Localized Microarchitecture-Level Voltage Management

YongKang Zhu
Department of Electrical and Computer Engineering
University of Rochester
Rochester, NY 14627, USA
Email: yozhu@ece.rochester.edu

David H. Albonesi
Computer System Laboratory
Cornell University
Ithaca, NY 14853, USA
Email: albonesi@csl.cornell.edu

Abstract— **Diminishing voltage margins, coupled with power and temperature constraints, call for microarchitecture-level runtime mechanisms for voltage control. This paper describes a localized approach for dynamic voltage management within the domains of a Globally Asynchronous, Locally Synchronous (GALS) processor design. Dynamic Voltage Scaling at this fine grain level permits effective temperature management with less performance impact than global voltage control.**

I. Introduction

For CMOS circuit timing margins to be met, temperature must be controlled within specified limits. In recent years, microarchitecture techniques for Dynamic Temperature Management (DTM) have been proposed for maintaining suitable operating temperatures with reduced packaging costs [1]. The premise of this approach is that only a tiny fraction of all applications produce thermal violations; therefore, a lower cost package can be employed, so long as the hardware can detect a potential thermal violation at runtime and react accordingly, for instance, through Dynamic Voltage Scaling (DVS).

Although DVS is generally the most effective thermal management response, it has the disadvantage of impacting global microprocessor performance due the global reduction in clock frequency, even if the thermal emergency is isolated to a small region of the die (such as the integer region). Table I shows the thermal characteristics of the SPEC2000 programs used in this paper. The simulation parameters, which are presented in Section III, were chosen to reflect the increased power density in future process generations, due to the slow scaling of supply voltage relative the transistor dimensions. One observation from this table is that the units in the front-end part of the processor are never among the hottest; therefore, there is little need to ever throttle performance in that domain for temperature purposes. Note also that the hottest units are located within at most two of the regions of the die (integer, floating point, or load-store).

These results indicate that a *localized* response to temperature emergencies may be effective in maintaining acceptable temperature levels while maintaining global performance. One such approach, localized throttling of the clock within the region of interest, was previously proposed [2]. However, this approach only impacts frequency, and therefore is often too gentle in addressing severe thermal emergencies [2]. Such severe thermal problems are expected to arise more often in the future with higher power densities and the continuing need for cost-effective and low profile chip packaging.

TABLE I
THERMAL CHARACTERISTICS OF A FULLY SYNCHRONOUS MICROPROCESSOR WITHOUT ANY DTM CONTROL FOR SPEC 2000 INTEGER (TOP) AND FLOATING POINT (BOTTOM) PROGRAMS. THE HOTTEST UNITS ARE THE INTEGER ISSUE QUEUE (INTQ), REGISTER FILE (IREG), AND EXECUTION UNIT (IEXEC); FLOATING POINT REGISTER FILE (FREG), ADDER (FADD), AND MULTIPLIER (FMUL); AND THE LOAD-STORE QUEUE (LSQ).

	max temp (degrees C)	3 hottest units
crafty	92.4	iExec, IntQ, iReg
eon	98.0	fAdd, fReg, LSQ
gzip	90.2	iExec, IntQ, iReg
mesa	92.8	fAdd, fReg, IntQ
equake	96.5	iExec, IntQ, iReg
facerec	89.6	fAdd, fReg, fMul
fma3d	91.7	iExec, iReg, IntQ
galgel	125.6	fMul, fAdd, fReg

In this paper, localized DVS-based DTM is proposed via a Globally Asynchronous, Locally Synchronous microprocessor called MCD [3]. In MCD, the major microprocessor functions are located in separate clock/voltage domains. The advantage of this approach, in terms of DTM, is that a localized response can be made to the particular unit which is overheating at any given point of execution. This permits other domains to maintain full speed operation, leading to less performance overhead compared to a fully synchronous processor with DVS-based DTM. The performance cost is inter-domain synchronization. This added cost is shown to be offset by the lower performance overhead afforded by localized DVS control.

II. DVS-based DTM algorithm

Due to the cubic relationship between dynamic power consumption and operating frequency, DVS has been shown to be highly effective in reducing chip power consumption. However, transitioning voltage and frequency is slow, which makes it difficult to determine the right voltage and frequency values for the future based on knowledge of current and previous thermal behavior.

