Adv. Radio Sci., 11, 219–225, 2013
www.adv-radio-sci.net/11/219/2013/
doi:10.5194/ars-11-219-2013

Advances in Radio Science

Modeling of temperature scenarios in a multicore processor system

E. Glocker and D. Schmitt-Landsiedel

Lehrstuhl für Technische Elektronik, Technische Universität München, Theresienstraße 90, 80333 München, Germany

Correspondence to: E. Glocker (elisabeth.glocker@tum.de)

Abstract. In modern CMOS integrated Systems-on-Chip global temperature variations arise as well as local fluctuations in regions of high activity, resulting in the arise of local hot spots. This in turn can greatly affect reliability and lifetime of a chip. Economically affordable processor packaging cannot be provided for the worst case hot spot scenario. In a multicore system a reciprocal influence between the temperatures of neighbouring cores occur leading to increasing core temperature compared to a single core. This results in the need to monitor and regulate the operating temperature during runtime in order to keep it at tolerable values. This can be done in an easy way in an invasive architecture. In this paper the temperature distributions of cores in a multicore system are simulated for various scenarios. Different task allocation techniques and application characteristics as well as different physical conditions such as package types, material parameters and cooling all result in different system power scenarios. The impact of different scenarios which affect the system temperature scenario is investigated. The results are analysed and compared to determine the worst case scenario. With regard to simulation results and practicability the best temperature levelling measures are chosen.

1 Motivation

Integrated circuits today and even more in the future are subject to significant variations: Between different components (resulting from fluctuations in manufacturing process), over space (e.g., "hot spots") and time (short-term: resulting from fluctuations in operating conditions, long-term: resulting from degradation effects due to ageing). This results in significant differences in processing capabilities and in susceptibility to degradation because of varying processing loads for different processing elements even on the same chip. Especially as we can foresee system-on-chip (SoC) architectures with 1000 or more processors on a single chip, static, central management concepts for execution control may meet their limits. So ways must be found to deal with these imperfections resulting in more and more unreliable components.

Invasive architectures for multicore systems provide the required self-organising behaviour: the main idea of invasive computing is a resource-aware programming support, where parallel programs get the ability to explore the system and make decisions for execution (e.g. number of processors to execute on) based on the current state – including physical hardware properties – of the hardware platform (Teich et al., 2011).

To realise invasive architectures, a closed-loop between applications and the underlying hardware is necessary with the need of physical hardware for monitoring and regulating the physical conditions of the processor system, including temperature.

In modern CMOS technologies core temperature in a multicore is increasing and distributed non-uniformly: power densities in modern CMOS technologies increase, because of decreasing feature size and continuing demand for higher performance. Dependent on the thermal conductivity of chip and package, this will influence chip temperature, leading to global variations and local fluctuations in regions of high activity. Therefore, in case of non-uniform workload of different circuit blocks, local temperature hot spots will arise. This together in turn can greatly affect reliability and life-time of a chip (Semenov et al., 2006; Brooks et al., 2007). Additionally, with the power density also the costs for packaging and cooling increase. As a result, economical, reasonably priced processor packaging cannot be provided for the worst case hot spot scenario anymore (Gunther et al., 2001).

In a multicore system, the temperature of a processor block not only depends on its own power density, but also on the power density of neighbour blocks. This leads to a reciprocal influence between the core temperature of neighbouring cores, and with that an even higher increasing core temperature can arise. By monitoring the operating temperature

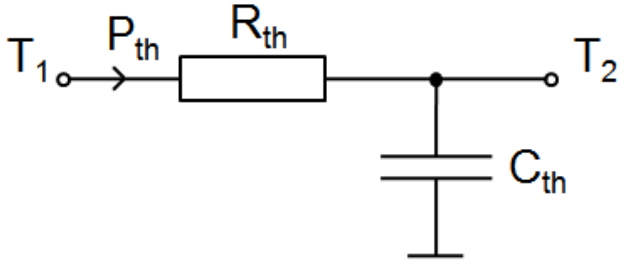

Fig. 1. Due to the duality between heat transfer and electrical phenomenas a RC circuit based thermal model is possible.

