ANALYSIS OF A DYNAMIC VOLTAGE AND FREQUENCY SCALING ALGORTIHM ON XSCALE.

CSC 714 Project.
Dr. Frank Mueller

Prasanth Ganesan, Shobhit Kanaujia, Ishdeep Sawhney
{pganesa, sokanauj, issawhne}@unity.ncsu.edu

Frequency Scaling:

Using the methods described in our previous report, we have been able to acheive frequency scaling on the IPAQ. We are using the mmap () - approach to set the CCCR register, and linking a function written in assembly to set the CCCLKCFG register. The effects of frequency scaling were observed by simply timing the execution time of a particular loop at various frequency settings.

We also explored various settings to observe the possible frequencies at which the IPAQ would work. From section 4.2 of [1], we plan to use the recommended operational voltage for various frequencies to calculate power-savings.

Note: It is still not clear as to what hardware suppport is available to actually change the voltages. The calculations will suffice for the simulation purposes, although we might not be actually changing the voltages.

The following is an excerpt of our test loop:

```
"
        .....................
        SetFrequency(i);
        dwStartTime = GetTickCount();
        for(k=0;k<1700000;k++);
        dwFinishTime = GetTickCount();
        diff = (int)(dwFinishTime-dwStartTime);
        ..............
        ..............
"
```

SetFrequency() is our function which sets the registers to do the frequency scaling. The above test-loop was executed at various frequencies and the observed results are as below:

Frequency	Execution Time (msec)
99.53	210
117.96	175
149.295	138
176.94	117
199.06	103
235.92	87
298.59	69
353.88	58
398.12	52
471.84	43
597.18	35

** *There is slight jitter in the execution times; we have observed it to be within +/- 5% max. We are currently exploring how to minimise other OS Activities in order to reduce this jitter.*

Static DVS Scheduling:

Implemented the static DVS scheduling algorithm and performed frequency scaling. Our tasks for the simulation were function calls that ran for values slightly less than the WCET of the task. The task set used for the simulation was taken from [2]. The maximum frequency at which we operate the Ipaq is 597.18 MHz. On computing alpha, the lowest frequency at which the task set becomes schedulable is 471.84 MHz.

To simulate actual execution time of the task less than the WCET, we generate a random value in the range of 0.3 < rand 0.8 and multiply it with the WCET.

We currently have a framework to implement the other frequency scaling algorithms mentioned in the above paper. We shall be running the dummy tasks as threads instead of function calls in future simulations. Our scheduler shall be separate thread in that case against the static case where our scheduler was part of the main thread.

By running our thread at the highest priority and not making any system calls, we are able to reduce jitter in our observations as of now.

Computing time issues:
We tried using the GetThreadTimes() API to observe compute time of threads, but unfortunately the structure that stores the execution time of the thread is destroyed when the thread exits. Hence we expect our scheduler to maintain our own time schedule with the help of the GetTickCount() call that gives the milliseconds elapsed

since the system was started.

Tasks at hand & Timetable:

- Implement the Look Ahead/Cycle Conserving DVS scheduling algorithm **(Nov. 18 – Nov. 25)**
- Come up with mechanisms to bench mark and perform measurements and analysis. We will also consider making any enhancements if feasible. **(Nov. 26 – Dec. 2)**

Distribution of Work:

Understanding of Algorithms and Design issues:
Investigators: Ishdeep, Prasanth, Shobhit. **(Status: In Progress)**

Implementing Frequency Scaling APIs:
Investigators: Prasanth, Shobhit. **(Status: Completed)**

Implementation of task-simulations:
Investigators: Ishdeep, Prasanth. **(Status: In Progress)**

References:

[1]. Electrical, Mechanical, and Thermal Specification Datasheet for PXA250. ftp://download.intel.com/design/pca/applicationsprocessors/manuals/278524-001.pdf

[2].
Padmanabhan Pillai, *Kang G. Shin*, "Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems (2001)", 18th ACM Symposium on Operating Systems Principles.