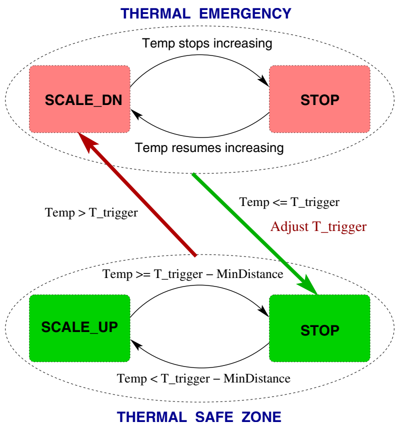

Fig. 1. DTM algorithm.

Control theory based schemes have been proposed to address this issue [2], yet their effectiveness is highly dependent on how well the controlled system is modeled and then linearized. Usually the transfer function of the controlled system varies when running different benchmarks or different phases of one benchmark, which makes it difficult to find a good set of controller parameters for a wide range of applications. Stability analysis shows how well the pre-specified target is being followed, but will not necessarily catch any non steady state characteristics of the system, for system behavior at the initial period. Finally, such DTM algorithms cannot guarantee that temperature will not exceed the thermal limit due to the variety of system response characteristics. Therefore, a fail-safe backup scheme is required.

The proposed DTM algorithm, shown in Figure 1, does not specify any target voltage. Rather, it samples temperature values at a fine enough granularity to catch small changes, assuming next generation thermal sensor technology. To prevent exceeding the thermal limit, a trigger temperature is specified for engaging DVS. Since the thermal behavior and characteristics of every unit is different at different (thermal) phases of execution, a single trigger temperature does not work optimally in all cases. Therefore, the algorithm sets the trigger temperature to an initial value and then dynamically changes it based on previous thermal behavior. For a fully synchronous machine, one trigger temperature is used for all units; for an MCD machine, each domain has one trigger temperature for all the units in that domain. In both cases, thermal sensors are assumed for all of the hotspot units.

Each domain is initially in a thermal safe zone (Figure 1). When any monitored unit temperature exceeds the trigger temperature, the domain enters the thermal emergency zone and the voltage and frequency are reduced at a constant speed. This continues until the maximum temperature is observed to stop increasing and is below the hard limit. As in Intel's XScale processor [4], all circuits operate during the period of voltage and frequency transition. After the scaling down process is terminated the temperature either remains between the trigger temperature and the hard limit; resumes increasing sometime later, in which case DVS resumes; or gradually decreases below the trigger temperature and enters the safe zone.

The trigger temperature is adjusted when the thermal safe zone is entered after an emergency. During the emergency period, a special register records the maximum temperature reached by any unit in that domain. If the value in this register is larger than the hard limit, the trigger temperature is set too high and needs to be lowered; if it is smaller, the trigger temperature can be potentially increased. The difference between the maximum temperature in an emergency and the hard thermal limit is calculated, and the trigger temperature is then adjusted in either direction by this difference. Increasing the thermal temperature may be risky in terms of absolute safe temperature control, since there should be some temperature buffer. This is handled by requiring a minimum separation (0.5 degrees in the algorithm) between the trigger temperature and the hard limit, and forcing the upper limit of the trigger temperature to be lowered dynamically; in this paper, the limit is lowered by 0.5 degrees for every three thermal violations.

If after returning to the safe zone the temperature still decreases, the voltage is increased until the temperature is close to the trigger temperature. This minimum temperature distance (called *MinDistance*) must be carefully chosen. If the value is too small, the temperature may oscillate around the trigger thus incurring an alternating scaling down and up process. Too large a value increases stability, but may degrade performance. While dynamic adjustment of the minimum distance is possible, a static value is used in the algorithm.

The last parameter is the voltage transition speed. In the proposed algorithm, the scaling up process occurs at the fastest practical speed, while scaling down occurs at a slower rate. This reduces the chances that a scaling down process results in a subsequent need to scale up, thereby incurring less oscillation and potentially benefiting performance.

