

---

Lecture 19:  
Quick Summary of EE282

Department of Electrical Engineering  
Stanford University

<http://eeclass.stanford.edu/ee282>

# Announcements

---

- PA-2 is due today
  - Electronic submission
- Review session: Friday 12/5<sup>th</sup>, 11am, Skilling 193
- Quiz 2: Monday, 12/8<sup>th</sup>, 8.30am, Skilling Auditorium
  - Maximum duration 3h (but exam may be shorter)
  - All lectures & required reading included
  - Bring 1 page of notes (double sided) & your calculator
  - If you are a local SCPD student, you must come to campus
- Fill in on-line course evaluation form for EE282

# Today's Menu

---

- Wrap-up discussion on energy
- EE282 review

# Review: Sources of Power Consumption

---

$$P = C \cdot V_{dd}^2 \cdot F_{0 \rightarrow 1} + T_{sc} \cdot V_{dd} \cdot I_{peak} \cdot F_{0 \rightarrow 1} + V_{dd} \cdot I_{leakage}$$

- Dynamic or active power consumption
  - Charging and discharging capacitors
  - Depends on switching activity
- Short circuit currents
  - Short circuit path between supply rails during switching
  - Depends on the size of the transistors
- Leakage current or static power consumption
  - Leaking diodes and transistors
  - Gets worse with smaller devices and lower Vdd
    - Quickly becoming a major factor
  - Gets worse with higher temperatures

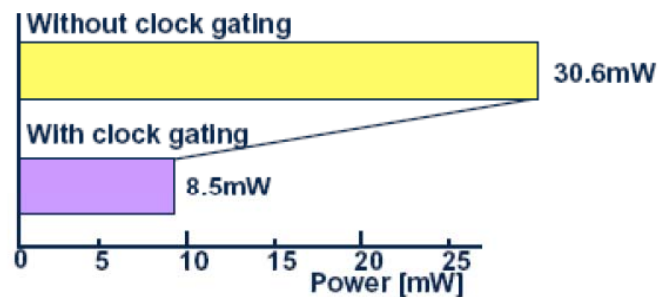
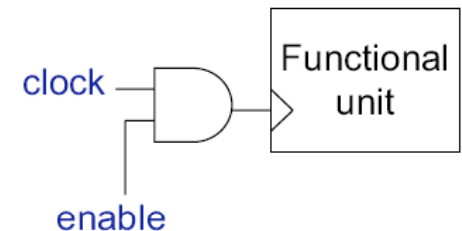
# Reducing Power at All Levels

---

- Process technology
  - Examples: reduce capacitances, provide low/high  $V_t$  transistors
- Circuits design (gates and blocks)
  - Example: reduce switching probability, use low/high  $V_t$  transistors, use multiple supply voltages
- Architecture
  - Examples: simpler architectures, use of parallelism
- Software
  - Examples: manage voltage/frequency scaling, better algorithms ...
- Some common principles
  - Reduce redundant work/components
  - Turn off unused components
  - Pick the implementation that best matches constraints

# Clock Gating

- Reduces dynamic power on clock network
  - One of the biggest power contributors
  - Can be done at various granularities
    - Whole core to a set of registers
    - Tradeoff: overhead vs benefit
- Challenge: enable signal generation
  - Control logic complexity
  - Timing issues to avoid glitches
- Example: power gating efficiency for MPEG-4 decoder [ISSCC'02]

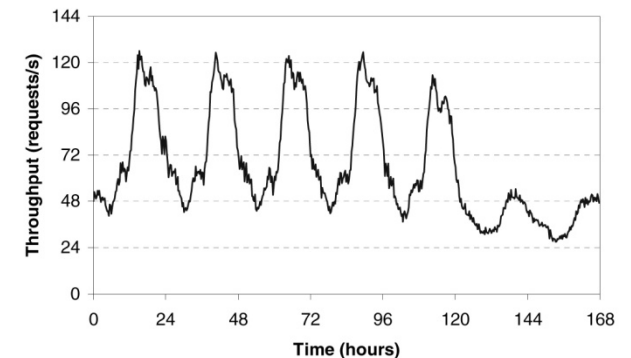


# Dynamic Voltage and Frequency Scaling

- Idea: consume as much as you need
  - Use lowest frequency that achieves perf. target
  - Use lowest Vdd that allows that clock frequency
- Regulated by software based on system load
  - Heavy load: frequency, voltage, power high
  - Light load: frequency, voltage, power low
  - Trade-off: power savings vs overhead of scaling