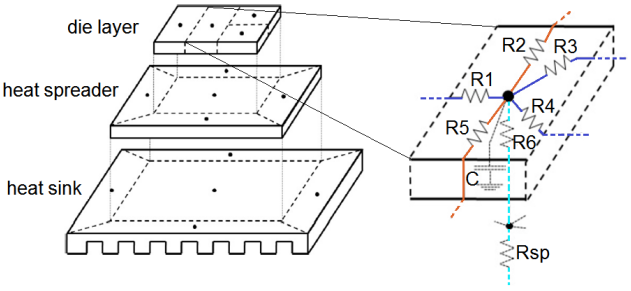

Fig. 2. Modelling of a whole microarchitecture with equivalent RC circuits.

during runtime, it is possible to keep it at tolerable values by load balancing.

After describing the used temperature model in Sect. 2, the simulation environment is presented (Sect. 3). The temperature distribution for various scenarios in a multi-processor system is presented in Sect. 4 and the best temperature limiting measures are presented in Sect. 5.

2 Temperature model

Due to the duality between heat transfer and corresponding electrical parameters – current and heat flow are described by the same potential difference differential equations – a thermal model based on thermal resistances and capacitances can be made: as in an electrical RC circuit, the thermal Rs and Cs lead to exponential rise and fall times characterized by thermal RC time constants. Figure 1 shows the basic components of the thermal model: the temperature difference ΔT can be considered as analogue to the voltage drop and the input power P_{th} as analogue to the current flow. So heat flow can be described as current passing through a thermal resistance R_{th} leading to a temperature difference ΔT. The capacitance C_{th} determines the transient behaviour, i.e. the delay before a power at the input results in reaching the steady state temperature (Boehm, 200; Skadron et al., 2004).

For modelling a whole microarchitecture with the thermal RC model, heat sink, heat spreader and die layer are broken up into several blocks as shown in Fig. 2 (left side). For the die layer the blocks directly correspond to the microarchitecture units. Heat convection is modelled at the package-air interface by a thermal resistance. Inside the package and the silicon only heat conduction has to be modelled, since this is the dominant mechanism there. For each of the blocks, the equivalent RC circuit consisting of lateral and vertical circuit elements is constructed as shown on the right side of Fig. 2.

In the shown example the resistances $R6$ and Rsp model the vertical heat flow. $R1$ to $R5$ model the lateral heat flow from the block center to the center of each edge: $R1$, $R3$ and $R4$ model the heat exchange with neighbour blocks and $R2$ and $R5$ model the heat flow to the next layer (heat spreader/sink in this case). In combination with the capacitance C the transient temperature changes can be modelled. The block node models the power source P as each circuit block is modelled as a heat (power) dissipator (Skadron et al., 2004).

3 Simulation environment

The temperature simulations are done with HotSpot, a thermal model tool suitable for architectural studies (Huang et al., 2004). This compact simulator shows good agreement with finite-element simulations: For steady-state temperature simulations the errors in HotSpot (with respect to ambient temperature of 45 °C) compared to the finite-element simulator Floworks were found to be always smaller than 5.8 % (usually smaller than 3 %) (Skadron et al., 2004). HotSpot was validated against experimental results obtained from a test chip and showed good agreement, with largest errors for steady-state temperature smaller than 5 % (Huang et al., 2004). As input for the temperature simulation, a floorplan and power values per block and time step are necessary. The power values are generated via power simulations with PTScalar, a cycle-accurate microarchitecture-level performance and power simulator for SuperScalar architectures (Liao et al., 2005).

