POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

An autotuning controller for CPU power/performance/thermal management

Laurea Magistrale in Automation and Control Engineering - Ingegneria dell'Automazione

Author: Riccardo De Rosi

Advisor: Prof. Alberto Leva

Co-advisor: Dr. Federico Terraneo

Academic year: 2021-2022

1. Introduction and contributions

The thesis proposes an autotuning controller for power/performance/thermal management in modern microprocessors — a matter that emerged in quite recent years but is gaining so much importance to rank among the enablers for high-performance computing solutions, and often to be vital for the safe operation of the processors themselves.

The world of processors has in fact entered the so-called "dark silicon era", in which the thermal power is so high that certain areas of the active silicon – i.e., the portion of the chip that actually performs operations – must remain unused, to prevent the processor from overheating. And to make the *scenario* even more complicated, the heat generated is subject to enormous, sudden, and hardly predictable changes. The resulting challenge is so tough that some researchers even cast doubt on a prosperous future – if not on survival – for the multicore era [2].

The thermal dynamics of active silicon is incredibly fast. Today it requires a control capable of reacting at a millisecond scale, and the situation will most likely worsen with the advent of three-dimensional architectures. For this reason, the use of fixed rate controls is no longer a viable option. Such fast, periodically computed controls require a non-negligible amount of computational power, and it is not clear where to best integrate them: if they were allocated in software, they would burden the operating system by stealing computational power. If they were implemented in hardware, they would require silicon area and power. In contrast, event-based controls have the characteristic of acting only when needed, mitigating the overall computation and power demand.

As proven in the previous work [6], event-based control is able to effectively manage temperature transients in a multicore system. The objective of this thesis is the development of a controller with a methodological approach to temperature management of individual cores in a multicore context, and more specifically, to endow such a controller with autotuning capabilities, to be invoked typically at the time of a system re-configuration, or if needed at each system startup; this is the first contribution.

The second contribution, obtained by an accu-

rate design and engineering, is an implementation of the said controller capable of running together with (a) a Modelica model of the heat dissipation system connected to the addressed CPU, and (b) a fine-grain chip simulator like 3D-ICE [9], This allows for a model-based design of cooling solutions comprehending the proposed autotuner, testing the said design from the viewpoints of both the thermal policies realised aboard the CPU and the system-level heat dissipation equipment.

2. Related work

The most effective way to counteract processor overheating is to act on the clock frequency by means of a mechanism called DVFS (Dynamic Voltage and frequency Scaling). Alternatives like task migration [3, 7, 8] can be considered, but with the fast dynamics seen nowadays, at least the co-presence of DVFS is inevitable [12]. This thesis builds on one side on the results presented in [6], where a fixed-parameters event-based controller is proposed, and in [5], where a mixed model- and relay-based autotuning technique is introduced to ensure model fidelity in the vicinity of the cutoff frequency.
However, a peculiarity of the addressed problem is that a core is always subject to disturbance from the neighbouring ones on the silicon die, which requires to bring into play relay identification techniques capable of rejecting such undesired *stimuli* — a matter discussed e.g. in [4] and in [1], on which the identification technique presented in the thesis is based.

3. The proposed autotuner

The presented autotuner applies to the control structure proposed in [6], which in turn relies on a PI loop per core (thanks to the very loose inter-core interactions observed at the time scale of interest) equipped with an override signal coming from the operating system governor, that requests an operating frequency based on the measured load and some configuration parameters. In a nutshell, as long as the core is cool enough the governor rules the frequency, while if the set temperature threshold is approached the thermal controller takes over (and reduces the frequency with respect to the governor request). The autotuner is made (for each core, given the adopted decentralised control approach) of the following components:

- the *low-level controller*, that is, the PI with override just mentioned;
- the *exciter*, devoted to managing the relay-based identification phase;
- the *analyser*, with the task of turning relay data into pointe of the frequency response of the dynamics seen by the controller to tune;
- the *high-level controller*, in charge of computing the control parameters and managing the entire autotuning procedure.

The high-level controller, that governs the operation of the entire system, operates according to a finite state machine that distinguishes the various phases of an autotuning operation and acts on the underlying low-level controller by means of its modulating and logic input (for example, to orderly replace the PI with the relay, the former is forced into tracking mode and the relay output becomes the track reference); a synthetic overview of the finite state machine just sketched is reported in Figure 1.