Web request for www.ibm.com for Feb. 5-11, 2001 (from J. Chase, SOSP 2001)

- Example: Transmeta's LongRun technology
  - 32 levels of frequency and Vdd
    - 200MHz – 700MHz, 1.1V-1.6V
  - 20 usec to change to next level



- Simulation results: 9-38% CPU energy savings for web workloads

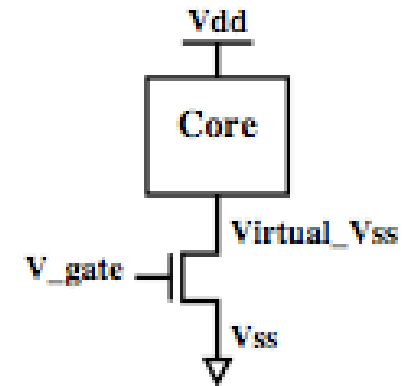
# DVFS and Concurrency

---

- Assume an initial processor with  $P_{ref} = C_{ref} * V_{ref}^2 * F_{ref}$
- Two-core system
  - Assume that applications get ideal speedup (2x)
  - We reduce the frequency by a factor of 2 ( $F_{tc} = F_{ref}/2$ )
  - This allows us to drop voltage by 1.7x ( $V_{tc} = V_{ref}/1.7$ )
  - Capacitance is higher ( $C_{tc} = 2xC_{ref}$ )
- Overall results:  $P_{tc} = 0.34 * P_{ref}$ , same application performance
  - Can get similar results with pipelining or other forms of parallel execution

# Power Gating

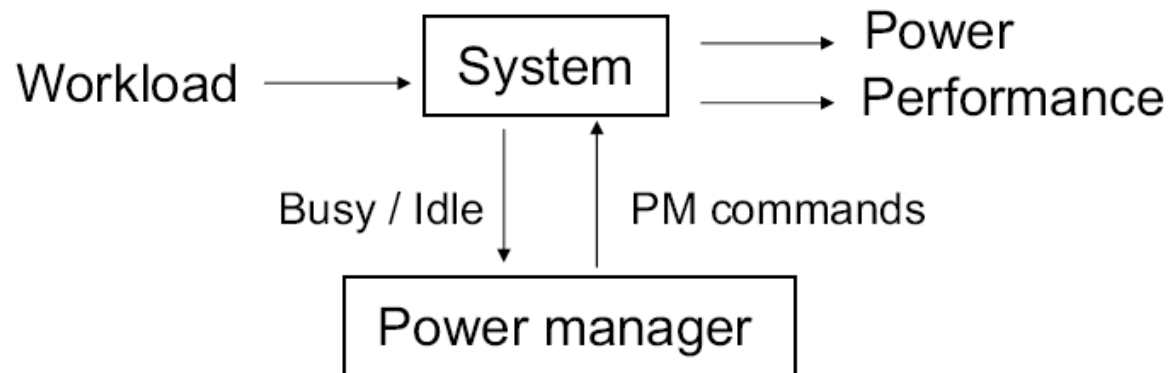
- Idea: turn off power completely to unused components (e.g., cores)
  - Minimizes leakage power on unused component
- Challenges
  - Area power consumption of power gate
    - High
  - Latency of transition (10s msec)
    - Saving SW state, flushing dirty cache lines, turning off clock tree
    - Carefully done to avoid voltage spikes or memory bottlenecks
- Opportunities
  - Use thermal headroom to overclock other cores...
  - Improves single-thread performance