The simulations are done at a base processor frequency of 3.47 GHz and a supply voltage of 1.5 V. Ambient temperature is 35 C and chip thickness is 0.5 mm. Convection capacitance is 140.4 J K^{-1}, convection resistance is 0.1 K W^{-1}, silicon thermal conductivity is 100 $\text{W m}^{-1}\,\text{K}^{-1}$ and silicon specific heat is 1.75 $\text{MJ m}^{-3}\,\text{K}^{-1}$ (Skadron et al., 2004). A sampling time step of 2.8818 µs, corresponding to 10 k clock cycles at 3.47 GHz, is chosen. This means the power values are simulated in each time-step, averaged over the last 10 k cycles. The temperature error is less than 0.1 C when using a time step every 10k clock cycles or lower (Skadron et al., 2004). The used processor floorplan is based on a 0.13 µm DEC ALPHA 21364 processor, where the entire periphery is treated as L2 cache (Skadron et al., 2004), as shown in Fig. 3. For the multicore floorplan 16 single DEC ALPHA's (4×4) were used. As benchmark example for the power simulations an anagram program was used. Based on the power

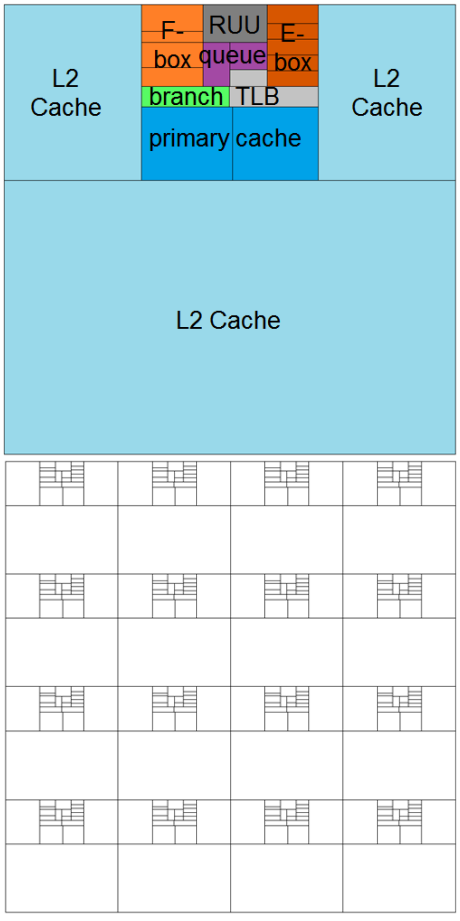

Fig. 3. Single core floorplan based on DEC ALPHA 21364 processor, where entire periphery is treated as L2 cache and resulting 4×4 multicore floorplan.

simulation results with the benchmark, a power scenario with a time step that results in an average power per clock cycle of 39 W (corresponding to the average power of the benchmark) is chosen.

4 Evaluation of temperature scenarios

In the following simulations the steady-state temperature is simulated. In general the steady-state temperature is based on average power dissipation and is a good estimation for the temperature present when the average task execution time is large enough. The rate of heat dissipation is affected by the duty cycles of the power sources where different thermal time constants are involved: for package/substrate and die, thermal time constants are non-critical since they are within seconds or milliseconds, respectively. The individual device temperature peak is within 100s of microseconds, so an instantaneous peak junction temperature much higher than the steady-state temperature is possible due to accumulated heat (Chandra, 2006). This is why at first the results of steady-state and transient temperature simulation for a single core (anagram program, runtime 0.28 ms) is evaluated: for calculating steady-state temperature an averaged power value over the whole runtime per core unit is considered. The transient temperature is calculated for every time step during the runtime by using the power values per core unit for the desired time step. The transient temperature per core unit over the whole runtime can be considered as constant, since it varies in all cases by a value smaller than 0.1 C. The steady-state temperature for all the core units normally differs by around 1 °C from its transient temperature. In the most extreme case the steady-state temperature for the IntReg core unit was found to be 3.67 °C lower than its transient temperature.

