Reliability in Warehouse-Scale Computing: Why Low Latency Matters

Nannarelli, Alberto

Published in:
Proceedings of MEDIAN Finale - Workshop on Manufacturable and Dependable Multicore Architectures at Nanoscale

Publication date:
2015

Document Version
Publisher's PDF, also known as Version of record

Reliability in Warehouse-Scale Computing: Why Low Latency Matters

Alberto Nannarelli
DTU Compute, Technical University, Kongens Lyngby, Denmark

***Abstract*—Warehouse sized buildings are nowadays hosting several types of large computing systems: from supercomputers to large clusters of servers to provide the infrastructure to the cloud. Although the main target, especially for high-performance computing, is still to achieve high throughput, the limiting factor of these warehouse-scale data centers is the power dissipation. Power is dissipated not only in the computation itself, but also in heat removal (fans, air conditioning, etc.) to keep the temperature of the devices within the operating ranges. The need to keep the temperature low within a minimal power envelope and to maintain high throughput and high reliability poses hard challenges.**
In this work, we show that by moving part of the computation to accelerators, not only we reduce the latency of operations, but also make the system more energy efficient and reliable.

I. Introduction

Warehouse sized buildings are nowadays hosting several types of large computing systems: from supercomputers to large clusters of servers to provide the infrastructure to the cloud, from web-hosting data centers, to ad-hoc computation centers (financial, scientific, etc.).
Although data centers are designed to achieve high-throughput, the main concern is power dissipation which has an high impact on the system reliability.
In nanometric sized devices many factors have an impact on the system reliability. Variability at manufacturing which makes impossible for two chips implementing the same functions to have precisely matching characteristics (inter-die and intra-die variations). Variation in the supply voltage due to drops in the power grid which causes reduction of the noise margins. Jitter and skew in clock trees resulting in uncertainties in the timing. Transient faults caused by particles upsetting the state of devices problematic even at sea level for nanometric devices.
However, high temperatures are probably the most serious threat to reliability, since temperature impacts several aspects, as explained next.
Electromigration. The Mean Time To Failure (MTTF) is the average time to which a wire fails due to electromigration. The reliability model is based on Black's equation [1]:

$$MTTF = \frac{A}{J^2} e^{\frac{E_a}{KT}} \tag{1}$$

where T is the metal temperature in Kelvin, J is the current density, A, E_a and K are constants with A being the cross-sectional area of the interconnect, E_a being the activation energy, and K being the Boltzmann constant.

Thermal runaway is caused by self-heating in conductors due to high power dissipation. One of the consequence of increased die temperature is an increased leakage power, and total power, that contributes to the rise of temperature. This is clearly a vicious circle

$$\begin{array}{ccc} T\uparrow & \longrightarrow & P_{leak}\uparrow \\ \nwarrow & & \swarrow \\ & P_{TOT}\uparrow & \end{array} \tag{2}$$

which can lead to device burn-down.
Aging is caused by negative biased temperature instability (NBTI) which affects pMOS transistors. NBTI effects are mainly dependent on temperature: transistors' threshold voltage increases with increasing temperature. The resulting effect is the progressive slow down of CMOS gates [2].
"Hot wires" (wires running over hotspots) are slower because the resistance of metal increases with temperature. This might cause delay mismatches among wires running on areas of the die at different temperatures (thermal gradient) [3].
For all these reasons, the key is to perform computation in a power efficient manner, not only to keep reliability at an acceptable level, but also to be economically viable.
For example, by projecting the increase in performance of the world's top supercomputers, the exa-scale level (10^{18} FLOPS) is likely to be reached by 2020. However, the goal is to obtain exa-FLOPS performance staying within a 20 Mega-Watt power envelope. This leads to a performance efficiency requirement of 50 GFLOPS/W, which is about 20 times higher than where we are today.
To increase power efficiency, high-performance computers are evolving towards hybrid systems where a large part of the computation is not done in CPUs but in accelerators such as GPUs (nVIDIA, AMD) or coprocessors (Intel Xeon Phi). For cluster of computers used to sustain the cloud (e.g., search engines) ad-hoc FPGA based accelerators are deployed on the motherboards next to the CPUs [4].