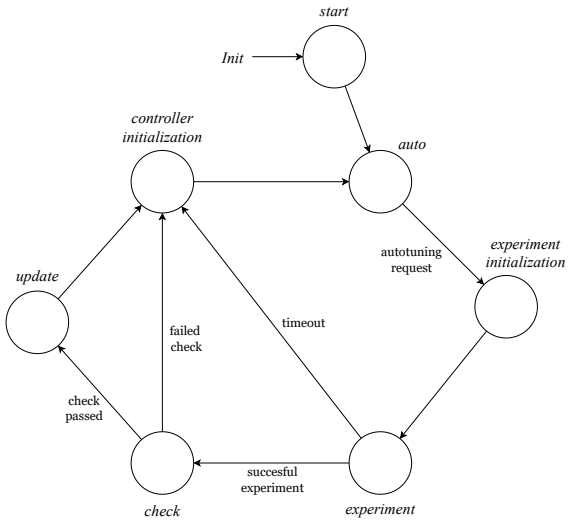

Figure 1: Finite state machine for the high-level controller.

Peculiar to the presented autotuner are three features. First, the exciter can make use of (i) a pure relay, (ii) a relay with a cascaded integrator and (iii) the technique suggested in [1]; this basically consists of augmenting the relay-process cascade with a high-pass filter, devoted to rejecting matched disturbances (as those due to power from adjacent cores on a measured temperature)

and with a low-pass filter devoted to recover the correct process information by compensating for the effect of the high-pass one.
Second, the analyser can be configured to employ the describing function approximation, a widely used technique in relay-based autotuning, or to employ a more complex method based on Fourier analysis, that is capable of yielding better precision at the cost of an increased computational demand.
Third, the tuning procedure can employ the standard IMC rule or the contextual one proposed in [5]. The contextual technique tunes a controller based on points of the process Nyquist curve, thus well complementing relay-based identification, and by using those points to parameterise a transfer function model of the process, allows for the use of model-based tuning formulæto the advantage of readability, especially concerning the way specifications are stipulated (in the addressed computer-centric domain, a settling time requirement is far better understood than e.g. a phase margin one). However, instead of first using the frequency response point to parametrise a model and then use this model to tune the controller, the equations corresponding to the two tasks just mentioned are solved jointly (or as the name says, "contextually"). This implies that the model used for the tuning is by construction "exact at the cutoff frequency", as this is taken (we omit details here) to match one of the found points, and has an important consequence: one can use the closed-loop model formed with the parametrised model and the tuned controller to predict the behaviour of the controlled variable reliably enough in the face of any input that enters the control system in a position such that its influence on the said variable depends only on the loop transfer function, provided that the tuning is made in such a way that at low frequency (with respect to the cutoff) the loop frequency response magnitude is "very large" — as incidentally any controller with integral action inherently guarantees.

4. Implementation

The proposed autotuner was implemented both as an all-Modelica application and a C++ library, capable of running together with 3D-ICE, and also suitable for future porting on the physical devices to address. The reason is that a completely Modelica realisation allows one to exploit the powerful solution capabilities – variable-step integration to mention just one – offered by the language and its tools, while a C++ realisation is necessary for the deployment-related reasons just mentioned.
The package structure of the created Modelica library is shown in Figure 2, while Figure 3 shows an example of the contained control schemes.

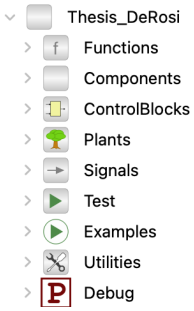

Figure 2: Overview of the created Modelica library.

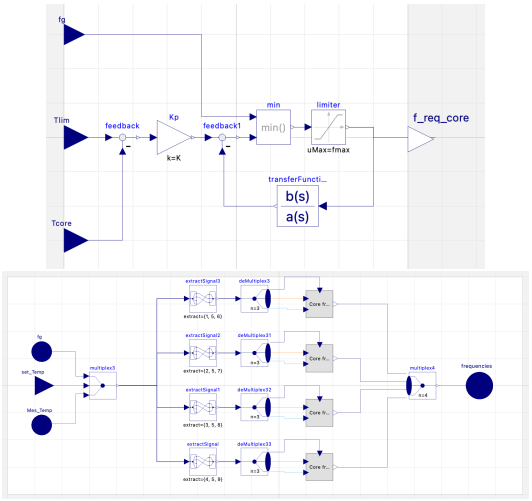

Figure 3: block diagrams of the override PI controller implemented in Modelica (top) and its quadrupled version (bottom) for a quad-core.

