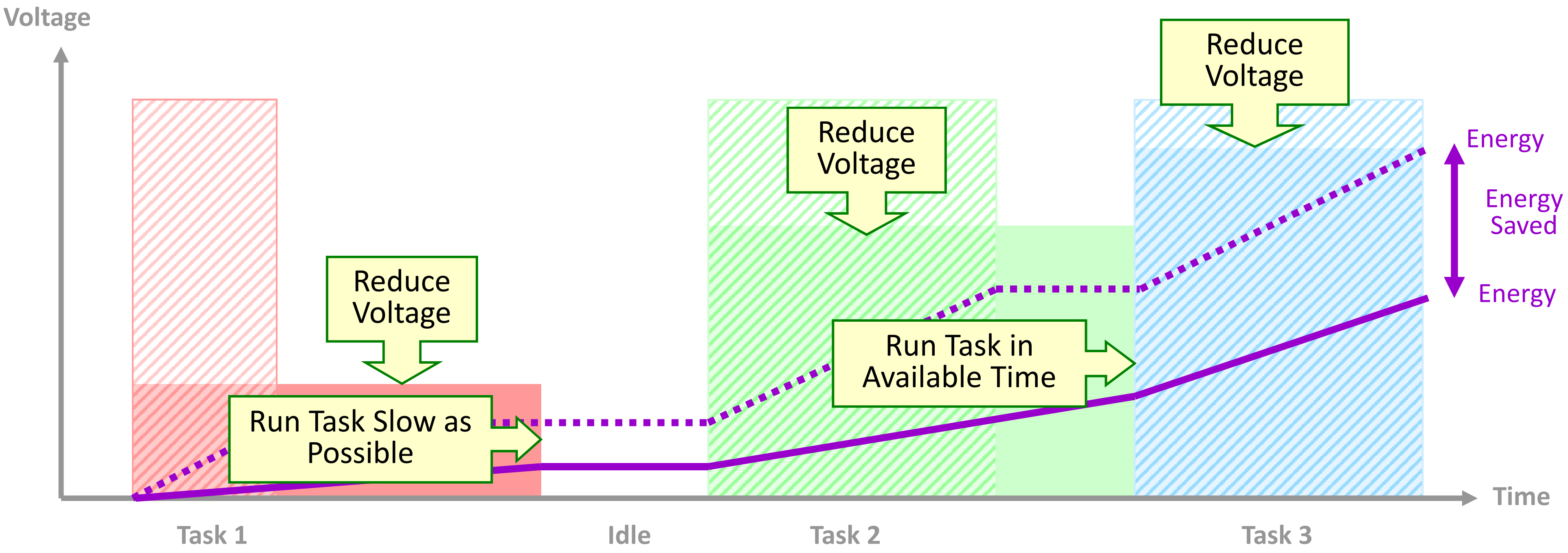
- IEM (Intelligent Energy Management) = AVS



- HPM: Hardware Performance Monitor
- APC: Advanced Power Controller
- PWI: PowerWise Interface
- CMU: Clock Management Unit

ARM IEM Principles

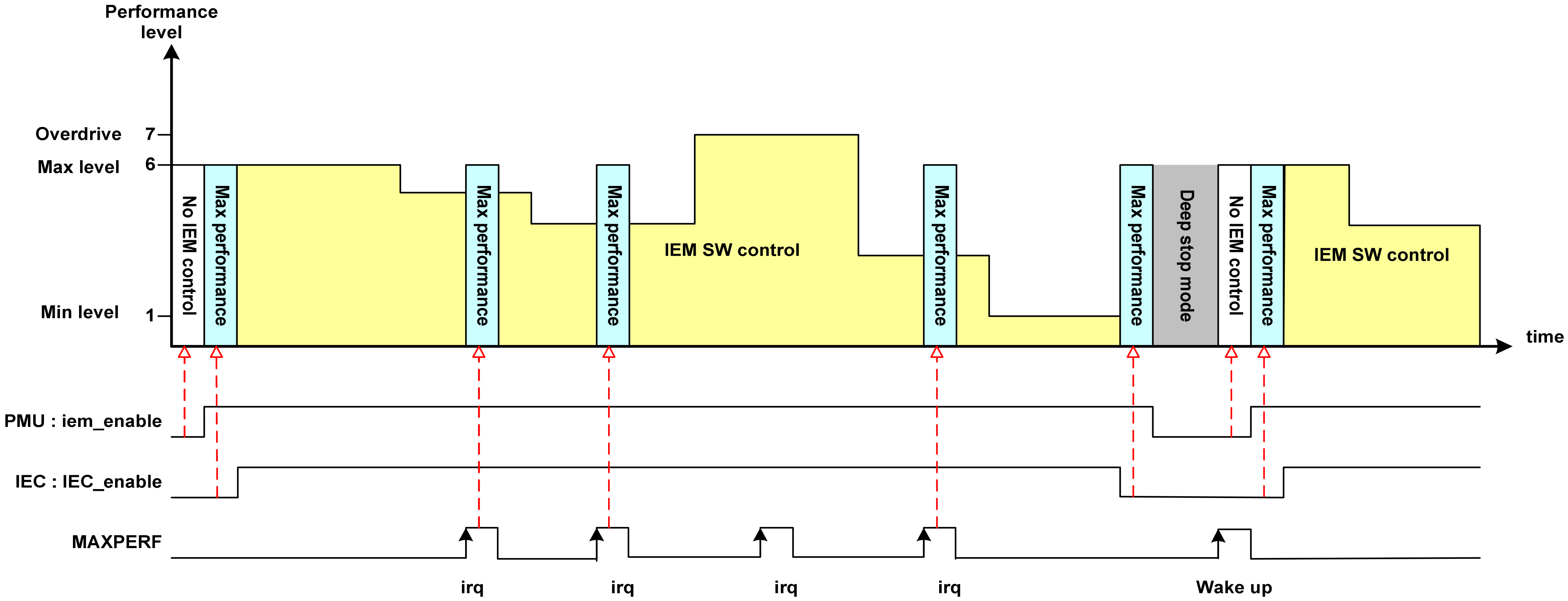
- Batteries have finite amounts of energy stored in them
- Running fast and then idling wastes energy