# Power Management

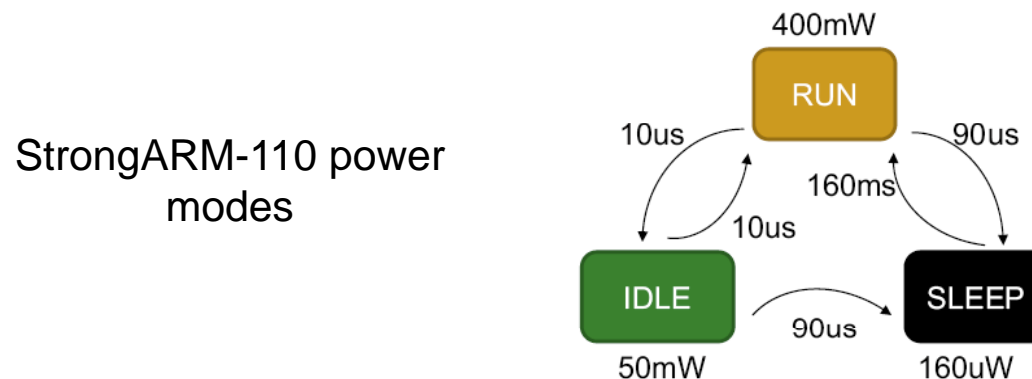
---

- Selectively set the power state of HW components to match software requirements
  - Exploit workload fluctuations
  - Done by the OS, the compiler, and/or the user
  - Trade-off: power saving versus speed of resuming



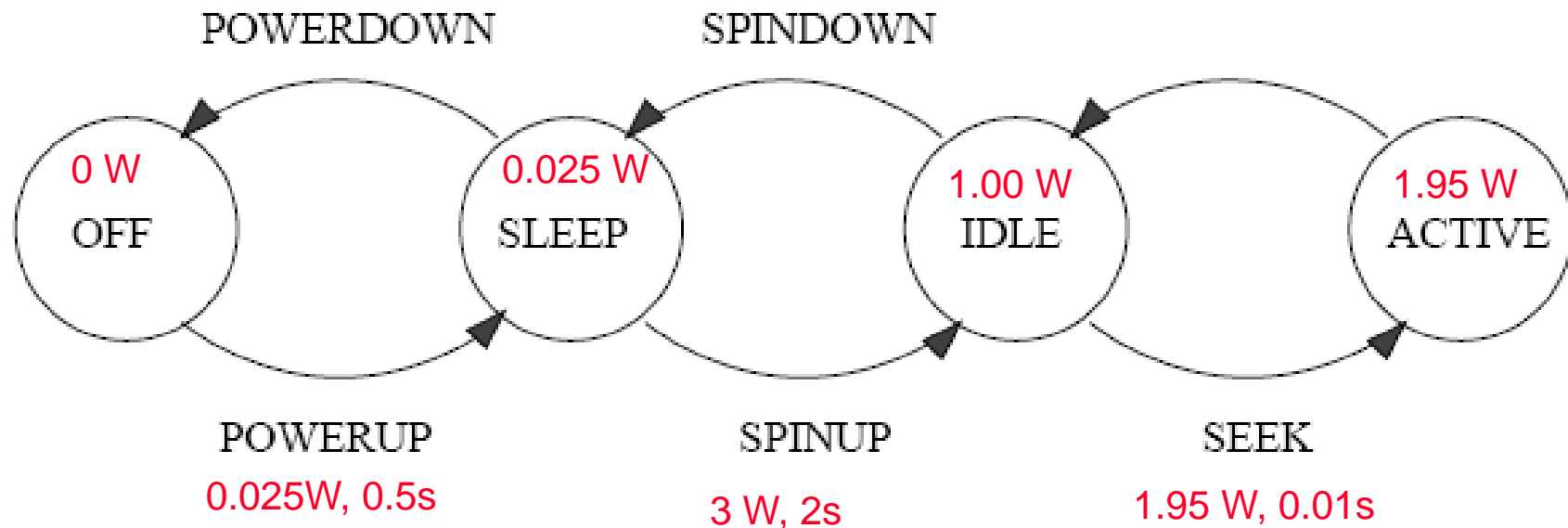
# Example: Processor Power States

- Example processor modes
  - Idle mode (short periods of inactivity – few hundred mW)
    - Clock distribution turned off; needs a few cycles to recover
  - Nap mode (medium periods of inactivity – 100mW)
    - Caches turned off
  - Sleep mode (long periods of inactivity – 20 mW)
    - Turn-off power to processor (including PLL)



# Example: Disk Drive Power Modes

- Common optimization for mobile computers: stop spinning the disk when it is unused for a certain period of time
- Sample: Toshiba notebook drive



State diagram from Li, 1993.

