Power-aware Computing on GPGPUs

Kiran Kumar Kasichayanula
University of Tennessee

Stanimire Tomov
University of Tennessee

Haihang You
University of Tennessee

Heike Jagode
University of Tennessee

Shirley Moore
University of Tennessee

Matt Johnson
University of Tennessee

INTRODUCTION

Recently, GPGPU accelerated computing systems have drawn the attention of researchers. Because GPGPUs have affluent cores and arithmetic computational units, they are inherently suited for massive parallel and computation intensive workloads. The most recently built supercomputer Tianhe-1A, equipped with Intel Xeon 5670 and NVIDIA GPGPU Tesla M2050, has reached up to 2.57 PFlop/s Linpack performance, winning the second top spot on the TOP500 list in June 2011. **Coming along with the exciting computational capability, the power consumption of supercomputers has become a serious issue.** For example, the average power consumption of the TOP 10 supercomputing centers was 1.32 Mw in 2008 and 3.2 Mw in 2010, translating to a multi-million-dollar electric bill. **Designers must employ aggressive power-management techniques to keep ballooning power cost under control.** A key challenge to effective runtime power management is estimating the real-time power consumption. Although the power estimation for processors, memories, disks, and fans has been introduced, the power estimation technique of GPGPUs is relatively less developed.

In this work, we explore the use of the NVML library to measure real-time power consumption of several fundamental BLAS libraries and LAPACK algorithms. We use implementations from the MAGMA library (http://icl.eecs.utk.edu/magma/).

MODEL

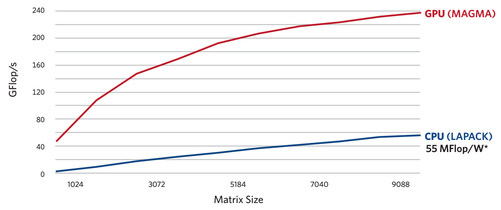
To measure the real time power consumption of a GPGPU, we use NVML (Nvidia Management Library) function calls like `nvmlDeviceGetPowerUsage` (Retrieve the power usage reading for the device, in milliwatts. This is the power draw for the entire board, including GPU, memory, etc. A device is a single GPU.) and `nvmlDeviceGetTemperature` which is a software approach. We use pthreads to run power or temperature measurements on one thread, and run MAGMA LU factorization or the MAGMA Hessenberg reduction on another and measure power in realtime. We are developing an analytical model, called Activity-based Model for GPGPUs to estimate activity factors and power for microarchitectures on GPGPUs. To measure the real time power consumption we have used the latest NVIDIA C2075 Fermi GPU which has built-in support for power management mode. To measure the actual power consumption in watts using NVML, the power management mode on the GPU needs to be enabled. However, even with a GPU similar to C2050 it's possible to measure real time power consumption in terms to power states (P0-P15) where P0 corresponds to idle state and P15 corresponds to the state when the GPU is running at full speed. In our model we show the trace of power consumption exactly when MAGMA kernel is executing by utilizing the latest feature provided in CUDA 4.0, which allows access to the same GPU by multiple threads.

RESULTS

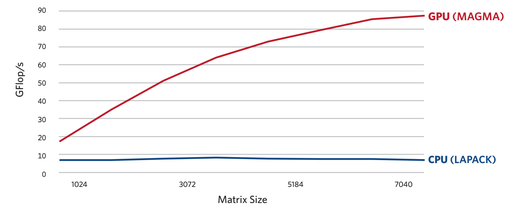
In this work we explore the use of the NVML library to measure real-time power consumption of several fundamental BLAS kernels and LAPACK algorithms. We analyzed the real-time power consumption of two fundamental linear algebra algorithms – the LU factorization (`magma_dgetrf`) for solving dense linear systems of equations and the upper Hessenberg reduction (`magma_dgehrd`) for solving the general eigen-value problem. Results show that the MAGMA implementations of these algorithms achieve astounding energy efficiency. Depending on the hardware and software configuration, we have demonstrated that MAGMA uses as little as 1/50th the power of traditional multicore CPUs. Shown are the performance charts for the two algorithms along with the real-time power consumption traces. The MAGMA LU factorization is a compute bound algorithm (expressed in terms of GEMMs) and the MAGMA Hessenberg reduction is memory bound (expressed in terms of GEMVs and GEMMS, correspondingly ~20% and 80% of the flops). The real-time power consumption for the two basic BLAS kernels (GEMM and GEMV) used is also shown.