III. Evaluation Methodology

The evaluation methodology is based on the SimpleScalar and Wattch toolkits [5], [6] and the HotSpot temperature modeling tool [1]. The microarchitecture and temperature modeling parameters are shown in Tables II and III. Temperature sensors are placed at all relevant units and assumed to have an accuracy of 0.5 degrees Celsius.

Of the SPEC2000 benchmark programs, the eight with the most severe thermal problems were chosen; the remaining benchmarks generated no or very few thermal emergencies. Each benchmark was fast-forwarded 2 billion instructions, followed by the warm up of various structures (like branch predictor and caches) for 100 million instructions, and then the warm up of different chip units to reach representative temperature values, for another 200 million instructions. Statistics were then gathered for the next 300 million instructions.

For each result with each benchmark, three simulation runs were conducted, with each run taking the steady state temperatures from the previous run as the initial temperatures, except for the first run which set the initial temperature at 80 degrees Celsius and operated without any DTM control.

TABLE II

ALPHA 21264–LIKE ARCHITECTURAL PARAMETERS.

Configuration Parameter	Value
Branch predictor:	
Level 1	1024 entries, history 10
Level 2	1024 entries
Bimodal predictor size	1024
Combining predictor size	4096
BTB	4096 sets, 2–way
Branch Mispredict Penalty	7
Decode/Issue/Retire Width	4/6/11
L1 Data Cache	64KB, 2–way set associative
L1 Instruction Cache	64KB, 2–way set associative
L2 Unified Cache	1MB, direct mapped
L1 cache latency	2 cycles
L2 cache latency	12 cycles
Integer ALUs	4 + 1 mult/div unit
Floating–Point ALUs	2 + 1 mult/div/sqrt unit
INT Issue Queue Size	20 entries
FP Issue Queue Size	15 entries
Load/Store Queue Size	64
Physical Register File Size	72 integer, 72 floating–point
Reorder Buffer Size	80

TABLE III

TEMPERATURE MODELING AND THERMAL MANAGEMENT PARAMETERS.

Temperature sampling interval	10000 cycles of a 3GHz clock
Thermal threshold	85 Degree (Celsius)
Nominal frequency	3.0 GHz
Nominal voltage	1.4 Volt
Ambient air temperature	45 Degree (Celsius)
Convection thermal resistance	0.8 K/W
Convection thermal capacitance	140.4 J/K
Die	0.5 mm thick
Heat spreader	1.0 mm thick, 3 cm $\times$ 3 cm
Heat sink	6.9 mm thick, 6 cm $\times$ 6 cm

For runs with DTM control, initial temperatures were clipped based on the pre-specified hard limit, which is set at 85 degrees Celsius. The maximum voltage is 1.4V and the frequency range is from 3GHz to 1GHz. The fastest voltage transition speed is 16.7 mV per μs.

The chip floorplan is shown in Figure 2, while the logical domain partitioning of the MCD processor (proposed in [7]) is shown in Figure 3. The integer and memory domains are merged to remove those most performance critical inter-domain channels. The unified L2 cache is in its own domain and always operates at half frequency (1.5 GHz).

IV. RESULTS

The DVS-based DTM algorithm described in Section II was applied to both fully synchronous and MCD processors (each domain independently implementing the algorithm). Figure 4 shows the corresponding performance degradation. Comparing the bars on the left (fully synchronous microprocessor with DVS-based DTM) with the maximum temperatures in Table I shows that for high temperature programs like *galgel*, *eon* and *equake*, the performance cost is high as well, more than or almost 10%. The worst performance cost is 27% for *galgel*, which is the program that has the highest temperature. For lower temperature programs such as *facerec* and *gzip*, the performance cost is also small.

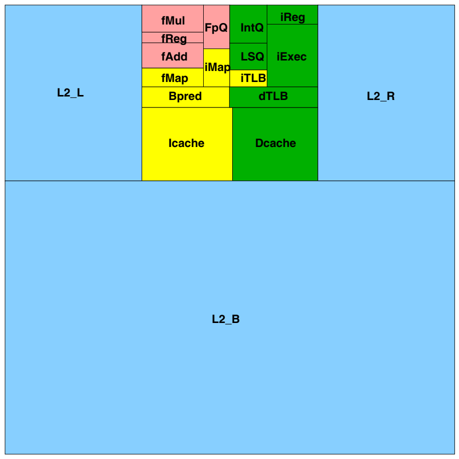

Fig. 2. Alpha21364 like floorplan showing the different MCD domains.

