View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Dagstuhl Research Online Publication Server

Monitoring and Control of Temperature in Networks-on-Chip

Tim Wegner, Claas Cornelius, Andreas Tockhorn, and Dirk Timmermann

Institute of Applied Microelectronics and Computer Engineering, University of Rostock
Richard-Wagner-Str. 31, 18119 Rostock-Warnemuende, Germany
tim.wegner@uni-rostock.de

Abstract

Increasing integration densities and the emergence of nanotechnology cause issues related to reliability and power consumption to become dominant factors for the design of modern multi-core systems. Since the arising problems are enforced by high circuit temperatures, monitoring and control of on-chip temperature profiles need to be considered during design phase as well as during system operation. Hence, in this paper different approaches for the realization and integration of a monitoring system for temperature in multi-core systems based on Networks-on-Chip (NoCs) in combination with Dynamic Frequency Scaling (DFS) are investigated. Results show that both combinations using event-driven and time-driven forwarding more than double overall execution time and considerably reduce throughput of application data. Regarding performance of notification and reaction to temperature development event-driven forwarding clearly outperforms time-driven forwarding.

Keywords and phrases Network-on-Chip, Reliability, Monitoring, Temperature, Control

Digital Object Identifier 10.4230/OASIcs.MEMICS.2010.124

1 Introduction

Aggressive downscaling of device sizes and ever increasing integration densities result in a fast growing number of processing and storage components per chip. This development gives rise to quickly growing systems with exceedingly high complexity, as it is well reflected in Systems-on-Chip (SoCs) combining multiple Intellectual Property (IP) cores. Against this background, NoCs provide an enabling solution to fulfill the communication requirements of such very-large-scale integrated systems [1]. However, this development causes issues related to reliability and robustness to become critical aspects for chip design. On the course of miniaturization, the transistor count per die increases, causing a generally higher probability of system failures on the one hand. On the other hand, probability for an individual transistor to fail is also raised, since the decreasing structural size of Integrated Circuits (ICs) leads to higher susceptibility to environmental influences and deterioration. Several physical mechanisms contributing to the these effects are known to be abetted by high temperatures. This leads to on-chip temperature distribution having considerable influence on various parameters of ICs like failure rate, lifetime, performance and power consumption. The correlation between temperature and deterioration is established by the Arrhenius model, describing the velocity of chemical reactions depending on temperature [10]. Two important mechanisms redounding to deterioration are Time Dependent Dielectric Breakdown (TDDB) and Electromigration (EM). TDDB describes the formation of charge traps in the gate oxide of a transistor and ultimately leads to gate oxide breakdown [9] rendering the transistor inoperative [3]. EM is defined as the transport of material caused by


© Tim Wegner, Claas Cornelius, Andreas Tockhorn, Dirk Timmermann;
licensed under Creative Commons License NC-ND
Sixth Doctoral Workshop on Math. and Eng. Methods in Computer Science (MEMICS'10)—Selected Papers.
Editors: L. Matyska, M. Kozubek, T. Vojnar, P. Zemčík, D. Antoš; pp. 124–131

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

ion movement in conductors and leads to the formation of locations of material loss (voids) and material accumulation (hillocks) [2]. Thus, current paths might be interrupted or short circuits might arise between adjacent wires.
The presented facts reason that adequate mechanisms for monitoring and control of on-chip temperature distribution are vitally important in order to mitigate the named effects and to delay failures caused by deterioration. Hence, in this paper methods to track temperature profiles of on-chip components and to react to temperature changes are examined with respect to the impact on performance of NoC-based multi-core systems, area costs and performance of temperature monitoring and control. More in detail, investigations are carried out for event-driven as well as time-driven forwarding of packets for temperature monitoring. Packets are sent to a Central Control Unit (CCU), which replies by giving instructions for DFS if necessary to lower the activity of the concerned IP core and thus relax its temperature.