Only need to run just fast enough to meet the application deadlines

System-level Low Power Design – IEM

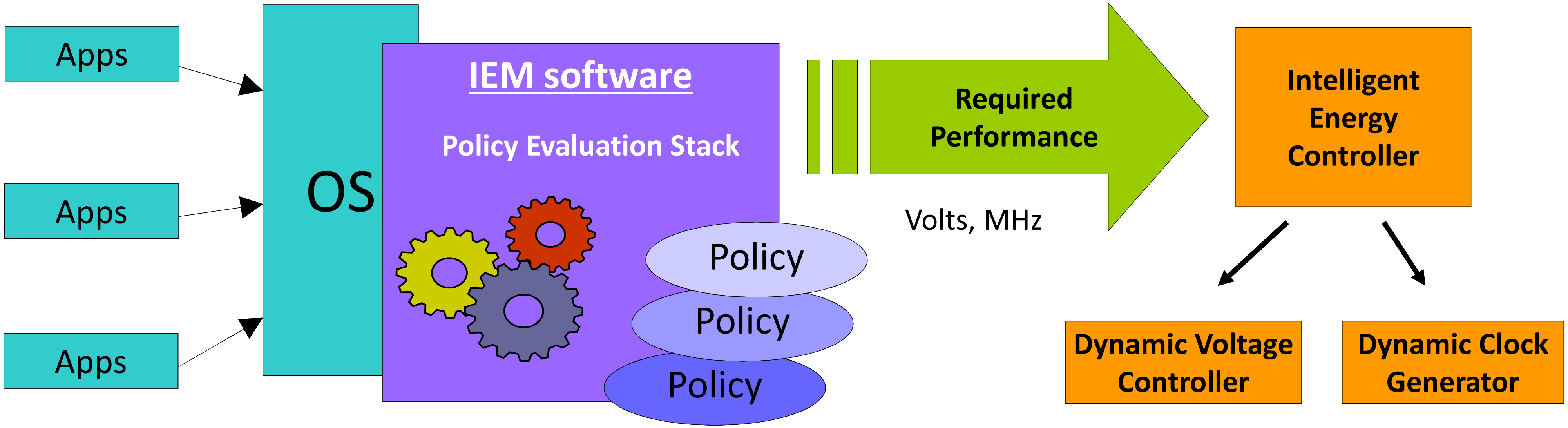
- IEM scenario example



ARM IEM Technology

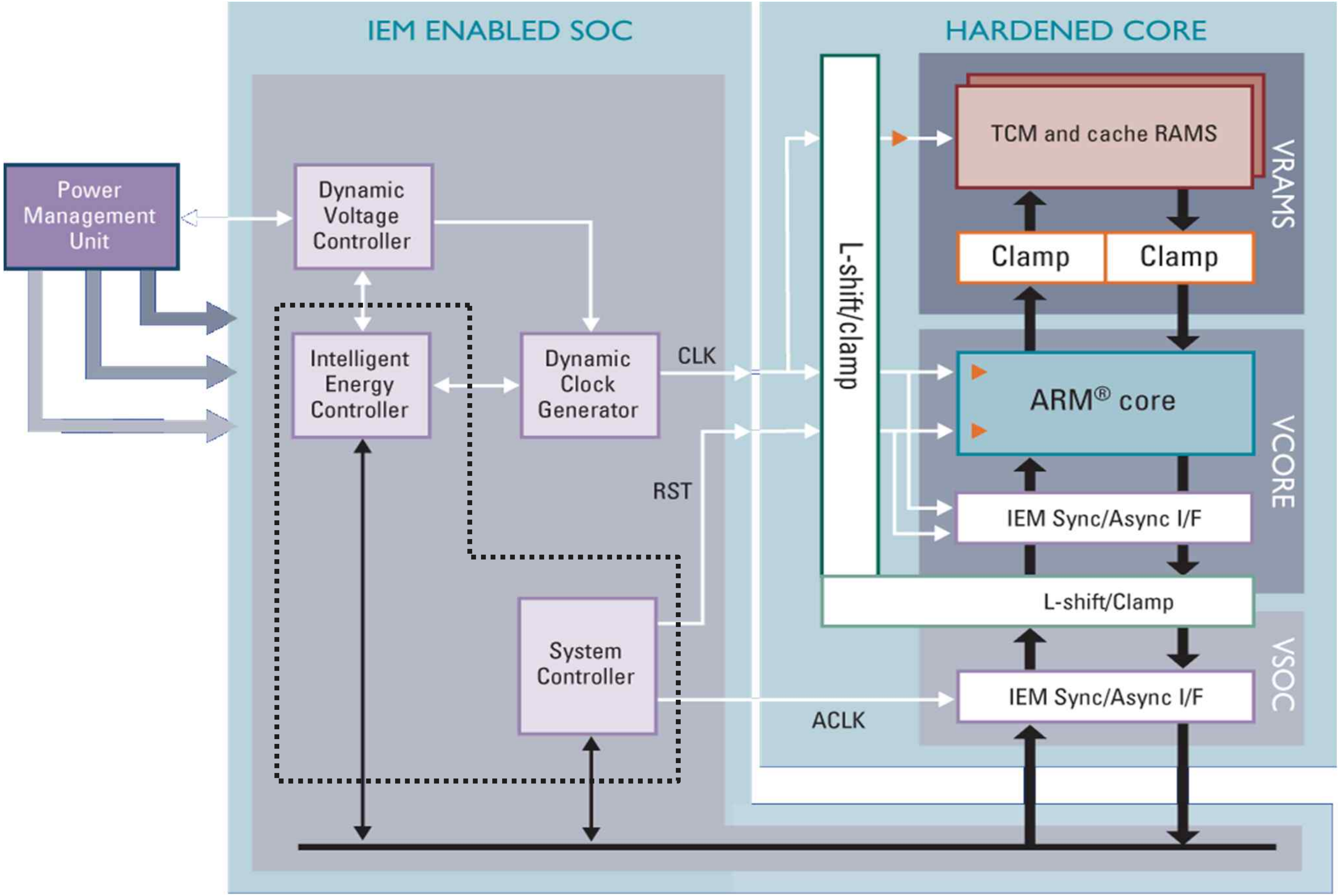
Hardware and software solution for energy management

Dynamic control of voltage and frequency scaling.



- IEM software connects to OS kernel and collects data
- Multiple policies categorize the software workload
- Prediction of future performance requirement is made
- Suitable operating point (Voltage and Frequency) is set

ARM IEM System Implementation



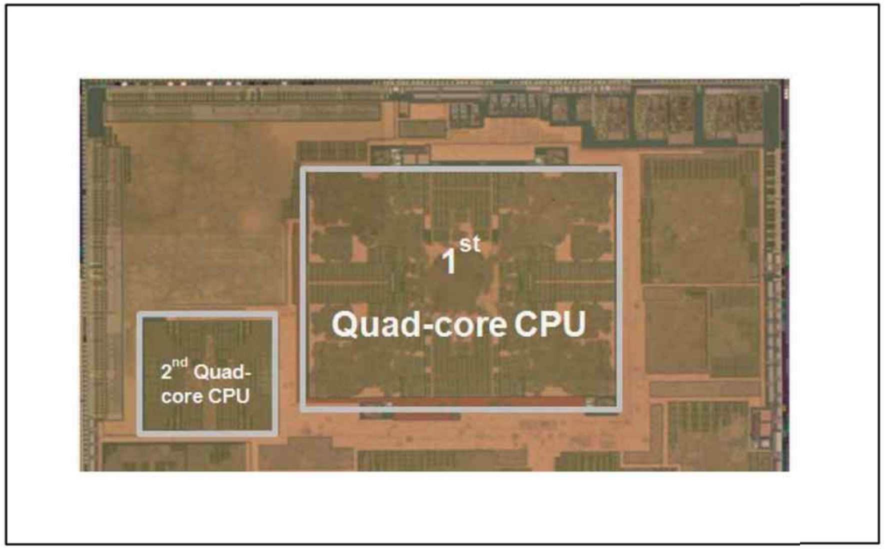
Announcement

- Homework #3
  - Reading assignment
    - http://dl.acm.org/citation.cfm?id=996798
    - Report (summary) format: MS Word, 3 page, 11pt
    - Due data: Oct. 7 (Mon) – in the classroom only at the beginning of class time

System-level Power Optimization Techniques: ARM big.LITTLE

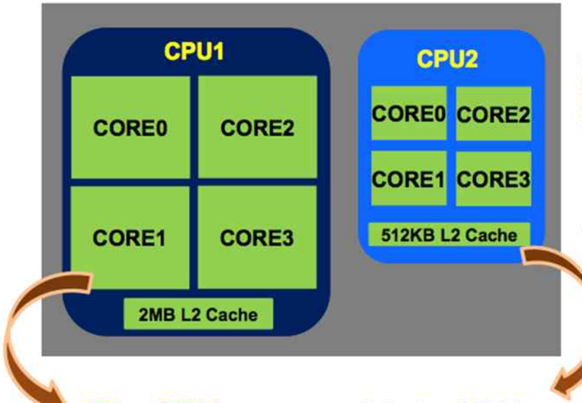
Recall: Samsung Exynos 5410 Octa - for Galaxy S4

Die Photo



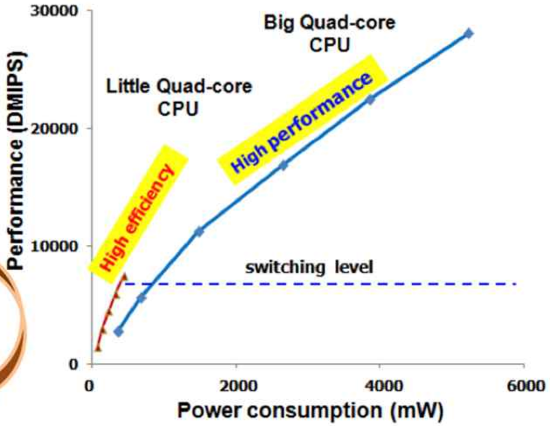
Heterogeneous CPU operation

- Two Heterogeneous Quad-core CPUs for
  - Can be switched based on task and work loads.
  - Efficient power consumption with Maximized performance.



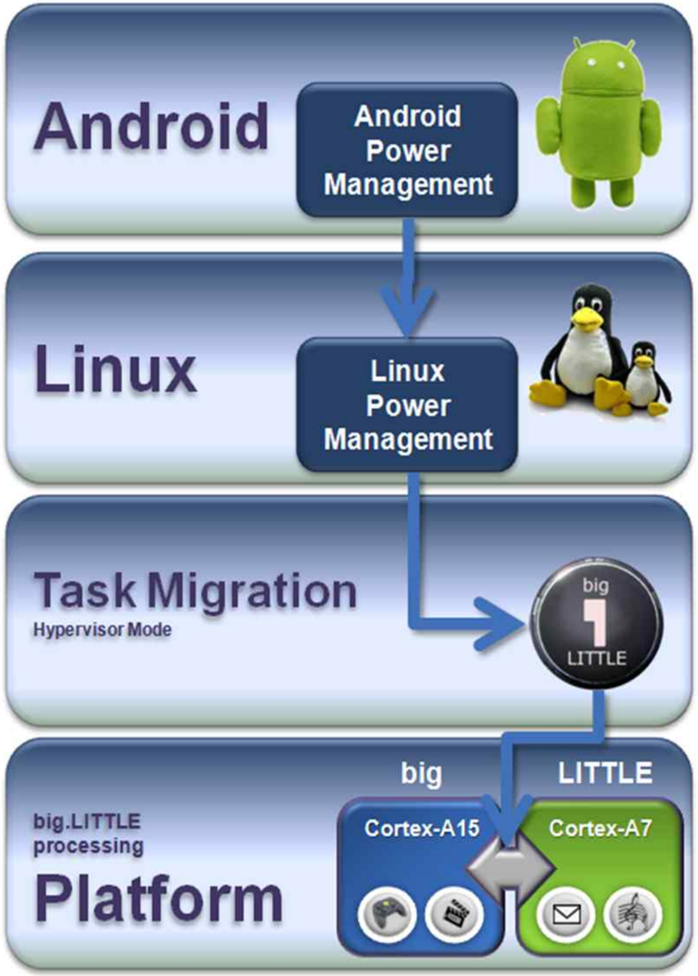
Big CPU
High performance for compute intensive applications