II. Metrics

We introduce some metrics to quantitatively define power, or energy, efficiency.
We define as t_{app} the execution time of a given application, e.g., a numerical simulation, a search query, etc. The energy necessary to run the application is

$$E_{app} = P_A \times t_{app} \quad [J] \tag{3}$$

where P_A is the average power dissipated to run the application on the platform A. In this context, the performance is defined as:

$$\text{performance} = \frac{1}{t_{app}} \qquad [s^{-1}] \tag{4}$$

and in the specific case, where floating-point operations are executed, is expressed in FLOPS.
To maximize the performance in a given power budget, the metric to consider is the performance–per–watt (PpW), defined as

$$\text{PpW} = \frac{\frac{1}{t_{app}}}{P_A} = \frac{1}{E_{app}} \qquad \left[\frac{\text{FLOPS}}{W}\right] \tag{5}$$

The temperature in a chip rises depending of the quantity of heat Q dissipated in the computation:

$$\Delta T = \frac{Q}{C_{heat}} = \frac{E_{app}}{C_{heat}} \qquad [^oC] \tag{6}$$

where C_{heat} is the thermal capacitance of the heat sink and

$$\Delta T = T_{dev} - T_{amb} \tag{7}$$

is the temperature rise of the device over the ambient temperature. By combining (6) and (7), we have

$$T_{dev} = \frac{E_{app}}{C_{heat}} + T_{amb} \qquad [^oC] \tag{8}$$

From (1) through (8), it is clear that reducing the latency t_{app} is the key to obtain energy/power efficiency, keep the temperature low, and, consequently increase reliability as well.

III. Power and Thermal Characterization of the CPU Sub-System

We run our experiments on a desktop PC equipped with a FPGA board connected to the host PC via the PCI bus.
The CPU of the host PC is an Intel processor. The CPU clock, managed by Intel's SpeedStep, can be run at two different frequencies: 2.0 GHz and 3.0 GHz.
The CPU is cooled by a fan with adjustable rotation speed in the range 1000–2000 RPM (rotations per minute). The fan's rotation speed is set by a Pulse Width Modulation (PWM) signal, and the actual rotation speed is monitored by the operating system through a monitoring signal.
The application chosen to run the experiments is a Monte Carlo simulation for option pricing algorithms [7] run on several sets of data. This type of application, is run in batch mode to statistically determine the strike price of options (a financial instrument) and use these strike values as thresholds to sell or buy options in the so called high-frequency trading market [8].
The experimental set up is shown in Fig. 1 (block diagram) and Fig. 2 (photo). In our characterization, we monitor the following parameters.

- Application execution time t_{app} recorded by the CPU timer and stored in a log file.
- CPU temperature read by an operating system's call (1 sample per second) and recorded on a log file.

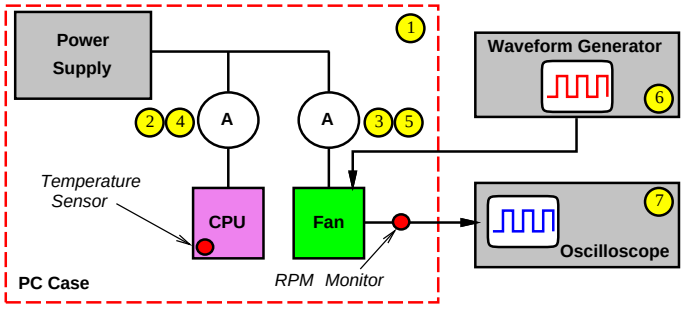

Fig. 1. Characterization set up: block diagram.

- CPU's fan rotation (RPM) read by an operating system's call (1 sample per 10 seconds) and recorded on a log file.
- Power dissipated in the CPU. The power dissipated in the CPU is monitored by a Hall's effect current sensor – (2) in Fig. 1 and Fig. 2 – which is connected between the power supply and the motherboard's ATX socket (detail in [6]). The Hall sensor is sampled every 100 ms by a digital multimeter (DMM) (4) and the readings are recorded in the DMM's internal memory.
  By averaging the power readings over t_{app}, we can determine the average power dissipated in the CPU while the application is running P_{CPU}.
- Power dissipated in the CPU's fan. Another Hall sensor (3) is placed in series with the power cable of the CPU's fan. A second DMM (5) samples the sensor every 1 s and records the data in the internal memory. By this second DMM, we can determine the average power dissipated in the fan while the application is running P_{fan}.