# Advanced Configuration and Power Interface (ACPI)

---

- A standard for power management of systems
  - Describes power stages for system, devices, cores, ...
    - Vendors may implement subset of states/options
  - Defines interface for SW to query and manage power states
- Global system states:
  - G0: working - system is responsive, user applications run
  - G1: sleeping - appears to be off, OS may still be silently running
    - Multiple S-states (sleep states) defined within G1 with different wake latencies/power consumption/retention of context
  - G2: soft off - neither user nor OS is running
  - G3: hard (mechanical) off

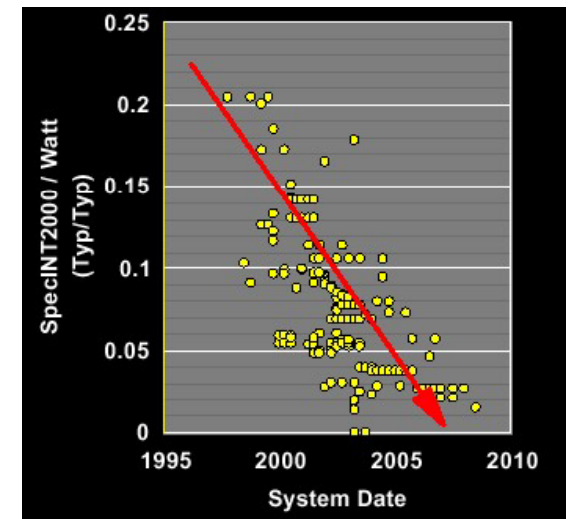
# Advanced Configuration and Power Interface (ACPI)

---

- Device states
  - D0 fully-on operating state
  - D1 and D2 are intermediate states (vary by design)
  - D3 is a powered off state (device is unresponsive)
- Processor states (P and C)
  - P states are DVFS stages
  - C0 is fully on
  - C1 to C3 are various idle modes
    - Clock may be stopped but state is maintained
  - C4 and beyond are various power off states
    - First the caches, then cores, and finally the whole chip

# Reducing Power with Simpler Hardware

- Complexity = higher power
  - Diminishing returns from power budget for higher performance
  - Pipeline depth
    - More stages increase register count and clock load
  - ILP
    - Parallel execution units, dependence check logic, VLIW instruction fetch increase power
  - Speculation
    - Speculation hardware burns power (predictors, etc.)
    - Guessing wrong wastes power
  - Caches
    - Large/complicated caches burn power to access
- Example
  - Pentium 4: 65W
    - OOO, 3-way issue, 22 pipeline stages, 12K \$I, 8K \$D, 512K \$L2, 4K entry predictor, 211 mm<sup>2</sup>. 0.13um, 2 GHz
  - ARM 920T: 0.09W
    - in-order, 1-way, 5 pipeline stages, 16KB \$I, 16KB \$D, 4.7mm<sup>2</sup>, 0.13um, 250 MHz



# Low Power & Memory Systems

---

- Caches reduce power
  - Avoid expensive off-chip (or remote) accesses
  - Anything that improves cache performance typically improves power consumption
  - But introduce an issue of leakage management
- Low power caches
  - Sequential tag – data accesses
  - Multi-bank caches
  - Configurable caches
  - Decay or drowsy caches (turn off portions of the cache)
  - Data compression

# EE282 Overview

---

# Top Ideas in Computer Architecture

---

- Pipelining
  - Overlap steps in execution; watch out for dependencies
  - Examples: processors, memories, communication, ...
- Caches
  - Keep close a copy of frequently used information
  - Examples: data caches, TLB, web/FS caches, ...
- Indirection
  - Go through a translation step to allow intervention & checks
  - Examples: VM, security, ...
- Prediction
  - Better to ask for forgiveness than permission...
- Parallelism
  - Replicate resources to execute independent tasks in parallel
- Redundancy
  - Additional information or resources to recover from errors