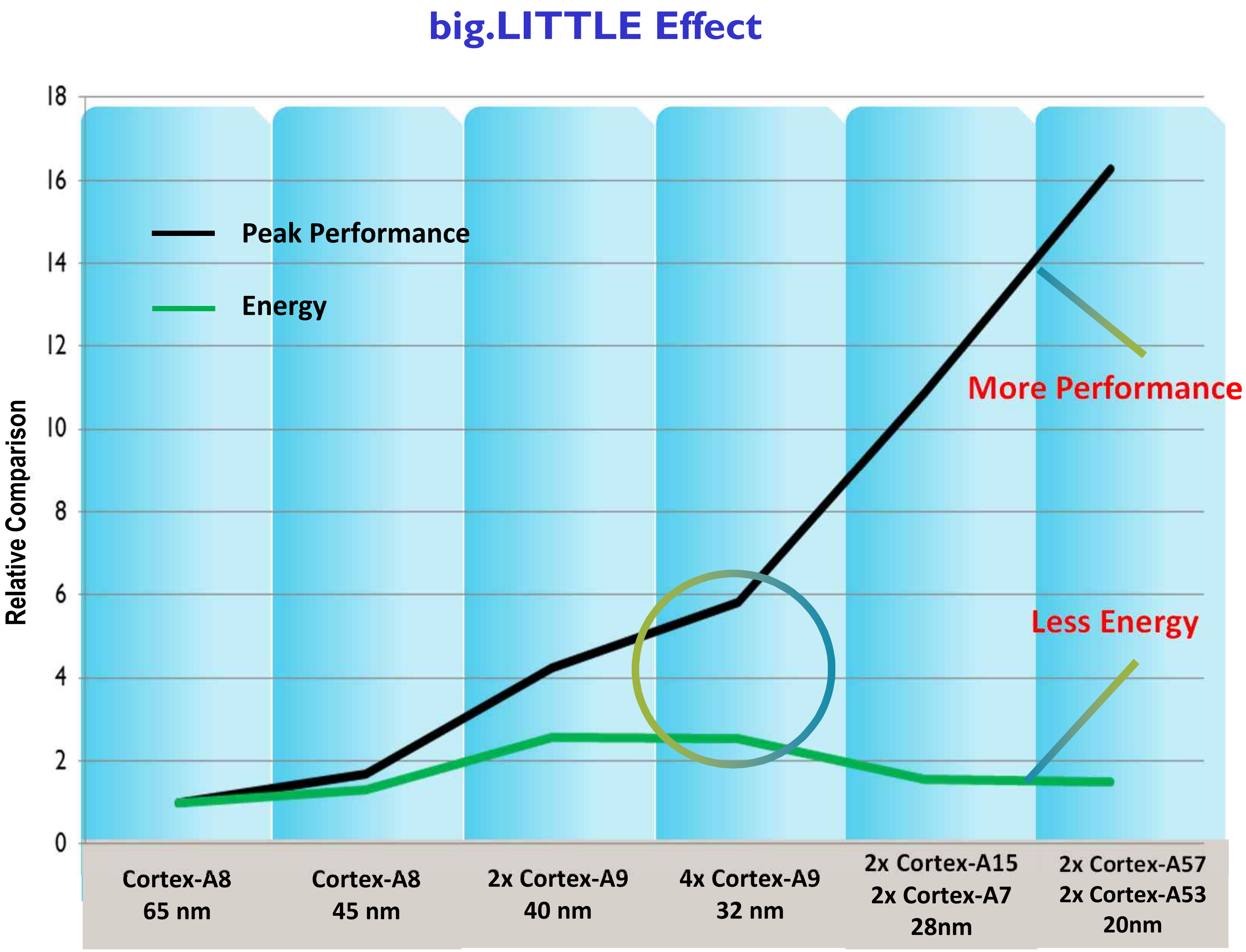
Little CPU
Low power execution of majority workloads



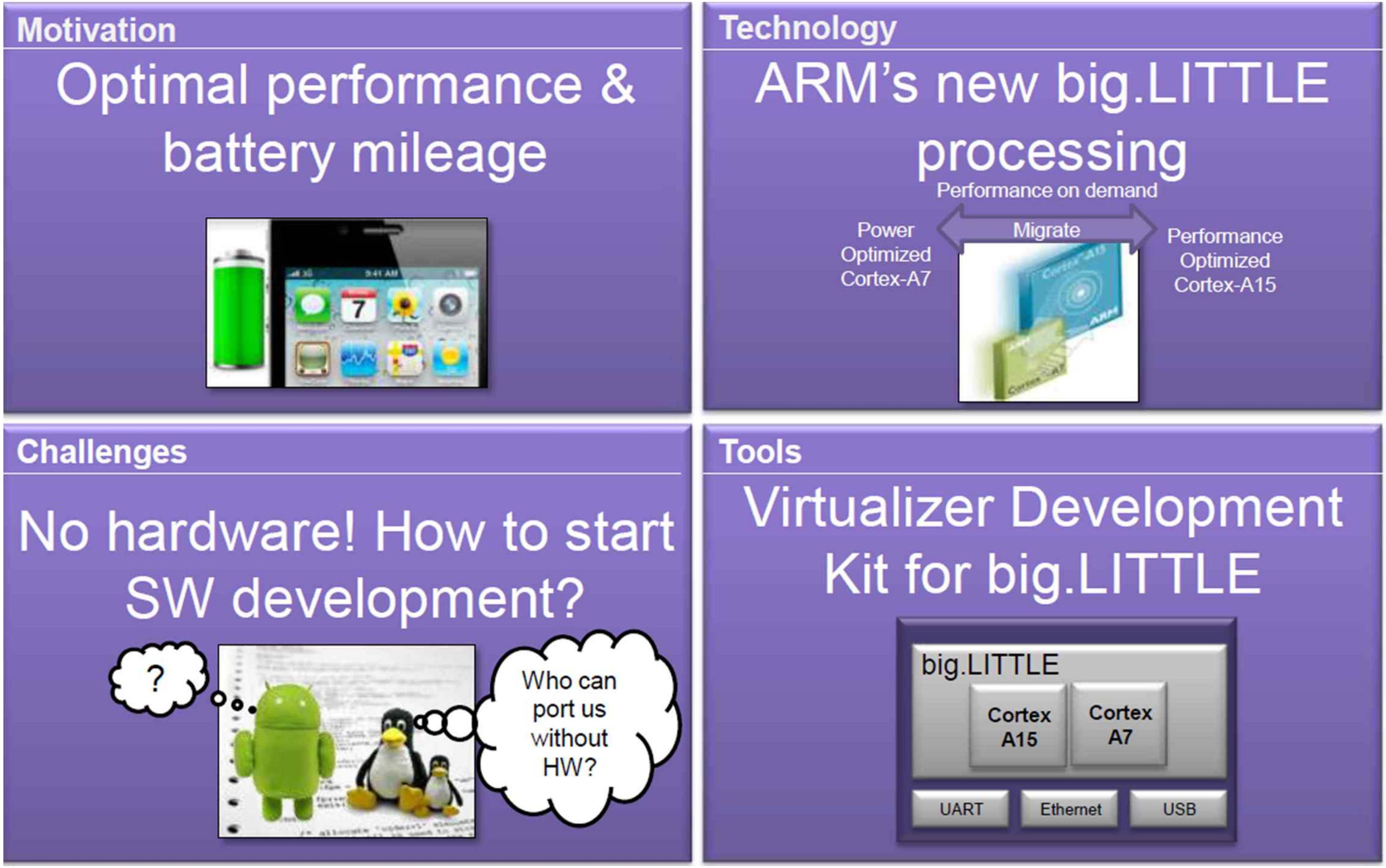
	1st Quad-core CPU	2nd Quad-core CPU
Architecture	ARM v7a	ARM v7a
Process	Samsung 28nm HKMG	Samsung 28nm HKMG
Frequency	200MHz~1.8GHz+	200MHz~1.2GHz+
Area	19mm^2	3.8mm^2
Power-ratio	1	0.17
L1 Cache size	32KB I/D cache	32KB I/D cache
L2 Cache size	2MB Data cache	512KB Data cache

Recall: ARM big.LITTLE Architecture for Low Power





big.LITTLE processing in a nutshell – high level



Motivation

Optimal performance & battery mileage

Technology

ARM's new big.LITTLE processing

Performance on demand
Migrate
Power Optimized Cortex-A7
Performance Optimized Cortex-A15

Challenges

No hardware! How to start SW development?