The characterization consists in running the application under different performance modes (set by SpeedStep) and under different cooling levels (CPU's fan rotation speeds). To set the fan rotation speed, we disconnected the fan control wire from the motherboard and connected it to an external pulse generator (6). The fan monitoring signal is a periodic signal with frequency proportional to the RPM that is read by a operating system's call, and that we monitor on an oscilloscope (7).
In summary, we run the application at two different clock frequencies: 2 and 3 GHz; and two fan's rotation speeds: 1100 and 2000 RPM.
As the first step, we calibrated the system by measuring all the parameters when the CPU is not running the application. We call this the *"rest"* state although several processes (operating system and basic services) are running. We report in Table I the readings in *rest* state.

Temperature	^{o}C	CPU Power	[W]
Ambient	24	@ 2 GHz	4.03
CPU	37	@ 3 GHz	4.40

TABLE I
CPU SUB-SYSTEM READINGS IN *rest state*.

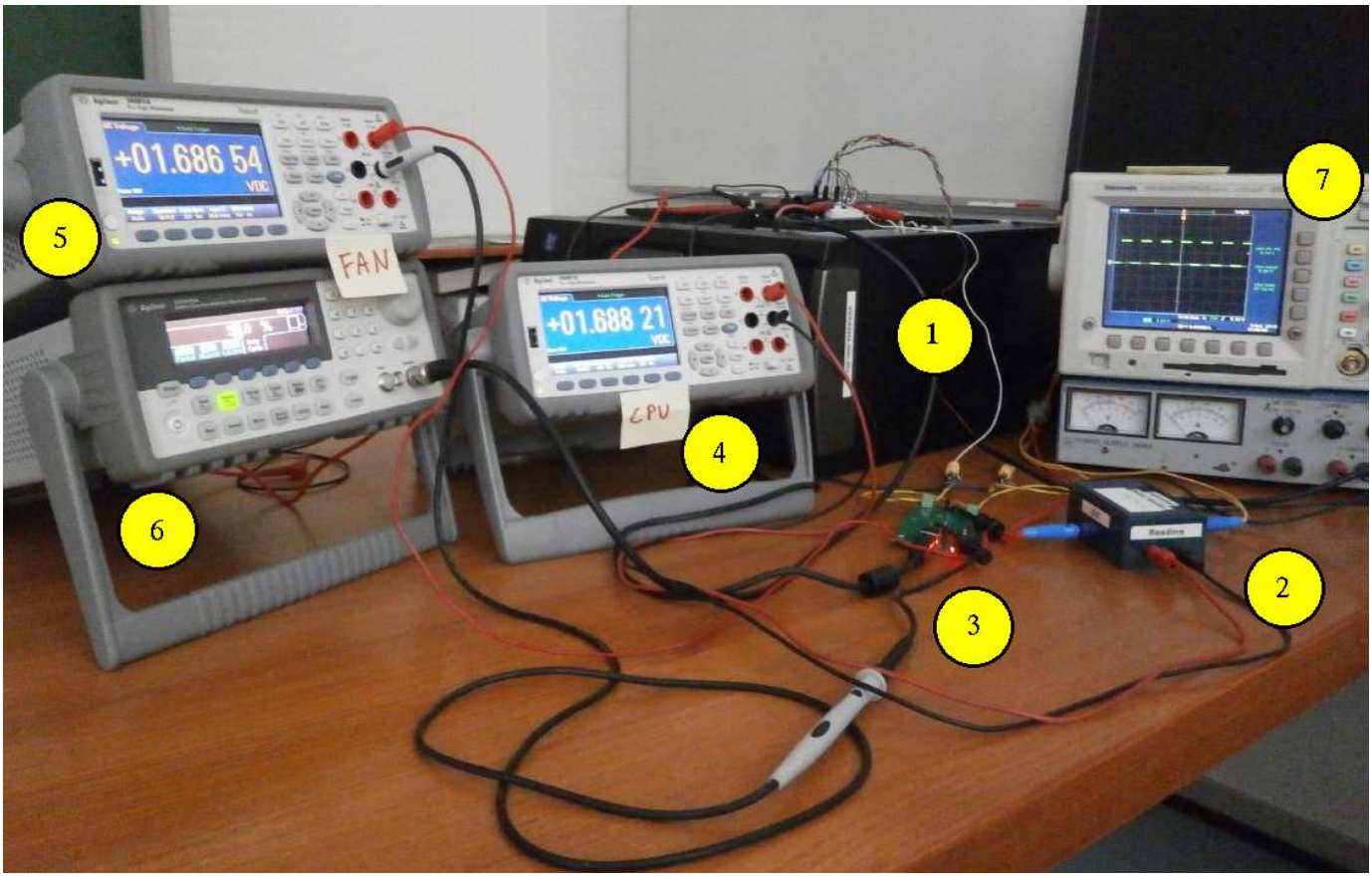

Fig. 2. Photo of characterization set up.

run	CPU freq.	Fan rot.	t_{app}	Temp. CPU ave.	Temp. CPU max.	Power P_{CPU}	Power P_{fan}	Power P_{TOT}	Energy E_{app}
LL	2	1100	267.4	42	44	5.683	0.545	10.265	2,745
LH	2	2000	267.4	40	41	5.704	1.091	10.831	2,896
HL	3	1100	177.3	**54**	**59**	**19.157**	0.545	23.739	4,210
HH	3	2000	177.3	50	53	18.733	1.091	**23.860**	**4,231**
	[GHz]	RPM	$[s]$	^{o}C	^{o}C	$[W]$	$[W]$	$[W]$	$[J]$

TABLE II
CPU SUB-SYSTEM SIMULATIONS (DATA SET SIZE 500).

