Single Chip Heterogeneous Multiprocessor Design

JoAnn M. Paul

www.ece.cmu.edu/~mesh

July 7, 2004

Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213

Carnegie Mellon

MESH

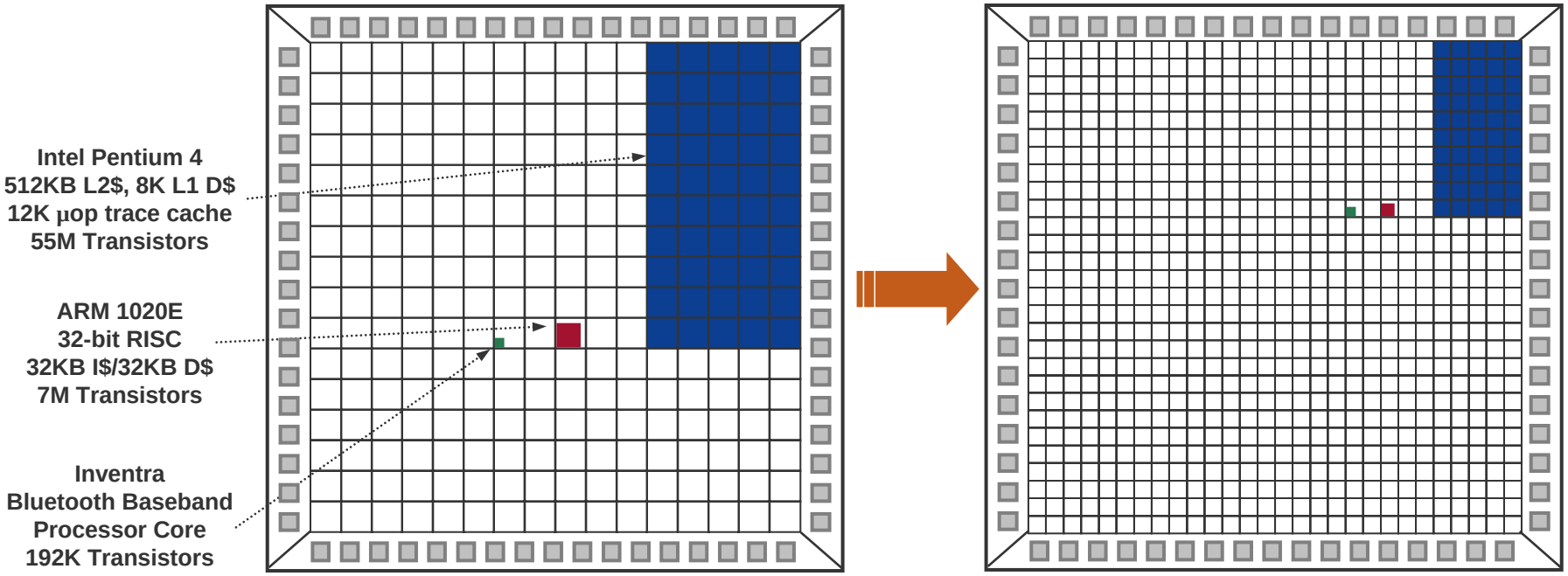
The Cell Phone, Circa 2010

MESH

- Cell Phone? It's probably more like an uber-PDA
  - Speech Recognition, Database, GPS, Software radio, Bluetooth, Voice Over IP, Image Processing, Ad Hoc Networking, Video, 3-D Graphic Shading, Security, Web Browser, MP3, Motion Sensor, 802.11, HCI
- More importantly, what's on the chip that's inside of it?
  - Single Processor?
  - Multiple processors?
    - Heterogeneous?
  - Custom hardware?
  - Bus
    - How many?
  - Central Network?
    - Network on Chip

How Do You Organize…

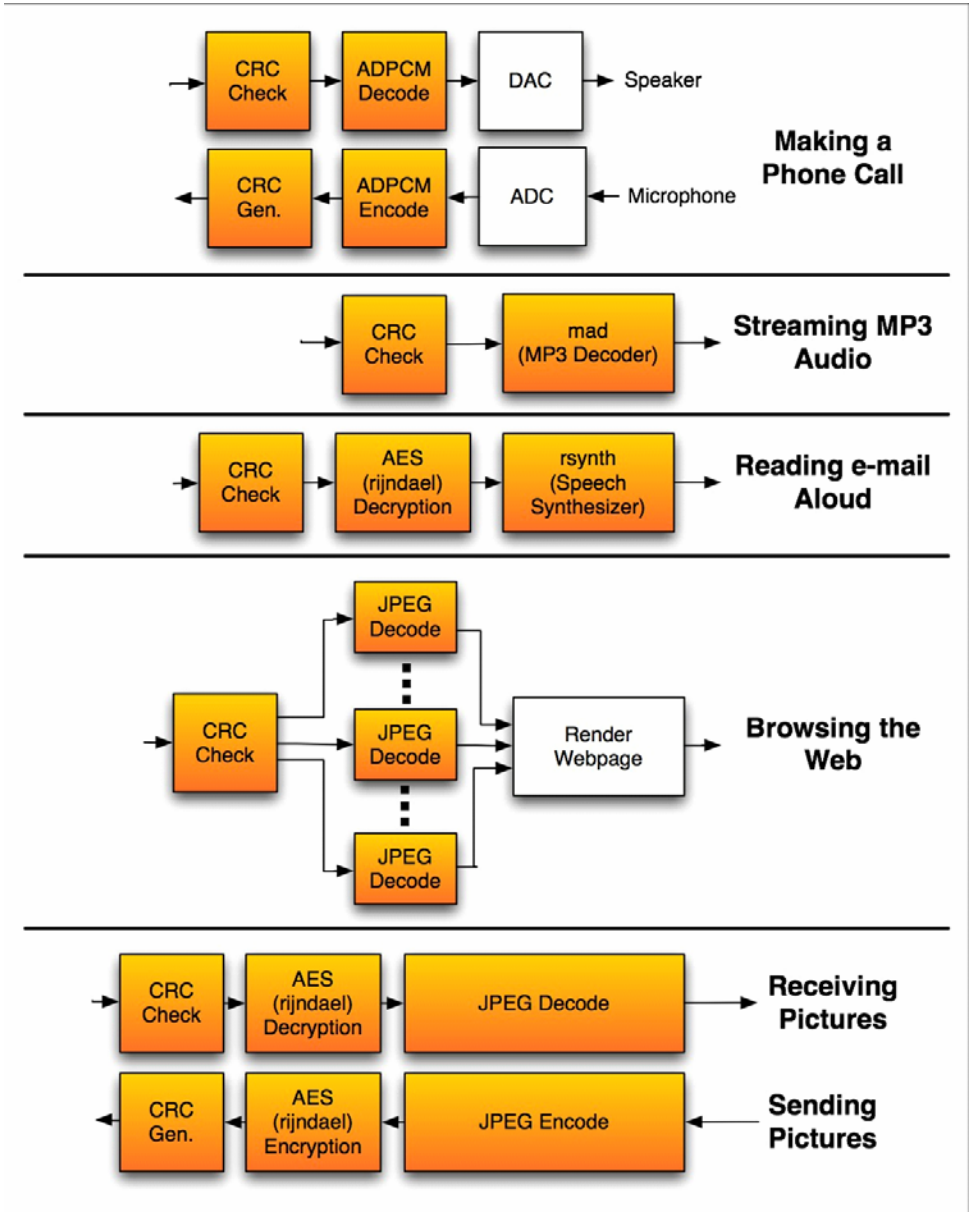
- … all those transistors!
- Is this board-on-a-chip design?
- "A processor of heterogeneous processors"
  - Single Chip Heterogeneous Multiprocessor (SCHM)
  - Overhead for inter-processor coordination is significantly smaller on a chip than on a board (between chips)



2002: ~243M Transistors
280 mm² die

2007: ~773M Transistors
280 mm² die

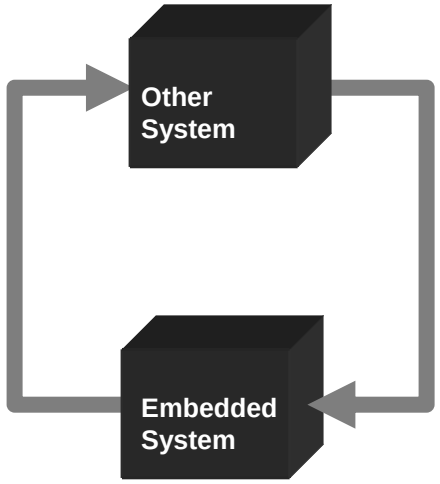
Some Cell Phone Applications

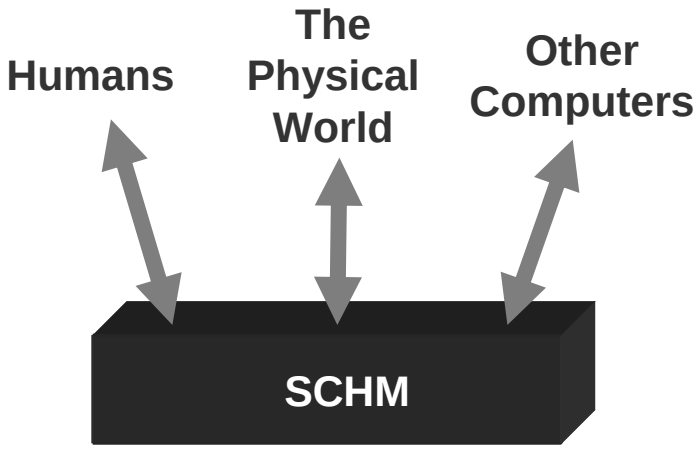


- Three levels
  - Application Elements
    - May be re-used
  - Applications
  - Something beyond… (later)
- A Hybrid of Responses
  - Constant
    - Additional resources won't help performance
  - Data-content dependent
  - Data-size dependent
    - Packets/jobs
  - Bounded
    - "leftover" resources
- Concurrent, heterogeneous, programmatic