?
Who can port us without HW?

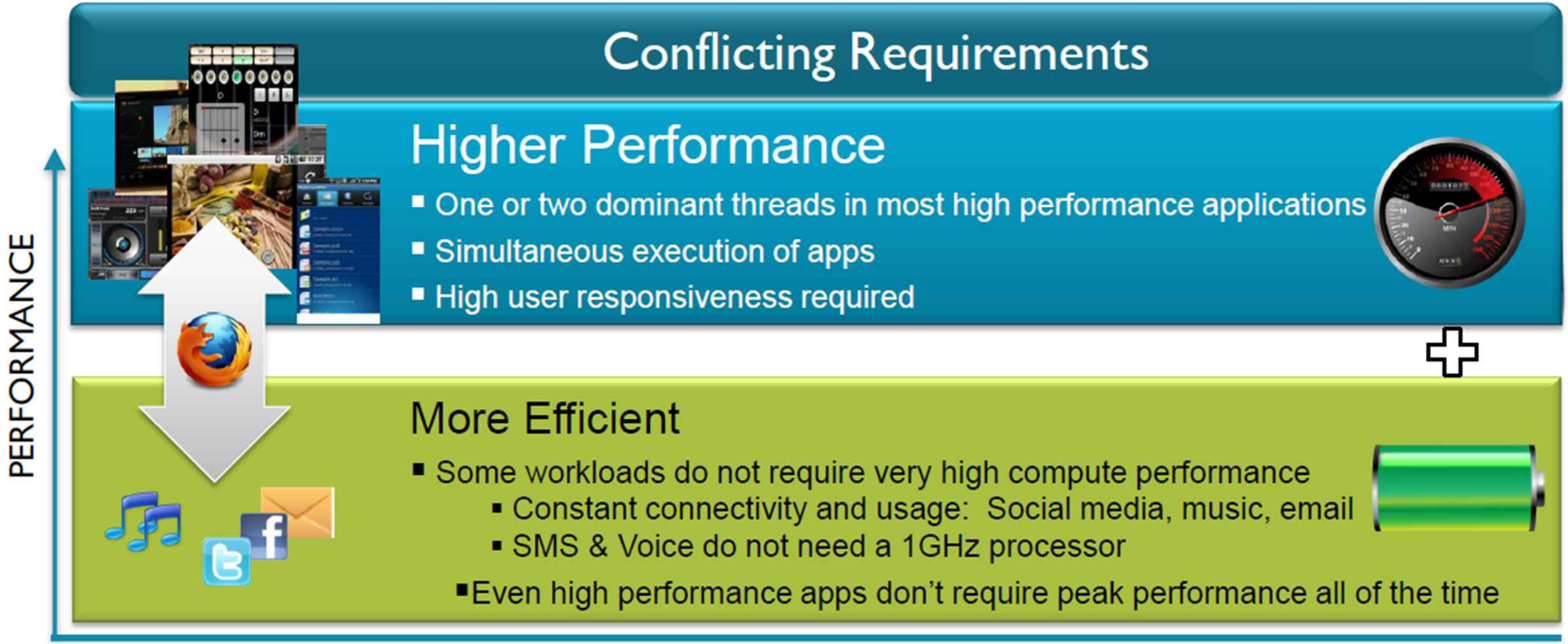
Tools

Virtualizer Development Kit for big.LITTLE

big.LITTLE
Cortex A15
Cortex A7
UART
Ethernet
USB

Why big.LITTLE?

- All workloads are not equal
- Applications do not require high performance all of the time
- One size core does not fit all



Conflicting Requirements

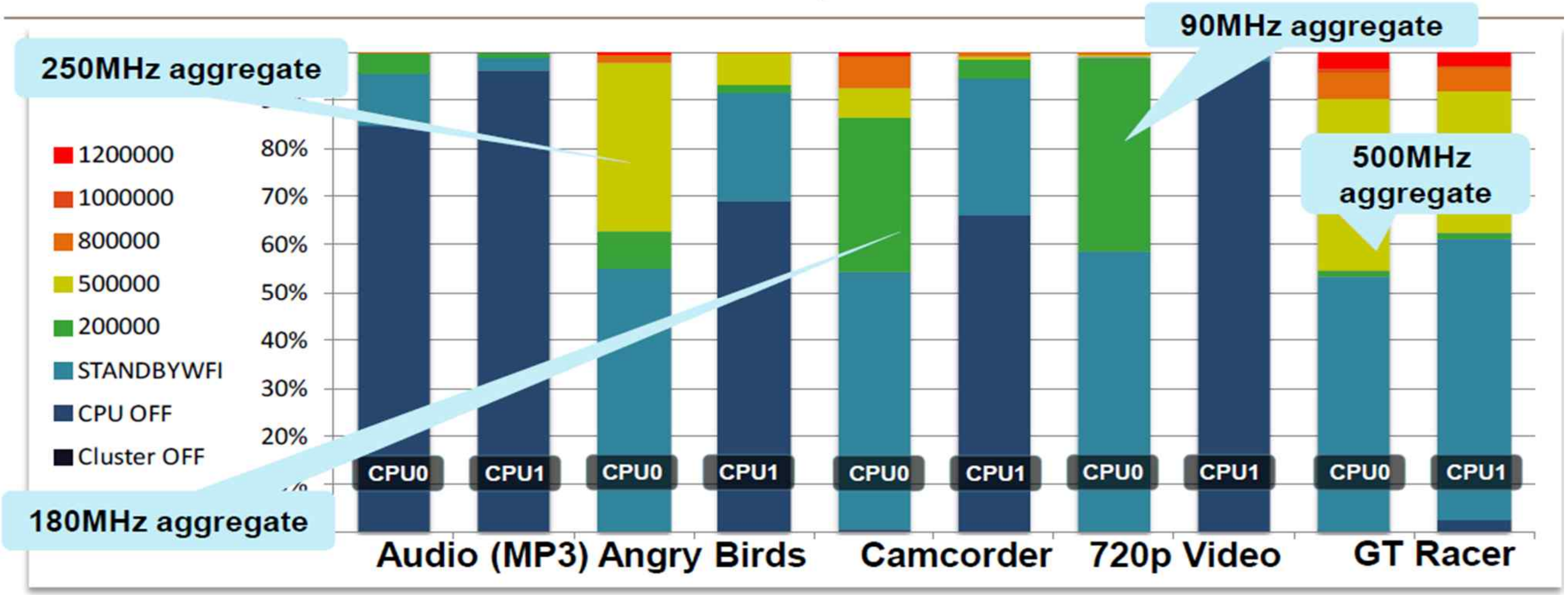
Higher Performance

- One or two dominant threads in most high performance applications
- Simultaneous execution of apps
- High user responsiveness required

More Efficient

- Some workloads do not require very high compute performance
    - Constant connectivity and usage: Social media, music, email
    - SMS & Voice do not need a 1GHz processor
- Even high performance apps don't require peak performance all of the time