In the first set of experiments, we run the application for 500 different data by changing clock frequency and fan's rotation speed for a total of four combinations (runs). The measured values of this first experiment are reported in Table II. In the table, we report with "H" the runs done with high values of clock/RPM, and with "L" the runs done with low values.
From Table II, we can notice the following.

- The ratio of execution times between 3 GHz and 2 GHz clock frequencies is 2/3, as expected.
- The highest average and maximum temperatures are measured for run "HL" (3 GHz and 1100 RPM), as expected.
- The highest power dissipation in the CPU is also for run "HL". Since the clock frequency is the same as in run "HH", the extra power dissipation in P_{CPU} depends on the higher temperature of the chip.
- The fan power dissipation is reduced by about 50% when running at 1100 RPM with respect to full speed (2000 RPM).
- The total power dissipation is higher in "HH" by about 0.5% with respect to the run "HL" because of the extra power in P_{fan}.

From Table II, the runs at 2 GHz clock are much more power efficient if we can afford the longer latency. The reason is that when the clock frequency is scaled, also the supply voltage is scaled by the SpeedStep governor, and the combination of the two scalings results in a reduction of P_{CPU} to about one third.
In contrast, for runs at 3 GHz, saving power in P_{fan} does not pays off since P_{TOT} is marginally affected, but the CPU

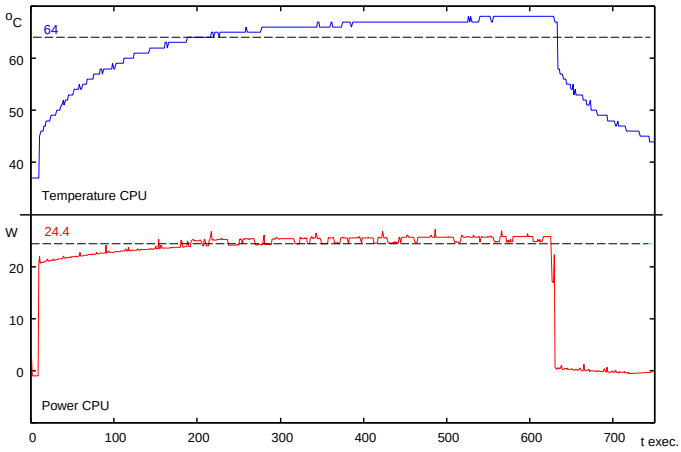

Fig. 3. Power dissipation and temperature rise in CPU in "HL" run (10 minute simulations).