A Hybrid of Design

- Old Views
  - Embedded, Reactive systems
    - computer systems interacting with Non-Computer physical Systems, with minimal human intervention
  - General purpose & desktop computers
    - Programmed to interact with humans and other computers
- Towards Ubiquitous Computing
  - A hybrid of application types, a hybrid of design





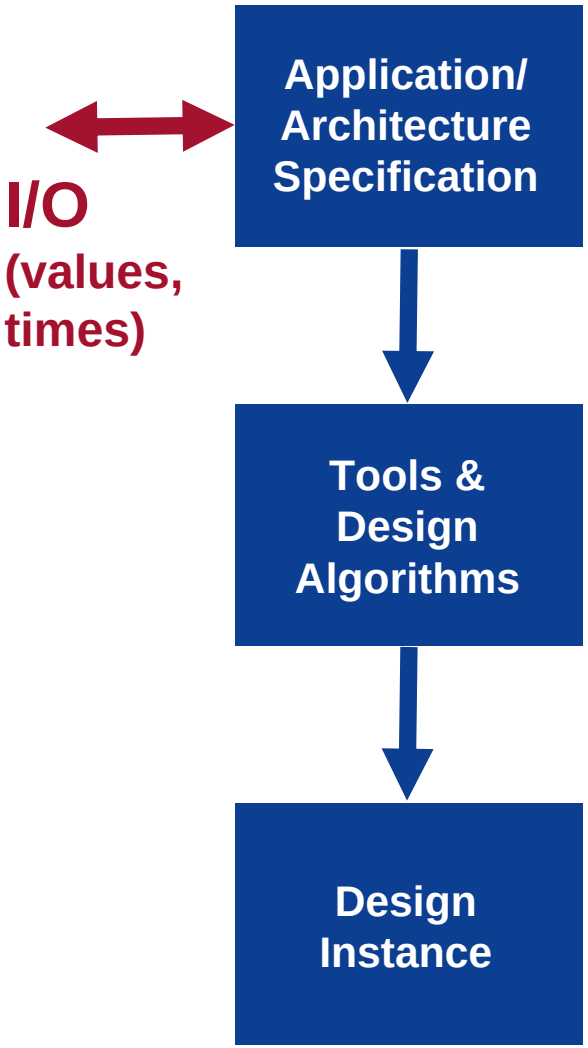
Processor or System Design?

MESH

- Physical vs. Mathematical
  - Non-ideal properties, not “models of computation”
- Facilitation vs. Necessity
  - Branch predictors, Caches
- The Common Case vs. the Worst-case
  - Rare performance may suffer
- Benchmarks vs. Applications
  - Anticipated completion of the system
- Trends
  - Changes in applications and technology over time result in different ways of organizing designs
    - When did branch predictors first make sense?

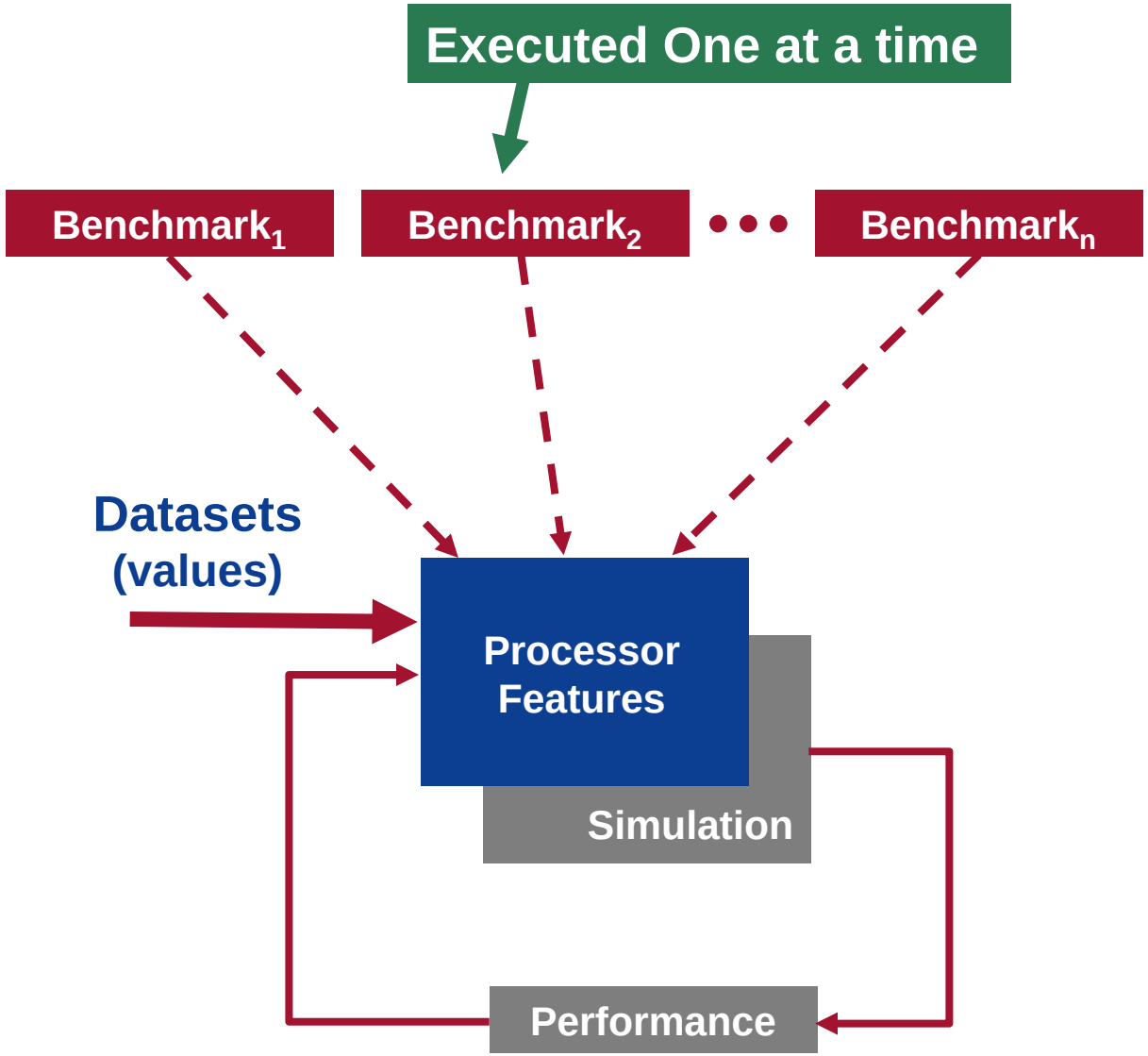
Design 101 – CAD

- Classic Hardware (ASIC), Embedded System, Real-time (RT) Design Automation (DA)
  - In goes the description
  - Out comes the design
- In CAD, we think in terms of
  - The system
    - not just a piece of a system
  - A specification, up-front
    - Top-down
    - Synthesis
    - Design Tools
    - Design Algorithms
  - Design Algorithms capture patterns
    - In the middle is a complex tool that may include a whole bunch of heuristics and/or some mathematical basis