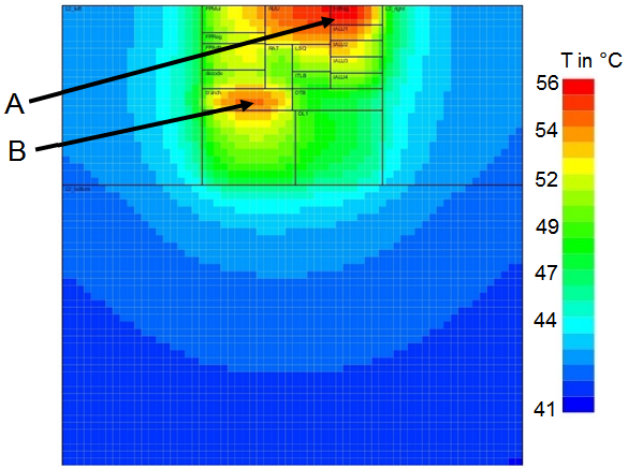

Fig. 4. Temperature distribution for single core. The hotspots are hotspot "A" in RUU/IntReg region (56 °C) and hotspot "B" in branch region (54 °C).

4.1 Single vs. multicore scenario

The temperature distribution for a single core is shown in Fig. 4. Temperatures range from 41 °C to 56 °C, where the lowest temperature is reached in the L2 cache. There are two hotspots: hotspot "A" in the RUU and IntReg region (56 °C) and hotspot "B" in the branch region (54 °C). IntReg region contains the integer register files and RUU is the register update unit that acts as a combination of register stages and reorder buffer. Branch is the branch prediction unit that predicts which way of a branch (e.g. if-then-else construct) is taken. For the single core scenario the hotspot with highest temperature is hotspot "A".

For the multicore system hotspot "A" is not as critical as hotspot "B": for single core hotspot "A" is located at the edge of the die. The die is sealed inside its package, so there is low airflow along the edge of the die. This results in limited heat removal via convection. Also the surface area for convective transfer is minimal since chip thickness is low. As a consequence temperature in hotspot "A" is higher than temperature in hotspot "B". In a multicore system hotspot "A" is usually

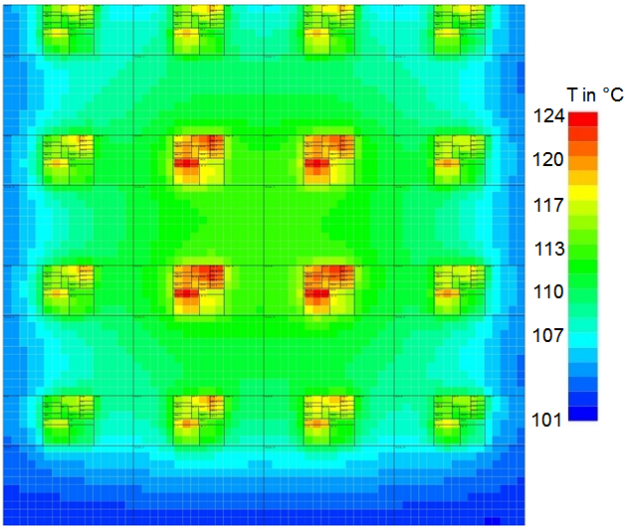

Fig. 5. Temperature distribution for a 4×4 multicore system. Hotspot "B" temperature of the middle cores increases by 124 % and hotspot "B" temperature of the corner cores increases by 112 % compared to single core.

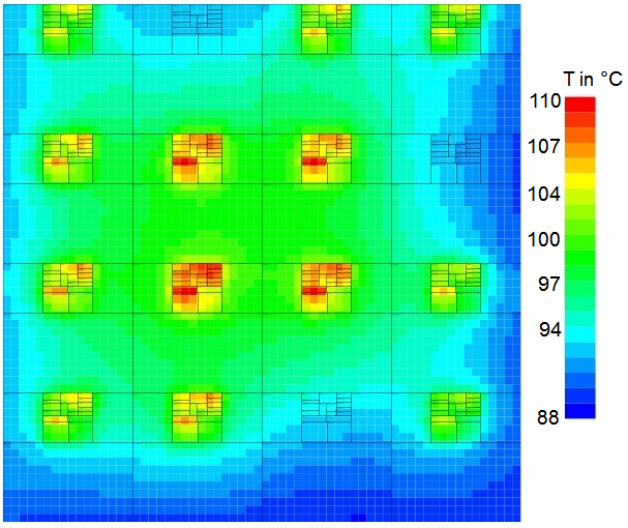

Fig. 6. For 80 % usage scenario, maximum and hotspot "B" temperature of active cores decrease by 11 %.