The C++ implementation relies on an articulated class hierarchy, represented in UML form in Figure 4 and impossible to describe here in detail.
As for 3D-ICE integration, the client-server setup made available by that tool was exploited, modifying the client in such a way as to allow 3D-ICE to accept the external control actions computed by the autotuning controller.

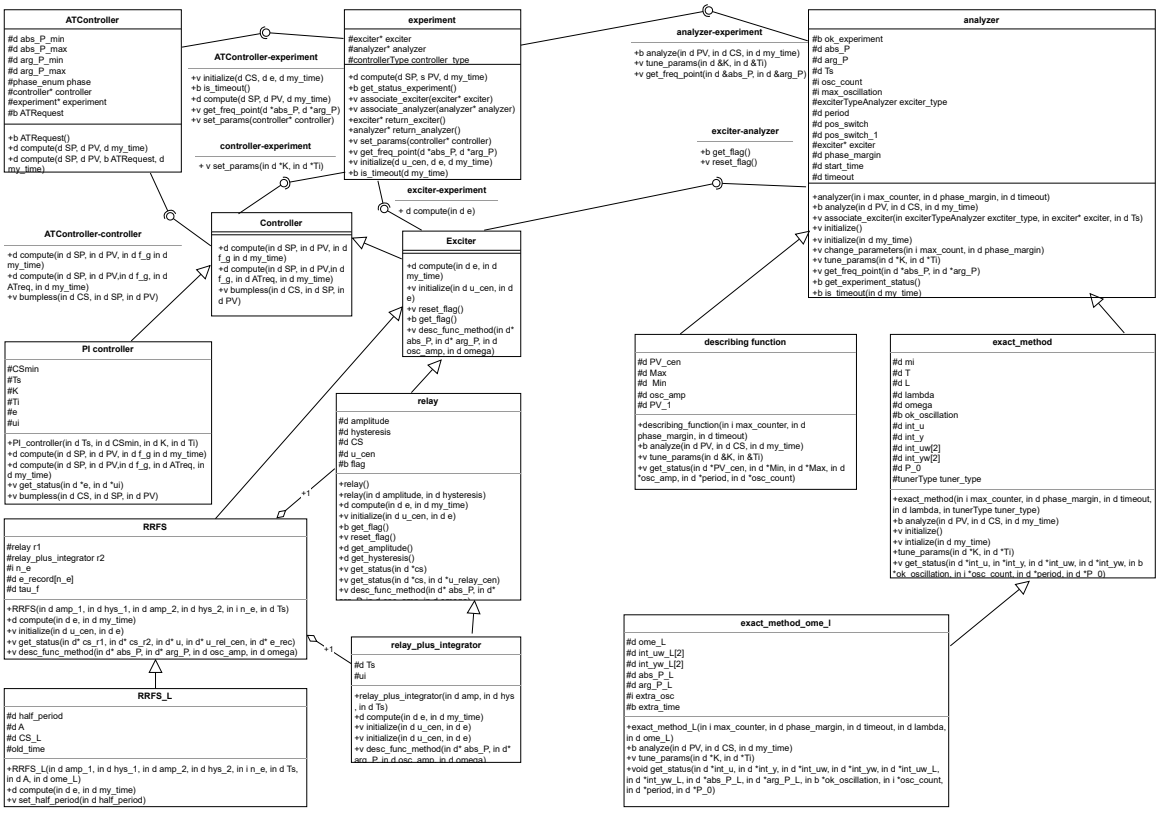

Figure 4: UML representation of the proposed autotuner as realised in C++.

5. Testing

The testing activity constitutes a relevant part of the thesis work. At present it was done only in simulation, since porting to physical devices is envisaged as a future activity, and was aimed at the following objectives.

1. Experiment – though in a somehow still idealised setting – with the different options implemented for excitation, analysis and tuning, in a view to critically compare them.
2. Assess the correct operation of the Modelica and the C++ implementations of the autotuner, as well as their mutual consistency.

The first set of experiments was carried out in Modelica with a 3-capacities model of a quad-core processor; tests were carried out with various combinations of the proposed stimulation, analysis and tuning policies, as well as in the presence of constant and time-varying thermal disturbances from the cores adjacent to the controlled one. Figure 5 shows an example of the obtained transients, in detail reporting the behaviour of the controlled temperature and the control signal during and after a tuning operation; after the tuning, evidenced by the relay excitation, a step set point modification and a step disturbance were applied (the latter by acting in open loop on the powers dissipated by the adjacent cores) to show the behaviour of the obtained controller.