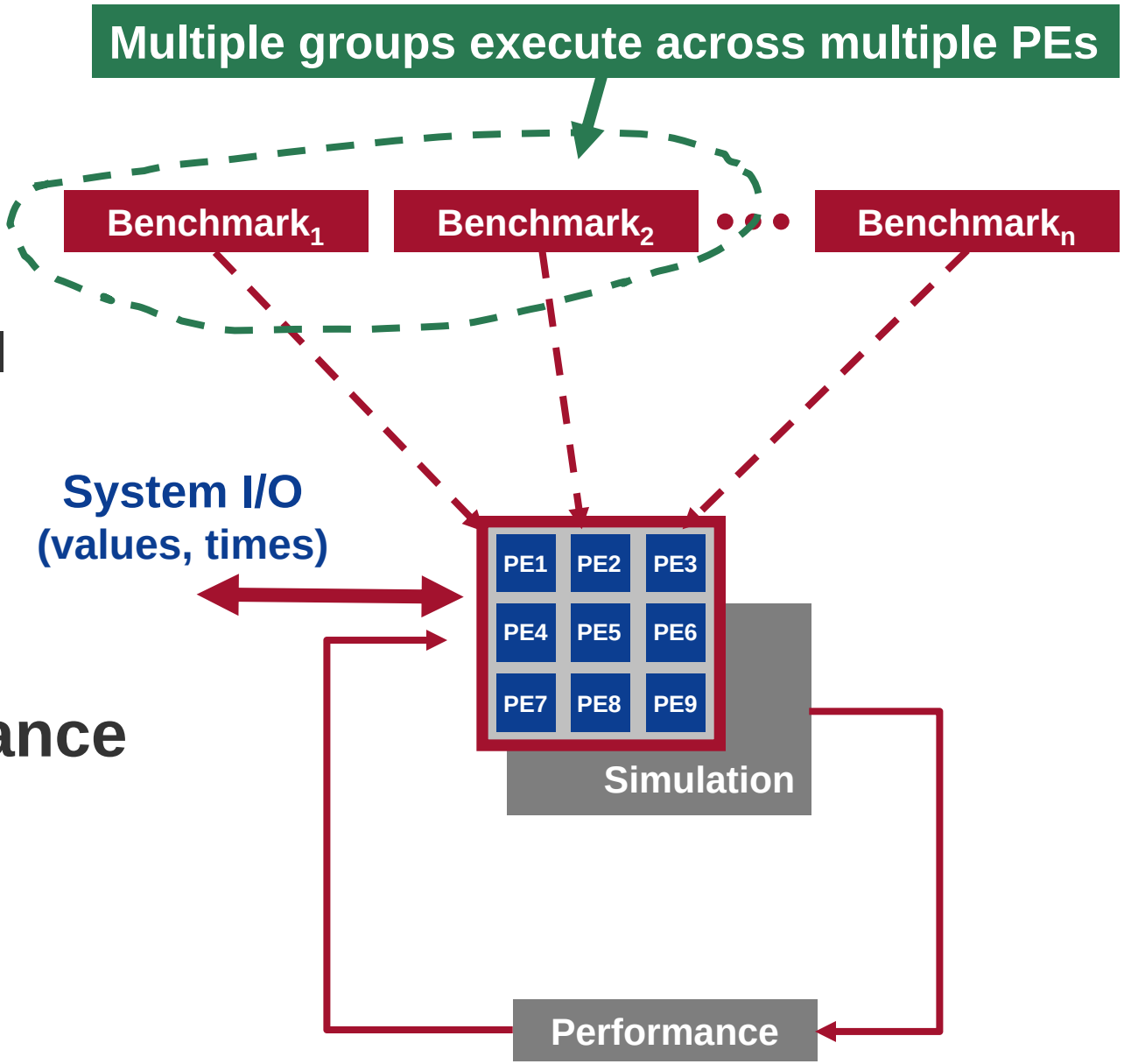
Design 101 – Architecture

- Conventional Processor Design
  - Designer is in the loop
  - Facilitate Creativity
    - Detail reduction, design element modeling
    - not complete automation!
  - RT-, IS-level
- Design for common, anticipated cases
  - Architectural infrastructure – caches, branch predictors



MESH

- Designer Creativity
  - Design Strategies
  - Architectural Features
- Timed Inputs
  - Size, arrival rates, data
- New performance trade-offs
- At Issue
  - ISS too low



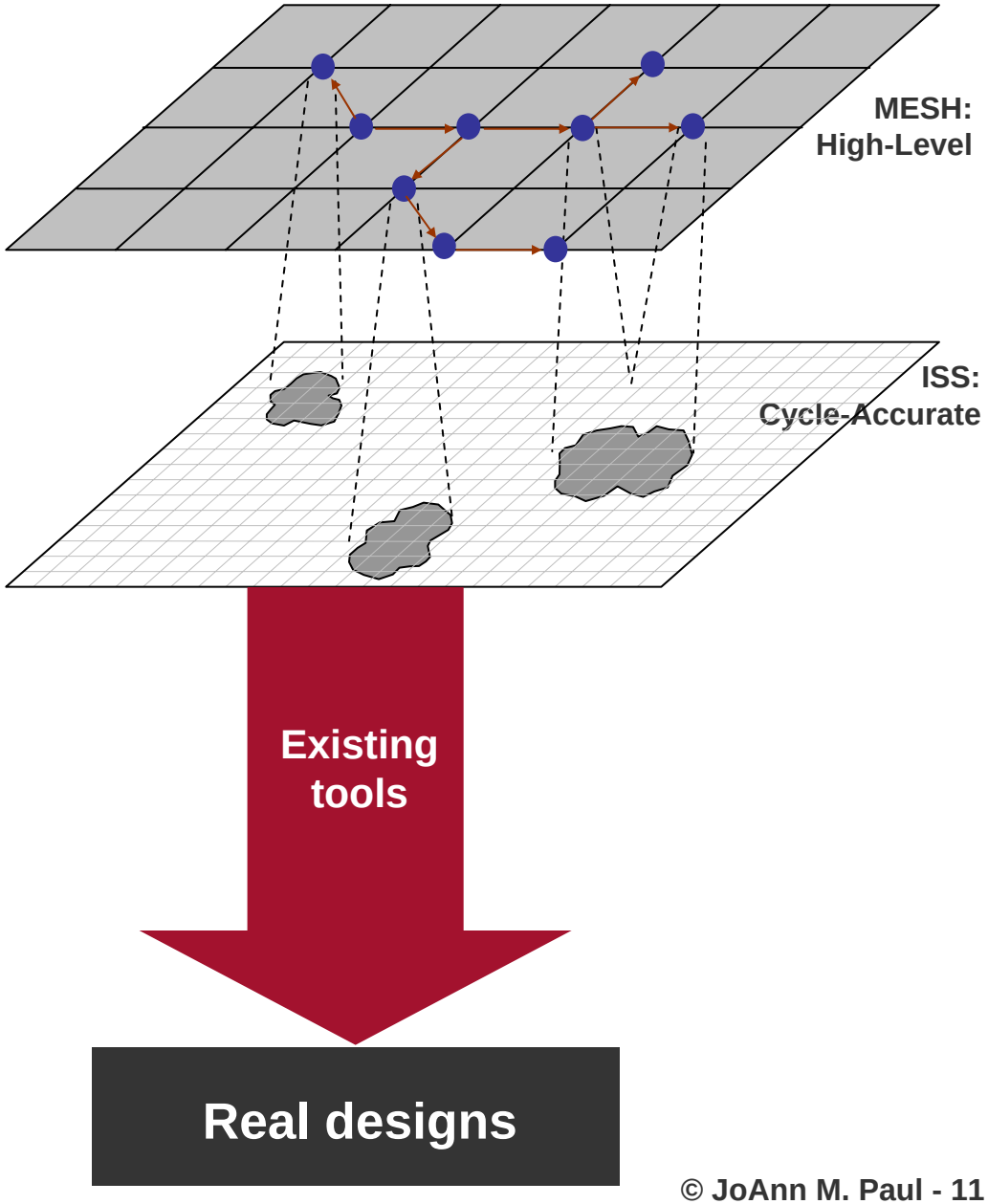
A Word on High-level Modeling

MESH

- How does it relate to real designs?
  - Age-old problem
  - A difficult problem!
  - Verilog to gates… gates to transistors…
- H.L. model + detail → real design
  - Formal models
    - Capture behavior as a mathematical problem
    - Restrict the kinds of systems that can be designed
  - Design tools
    - Capture physical systems as design artifacts
    - Classic modeling and simulation
- At issue
  - so long as the high-level model tracks real designs in a predictable way, then bounded detail can be filled in according to assumptions
  - And a much larger design space can be explored!

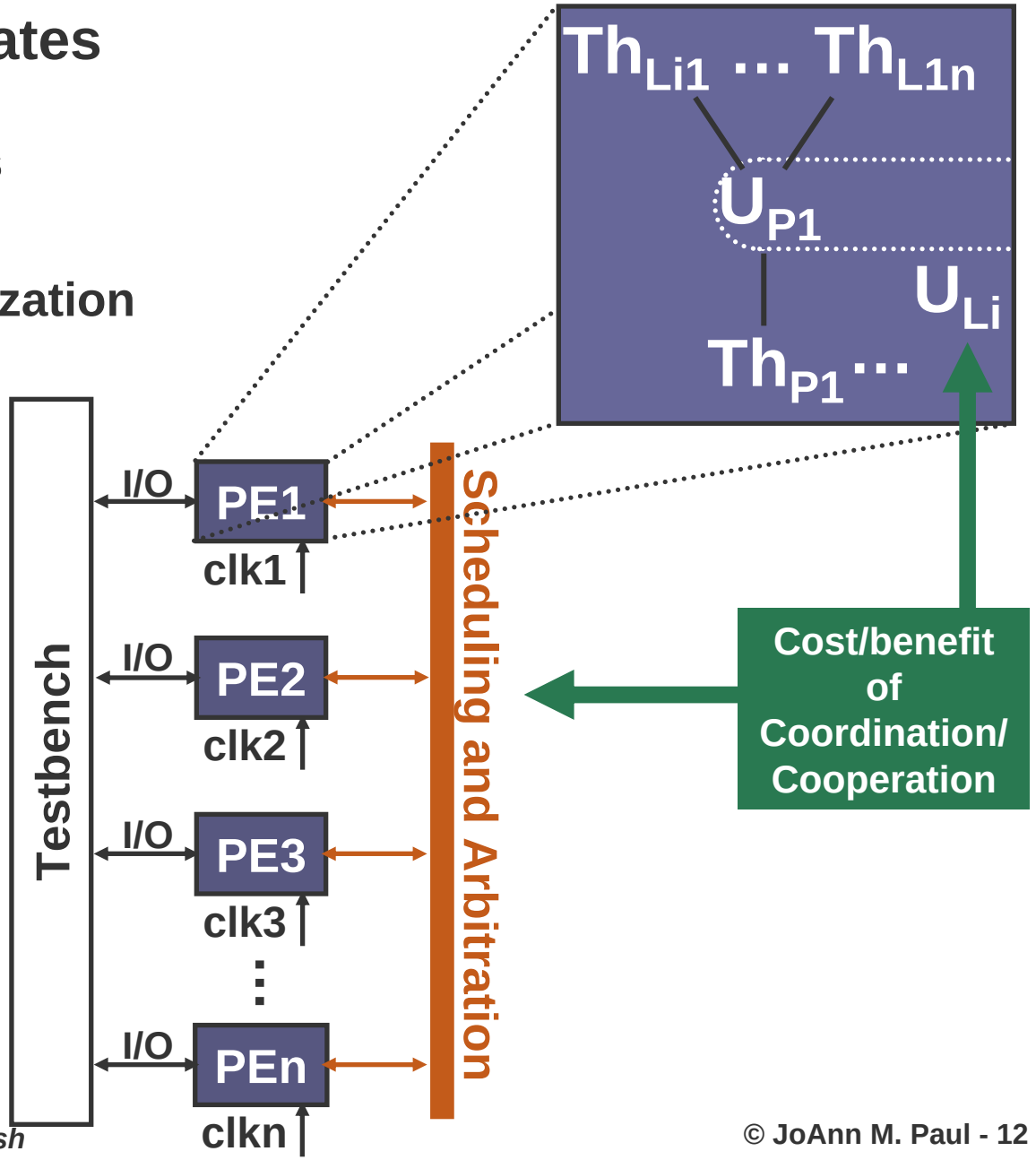
Must Include the Machine!

