Different DVFS Techniques for Multitasking Systems

Background

Dynamic Voltage and Frequency Scaling is a technique currently being researched in embedded system design to reduce power, and thus, energy consumption. In general, DVFS alters the voltage and frequency of the CPU according to processing needs. Because energy is quadratically dependent on voltage, DVFS changes the voltage and frequency dynamically to only use the minimum necessary levels to still yield high performance.

Goals

In the recent mini-project, we used a naïve DVFS technique to create an energy-efficient media player through calculating the CPU utilization over given intervals and adjusting the voltage-frequency level accordingly. For the final project, our goal is to extend this energy minimization over the entire system, rather than just this specific application. We will implement and compare two previously researched DVFS techniques in the Linux kernel to accommodate multi-tasking environments. Specifically, in [1], DVFS is implemented using machine learning techniques based on runtime statistics (CPI) for each individual task.

Implementation

We will first research different DVFS techniques and familiarize ourselves with the Linux kernel. We will then select two algorithms to implement. The implementation will be done by either creating our own module to be integrated with the Linux kernel or modifying an existing module with our enhancements. After the completion of our implementations, we will run several benchmarks and record the performance of our two implementations.

Resources Needed

We will be using the Intel PXA27x platform and developing on the Linux kernel version 2.6.9. We will also need some benchmarks to test the performance of our implementation.

Expected Results

While our mini-project was energy efficient for a specific application, we expect that our final project implementations will be more accurate and energy efficient over a wide range of tasks. We expect that the two implementations will work equally well for certain tasks, differ in their performance for other tasks, or that one will just work better overall. In the case for different performance for different tasks, it is possible perhaps in a future project to implement a controller that selects the implementation that is best suited for specific tasks.
Project Outline

2/12 Proposal due
2/12 - 2/19 Research DVFS policies, read related papers, and become familiar with Linux structure
2/19 - 2/26 Devise a plan to implement two of these policies
2/26 Mid-report due
2/26 – 3/04 Implement DVFS policies
3/04 – 3/11 Run tests, compare, and fine-tune policies
3/13 Final Report due

References