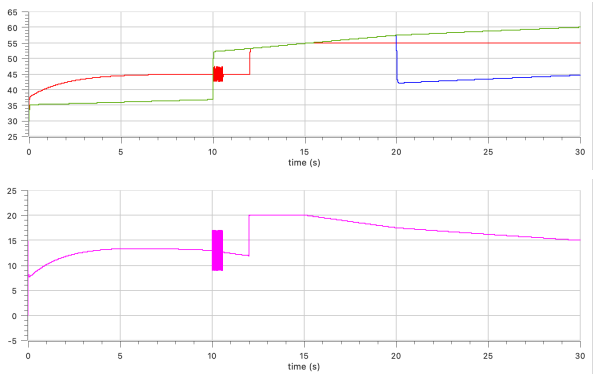

Figure 5: Autotuning operation simulated in Modelica: controlled variable (top, red, limit set to 55°C), temperatures of the other cores (top, green and blue) and control signal (bottom).

Based on the obtained results we conclude that the describing function and the Fourier analysis are both viable approaches, that the former is less accurate as expected, but also that the latter provides its better accuracy at the cost of an increased number of required points — i.e., of a smaller sampling time. We also conclude that none of the proposed tuning policies significantly outperforms the others, hence CLA can be used if maximum simplicity is a must, while the contextual IMC is preferable in a view of the future extensions envisaged thanks to its capabilities of providing reliable forecasts of the controlled variable.

The second set of experiments involves co-simulation with 3D-ICE, exploiting its so-called "pluggable heat sink" capability [11] to employ, together with a fine-grained chip description, also accurate heat sink models. The purpose of these experiments is twofold: (i) verify that the behaviour of the C++ autotuner implementation reasonably matches that of the Modelica one – of course from a qualitative standpoint, as the processor model used in 3D-ICE is different from the 3-capacities Modelica one, and far more accurate as it also includes a detailed representation of the heat sink; (ii) check the co-simulation capabilities of the proposed C++ realisation. An example of the obtained results, with a Cuplex waterblock as sink, is shown in Figure 6. The shown time scale is short as the simulation is more computing-intensive than the all-Modelica case; note however that the time scale of tuning and responses do correspond.

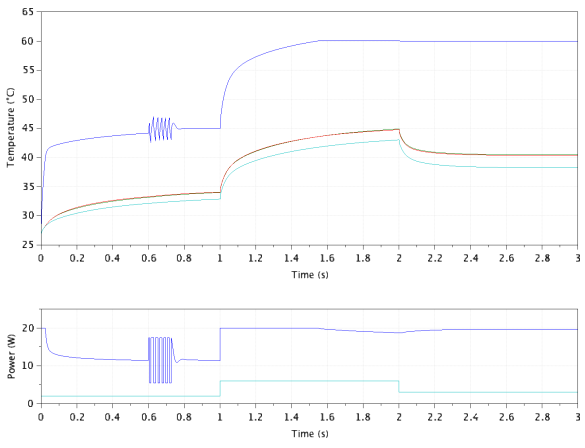

Figure 6: Autotuning operation in co-simulation with 3D-ICE (tuning, set point change, disturbance step).

In addition, to show the added value of co-simulating with 3D-ICE, in Figure **??** we show a thermal map of the chip (4 cores in a square 2×2 layout) at the end of the tuning operation (the simulation produces a sequence of such maps to help evaluate the spatial behaviour of temperature over time).

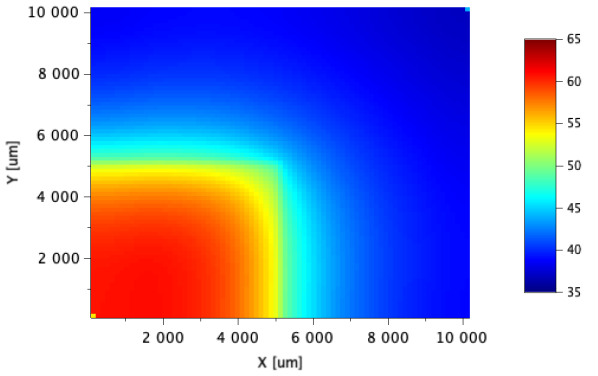

Figure 7: Thermal map obtained from an autotuning co-simulation with 3D-ICE.

The activity carried out indicates first that the C++ implementation of the autotuner operates correctly, and as a consequence that it can be used in conjunction with 3D-ICE (or in principle with any tool capable of co-simulation) for the intended integrated studies aimed at a joint verification of heat dissipation equipment and on-chip thermal policies, as envisaged e.g. in [10].