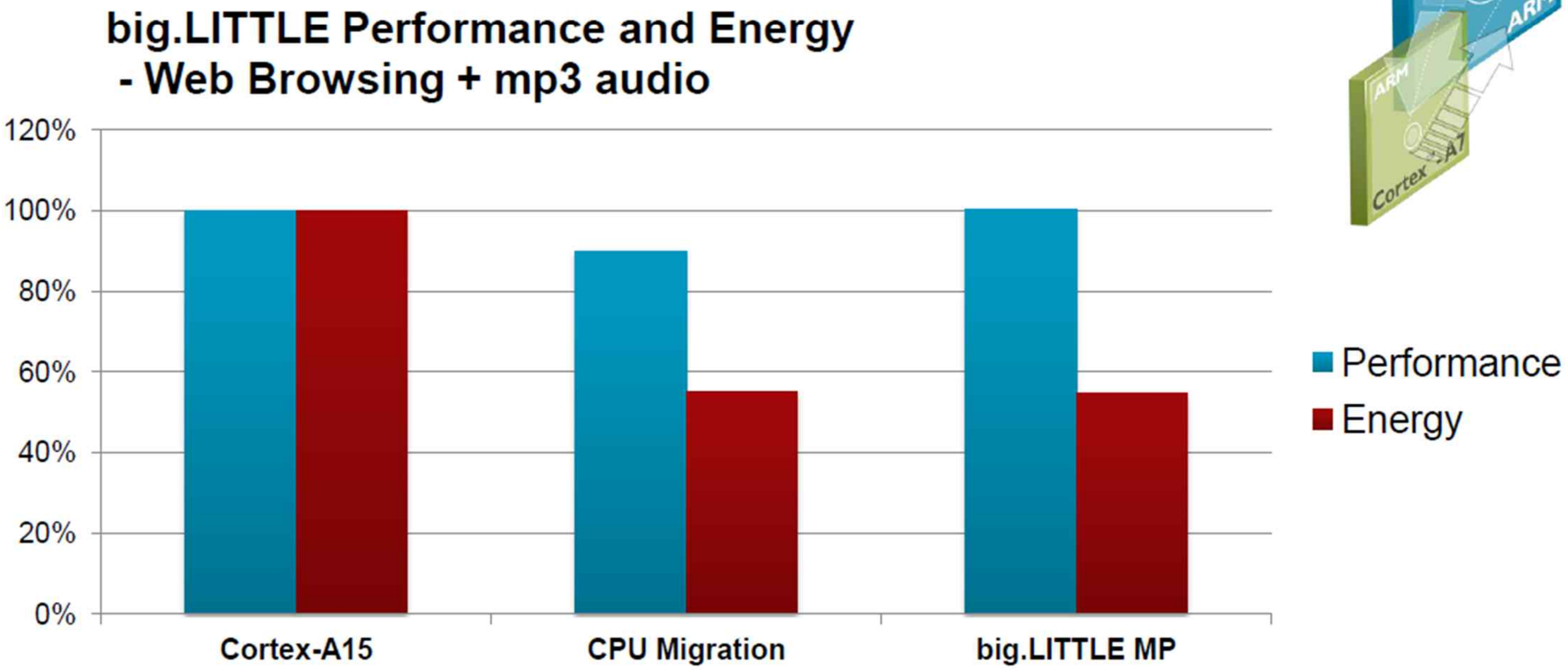
Low to Medium Intensity Use Cases



- DVFS profiles from leading Dual Cortex-A9 Smartphone
- Demonstrates that many common applications require only low to moderate processing power
- All of these use cases will run predominantly on the LITTLE cores
- The small % peaks to high MHz don’t necessarily require migration to big cores
  - The short term DVFS system response to any increase in average load is to go to the highest MHz

big.LITTLE Measured Results

~50% energy saving for same performance

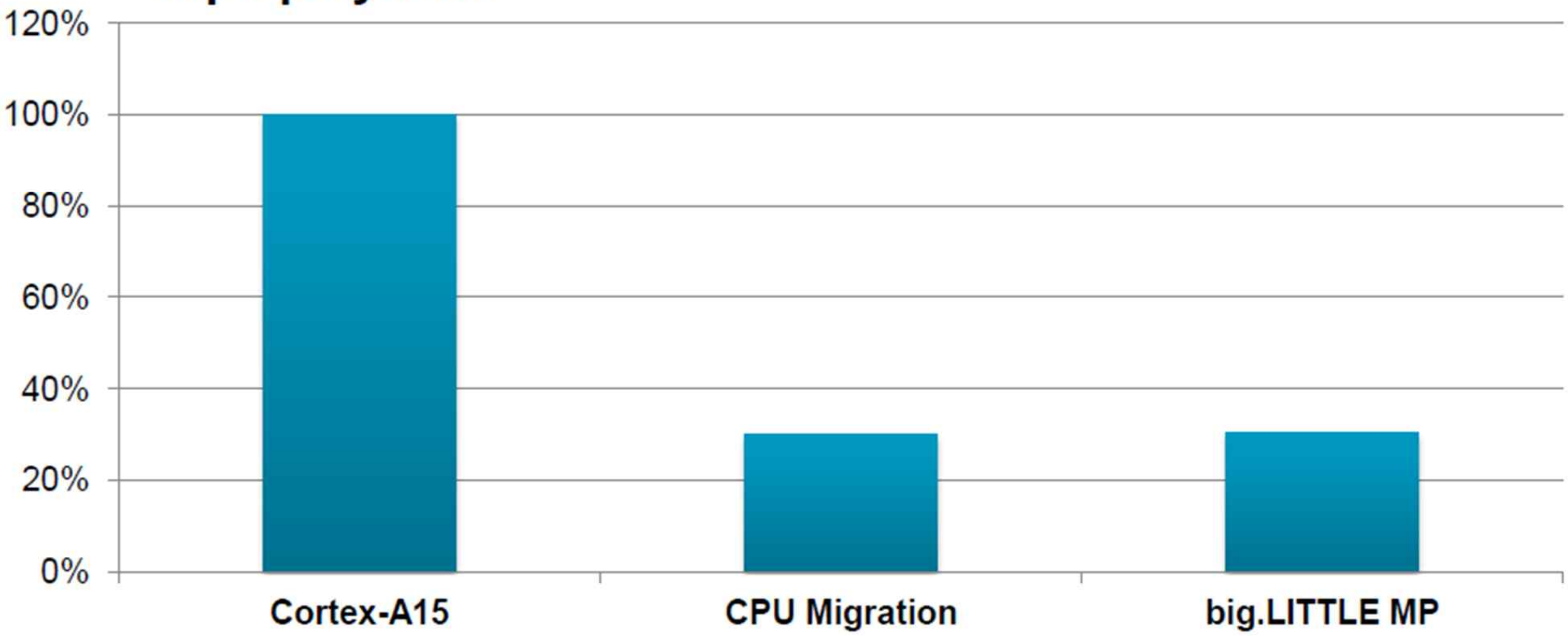


Optimized Power and Responsiveness

big.LITTLE Measured Results

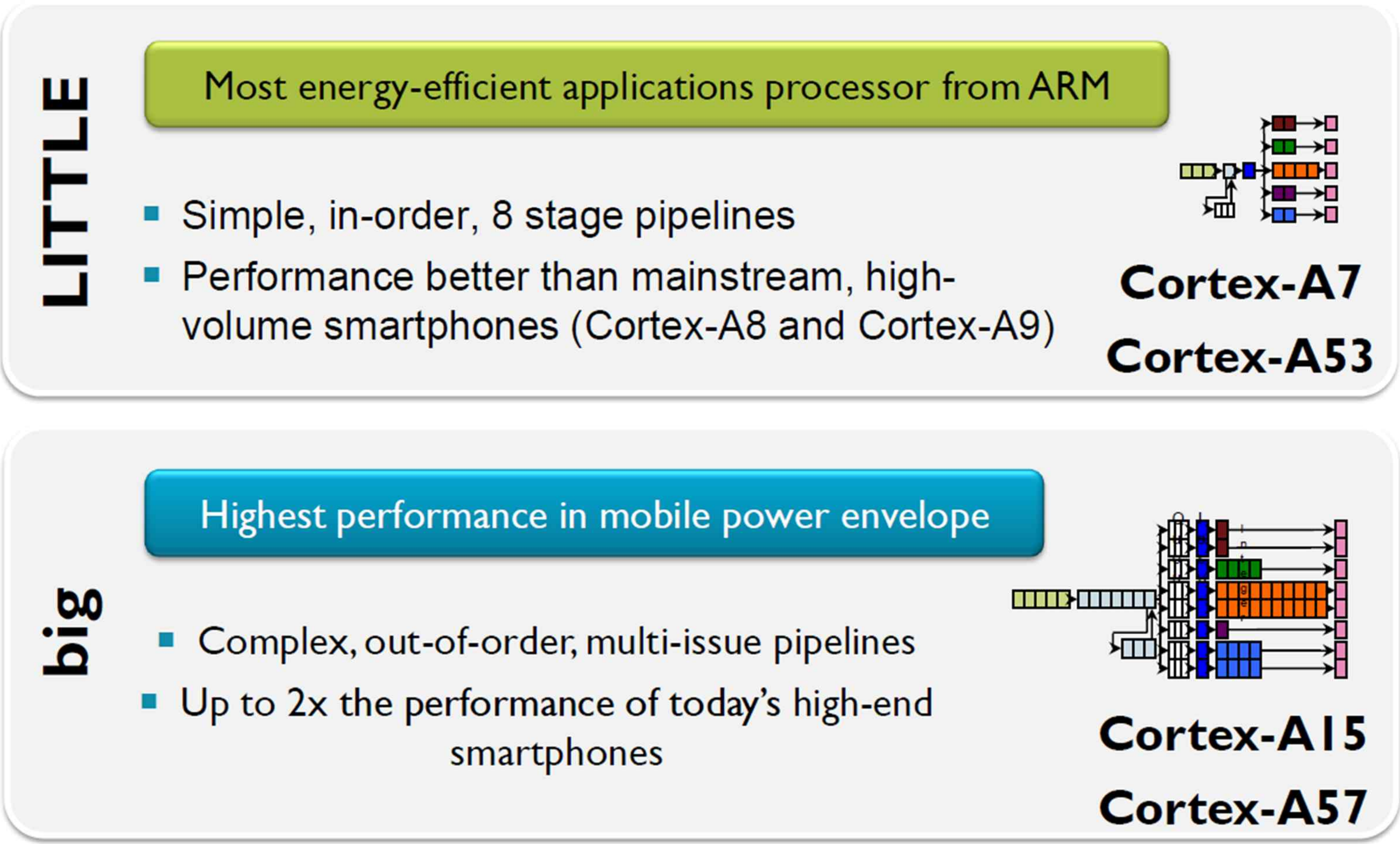
~70% energy saving for same performance

big.LITTLE Energy Consumption
- mp3 playback



Optimized Power and Responsiveness

Fine-Tuned to Different Performance Points



LITTLE

Most energy-efficient applications processor from ARM

- Simple, in-order, 8 stage pipelines
- Performance better than mainstream, high-volume smartphones (Cortex-A8 and Cortex-A9)