not located at the edge of the die (only for the four cores in the north edge of the die). In contrast to the fact that for a single core hotspot "A" has a higher temperature, hotspot "B" is the critical one in the multicore scenario (see Fig. 5). Temperature distribution for a 4×4 multicore system is shown in Fig. 5. Temperatures range from 101 °C to 124 °C. Since the temperatures of individual cores influence each other, the highest temperature is reached in the middle of the multicore system. The overall maximum temperature of the multicore scenario increases by 120 % (to 124 °C) compared to the single core scenario. The hotspot "B" temperature of the middle cores increases by 124 % (to 121 °C) and the hotspot "B" temperature of the corner cores increases by 112 % (to 114 °C).

4.2 Other usage scenario

In a realistic multicore scenario only about 80 % of the processors (13 cores in a 4×4 multicore system) are used. Figure 6 shows the temperature distribution for 4×4 multicore system with 13 used cores: the temperature distribution has changed depending on whether neighbouring cores are active or non-active. The overall temperature of active cores decrease by about 11 % (about 13 °C). The change between active and non-active mode of a processor depends on the application and/or the operating system. It is a typical procedure to use power gating in a multicore system to turn off not used cores (Lee et al., 2009). Since additional energy is needed to turn them on again, from an energy point of view it is best to not use cores that are turned off, when there is the possibility to use a not used – but still on – processor instead. This may lead to an unbalanced use of the cores, which is disadvantageous from a reliability, life-time and temperature point of view.

4.3 Other usage scenario and intelligent task allocation

During times of lower usage rates, it is possible to choose the placement of active and non-active cores. For a 50 % usage scenario (8 cores, 4×4 multicore system) the resulting temperature distribution for square configuration and checkerboard configuration is shown in Fig. 7: in square configuration the bottom 8 cores are active, the upper 8 cores are inactive: Maximum and hotspot "B" temperature of active cores decrease by about 29 % (about 34 °C) compared to full usage. In a lower usage scenario an intelligent task allocation and scheduling is possible, resulting in suitable combinations of application needs/characteristics and individual core health status to reduce overall temperature values. In our simulation every active core performs the same application. In this scenario intelligent task allocation is possible by placing active cores as far away from each other as possible to lower the overall temperature: In checkerboard configuration maximum and hotspot "B" temperature of active cores decrease by about 31 % (about 37 °C), the lowest temperature decrease by about 33 % (about 33 °C) compared to a full-usage scenario. Maximum and hotspot "B" temperature of active cores decrease by about 3 % (about 3 °C) compared to square configuration.

For a 25 % usage scenario (4 cores, 4×4 multicore system) the resulting temperature distribution for square configuration and no-middle configuration is shown in Fig. 8: in square configuration the left bottom 4 cores are active, all others are inactive: maximum and hotspot "B" temperature of active cores decrease by about 44 % (about 54 °C) compared to full usage. In no-middle configuration all middle