- MESH models include some non-idealized properties of the machine
  - Contention (shared resource access)
  - Less than ideal resources!
- Thus performance simulation
  - Data-dependent computation time
  - Dynamic decision-making
- New design elements
  - The art of the design of systems with non-ideal properties must be captured in simulations



Physical, Logical, Layered

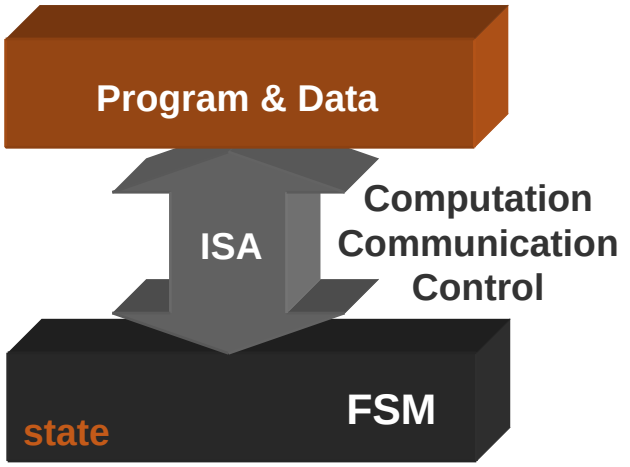
- Scheduling coordinates a set of resources
  - Key design elements
- Design layers
  - Cooperation, Globalization
- Must evaluate the cost/benefit of the layering
  - Trading the cost of cooperation, globalization vs. more powerful, local
  - This is evaluating the HM as a processor

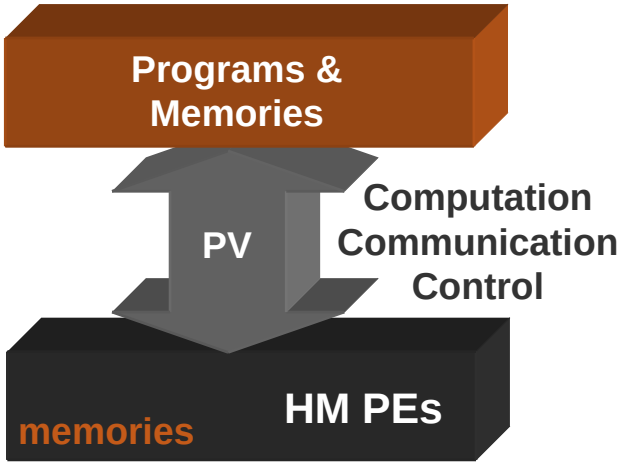


What Makes a Processor?

MESH

- Consider an ISA
  - Three classes of instructions
    - Computation
    - Communication
    - Control
- "Loading the program counter"
  - For a SCHM, this means coordination of a group of processors in response to data
- The Trade-offs
  - The cost of dynamic coordination
    - Global v. local

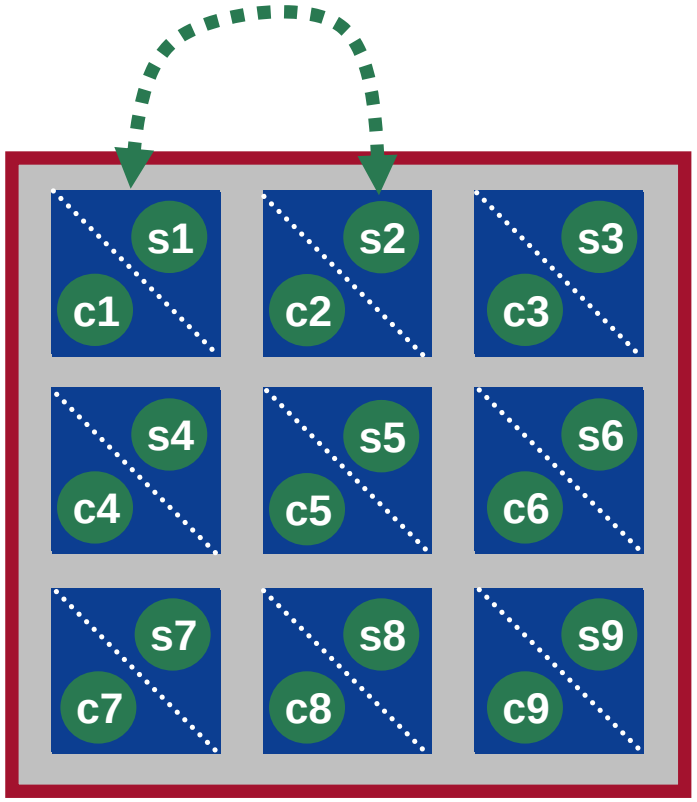




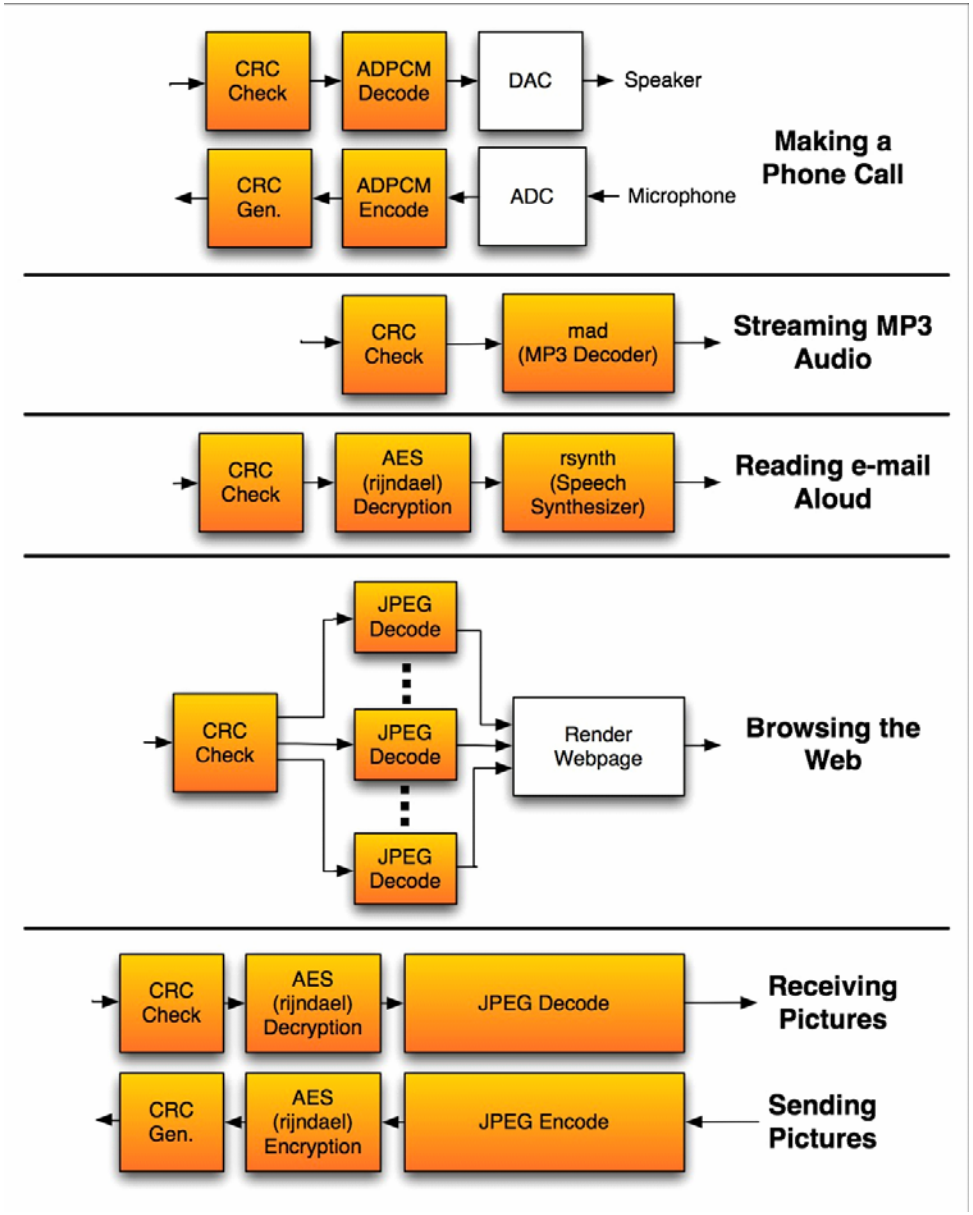
Two Kinds of Partitioning

MESH

- The selection of the PEs
  - Side by side
- The coordination of the PEs
  - Coordinating Program (CP)
  - Chip-wide, a design layer
- Layered Partitioning
  - Local, Computation State (C_i)
  - Global, System State (S_i)
    - Used to make decisions about how system resources are coordinated in chip-level programming
    - May also be distributed
- Modeling, Supporting, and Evaluating
  - the Cost/Benefit of Dynamic, Global Coordination is the key!!