Cortex-A7

Cortex-A53

big

Highest performance in mobile power envelope

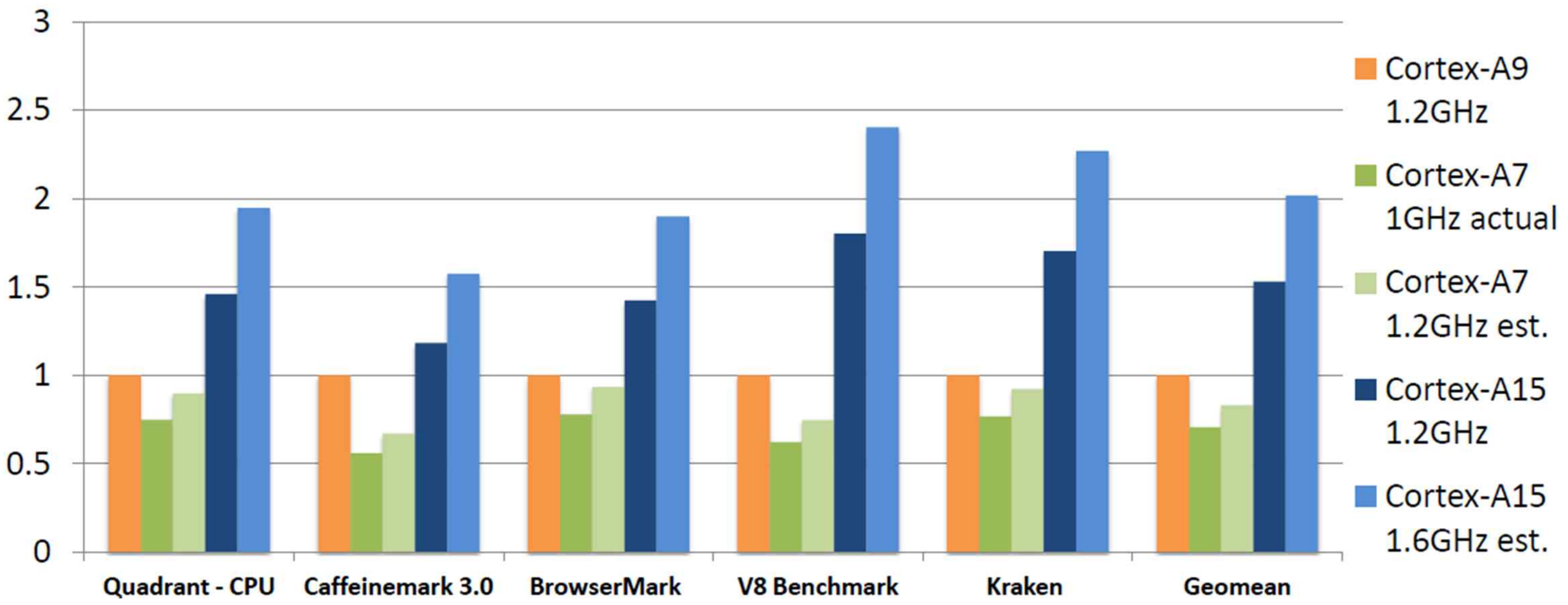
- Complex, out-of-order, multi-issue pipelines
- Up to 2x the performance of today's high-end smartphones

Cortex-A15

Cortex-A57

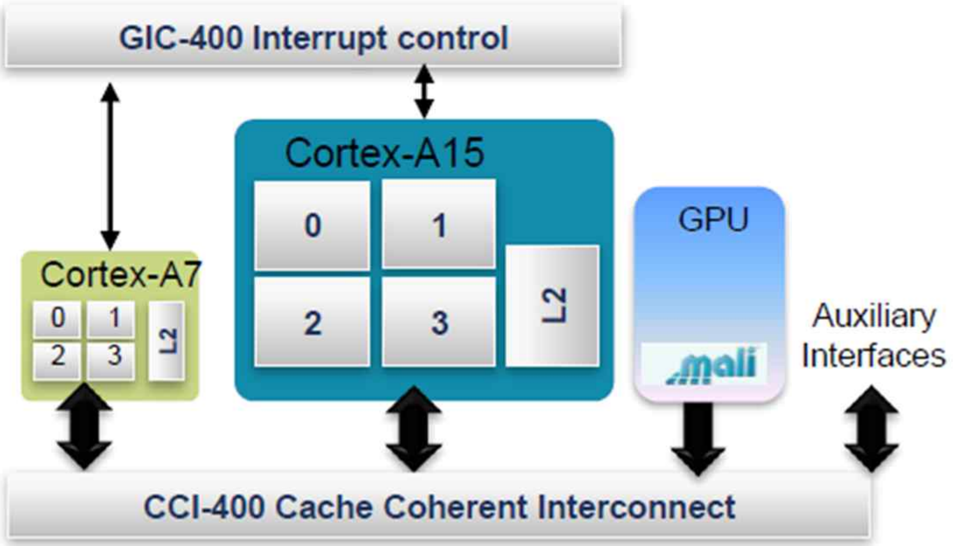
big and LITTLE CPU Performance

- Cortex-A9 powers high end mobile devices today
- Cortex-A7 delivers comparable performance
  - ...at lower power and area
- Cortex-A15 delivers significantly higher performance



Putting Together a big.LITTLE System

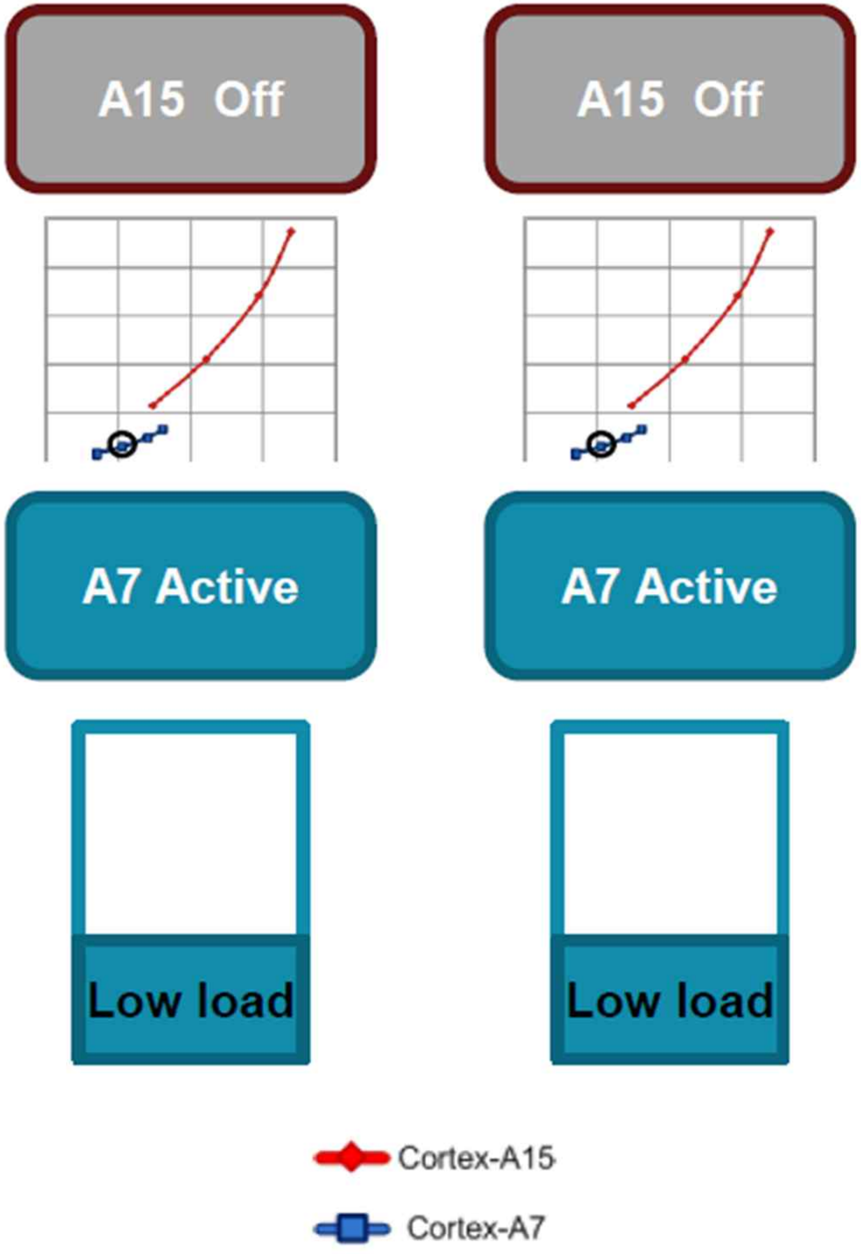
- Programmer's view of hardware
  - High performance Cortex-A15 CPU cluster
  - Energy efficient Cortex-A7 CPU cluster
  - CCI-400 Cache Coherent Interconnect maintains cache-coherency between clusters
  - GIC-400 provides transparent virtualized Interrupt control
- Same Hardware Supports all big.LITTLE Software Models