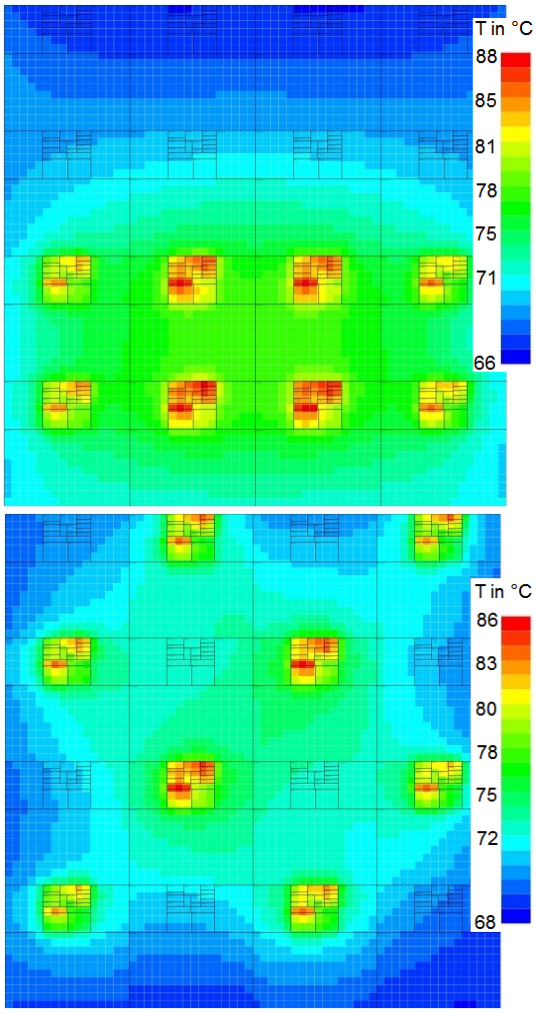

Fig. 7. 50 % usage scenario: in square configuration (left side) maximum and hotspot "B" temperature (active cores) decrease by 29 %. In checkerboard configuration (right side) maximum and hotspot "B" temperature (active cores) decrease by 31 % compared to full-usage scenario and by 3 % compared to square configuration.

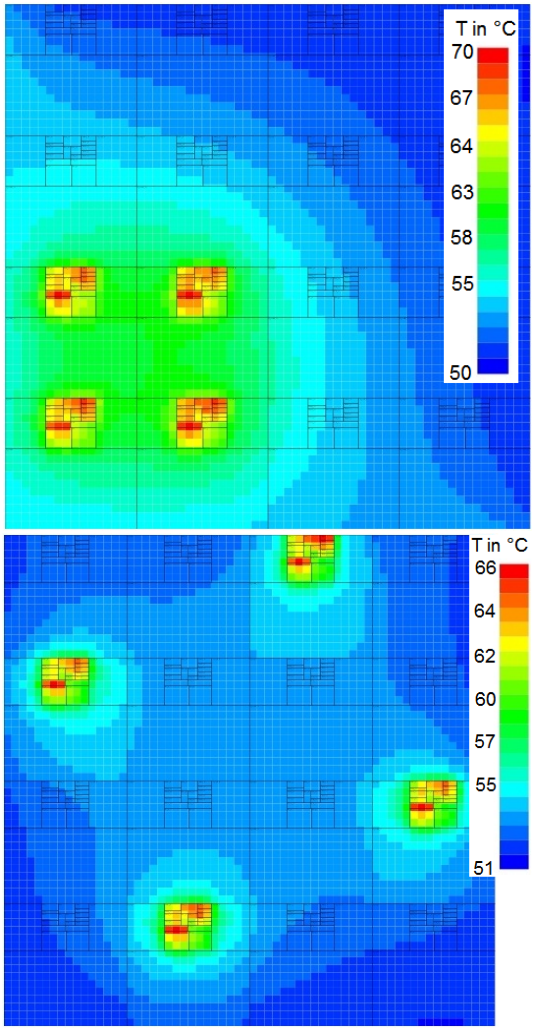

Fig. 8. 25 % usage scenario: in square configuration (left side) maximum and hotspot "B" temperature (active cores) decrease by 44 %. In no-middle configuration (right side) maximum and hotspot "B" temperature (active cores) decrease by 47 % compared to full-usage scenario and by 5 % compared to square configuration.

cores are inactive. Maximum and hotspot "B" temperature of active cores decrease by about 47 % (about 57 °C), the lowest temperature by about 50 % (about 50 °C) compared to full-usage scenario. Maximum and hotspot "B" temperature of active cores decrease by about 5 % (about 3 °C) compared to bottom square configuration.