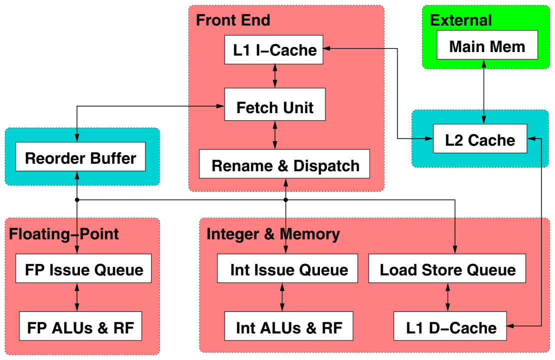

Fig. 3. MCD processor with logical domain partitions.

The performance cost of localized DVS within MCD (relative to the baseline MCD performance) is on average less than that of global DVS (relative to the performance of the fully synchronous processor) and significantly so, for some benchmarks. Since domains in an MCD processor are independent, the performance impact of DVS is largely confined within the domain where the hot spots are located. Maintaining full speed operation in other domains is especially important when one or all of the other domains are very performance critical. If there is execution slack in the hot spot domain when DVS is applied, then the performance loss is further reduced.

The difference in performance cost between localized DTM on MCD and global DTM is particularly striking for *galgel*, the program with the most severe thermal problem. As shown in Figure 5, for the MCD DVS-DTM case, only the floating point domain frequency is reduced, as the other two domains do not contain hot spots. For Global DVS-DTM, all three domain frequencies must be reduced by the same amount. This has a huge performance cost for *galgel* since for this benchmark, all three domains are performance critical, containing critical paths through the execution dataflow graph. Therefore, the ability to maintain almost full speed operation in two of the three domains through localized DTM in MCD has a huge performance advantage in programs like *galgel*.

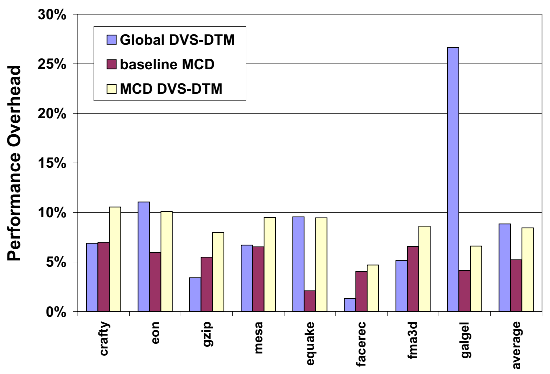

Fig. 4. Performance degradation relative to a fully synchronous machine without any DTM control. The bar in the middle shows the performance degradation of MCD without any DTM control.

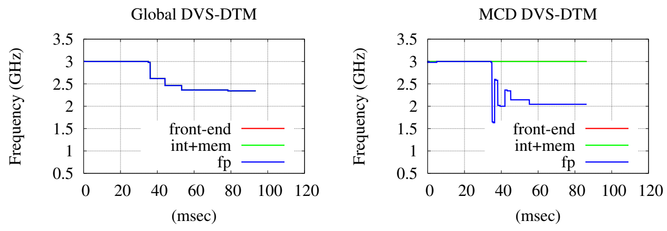

Fig. 5. Frequency profiles for *galgel*, on the left for a fully synchronous machine (all curves overlap), and on the right for MCD.

An advantage of global DVS is the cooling of neighboring units results in better heat removal from the hot spot due to a steeper temperature gradient; therefore, the voltage does not have to be lowered as much in the affected domain as in the MCD case, as seen for the floating point domain in *galgel* (Figure 5). However, this factor did not have as large a performance impact as the ability to maintain high speed operation of the front-end and integer/memory domains.