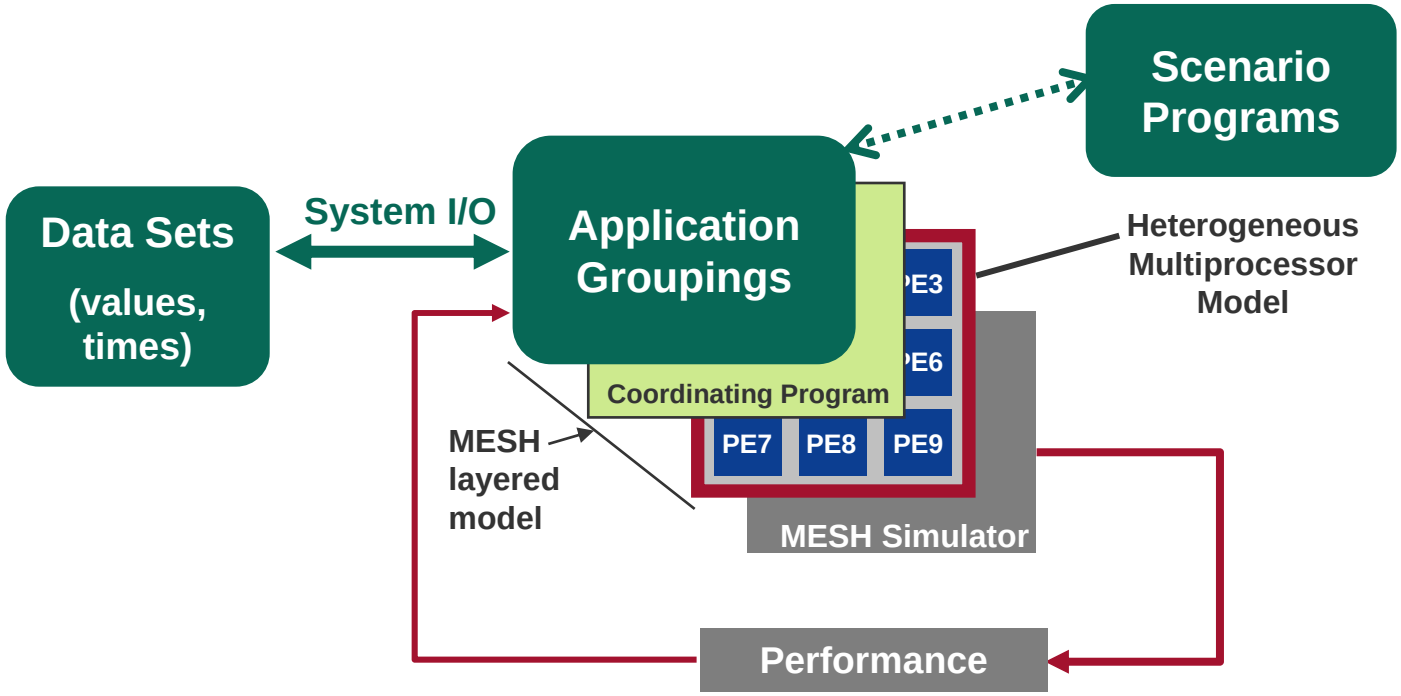
Example: Cell Phone



- A set of application elements
- Forms an application
- Multiple applications for application groups
- That can be exercised under different loading conditions
- These can be used to evaluate different
  - architectures
  - Chip-level scheduling strategies

Scenario-Oriented Design

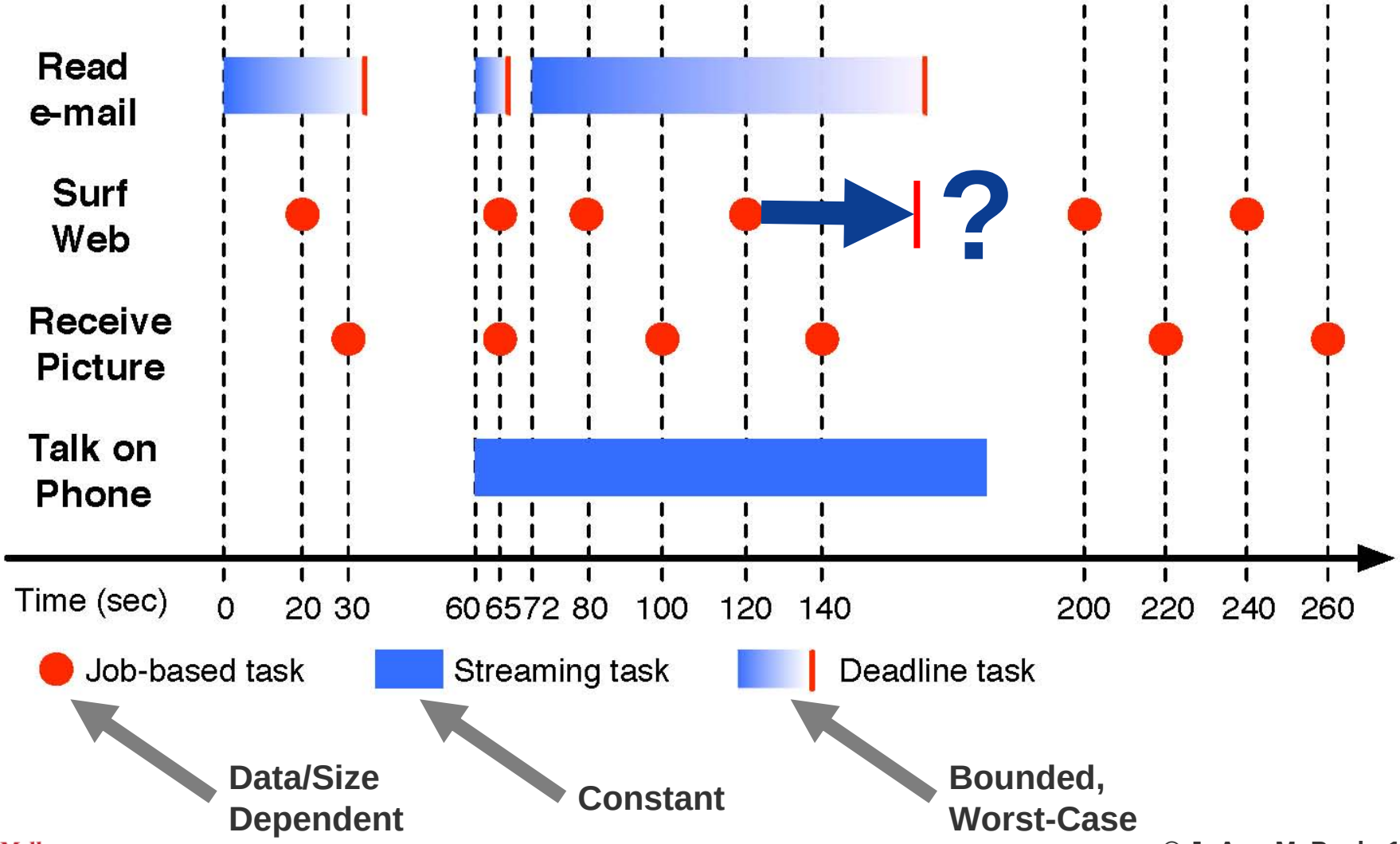
- Contain
  - Application groups, Datasets (time and value), Scenario Programs
- Evaluate
  - Architecture, Chip-level Coordinating Program (CP), Global/Local Partitioning of State



