ECE471: Embedded Systems – Homework 5

Power and Energy

Due: Tuesday, 10 December 2013, 9:30AM EST

Background:

- Use your Gumstix board for this homework.
- Download and install the HW5 files

  Download the template code from the ECE471 website:
  `http://web.eece.maine.edu/~vweaver/classes/ece471_2013f/ece471_hw5_code.tar.gz`

  Uncompress the file:

  ```
  tar -xzvf ece471_hw5_code.tar.gz}
  ```

  Install the included cpufrequtils packages (as root):

  ```
  dpkg -i ./cpufrequtils_008-1_armhf.deb libcpufreq0_008-1_armhf.deb
  ```

  Check that it installed properly by running:

  ```
  cpufreq-info
  ```

1. **Part 1: Measuring Energy of Naïve Matrix-Matrix Multiply**

   Measure the time needed to compute a naïve 300x300 matrix-matrix multiply (the code repeats 10 times internally). Use the provided `dgemm_naive` program run like this:

   ```
   time ./dgemm_naive 300
   ```

   Record the "user" time and use it with the provided under-load power in the table below to calculate total energy. (Be sure to include units!). You need to measure the time for each frequency available.

   To change the frequency, use the `cpufreq-set` command as root. For example:

   ```
   cpufreq-set -f 250MHz
   ```

   For background, this type of matrix-matrix multiply calculates $C = A \times B$ with the following simple algorithm:

   ```
   void naive_dgemm(char order, char transa, char transb,
       int rowac, int colbc, int colarowb, double scale,
       double *A, int astride,double *B,int bstride,
           double what,double *C,int cstride) {

       int i,j,k;
       double s;

       for(j=0;j<rowac;j++) {
   ```

```
        for(i=0;i<colbc;i++) {
            s=0.0;
            for(k=0;k<colarowb;k++) {
                s+=A[(j*rowac)+k]*B[(k*colbc)+i];
            }
            C[(j*colarowb)+i]=s;
        }
    }
}
```

Frequency	Idle Power	Load Power	Time	Energy
125MHz	2.3W	2.45W		
250MHz	2.4W	2.45W		
500MHz	2.5W	2.75W		
550MHz	2.6W	2.85W		
600MHz	2.6W	3.0W		

Questions:

(a) Which frequency setting used the least amount of Energy?

2. **Part 2: Measuring Energy of ATLAS Matrix-Matrix Multiply**

Measure the time needed to compute an ATLAS optimized 300x300 matrix-matrix multiply (the code repeats 10 times internally). Use the provided `dgemm_atlas` program run like this:

```
time ./dgemm_atlas 300
```

Record the "user" time and use it with the provided under-load power in the table below to calculate total energy.

The ATLAS library is optimized for performance. It uses the CBLAS (basic linear algebra) interface, a standard interface for doing matrix calculations. The call looks something like this:

```
cblas_dgemm(CblasRowMajor, CblasNoTrans, CblasNoTrans,
    m, n, k, 1.0, A, m, B, k, 0.0, C, m);
```

Frequency	Idle Power	Load Power	Time	Energy
125MHz	2.3W	2.5W		
250MHz	2.4W	2.5W		
500MHz	2.5W	2.7W		
550MHz	2.6W	2.8W		
600MHz	2.6W	2.9W		

Questions:

(a) Which frequency setting used the least amount of Energy?

(b) Does the ATLAS optimized code perform better than the naïve code (speed wise)?

(c) Does the ATLAS optimized code perform better than the naïve code (energy wise)?

3. **Part 3: Complex Usage Case**

   You are desiging an embedded system which takes a picture with a small camera and then processes it using a naïve matrix-matrix multiply as used in Part 1. The system needs to take a picture every 4 minutes. Calculate the total Energy used in a 4-minute time period. Re-use the time values you measured for Part 1. Assume that time spent not calculating the matrix multiply is idle.

Frequency	Idle Power	Idle Time	Idle Energy	Load Power	Load Time	Load Energy	Total Energy
125MHz	2.3W			2.45W			
250MHz	2.4W			2.45W			
500MHz	2.5W			2.75W			
550MHz	2.6W			2.85W			
600MHz	2.6W			3.0W			

   (a) Which algorithm and frequency results in the lowest Energy use?

   (b) Why might you use a configuration other than lowest Energy in your design?

4. **Part 4: x86 Comaprison**

   On an x86 Sandybridge machine running recent Linux the default power governor does not allow userspace setting! For naïve matrix multiply we find these results:

Frequency	Idle Power	Load Power	Time	Energy
Turboboost	15.0W	24.3W	0.372s	9.0396J

   And for ATLAS we find these results:

Frequency	Idle Power	Load Power	Time	Energy
Turboboost	15.0W	25.0W	0.112s	2.8J

   Questions:

   (a) Which machine, Gumstix or Sandybridge, uses the least Energy per matrix computation?

   (b) Which would you use in an embedded situation, the Gumstix or the Sandybridge? Why?

   (c) If you were designing a supercomputer using 1-million processors that would do nonstop matrix multiplications all day, which processor would you use, a Gumstix Cotex-A8 or the Sandybridge? Why?

   (d) If you were designing a system that used 1-million processors there were idle most of the day, which processor would you use, a Gumstix Cortex-A8 or the Sandybridge? Why?

   (e) The Sandybridge machine's power supply can supply 85W of power. Why is our simple matrix multiply benchmark not using the maximum power?

5. **Submitting your work**

   - Create a document including the tables and answers to the questions. This can be in plain text, pdf, or MS/Libre Office format.
   - e-mail your assignment to me by the deadline. Be sure to send the proper file!