2 Related Work

There has been a lot of work towards monitoring methods for NoC-based SoCs and management strategies for on-chip temperature, but little on the combination of the two delivering conclusions on the strengths and weaknesses of certain approaches. The concept of an event-based online monitoring service for NoCs is proposed in [4]. This service is based on reconfigurable event-based hardware probes attached to the NoC components (i.e. the routers) and offers runtime observability of the NoC behavior. In [5] the concept depicted above is examined on system level. Three different alternatives to integrate the monitoring service into a NoC are proposed and evaluated with respect to selected aspects. In [6] a NoC design flow, which takes monitoring into account at design time, is proposed. The paper focuses on the integration of monitoring into the overall NoC design process and makes proposals for the placement and the number of probes based on the underlying application. Moreover, design flow and system architecture for hierarchical power monitoring of on-chip networks are outlined in [7]. Thereto, a hierarchy of adaptive and scalable modules is used to handle various power monitoring services with different granularities. The hierarchy consists of multiple cell and cluster agents as well as a platform and an application agent. Similarly, in [8] a hierarchical agent-based concept is used to provide for reconfigurable NoCs with an increased fault tolerance on the architectural level. In case of a failure, communication and application execution are dynamically relocated based on given latency requirements.
In this paper the idea of a modular monitoring concept is adopted in order to monitor and control temperature in NoC-based SoCs. For this purpose, hardware probes, attached to all IP cores, forward the temperature to a CCU, which takes appropriate actions if necessary. Furthermore, values for area costs, system performance and performance of temperature monitoring and control are provided for both event-driven and time-driven temperature forwarding.

3 The Monitoring and Control System

In this section the monitoring system consisting of probes attached to every IP core and the CCU are introduced. The probes track the temperature of the associated IP core and the CCU gives instructions for DFS to the probes if necessary. Here, it is assumed that the temperature values are already available to the probes. Hence, this paper does not focus on the physical capture of the temperature value but on its further processing. The targeted NoC uses wormhole switching in combination with distributed XY-routing and

deploys a packet-based communication protocol, in which packets consist of a varying number of flow control digits (flits) representing the basic unit of flow control in the NoC. The target SoC is expected to be a general purpose system. This renders further assumptions on the incorporated applications dispensable. First, an event-driven and a time-driven solution for forwarding temperature values to the CCU are described. The targeted temperature range is from 20 to 127 °C. Furthermore, to ensure timely monitoring and control probe- and CCU-generated packets are prioritized over regular packets.

The event-driven probe design is based on [4]. The basic concept is that if the predefined conditions of an event are satisfied, further actions are triggered. The conditions here account for a maximum temperature or a large change within a given interval. The subsequent action is the forwarding of the received temperature value. In detail, the flow works as follows. The probe periodically reads temperature values and compares them to the value lastly reported to the CCU (T_{old}). If temperature forwarding is triggered, the temperature value T_{curr} is saved and a packet is generated containing T_{curr} as well as the address of the related IP core. This packet is then forwarded to the CCU. Figure 1 (a) depicts this flow exemplarily for a variation limit of $\Delta T \geq 10\,°C$, which can be freely chosen at design time in order to adapt the probe to different requirements and application settings. Relaxed conditions for ΔT will result in fewer monitoring packets less affecting regular traffic but leading to more intermittent temperature monitoring. If small changes of temperature shall be detected though, the conditions need to be more stringent leading to more packets potentially interfering with regular traffic. In general, a trade-off between quality and quantity (granularity of temperature forwarding versus number of generated packets) of event-driven temperature monitoring has to be made. Since packet generation for this approach depends on temperature gradients and therefore is non-deterministic, making statements about the optimal value for ΔT is nearly impossible. In contrast, the time-driven probe forwards an incoming temperature value periodically to the CCU based on a given period of time Δt. This is done independently from the current temperature, thus potentially causing redundant packets. The packet to be forwarded is identical to that from the event-driven approach. Figure 1 (b) illustrates the time-driven scheme of forwarding, in which Δt can be chosen at design time. This renders the time-driven approach deterministic as the amount of probe-generated traffic can be predicted in advance independently from temperature gradients. Therefore, Δt might be adopted to the prospective requirements.

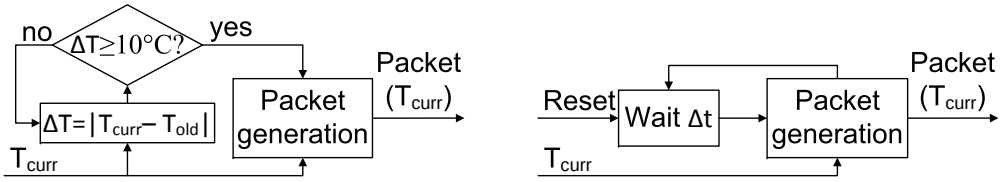

Figure 1 (a) Event-driven and (b) Time-driven temperature forwarding

All probe-generated packets are sent to the CCU, which is responsible for providing control instructions to the probes based on the reported temperatures. The frequency adjustment for an IP core is executed by the probe associated to this component once the probe has received instructions. Due to the fact that a considerable part of power consumed by ICs is dissipated as thermal energy, control instructions given by the CCU focus on methods to primarily reduce dynamic power consumption P_{dyn}. Based on the well-known equation $P_{dyn} = \alpha * \ C_L * \ V_{DD}^2 * \ f$, Dynamic Voltage and Frequency Scaling (DVFS) are the most