HPC @ 1/10TH THE COST & 1/20TH THE POWER

LU Factorization in double precision (DP)
for solving a dense linear system



Hessenberg factorization in DP
for the general eigenvalue problem

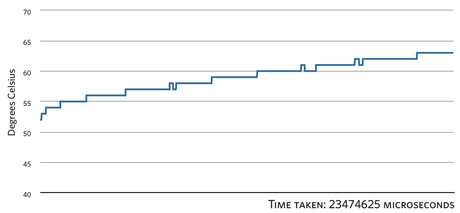


GPU Fermi C2075 [448 CUDA Cores @ 1.15 GHz]
DP Peak **515 + 40** GFlop/s
System Cost ~**$3,000**
Power ~**220** W

CPU AMD ISTANBUL [8 sockets x 6 cores (48 cores) @ 2.8 GHz]
DP Peak **538** GFlop/s
System Cost ~**$30,000**
Power* ~**1,022** W

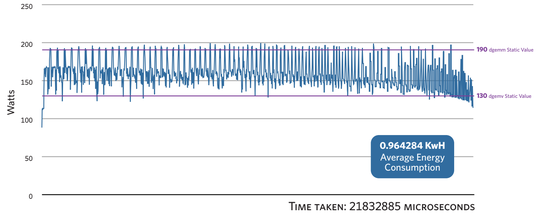
* Computation consumed power rate (total system rate minus idle rate), measured with KILL A WATT PS, Model P430

Temperature Measurment of MAGMA **dgehrd** on C2075
using NVML library for a matrix of size 8K



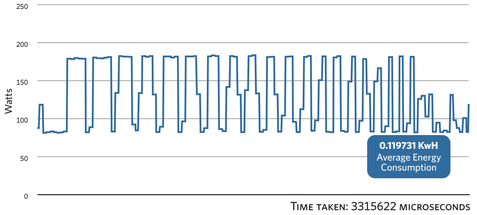
Time taken: 23474625 microseconds

Power Consumption of MAGMA **dgehrd** on C2075
using NVML library for a matrix of size 8K



Time taken: 21832885 microseconds

Power Consumption of MAGMA **dgetrf** on C2075
using NVML library for a matrix of size 8K



Time taken: 3315622 microseconds

CONCLUSION & FUTURE WORK

The current results have shown that maximizing performance leads to reduced execution time which results in proportional reduction in the energy consumption. A more detailed analysis is needed, though, and libraries like PAPI, CUPTI, NVML, and TAU will be crucial in enabling it. Other parameters influencing energy saving must be identified and energy saving hardware features must be integrated into MAGMA (e.g., what hardware context – how many CPU cores and how many GPUs – to be used to solve a problem, etc.). First and foremost this requires an investigation of the energy consumption effects of various algorithms. Future work includes the expansion of the current infrastructure for precise measurements and the development of energy consumption models. Tuning parameters must be identified and added to the MAGMA auto-tuning frameworks. The newly released PAPI (http://icl.eecs.utk.edu/papi/) CUDA Component offers a promising way to estimate structure power at runtime by extending methodology from design-time CPU power models like Wattch. This kind of technique counts utilization of structures (e.g., cache hits) and compute power estimates based on a per-event energy model. However, such direct computation of structure power on GPGPUs would require hundreds of utilization statistics. To address the challenges of estimating per-structure power in hardware, we would like to extend our Activity-based Model for GPGPUs (AMG), to estimate activity factors and power for microarchitectural structures on GPGPUs. This model will not only rely on real-time current monitoring, or simulating hundreds of utilization statistics, but also predict performance vs power consumption. We expect only a few input statistics are sufficient to estimate per-structure dynamic power of a GPGPU because the myriad of per-structure events are related to a small set of global parameters, such as IPC and load rate. We use this key observation to drive the development of AMG. We would first analyze the co-relation of variety of performance metrics. Then we monitor only the least co-related metrics and use monitored metrics to extrapolate the concerned metrics. After we obtain all of the relevant metrics about the structure events, we apply a per-event energy model derived from a circuit model to those structures to calculate the power of each structure. We believe that AMG makes a further step towards understanding and reducing the power of GPGPU systems through the usage of architecture level performance counters.

This work was supported by the National Science Foundation [NSF CNS-0910899] and the Department of Energy [DoE DE-FC02-06ER25761] [DoE DE-SC0004983]