Saving Power through Explicit Mechanisms
Dave Maze, Edwin Olson
6.893, Fall 2000

The Power/Performance Dilemma
• Mainstream architectures focusing on high performance
• Some applications require low power
• Usage patterns of some devices call for short periods of high performance and long periods of low-power performance.

Our Solution
• Build a machine which can yield adequate performance, but can switch to a low power mode.
• Turn functional units and capabilities on or off depending on performance/power needs.
• Have the compiler make as many decisions as possible (minimize hardware profiling required)
• Explicit instructions for coarse granularity, extra bits in instructions for fine granularity.

Areas that can be optimized
• Turn off sections of cache to reflect size of “working set”
• Put functional units into a “coma” mode
• Turn on/off out-of-order/speculative issue logic (resulting in an in-order microprocessor)

Things to worry about…
• Existing work exists. Try not to duplicate. Try to do something novel.
• Code density – perhaps we’ll ignore code density for this phase of research.
• Compiler hacking – perhaps we can sidestep the issue by doing some assembly hacking to get approximate results.

Execution Plan
• Begin by testing ideas/profiling benchmarks to determine effectiveness of particular ideas.
• Select an idea, hack together an approach, and measure results. (Checkpoint 1, Oct. 19)
• Refine, extend, automate optimization (Checkpoint 2, Nov. 9)
• Tools: SyCHOSis, MIPS ISA simulator