commonly used approaches. However, for the sake of simplicity, DVS is not implemented. Basically, the CCU waits until a packet arrives containing the temperature T_n of an IP core n. The desired operating frequency $f_{n,new}$ is then calculated based on the received temperature T_n and the current frequency $f_{n,curr}$. In case $f_{n,new}$ and $f_{n,curr}$ are identical, no action is taken. Otherwise, a packet containing DFS instructions is sent to the probe observing IP core n in order to reduce its activity and thus its temperature. Currently, the control mechanism is a simple reaction to temperature changes (DFS with five levels of frequency) to test the functionality of the probes. More advanced algorithms might be integrated prospectively considering the sophisticated correlations of temperature, reliability and further design metrics. The impacts on execution time of applications and operation conditions are not considered by the current simple DFS algorithm, which implies frequency reduction in case of raised temperatures as well as an incremental return to the maximum operating frequency in case of temperature normalization.
Since the probe is designed as an independent module, it has to be integrated accordingly into the NoC. Basically, three noteworthy possibilities exist. The probe's integration into an IP core promises to be the most straightforward and inexpensive solution as the probe uses the existing interface of the core for communication purposes (see Fig. 2 (a)). However, this precludes simultaneous packet transmission of probe and IP core. Furthermore, in case the IP core is unavailable (e.g. power down mode, failure), the probe becomes inaccessible and is no longer able to operate as it lacks of communication resources. Moreover, the integration of a probe into an IP core conflicts with the principle of strict modularity, which is one of the main intentions of NoCs. For the second approach the probe is placed outside the associated IP core (see Fig. 2 (b)). Here, both the probe and the IP core possess a dedicated communication interface, thus eliminating the effect of a probe being inoperable in case of core unavailability. Only the port of the router connecting the IP core to the NoC is shared among the probe and the core by using a Mux/Demux module. This module is responsible for correctly forwarding the traffic from and to the port of the router and always prioritizes probe-generated packets (current transmission of a packet containing multiple flits is finished first). Unfortunately, the Mux/Demux module and the additional interface induce extra area. As still only one router port is used, parallel communication of the IP core and the probe is not supported. The third alternative adds an extra port to the router in order to connect a probe to the NoC (see Fig. 2 (c)). Thereby, the probe is connected to the NoC completely independent from the IP core allowing full parallel communication, sustainment of modularity and operational readiness of the probe in case of IP core outage. The extra port and the raised complexity are supposed to induce the biggest area overhead though.

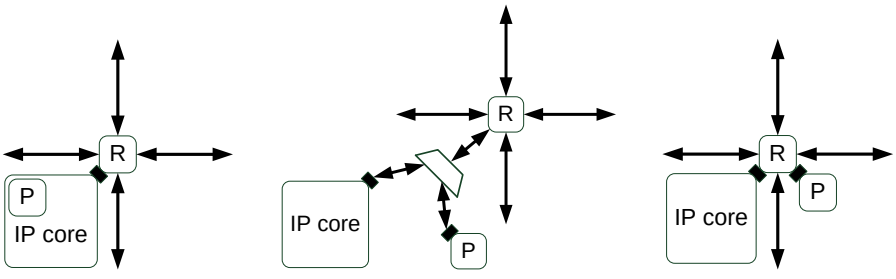

Figure 2 Integration of a probe (P) into the NoC: (a) Integration into the IP core, (b) Using the router port of the IP core, (c) Using an extra router port

4 Results and Discussion

All introduced proposals for probe design, the CCU and the integration into the NoC were synthesized with Xilinx Ise 10.1 for a Virtex5-FPGA to be comparable regarding area costs and frequency (see Tab. 1). Values for area and frequency of an unmodified NoC router serve as reference. Note that the required area is calculated as the number of pairs consisting of Look-Up Tables (LUT) and Flip-Flops (FF).