Airport Scenario Timeline

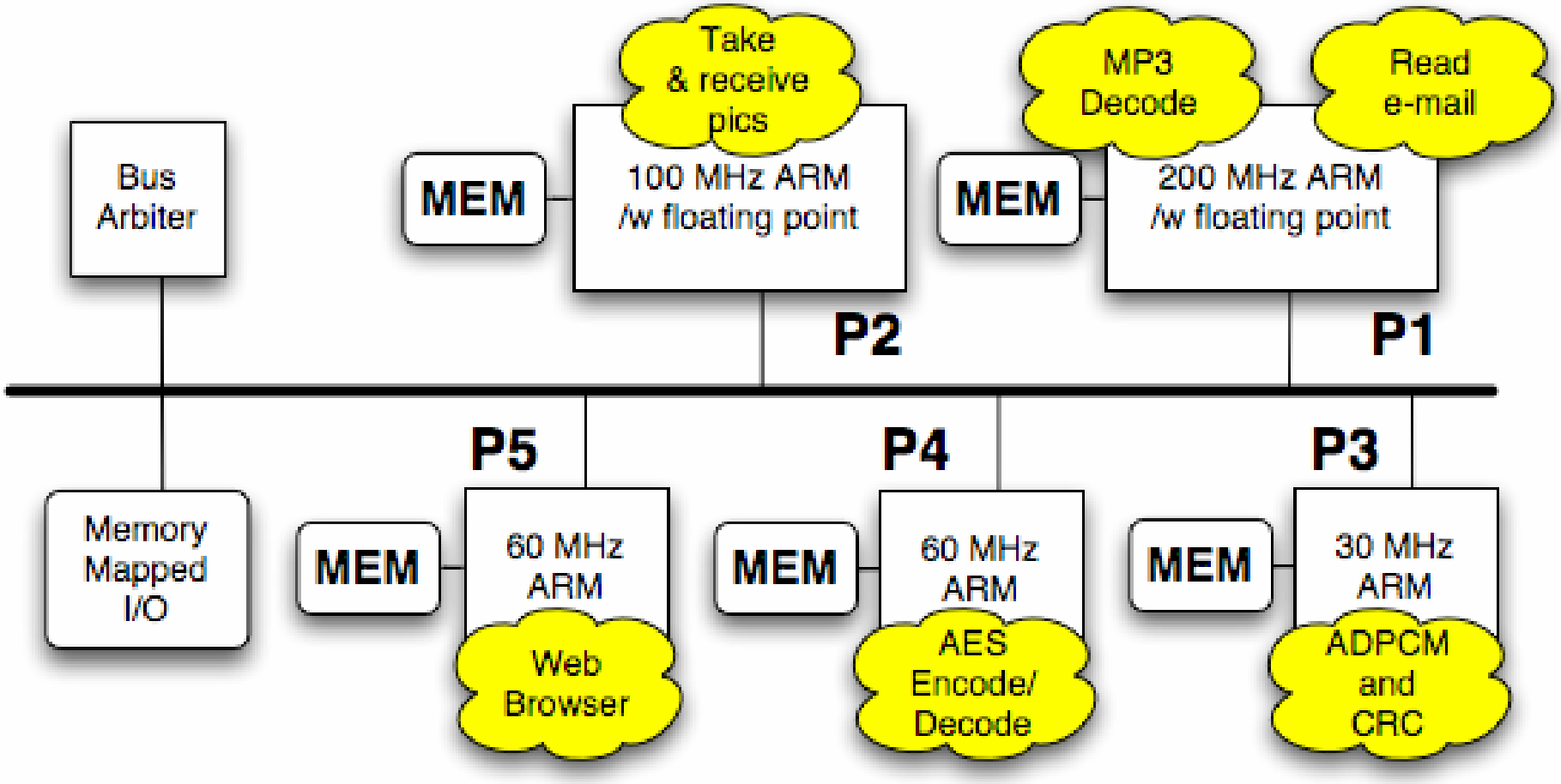
MESH

- Others are possible for the same cell phone
  - Grandma, Teenager…

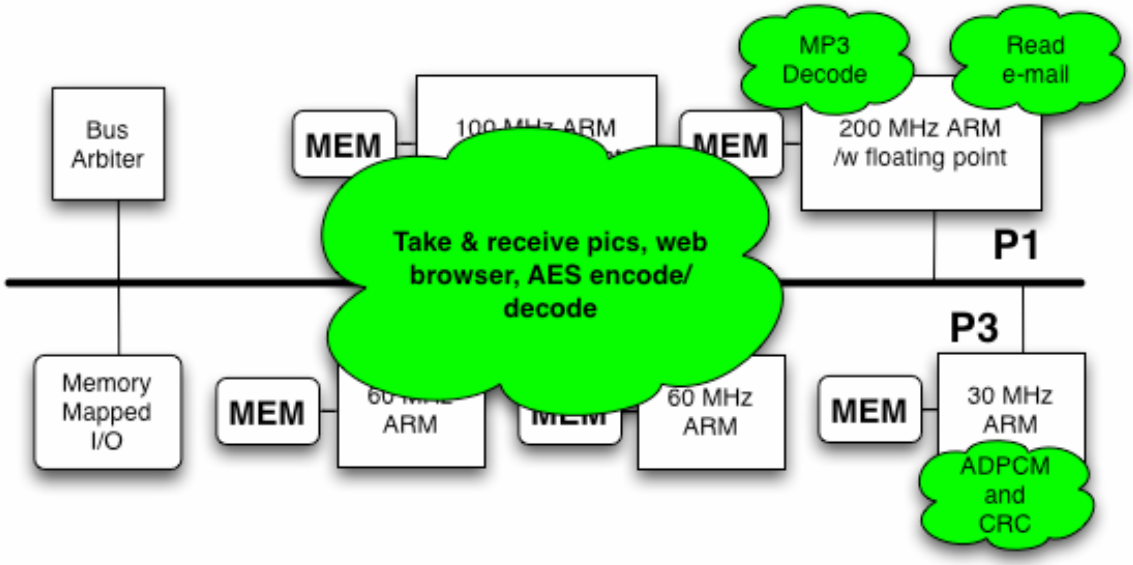


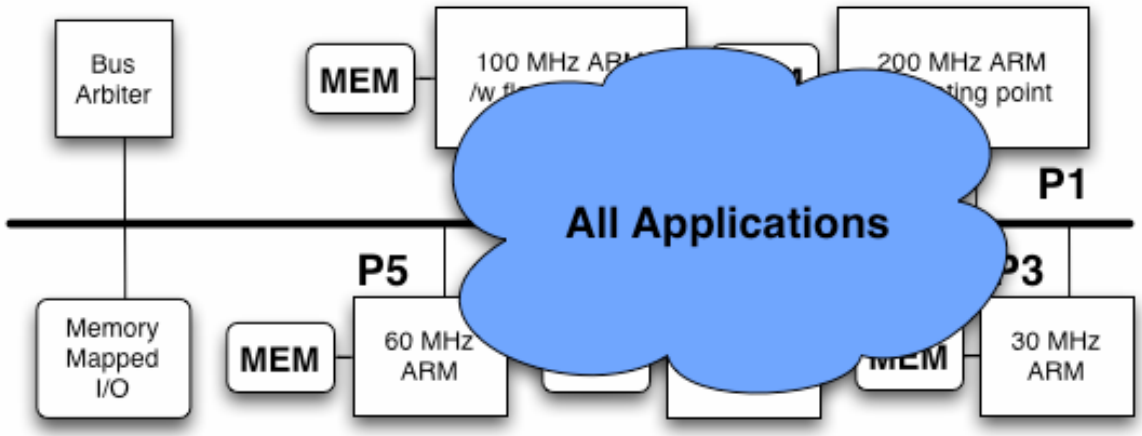
Naïve SCHM

- **Static Scheduling is Traditional**
  - **Everything is worst case or compute bound while resources go idle**