run	Temp. CPU ave.	Temp. CPU max.	Power P_{CPU}	Power P_{fan}	Power P_{TOT}	Energy E_{app}
HL	**64**	**68**	**24.436**	0.617	**29.089**	**18,069**
HH	58	60	23.149	**1.091**	28.276	17,588
	^{o}C	^{o}C	$[W]$	$[W]$	$[W]$	$[J]$

$t_{app} = 621\ s$ at 3 GHz

TABLE III
CPU SUB-SYSTEM SIMULATIONS (10 MINUTES).

temperature is considerably higher (4^{o}C on average).
During the experiments of Table II, we noticed that for 3 GHz runs the CPU temperature did not settle for execution times of about 3 minutes.
Therefore, we extended the data set to have approximately 10 minutes of computation. We only ran experiments for "HL" and "HH". We show in Fig. 3 the plots of the measurements of the CPU temperature and P_{CPU} while the simulation "HL" is progressing. The measured values (average and maximum) are reported in Table III
Fig. 3 clearly shows the dependency power-temperature sketched in (2). The temperature rise in the first 200 seconds of the simulations result in an increased power dissipation. Afterwards, the CPU-fan sub-system reaches a thermal equilibrium in the range 64–68^{o}C and the power dissipation plateaus as well.
In this longer simulation, the "HL" run is definitely less power efficient than the run with the fan at full rotation speed: higher power dissipation and higher temperatures.
In summary, for the Monte Carlo simulations run in software (CPU), there is a large trade-off between execution time and power dissipation (Table II). For runs at 3 GHz clock, by spinning the fan at 2000 RPM provides the best power efficiency and lower temperatures.

IV. Hardware Acceleration of Monte Carlo Simulations

In this section, we show how FPGA based Application Specific Processors (ASPs) can accelerate the Monte Carlo simulations.

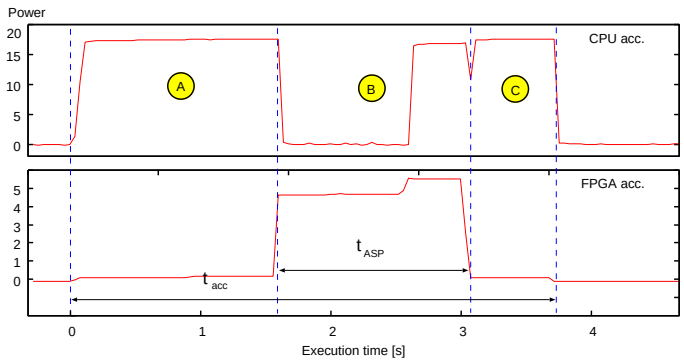

Fig. 4. Energy consumption to execute application in software (top), and in the accelerator (bottom).

The hardware accelerator is implemented in a Xilinx FPGA embedded in the Alpha Data ADM-XRC-5T2 board. The board is connected to the host PC via the PCI bus. The detailed hardware description of the FPGA accelerator is given in [5] while the system CPU+accelerator is described in [9].
In addition to the power dissipation measurements performed on the CPU, we also measure the power dissipated in the FPGA board (P_{FPGA}) when executing the computation core of the simulations in the ASP.
To monitor the power consumption in the FPGA accelerator with a good resolution, we opted for a data set of size 200.
Fig. 4 shows the plots for the simulation when executed on the FPGA accelerator. In the top part of Fig. 4, we report P_{CPU} when executing the simulation on the accelerator, in the bottom part, P_{FPGA}. We can identify three regions:

- (A) A first region where the CPU configures the ASP on the FPGA and transfers the simulation parameters.
- (B) A second region where the CPU is almost idle and the ASP is busy performing the computation. Toward the end of this period, the CPU wakes up to receive the results of the simulation.
- (C) A final region where the CPU closes the communication with the FPGA. The simulation is quit.