On the one hand, synthesis results for the temperature forwarding show that event-driven forwarding (66 LUT/FF pairs) occupies 17,5 % less area than the time-driven method (80 LUT/FF pairs). This is due to the fact that although more combinational logic is required by the event-driven probe, total area usage of the time-driven approach is dominated by the size of the counter that is necessary to trigger the periodic packet generation. To provide for a reasonable period, this results in a relatively large counter, which represents an area penalty compared to the event-driven scheme (e.g. $\Delta t = 2{,}5\,\text{ms}$ requires 18 bit wide counter based on a frequency of 100 MHz). Note that also existing trigger signals might be used for periodic packet generation. However, in this paper it is assumed that no such signals are available. On the other hand, reduced effort for combinational logic yields an operating frequency for the time-driven probe (338 MHz) that is 48,9 % faster with respect to the event-driven probe (227 MHz). From a cost-oriented point it can be concluded, that in case maximum frequency and determinism of packet-generation are the major design criteria, time-driven forwarding is the better choice. In case that minimal area and traffic overhead are required, event-driven forwarding is favored. Synthesis results for the CCU reveal that this module (507 LUT/FF pairs) requires only 28,6 % of the area needed for an router. Since an IP core is expected to require a multiple of this area, feasibility of positioning the CCU in a location regularly reserved for an IP core is affirmed. Admittedly, the CCU could also be implemented in software so that the kind of control could be adapted at runtime. However, the CCU is required to be implemented in hardware since the performance of communication between probes and CCU and its impact on system performance are of major interest in this paper. Due to its nature, the integration of a probe into an IP core causes no area overhead, whereas the maximum frequency of the network remains constant. Furthermore, the integration using the IP core's router port causes an area overhead of 7,34 % with respect to an unmodified router. Although the maximum frequency remains unchanged, the Mux/Demux module causes additional latency for packet transmission. The extra router port and the resulting raised complexity of the router not only cause the biggest area growth of 30,55 % but also lower the frequency to 112 MHz. Unfortunately, the most inexpensive method with the probe being integrated into the IP core cannot always be considered feasible, because IP cores often remain black boxes to the system designer when they are IP of third parties or fixed down to the layout. Regarding the two remaining possibilities, both methods do not necessitate IP core modification and guarantee probe operability in case the associated IP core becomes unavailable. However, the integration via an extra port causes unacceptable area overhead without delivering advantages (except parallel communication of probe and IP core) compared to the integration using the IP core's router port. Since the number of packets sent from and to the probe is comparatively small, the need for parallel communication is expected to be marginal. Therefore, the integration of a probe using the IP core's router port can be identified as the method delivering the best compromise between performance, overhead and feasibility.

Table 1 Synthesis results for monitoring, control as well as integration (unmodified NoC router: 1771 LUT/FF pairs, 122 MHz)

	Component			Integration method		
	Event-driven probe	Time-driven probe	Central Control Unit	Into IP core	Using IP core port	Extra port
Frequency [MHz]	227	338	165	122	119	112
Area [LUT/FF pairs]	66	80	507	1901	1896	2312

In order to investigate the impact of the examined methods on selected parameters of regular system operation, an 8×8 NoC was simulated both with and without enhancements for monitoring and control. Simulation was performed using both event-driven forwarding (with $\Delta T = 5\,°C$ and $10\,°C$) and time-driven forwarding (with $\Delta t = 0{,}1$ ms and 0,2 ms). The monitoring and control mechanisms were integrated using the approach proposed in Fig. 2 (b). During simulation 200000 regular packets with a maximum packet length of 5 flits were generated and sent to random destination addresses. Packet generation was uniformly distributed over all IP cores with an initial packet injection rate of 20 %. For simulation including monitoring and control the CCU replaced the most centric IP core at address 3,3 in the NoC. From Fig. 3 (a) it can be seen that mean packet delay of regular packets traversing the NoC is considerably decreased both for event-driven (up to 9,72 %) and time-driven (8,53 %) temperature forwarding combined with DFS. At first sight, this seems to be incorrect, since packets for monitoring and control additionally stress the NoC and therefore should have negative impact. Considering the overall execution time, which is drastically extended by at least 125,16 % for all tests, this phenomenon can be explained. In contrast to the reference, the applied DFS mechanism reduces the injection rate of the IP cores if necessary. Therefore, the overall number of packets (monitoring and control packets included) simultaneously crossing the NoC is reduced, yielding relaxed conditions for packet transmission. Thus, a lower utilization of the communication resources is traded off against an extension of overall execution time. This conforms to the principle of algorithms like DFS of maintaining operability at the expense of reduced performance. As expected, with a reduction of more than 55 %, integration of monitoring and control has a negative impact on the throughput of regular data, since fewer packets located in the NoC lead to a smaller number of flits that can be transmitted per clock cycle. Concluding, concerning effects on performance event-driven and time-driven forwarding do not differ notably from each other when combined with DFS. Furthermore, variation of ΔT (event-driven) and Δt (time-driven) has only minor influence on performance parameters. Concerning the results for notification delay (time for packet transmission from probes to CCU) and response delay (notification delay + time for packet transmission from CCU to probes) the event-driven approach outperforms the time-driven approach (see Fig. 3 (b)). In the former packets for notification arrive about 83 % faster than in the latter and response packets arrive about 77 % faster. This is due to the fact that for event-driven forwarding notification packets are only generated when temperature exceeds or goes below defined thresholds. This leads to a relatively even distribution of monitoring packets over time avoiding congestions around the CCU. In contrast, in the time-driven approach all probes simultaneously transmit a packet to the CCU after Δt has expired. As a consequence, bursty occurrence of monitoring