# Lessons from EE282 (1)

---

- Processor microarchitecture
  - Superscalar, out-of-order, multiple levels of caching, high frequency,...
  - Needs high memory bandwidth (always) and low memory latency (often)
  - Moving to multiple CPUs mostly because of power issues
- Performance analysis techniques
  - Averages hide interesting behavior and problems
  - Use distributions, calculate deviation/covariance/confidence/...
- Advanced caching
  - Reduce hit time: simple caches, wide interfaces, pseudoassociativity, multi-ported and multi-banked caches
  - Reduce miss rate: larger caches, higher associativity, skew associativity, victim cache, prefetching
    - Keep in mind the 3 Cs
  - Reduce miss penalty: multi-level caches, sub-blocks, critical word first, write buffers, non-blocking caches
    - Watch out for bandwidth bottlenecks

# Lessons from EE282 (2)

---

- Software optimizations for enhanced locality
  - Loop transformations, blocking, prefetching, eliminate redundant ops...
  - Your mileage may vary (see PA-1)
- Main memory systems
  - SRAM for caches, DRAM for main memory
    - DRAM optimized for low cost & bandwidth
  - Improving memory BW: wider buses, interleaving, multi-banking
  - Improving memory latency: caching, removing redundant steps, integration
- IO systems
  - Current: hierarchies of buses, controller, and devices
  - Future: point-to-point or switched interconnects
  - DMA, direct user access, acceleration, off-loading, on-loading, I/O integration

# Lessons from EE282 (3)

---

- OS & system software support
  - Operating modes, processes, mutual exclusion
  - VM: TLB, hierarchical and inverse page tables, ...
    - Watch out for interactions with memory hierarchy and IO
- Virtual machines
  - Full system virtualization for security, load-balancing, migration, ...
  - Architecture is efficiently virtualizable if all sensitive instructions trap when executed in user mode
  - Can also virtualize and ISA for which the sensitive instructions in user mode are a subset of kernel mode instructions
- Parallel architectures
  - Performance, scalability, reliability, ...
  - Shared memory and message passing models
  - Limitations: Amdahl's law and communications latency
- Clusters
  - Message passing machines from commodity hardware
  - Low cost, good scalability, but higher overheads
    - Reduce with better integration & smarter communication assist/software

# Lessons from EE282 (4)

---

- MapReduce
  - A simple (functional) model for parallel programming
  - System implement scheduling, load balancing, data distribution
- MPI: message-passing interface
  - Performance challenges: communication overhead & imbalance
  - Non-blocking send/receives, scheduling, work-queues
- Reliable systems
  - Faults and failures are unavoidable
  - Approaches: improve MTTF or MTTR
  - Fault-tolerant systems are based on redundancy
    - RAID, error codes in memory, redundancy in computation
- Low-power architectures
  - Turn off unused components
  - Scale power supply & clock frequency to performance needs
  - Simpler architectures

# Where to You Go from Here

---

- EE382A: Advanced processor architecture (Spring 09)
- CS315A: Parallel computer architecture & programming (Winter 09)
- EE382C: Interconnect networks (?)
- EE482: Special topics in computer architecture (?)
  - Watch out for announcements
- Also
  - OS and Compilers: CS243, CS140, CS 240, ..
  - Networks: CS244/EE284, EE384X/Y, ...
  - Design: EE271, EE273, EE371, ...
  - CAD: EE318, EE319
  - Fault-tolerant systems: EE489

# EE382A: Advanced Processor Architecture

---

- Topics
  - Advanced processor architecture (how to build Core 2)
    - Superscalar, deep pipelines, caching, fetching, prediction, OOO, renaming, dependencies, precise exceptions, ordering, bypassing, speculation, ...
  - Multithreading, multi-core, dynamic optimizations
  - Research-related topics
- Research-oriented project
  - EE382a projects frequently lead to paper publications...
- Who should take EE382A
  - Students interested in architecture or related areas, with solid background
  - Systems students that want to do research
- This will be a fast paced class

# Thanks for Taking EE282

---

- Quiz 2: Monday, 12/8<sup>th</sup>, 8.30am, Skilling Auditorium
  - Maximum duration 3h (but exam may be shorter)
  - All lectures & required reading included
  - Bring 1 page of notes (double sided) & your calculator
  - If you are a local SCPD student, you must come to campus
  - Don't stress, it's just a quiz and you will do well
- Enjoy the Christmas break