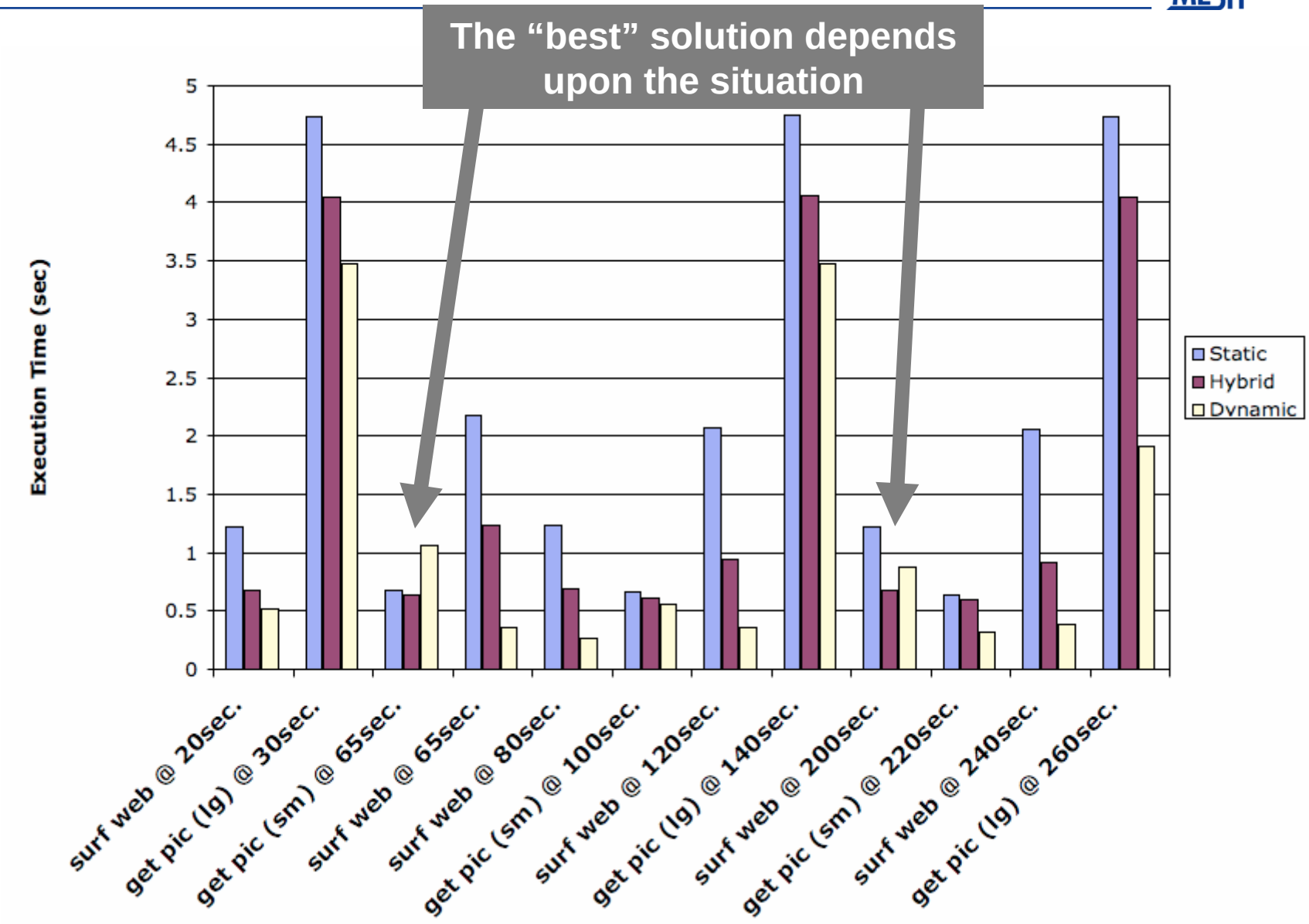
Other Candidates





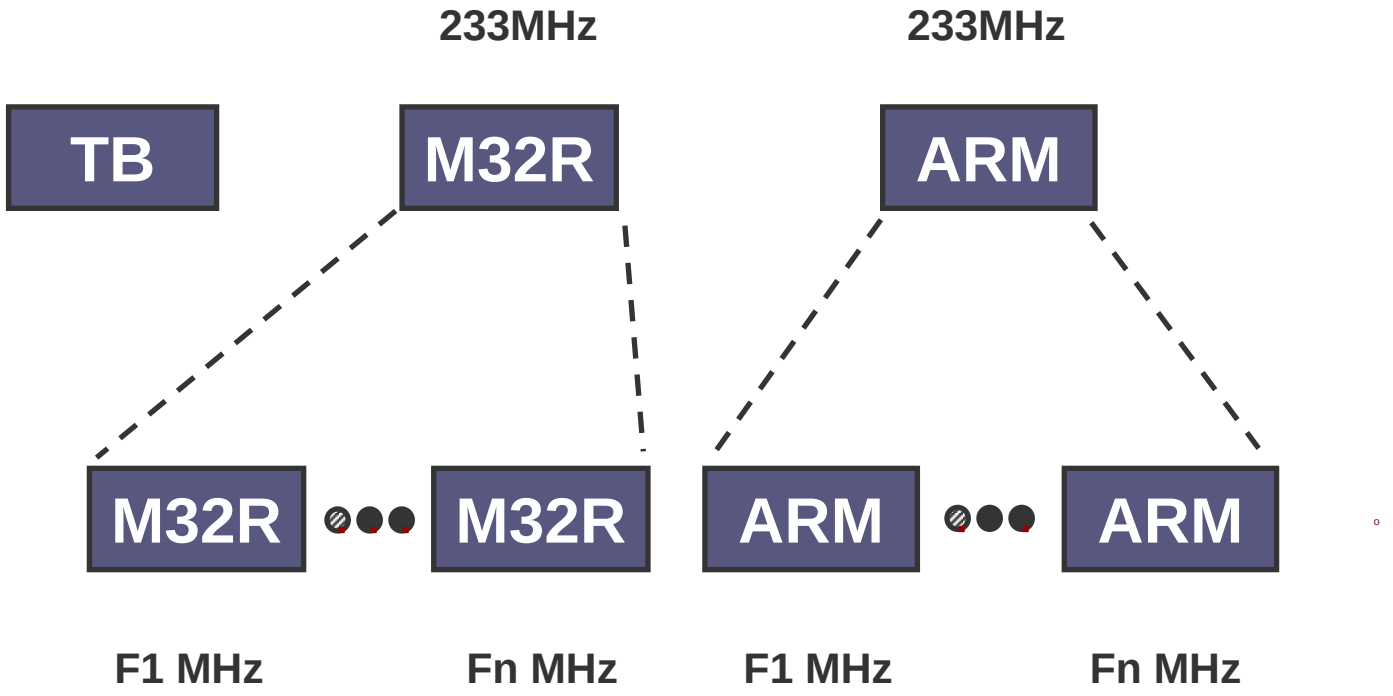
- **Hybrid CP**
  - Some applications are statically scheduled
  - Others are dynamically scheduled across half the resources
- **Dynamic CP**
  - All tasks are dynamically scheduled across all resources

Airport Scenario Results



What about Power?

- The substitution of *n* processors
  - executing at frequencies *f/n* for a single processor executing at frequency *f* can potentially result in significant power savings is *processor-rich* (PR) design
- Spatial Voltage Scaling
  - Instead of a set of (10, 10, 10) MHz for a 30MHz processor, how might (5, 10, 15) work?



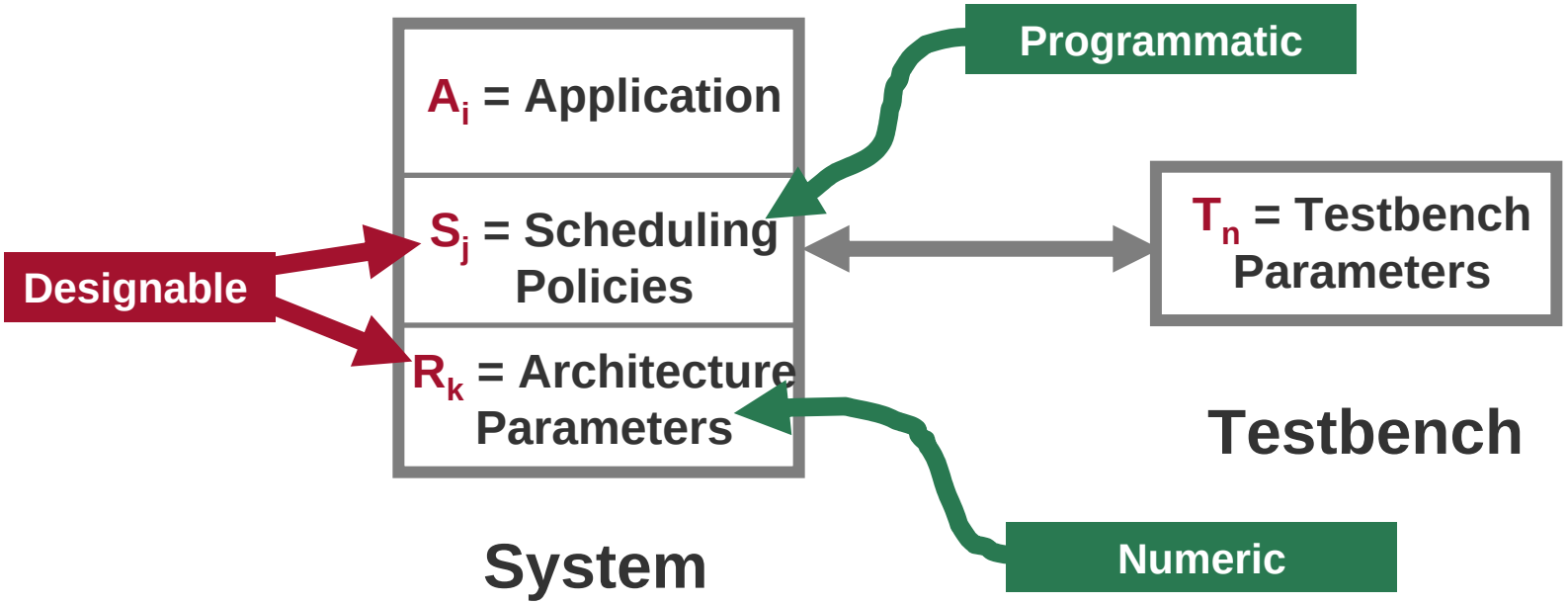
Instance of the PR Design Space

A = document management == {gzip, wavelet transform, zerotree quantization, management} which can execute on *any* processor at different levels of performance

T = { T_1= light load, T_2 = medium load, T_3 = heavy load}

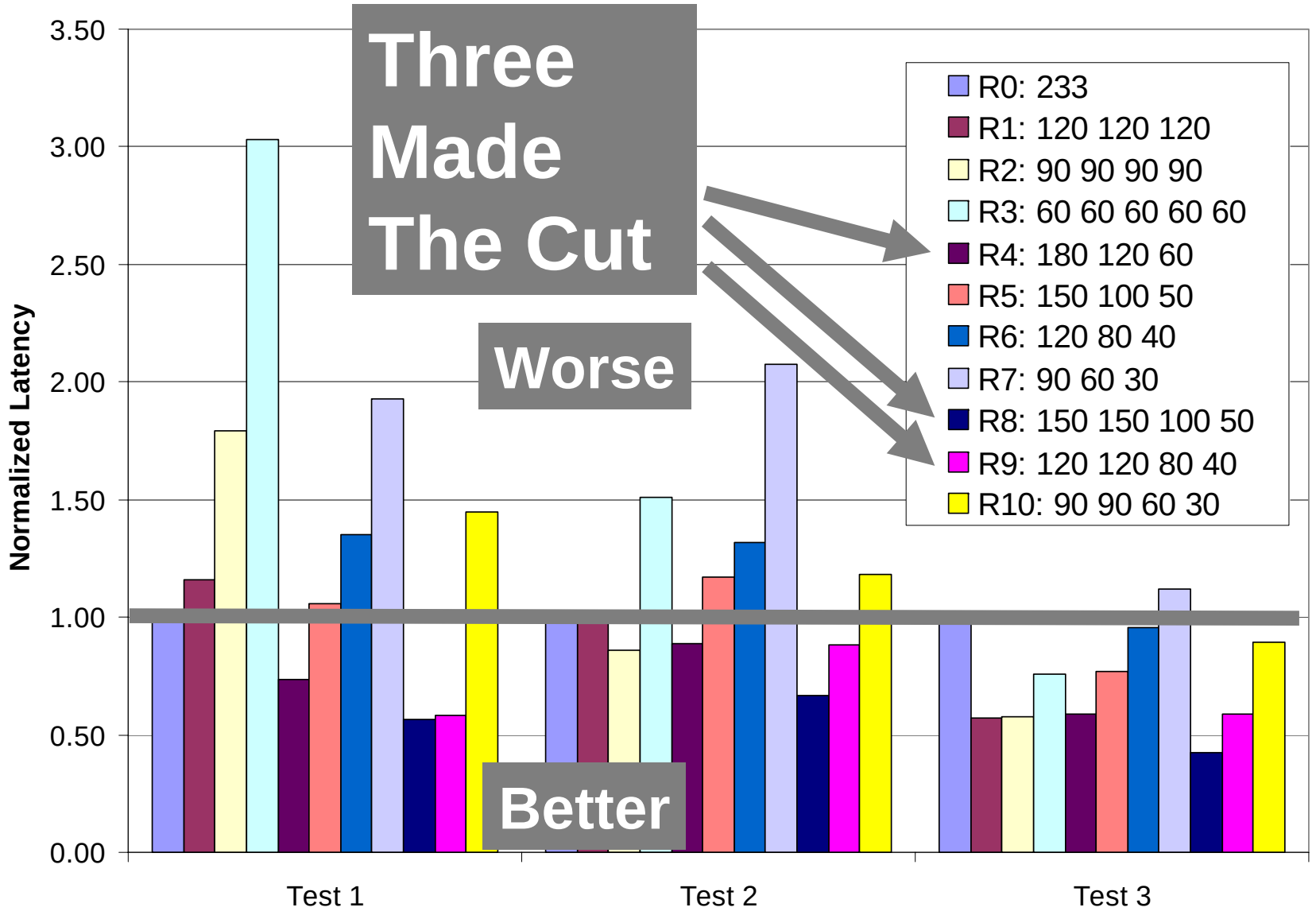
S = {job-size-and-type-aware (S_{JS}), with dynamic shutdown (S_{JSDS})}