packets causes congestions within the area containing the CCU. To solve this issue the probes' counters for Δt might be reset at different points in time resulting in timely staggered generation of monitoring packets. Although delays for time-driven triggering might be significantly reduced hereby, the risk of transmitting identical temperature values repeatedly and causing unnecessary traffic remains. Therefore, the event-driven approach promises to be the more practicable method.

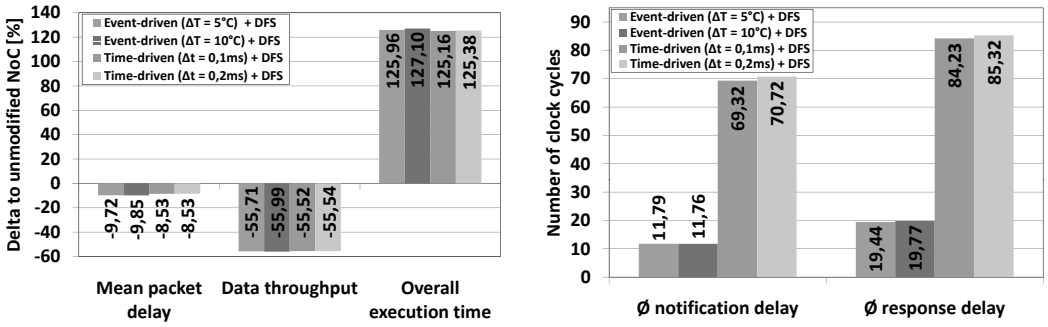

Figure 3 (a) Performance results for a NoC enhanced by monitoring and control in comparison with an unmodified NoC, (b) Delay results for notification (packets from probes to CCU) and response (packets from probes to CCU and back to probes)

5 Conclusion

In this paper the idea of a modular monitoring concept for NoCs is adopted for temperature monitoring in NoC-based SoCs and combined with DFS for control. For this purpose, probes inherit the function of monitoring temperature of the System-on-Chip, consisting of IP cores, and a CCU assumes the task of applying DFS to the IP cores. Regarding the integration of monitoring and control into the NoC we argued that using an IP core's router port poses the best trade-off between feasibility, performance and additional costs. Furthermore, an event-driven and a time-driven approach for forwarding temperature values from probes to CCU in combination with DFS were examined regarding their impact on performance of regular system operation and monitoring and control. Results show that both approaches similarly extend overall execution time by up to 127,1 % and reduce throughput of application data by up to 55,99 %. In return NoC utilization is reduced and mean packet delay is decreased by up to 9,85 %. Using the event-driven approach packet delay from probes to CCU is shortened by almost 83 % and response delay (packet from probe to CCU back to probe) is abbreviated by 77 % with respect to time-driven forwarding.

References

1 L. Benini and G. De Micheli, "Networks on Chips: A New SoC Paradigm", IEEE Computers, Jan. 2002, pp. 70-78.
2 J. Lienig, "Introduction to Electromigration-Aware Physical Design", in Proc. of ISPD 2006.
3 E. Vogel, et al., "Reliability of Ultra-Thin Silicon Dioxide Under Combined Substrate Hot Electron and Constant Voltage Tunnel Stress", in Trans. of Electron Devices, vol. 47, no. 6, 2000.
4 C. Ciordas, et al., "An Event-based Monitoring Service for Networks on Chip", in ACM TOADES 2005, vol. 10, no. 4, pp. 702-723.

5 C. Ciordas, et al., "NoC Monitoring: Impact on the Design Flow", in Proc. of IEEE ISCAS 2006.

6 C. Ciordas, et al., "A Monitoring-Aware Network-on-Chip Design Flow", J. Syst. Archit., vol. 54, issue 3-4 (March 2008), pp. 397-410.

7 L. Guang, et al., "Hierarchical Power Monitoring for On-chip Networks", in Proc. of PDP 2009.

8 P. Rantala, et al., "Novel Agent-Based Management for Fault-Tolerance in Network-on-Chip", in Proc. of DSD 2007.

9 J. Stathis, et al., "Reliability Limits for the Gate Insulator in CMOS Technology", IBM J. of Research and Development, 2002.

10 J. Srinivasan, et al., "RAMP: A Model for Reliability Aware Microprocessor Design", IBM Research Report, RC23048, 2003.