GIC-400 Interrupt control
Cortex-A15
0
1
2
3
L2
Cortex-A7
0
1
2
3
L2
GPU
mali
Auxiliary Interfaces
CCI-400 Cache Coherent Interconnect

big.LITTLE Software

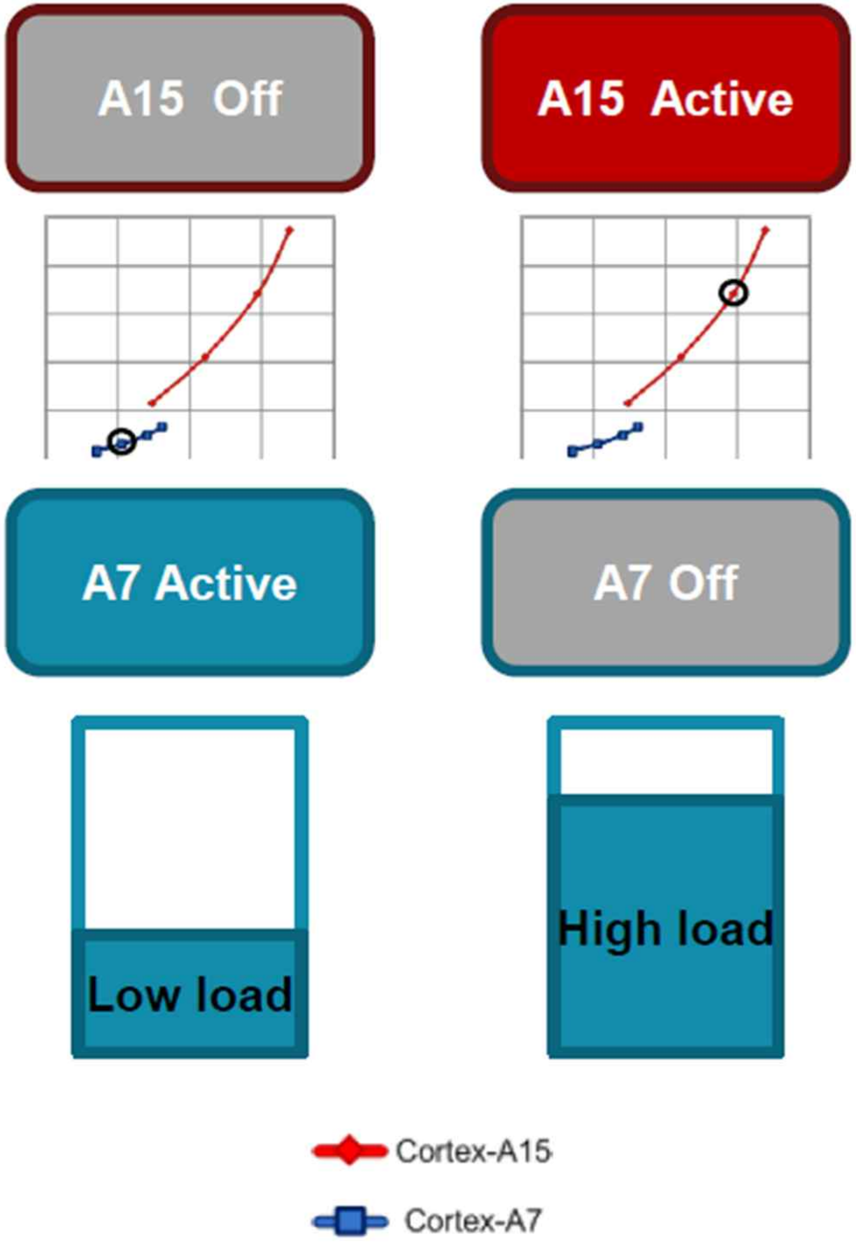
- CPU Migration
  - Migrate a single processor workload to the appropriate CPU
    - Migration = save context then resume on another core
    - Also known as Linaro “In Kernel Switcher”
  - Implemented via modified DVFS driver modifications and kernel modifications
    - Based on standard power management routines
    - Small modification to OS and DVFS, ~600 lines of code
- big.LITTLE MP
  - OS scheduler moves threads/tasks to appropriate CPU
    - Based on CPU workload
    - Based on dynamic thread performance requirements
  - Enables highest peak performance by using all cores at once

CPU Migration



- big.LITTLE extends DVFS
- DVFS algorithm monitors load on each CPU
- When load is low it can be handled on a LITTLE processor
- When load is high the context is transferred to a big processor
- The unused processor can be powered down
- When all processors in a cluster are inactive the cluster and its L2 cache can be powered down

CPU Migration



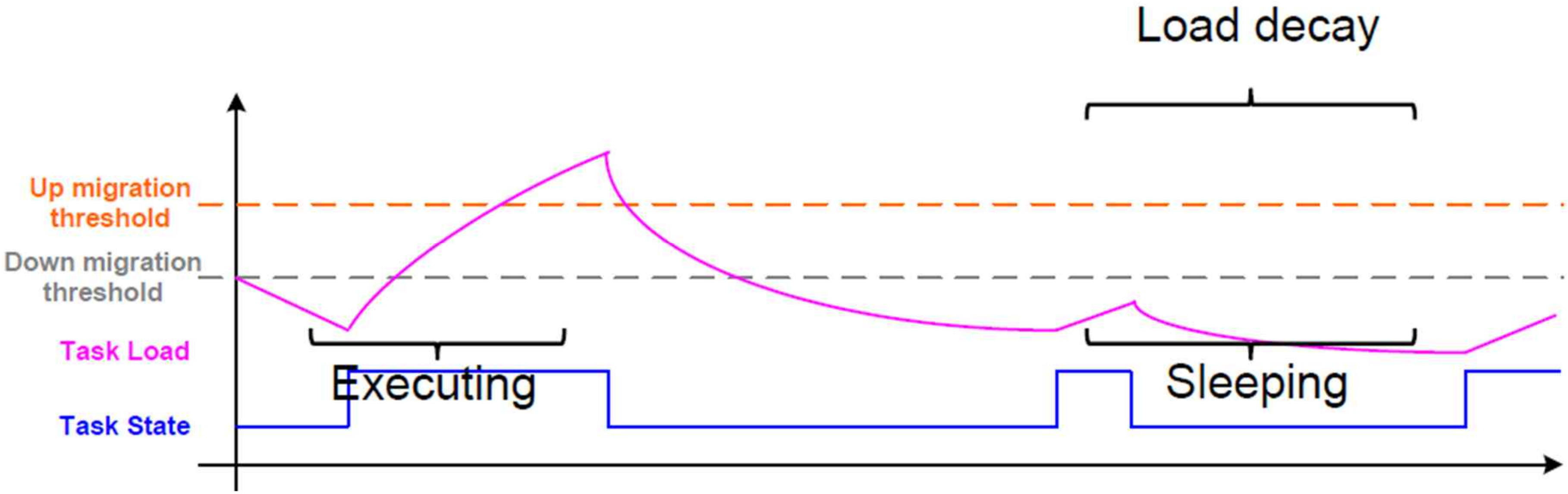
- big.LITTLE extends DVFS
- DVFS algorithm monitors load on each CPU
- When load is low it can be handled on a LITTLE processor
- When load is high the context is transferred to a big processor
- The unused processor can be powered down
- When all processors in a cluster are inactive the cluster and its L2 cache can be powered down