The *rest* state power, marked as zero in Fig. 4, are about 4 W for the CPU and about 9 W for the FPGA board.
For the execution in the ASP, we run the simulations at CPU frequencies 3 GHz ("HH") and at 2 GHz ("LL"). In the latter case, we also reduced the fan speed. The ASP is clocked at 80 MHz in both simulations.
In Table IV, we list the timing measurements and the energy consumption for the software execution (SW CPU) and for the accelerator. We also list the average CPU temperature and the fan's RPM, Moreover, we report the speed-ups and energy ratios.
The results show that the speed-up by hardware acceleration is between 12 and 19 times, and that the ASP solution is about 14 times more power efficient than running simulations in software.
By running the simulation "LL" (2 GHz and 1100 RPM), we can save a large amount of power in the CPU, and the

run	CPU freq.	Fan rot.	t_{app}	Temp. CPU average	Power P_{CPU}	P_{fan}	P_{FPGA}	P_{TOT}	Energy E_{app}	speed-up	ratio E_{app}
SW CPU	3	2000	71.04	48	17.76	1.09	–	22.88	1,626	1.0	1.0
CPU+ASP HH	3	2000	3.73	40	13.14	1.09	11.30	29.57	110	19.1	14.7
CPU+ASP LL	2	1100	5.81	39	4.07	0.55	11.30	19.95	116	12.2	14.0
	[GHz]	RPM	$[s]$	^{o}C	$[W]$	$[W]$	$[W]$	$[W]$	$[J]$		

TABLE IV
RESULTS OF MONTE CARLO SIMULATIONS RUN IN SOFTWARE AND IN THE ASP. P_{TOT} ALSO INCLUDES THE POWER IN REST STATE (CPU).

CPU's fan, at expenses of a longer latency than the simulation "HH". However, the energy efficiency of the two CPU+ASP simulation is about the same.

Moreover, the data in Table IV show that the software execution results in higher CPU temperature, since the CPU is consuming power for a longer period of time.

As described by equation (8) to keep the temperature in a medium/low range we need to increase C_{heat} or to lower T_{amb}.

The heat sink capacity can be increased by having the fan rotating faster. The ambient temperature can be lowered by air conditioning the room containing the computer. In both cases, to keep the temperature low, additional power needs to be consumed for the cooling.

By using accelerators, we can save both power in the computation and in the cooling.

Moreover, from an economical point of view, accelerators can help reducing the energy bills, as well.

V. CONCLUSIONS

In this paper, we presented the rationale of why an accelerator intended to reduce latency, is also beneficial to reduce the energy consumption, the temperature rise, and therefore, increase the reliability.

To support these claims, we presented a case study in which the application, Monte Carlo simulations for financial computing, is entirely run in the CPU, or run, the computing intensive parts, in an accelerator implemented in a FPGA board.

REFERENCES

[1] J. R. Black, "Electromigration - a brief survey and some recent results," *IEEE Transactions on Electron Devices*, vol. 16, no. 4, pp. 338–347, Apr. 1969.

[2] A. Calimera, W. Liu, E. Macii, A. Nannarelli, and M. Poncino, "Power and Aging Characterization of Digital FIR Filters Architectures," in *Proc. of the First MEDIAN Workshop*, Annecy, France, Jun. 2012, pp. 1–6.

[3] A. H. Ajami, K. Banerjee, and M. Pedram, "Modeling and analysis of nonuniform substrate temperature effects on global ULSI interconnects," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 6, pp. 849–861, Jun. 2005.

[4] A. Putnam *et al.*, "A reconfigurable fabric for accelerating large-scale datacenter services," in *Proc. of ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, Jun. 2014, pp. 13–24.

[5] J. S. Hegner, J. Sindholt, and A. Nannarelli, "Design of Power Efficient FPGA based Hardware Accelerators for Financial Applications," in *Proc. of the 30th NORCHIP Conference*, Nov. 2012.

[6] P. Albicocco, D. Papini, and A. Nannarelli, "Direct Measurement of Power Dissipated by Monte Carlo Simulations on CPU and FPGA Platforms," in *IMM-Technical Report-2012*, no. 18, 2012. [Online]. Available: http://orbit.dtu.dk/services/downloadRegister/51818668/1.pdf

[7] J. C. Hull, *Options, Futures and other Derivatives*, 8th ed. Prentice Hall, 2012.

[8] J. W. Lockwood, A. Gupte, N. Mehta, M. Blott, T. English, and K. A. Vissers, "A Low-Latency Library in FPGA Hardware for High-Frequency Trading (HFT)," in *Hot Interconnects'12*, 2012, pp. 9–16.

[9] J. K. Toft and A. Nannarelli, "Energy Efficient FPGA based Hardware Accelerators for Financial Applications," in *Proc. of the 32th NORCHIP Conference*, Nov. 2014.