4.4 Better cooling

When reducing the ambient temperature from 35 C to 30 C, the overall temperature values decrease by about 4 % (about 5 °C). When choosing a better packaging by doubling the thickness of the heat sink, the maximum and hotspot "B" (middle cores) temperature decrease by about 4 % (about 5 °C). Hotspot "B" (corner cores) and lowest temperature stay nearly the same. However, both techniques are cost intensive: as power increases there is no longer a linear relationship between cooling capabilities and cost of solution (Gunther et al., 2001).

4.5 Other power scenario

Temperature is directly depending on input power. By using lower input power for the individual cores, temperatures are decreased. A second power scenario with a time step that results in an average power per clock cycle of 32 W (corresponding to the start-up phase of the benchmark) is evaluated. Compared to the standard power scenario with average power per clock cycle of 39 W temperature distribution stays the same, but overall temperature values decrease by about 13 % (about 15 °C).

Table 1. Comparison of temperature limiting measures.

Measures	Temperature improvement	Implementation
other power scenario: 1) average power per cycle 32 W 2) voltage scaling ($V_{DD} = 1.3V$) ⇒average power per cycle 29 W	overall temperature decrease by about 13 % (15 °C) by about 18 % (22 °C)	e.g. improving task fragmentation & scheduling to decrease individual input powers implementing of (adaptive) voltage /frequency scaling
80 % usage	overall temperature decrease by about 11 % (13 °C)	may lead to unbalanced core use if always the same cores stay (non) active; performance loss; choose of non-active cores
50 % usage, intelligent task allocation & scheduling	overall temperature decrease by about 31 % (37 °C) (compared to full-usage) 3 % (3 °C) (compared to square conf.)	see implementation of 80 % usage; also: implementation: intelligent task allocation & scheduling, choose suitable application-core pairs

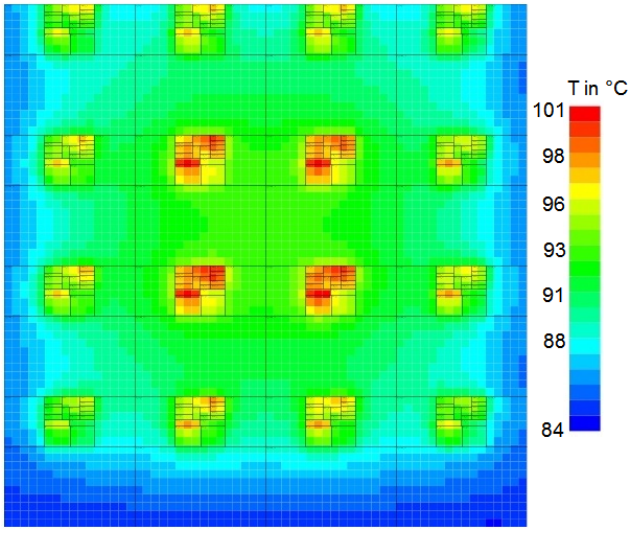

Fig. 9. For voltage scaling with reduced supply voltage of 1.3 V, maximum and hotspot "B" temperature (middle cores) decrease by 18 %.

There are various possibilities to change the power scenario e.g. using lower supply voltage: Fig. 9 shows the resulting temperature for reduced supply voltage of 1.3 V (instead of 1.5 V). This results in the use of a time step that results in a lowered average power per clock cycle of 29 W (instead of 39 W). The overall temperature distribution stays the same, the overall temperature values decrease by about 18 % (about 22 °C). Lowering supply voltage is realizable by e.g. using adaptive voltage (or frequency) scaling. In the approach presented in Wirnshofer (2011), the supply voltage is changed depending on the current system situation during runtime by detecting timing pre-errors. The supply voltage is regulated to a value where no pre-errors occur based on this information.

Improving the task execution can also realize lowered input powers per core: when parallel applications are executed on a multicore system, an improved task fragmentation and scheduling for an application lead to lower input powers for the individual cores. This is one of the basic principles of Invasive Computing (Teich et al., 2011).