R = {numbers, types, frequencies of processors}



Latency Across All Testbenches

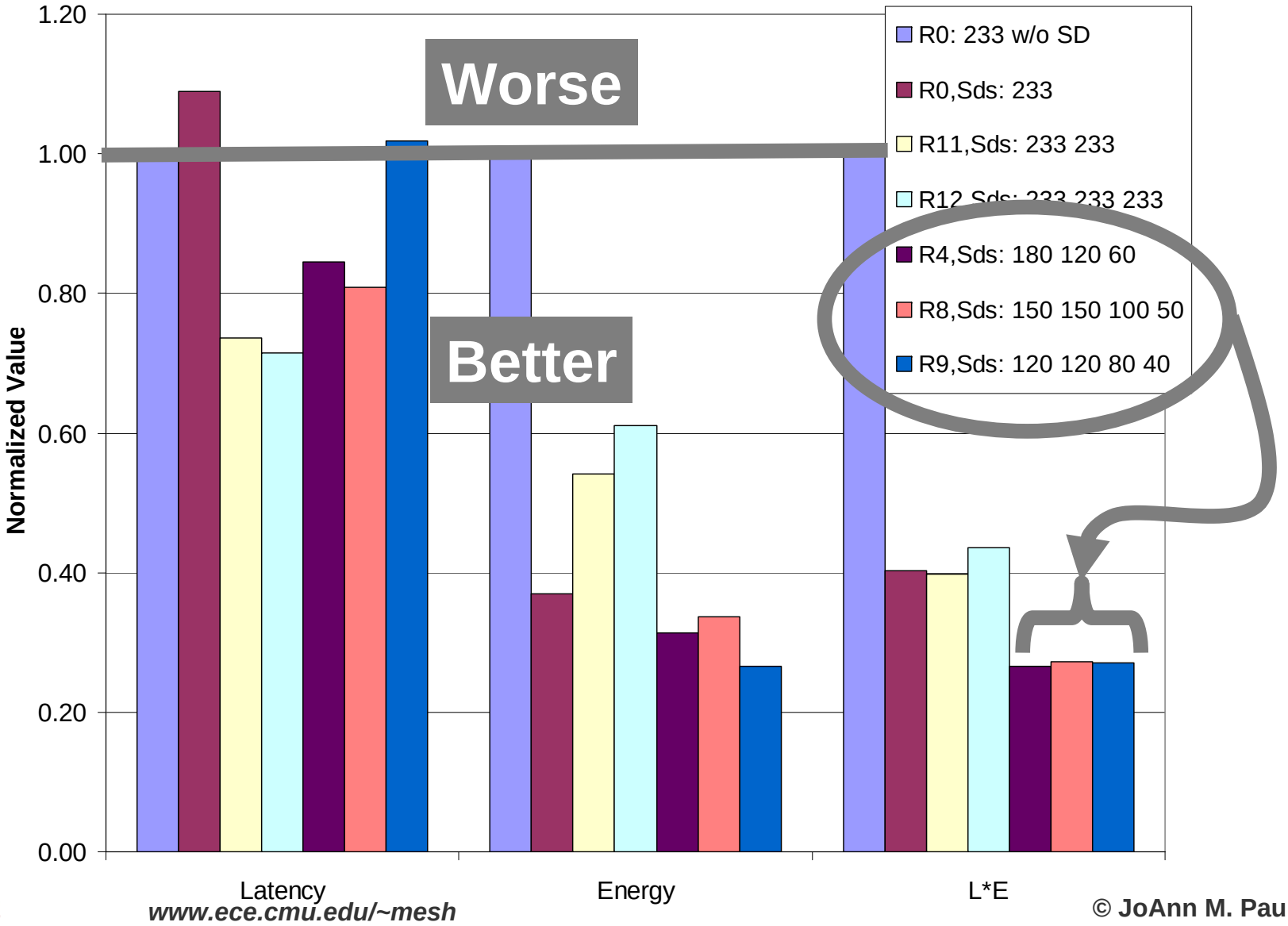
MESH



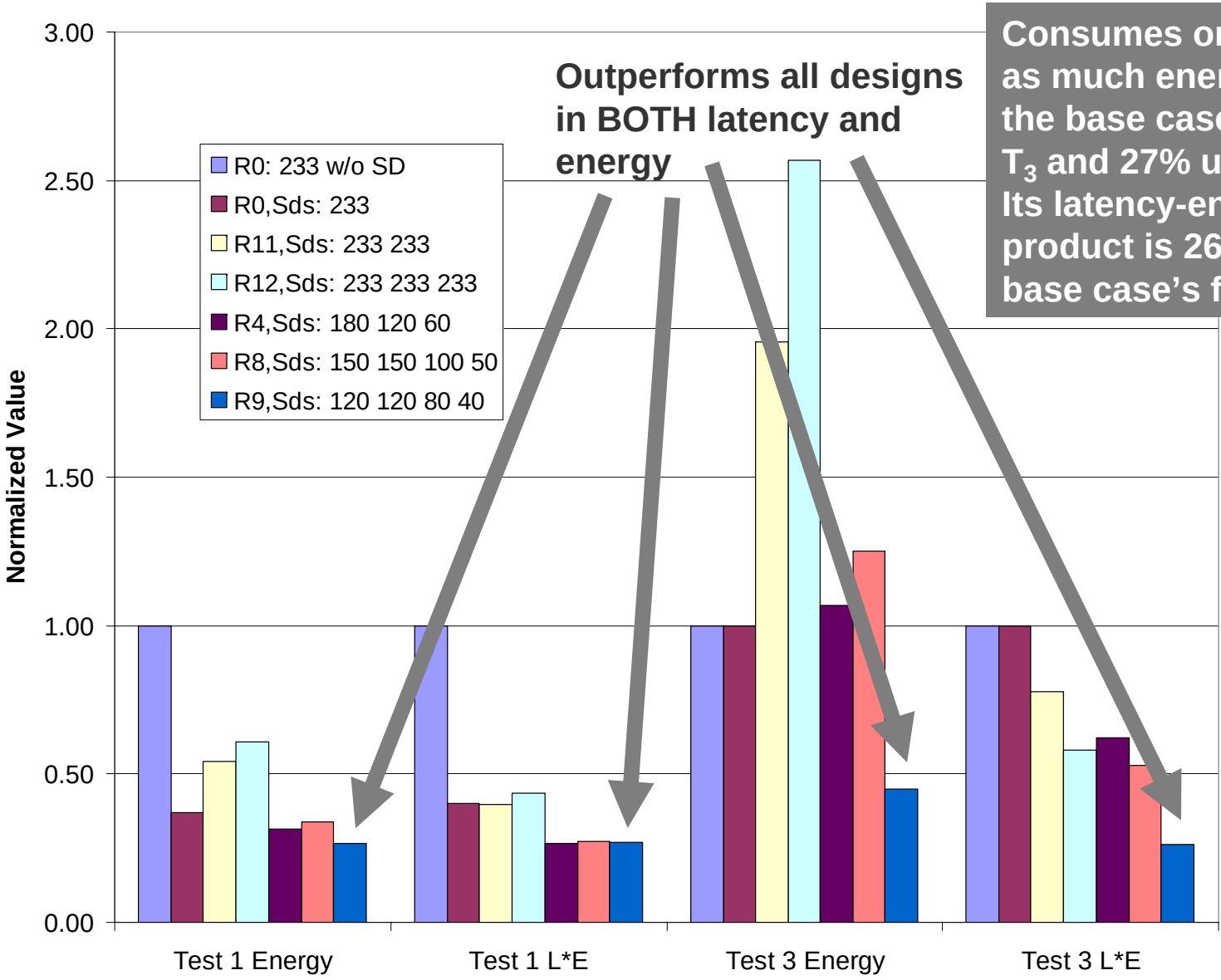
Add Dynamic Shutdown Over T_1

MESH

Some additional, baseline designs are included as well



Across Testbench Categories



Lots of further work to do

MESH

- Power over a “fragment”
  - Entire program, sets of programs or program fragments
  - How many are needed per processor over a set of fragments?
    - In most cases **ONE** is enough!
    - Intuitive – performance dominates performance-power modeling
- Bounding detail
  - Relationships assumptions and real designs
- Design elements
  - User-friendliness of modeling
- Design strategies
  - This is the key piece, because it enables designers to think in different ways, applying creativity in different way
  - Benchmarks, Model-Based Schedulers, Spatial Voltage Scaling, Chip-Level Programming… more to come!