Figure 6 shows the temperature profiles without and with DTM control, for *galgel*, *eon*, *equake*, and *mesa*, which are the four benchmarks that have the most severe thermal problems. The temperature is effectively maintained below the thermal limit. There are at most only two cases at the very beginning of the simulations (at the beginning of the second warm-up phase and before the mechanisms are able to fully react) where the limit is exceeded by less than 0.05 degrees. The only exception is *facerec*, which is the only program where the upper limit of the trigger temperature is lowered three times (nine thermal violations) at the early time of the second warm-up phase and then no extra thermal violation occurs. Such rare, mild violations are unlikely to result in operation problems due to timing violations or lifetime reliability.

V. Summary

While localized dynamic temperature management has the potential for maintaining safe operational temperatures with less peformance overhead, doing so is problematic within a full synchronous processor. This paper proposes localized DTM within the domains of a GALS microprocessor. The performance overhead is considerably less for MCD versus global DTM, and the difference is particularly striking for applications in which all domains are performance critical. With increasing power density levels causing more thermal violations for the same packaging cost in the future, such localized techniques will become increasingly necessary to achieve acceptable performance.

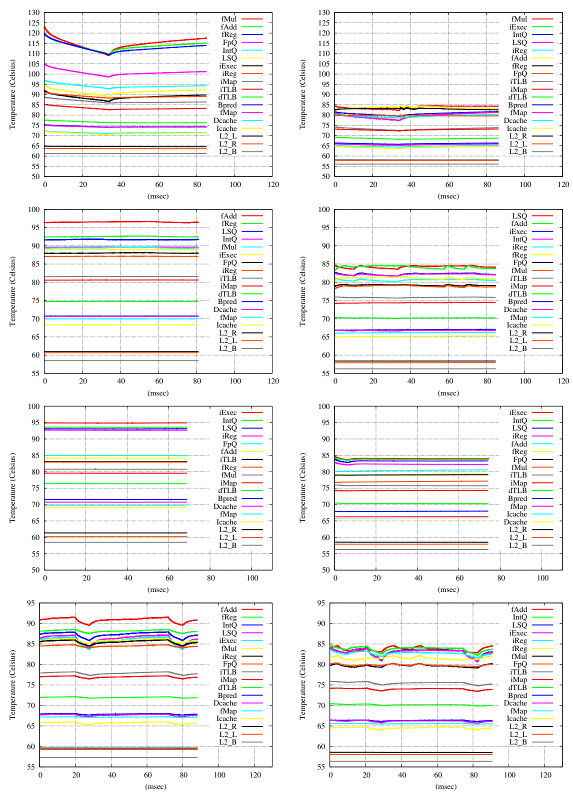

Fig. 6. Temperature profiles without (left) and with (right) DTM control, for *galgel*, *eon*, *equake* and *mesa* (top to bottom) on an MCD machine.

References

[1] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *Proceedings of the 30th International Symposium on Computer Architecture*, June 2003.

[2] K. Skadron, T. Abdelzaher, and M. Stan, "Control-theoretic techniques and thermal-RC modeling for accurate and localized dynamic thermal management," in *Proceedings of the 8th International Symposium on High Performance Computer Architecture*, February 2002.

[3] G. Semeraro, G. Magklis, R. Balasubramonian, D. H. Albonesi, S. Dwarkadas, and M. L. Scott, "Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling," in *Proceedings of the 8th International Symposium on High Performance Computer Architecture*, February 2002.

[4] L. T. Clark, "Circuit design of XScale microprocessors," in *2001 Symposium on VLSI Circuits, Short Course on Physical Design for Low Power and High Performance Microprocessors*, June 2001.

[5] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level analysis and optimization," in *Proceedings of the 27th International Symposium on Computer Architecture*, June 2000.

[6] D. Burger and T. Austin, "The SimpleScalar tool set, version 2.0," Department of Computer Science, University of Wisconsin-Madison, Technical Report 1342, June 1997.

[7] Y. Zhu, D. H. Albonesi, and A. Buyuktosunoglu, "A high performance, energy efficient, GALS processor microarchitecture with reduced implementation complexity," in *Proceedings of the International Symposium on Performance Analysis of Systems and Software*, March 2005.