6. Conclusions and future work

We presented an autotuning controller conceived for joint power/performance/thermal control in modern microprocessors. The necessity of such controls is nowadays testified by the increasing importance of the "dark silicon" problem, and as quenching thermal stress inherently comes at the cost of reducing performance, the said controls must be capable of limiting that detriment to the minimum required. In turn, given the various installation settings and ambient conditions that a microprocessor can experience, the need for so effective control performances calls for adaptation capabilities.
An autotuning controller is therefore highly desired, but needs to be (i) very simple so that computer (not control) personnel can operate it, (ii) not too invasive to not upset the operation of the overall system where it resides, (iii) robust in the face of disturbances from neighbouring parts

of that system such as adjacent cores, and (iv) computationally light so as to make it possible to invoke it e.g. at every system startup.
The proposed solution, thanks to the combination of purpose-specific stimulation, analysis and tuning policies, fulfills all the needs just set forth. After a theoretical motivation, it was realised both as a Modelica library, for system-level testing, and as a C++ application, for experimentation with fine-grain chip simulators like 3D-ICE and to run in conjunction with other on-chip policies such as load-based frequency/voltage governors, so as to be assessed as ready for porting on a real device.
The autotuner realisation on a physical processor is thus the first activity planned for the future, together with refinements of the tuning procedure and further assessment in simulation, for example within studies aimed at a joint design, based on virtual prototyping, of heat dissipation equipment and on-chip power/performance/thermal policies.

References

[1] T. da Silva, Moisés and Péricles R. Barros. A robust relay feedback structure for processes under disturbances: Analysis and applications. *Journal of Control, Automation and Electrical Systems*, 30:850–863, August 2019.

[2] Hadi Esmaeilzadeh, Emily Blem, Renée St. Amant, Karthikeyan Sankaralingam, and Doug Burger. Power challenges may end the multicore era. *Communication of the ACM*, 56(2):93–102, February 2013.

[3] N. Gomathi and K. Nagalakshmi. Criticality-cognizant energy-efficient task scheduling on heterogeneous multicore processor. *International Journal of Engineering Trends and Technology*, 70(4):203–214, April 2022.

[4] Jietae Lee, Jin-Su Kim, Jeonguk Byeon, and Whan Sung. Relay feedback identification for processes under drift and noisy environments. *AIChE Journal*, 57(7):1809–1816, July 2011.

[5] Alberto Leva, Sara Negro, and Alessandro Vittorio Papadopoulos. Pi/pid autotuning with contextual model parametrisation. *Journal of Process Control*, 20(4):452–463, 2010.

[6] Alberto Leva, Federico Terraneo, Irene Giacomello, and William Fornaciari. Event-based power/performance-aware thermal management for high-density microprocessors. *IEEE Transactions on control systems technology*, 26(2):535–550, March 2018.

[7] Mohammed Sultan Mohammed, Ali A. M. Al-Kubati, Norlina Paraman, Ab Al-Hadi Ab Rahman, and M. N. Marsono. Dtapo: Dynamic thermal-aware performance optimization for dark silicon many-core systems. *Electronics*, 9(11), 2020.

[8] Sanjay Moulik. Reset: A real-time scheduler for energy and temperature aware heterogeneous multi-core systems. *Integration*, 77:59–69, 2021.

[9] Arvind Sridhar, Alessandro Vincenzi, Martino Ruggiero, Thomas Brunschwiler, and David Atienza. 3d-ice: Fast compact transient thermal modeling for 3d ics with inter-tier liquid cooling. In *2010 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 463–470, 2010.

[10] Federico Terraneo, Alberto Leva, William Fornaciari, and David Atienza. Modeling and simulation challenges and solutions in cooling systems for nanoscale integrated circuits[feature]. *IEEE Circuits and Systems Magazine*, 23(1):36–56, 2023.

[11] Federico Terraneo, Alberto Leva, William Fornaciari, Marina Zapater, and David Atienza. 3d-ice 3.0: efficient nonlinear mpsoc thermal simulation with pluggable heat sink models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(4):1062–1075, 2021.

[12] Hai Wang, Jian Ma, Sheldon X.-D. Tan, Chi Zhang, He Tang, Keheng Huang, and Zhenghong Zhang. Hierarchical dynamic thermal management method for high-performance many-core microprocessors. *ACM Trans. Des. Autom. Electron. Syst.*, 22(1), aug 2016.