5 Temperature limiting measures

Full usage of a multicore system is the worst case scenario. In a realistic usage scenario about 80 % of the cores are used, which makes it possible to e.g. choose active cores, choose suitable cores for an application to execute on and implement techniques that allow different treatment for cores in a multicore system like power gating, where not used cores are shutdown or adaptive voltage scaling. With regard to our simulation results and practicability the best temperature limiting measures are either changing the power scenario or changing the usage scenario combined with intelligent task allocation (see Table 1): with enforced lower usage scenarios, the temperature improvement is good. But performance may be lost, since not all cores are used. It may also lead to an unbalanced core use, if always the same cores stay active and inactive, respectively.

When adding intelligent task allocation to lower usage scenarios a further improvement of temperature is possible. Using another power scenario leads to good temperature improvement but some effort for implementation of e.g. intelligent task fragmentation and scheduling or (adaptive) voltage/frequency scaling is necessary. In case of using adaptive voltage/frequency scaling also supply voltage or frequency may be lowered. Since the temperature should be monitored and regulated during runtime, different techniques can and should be implemented in the same system. This makes it possible to choose the best temperature limiting measure in the current system state during runtime.

6 Summary and conclusions

Economically affordable processor packaging cannot cover the worst case hot spot scenario anymore. In this paper

temperature distributions of different scenarios in a multicore system are analysed. With regard to simulation results and practicability the best temperature limiting measures were investigated and compared to each other: The best solution is either intelligent core choice combined with lower usage-rates or lowering of the input power e.g. by implementing supply voltage or frequency scaling or using parallel programs with intelligent task fragmentation techniques. To perform this, the temperature should be monitored and regulated during runtime, so we propose an implementation of different concepts and choosing a temperature limiting measure during runtime for individual situations. To conquer worst case hot spot scenarios in multicore systems, a collaboration of intelligent software, operating system and hardware solutions is necessary. Future work will include the simulation of bigger multicore areas, different applications on the same invasive multicore system and the investigation of the closed-loop interaction of software, operating system and hardware for invasive architectures.

Acknowledgements. This work was supported by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Centre "Invasive Computing" (SFB/TR 89).

References

Boehm, R. F.: Conduction Heat Transfer, in: The CRC Handbook of Thermal Engineering, edited by: Keith, F., CRC Press LLC, 2000.

Brooks, D. et al.: Power, Thermal, and Reliability Modeling in Nanometer-Scale Microprocessors, In IEEE Micro, Vol. 27, No. 3, 49–62, 2007.

Chandra, R.: Transient Temperature Analysis, Robuspic Workshop on European Solid-State Device Research Conference (ESSDERC), 2006.

Gunther, S. H. et al.: Managing the Impact of Increasing Microprocessor Power Consumption, In Intel Technology Journal first quarter 2001, 2001.

Huang, W., et. al.: Compact Thermal Modeling for Temperature-Aware Design, in: Proceedings Design Automation Conference, 878–883, 2004.

Lee, J. et al.: Optimizing throughput of power- and thermal-constrained multicore processors using DVFS and per-core power-gating, in: ACM/IEEE Design Automation Conference, 47–50, 2009.

Liao, W., et al.: Temperature and supply Voltage aware performance and power modeling at microarchitecture level, in: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1042–1053, 2005.

Semenov, O et. al.: Impact of Self-Heating Effect on Long-Term Reliability and Performance Degradation in CMOS Circuits, in: IEEE Transactions on Device and Materials Reliability, Vol. 6, No. 1, 17–27, 2006.

Skadron, K., et al.: Temperature-Aware Microarchitecture: Modeling and Implementation, In ACM Transactions on Architecture and Code Optimization, Vol. 1, No. 1, 94–125, 2004.

Teich, J. et al.: Invasive Computing: An Overview, in: Multiprocessor System-on-Chip – Hardware Design and Tool Integration, edited by: Hübner, M. and Becker, J., 241–268, Springer, Berlin, Heidelberg, 2011.

Wirnshofer, M., et al.: A variation-aware adaptive voltage scaling technique based on in-situ delay monitoring, In IEEE International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS), 261–266, 2011.