big.LITTLE MP

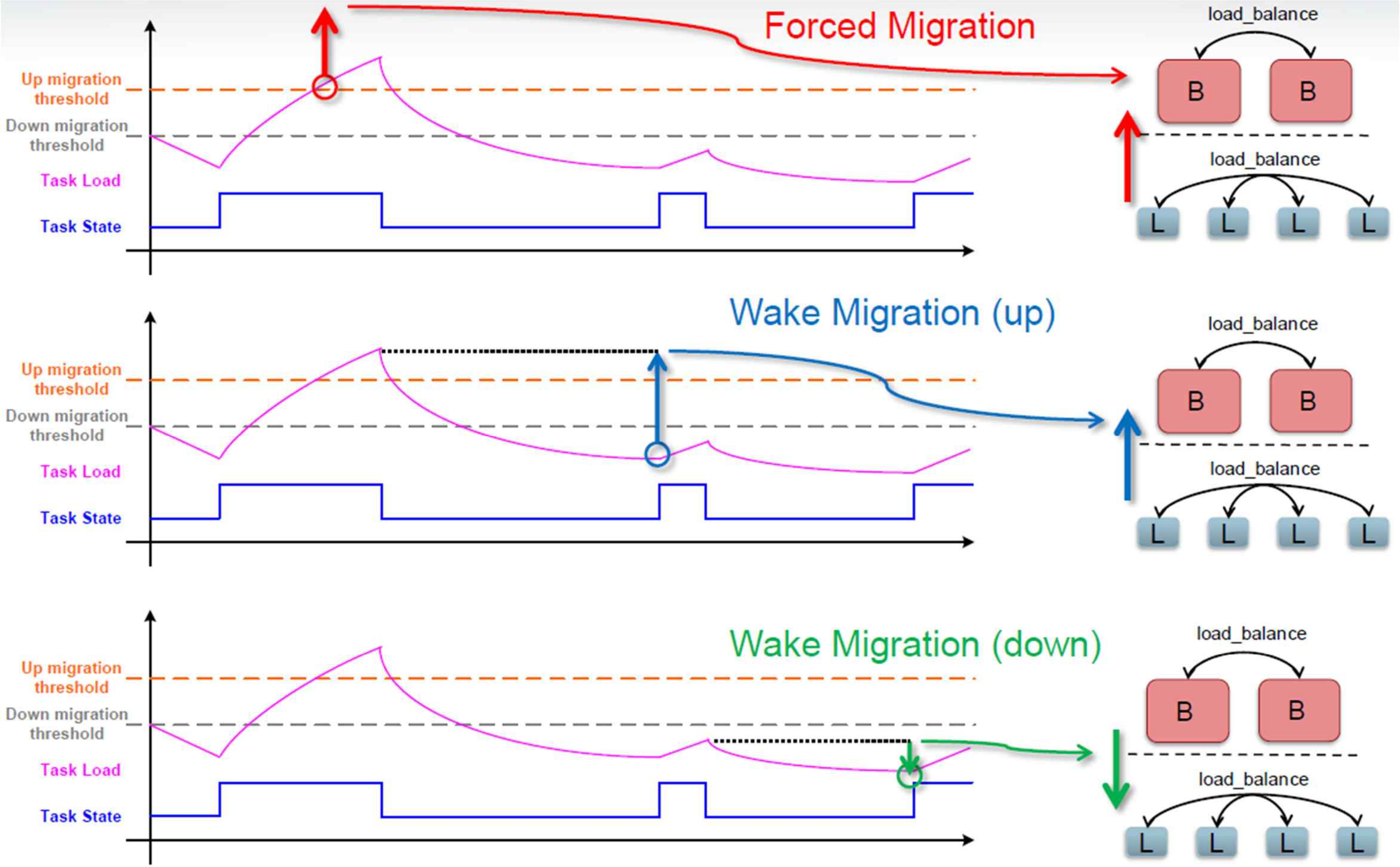
- The OS scales across all cores in the system
  - Any thread can run on any CPU
  - OSPM will idle unused CPUs
- OS scheduler is aware of the system topology
  - Scheduler is aware of big and LITTLE clusters with different performance characteristics
  - Scheduler chooses appropriate CPU for each individual thread based on historical measured workload
  - Works well with imbalanced workloads
  - Support clusters with different numbers of cores
  - Can use all cores or any combination of cores

big.LITTLE MP in Action

- The scheduler maintains a history of load on a per thread basis
- Migration thresholds are used to decide whether a thread should be placed on a big or LITTLE core;
  - **Wakeup migration:** The load level at the last time the thread exited the run queue to select appropriate CPU
  - **Forced migration:** Periodically, the load of current task on each LITTLE core is updated and if needed the task is migrated to the big core



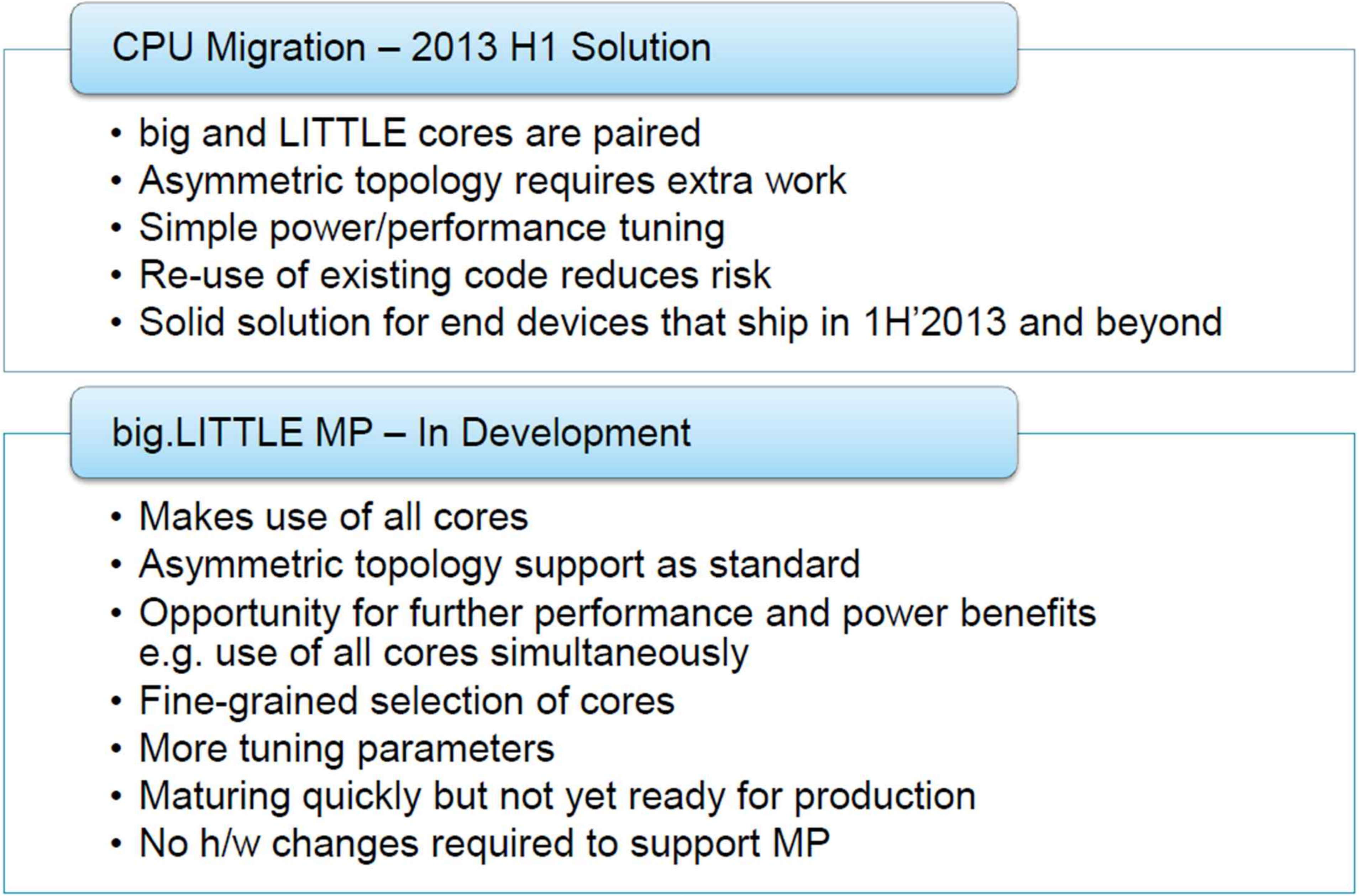
big.LITTLE MP in Action



Accessing big.LITTLE Software

- Big.LITTLE MP
  - Repository
    - http://git.linaro.org/gitweb?p=arm/big.LITTLE/mp.git;a=summary
    - Branches have version names „-Vx" to indicate major revisions of patch sets
  - Join the discussion forum
    - http://lists.linaro.org/mailman/listinfo/linaro-sched-sig
- CPU Migration
  - Available to Linaro members only, currently
  - Scheduled to be upstreamed in 2013

Which big.LITTLE model to choose?



CPU Migration – 2013 H1 Solution

- big and LITTLE cores are paired
- Asymmetric topology requires extra work
- Simple power/performance tuning
- Re-use of existing code reduces risk
- Solid solution for end devices that ship in 1H’2013 and beyond

big.LITTLE MP – In Development

- Makes use of all cores
- Asymmetric topology support as standard
- Opportunity for further performance and power benefits e.g. use of all cores simultaneously
- Fine-grained selection of cores
- More tuning parameters
- Maturing quickly but not yet ready for production
- No h/w changes required to support MP

big.LITTLE Summary

- Modern applications require a mix of high performance and high efficiency
- big.LITTLE technology provides the best user experience for the lowest possible energy
- Commercial chips employing ‘big.LITTLE’ are available now
  - Energy savings confirmed on a range of workloads
  - Further tuning and optimization underway
- ARMv8 next generation processors designed for big.LITTLE