Using Event-B and Modelica to evaluate thermal management strategies in many core systems

Colin Snook and Tom Kazmierski
Electronics and Computer Science
University of Southampton, UK
{cfs,tjk}@ecs.soton.ac.uk

***Abstract*—Dynamic thermal management is an increasingly critical and complex part of the run-time management of many-core systems. Methods of controlling temperature include thread migration, dynamic voltage and frequency scaling and power gating using various strategies and combinations of each. In the PRiME project we are developing run-time management systems to sustain the scaling of many-core systems. As part of this development we are investigating the relative benefits of different thermal management strategies by co-simulating a Modellica model of the characteristics of a many-core device with a discrete Event-B model of the run-time manager. The results enable us to efficiently design more elaborate experiments on real hardware platforms in order to validate the run time management.**

I. Introduction

Many-core processors require *Dynamic Thermal Management* (DTM) via *Dynamic Frequency and Voltage Scaling* (DVFS) and *Thread Migration* (TM) to control the temperature of individual cores. As the number of cores increases, dynamic thermal management becomes increasing complex due to physical topological effects such as inter-core heating. Within the PRiME project [11], we are developing a *Run-Time Manager* (RTM) [4] for many-core systems using a formal modelling language. In order to illustrate and understand the issues involved in DTM and to provide preliminary validation of DTM strategies, we model the physical thermal properties of a multi-core processor and co-simulate it with a formal discrete model of an RTM. We use the open source *Modelica Modelling Language* (Modelica) language, supported by the *Dynamic Modeling Laboratory* (Dymola) modelling tool, for modelling the physical thermal properties. Dymola supports export of independent *Functional Mock-up Unit* (FMU) simulation units using the *Functional Mock-up Interface* (FMI) standard format.

We use the *Event-B modelling noation* (Event-B) [1] discrete, state-transition modelling notation, supported by the *Rodin Platform* (Rodin) tool, to model the RTM. The Event-B model is animated using the *Pro-B model checker and animator for Event-B* (Pro-B) plug-in for Rodin to obtain the outputs to the physical environment, interleaved with running the FMU to obtain new input values from the physical environment.

The motivation for using Event-B is that it provides strong mathematical verification of abstract properties using proof and refinement and supports generic abstract specification with specialisation for e.g. different platforms. For example, other work within the PRiME project [6] has focused on automatically generating RTM code for different hardware platforms from a generic Event-B model. The motivation for using Modelica is that it is suitable for modelling various physical domains such as heat and electrical characteristics in a user-friendly way. In previous work [7] we have developed a framework based on Rodin, Event-B and Modelica, for validating formal models of RTM systems.

The contribution of this paper is to introduce a modelling method and illustrate how it is used to explore issues involved in RTM of hardware platforms and specifically DTM. The paper does not aim to present extensive results about the best DTM strategies. This will be the subject of future papers.

The rest of the paper is structured as follows. Section II gives some background on the methods and tools that we use and introduces possible thermal management strategies. Section III describes the Modelica and Event-B models. Section IV describes the results obtained from running the simulations with different thermal management strategies. Finally, we summarise and conclude in Section V.

II. Background

a) Event-B: Event-B [1] is a formal method for system development. The main features of Event-B include the use of *refinement* to introduce system details gradually into the formal model. An Event-B model contains two parts: *contexts* and *machines*. Contexts contain *carrier sets*, *constants*, and *axioms* describing the static parts of a model. Machines contain *variables*, *invariants* constraining the variables, and *events*. An event comprises a guard describing the conditions under which it is enabled and an action describing how the variables are modified when the event is executed. A machine in Event-B corresponds to a transition system where *variables* represent the state and *events* specify the transitions. More information about Event-B can be found in [9]. Event-B is supported by Rodin [2], an extensible toolkit which includes facilities for modelling, verifying the consistency of models using theorem proving. A plug-in model checker, Pro-B [10] is available for validating models with simulation-based approaches. Another plug-in that we utilise *iUML-B modelling notation* (iUML-B) [13], [14] provides a diagrammatic front-end modelling interface supporting notations such as state-machines

b) Co-Simulation: The Rodin tools and plug-ins are aimed at modelling discrete state-changing events; they are

not so good at validating continuous behaviour. Despite this we often need to model the requirements for a system that periodically controls some continuous dynamic behaviour. In order to validate such models a *MultiSim* plug-in [12] was developed by Savicks et al. The plugin allows an Event-B model and a continuous model to be simulated synchronously using the Functional Mock-up Interface (FMI 1.0) [8]. Typically the Event-B part will model a cyclic control system that monitors process variables from the continuous model and calculates a controlled output variable. The Event-B model is simulated programmatically using ProB and the continuous model is a FMU which has been exported from a continuous domain modelling tool such as Dymola [5]. The plug-in controls the coordination and communication between the co-operating simulations.

c) Thermal Management Strategies: Various methods of thermal management of multi-core processors have been proposed in the literature. We draw on those described and assessed by Chaparro et al. [3]. The proposed methods fall into the categories, i) stop-go power-gating, ii) DVFS management, iii) thread migration. We use stop-go power-gating in our RTM as a backup response to limit the temperature of a particular core when more pro-active management has not succeeded. While this is a realistic requirement, we should bear in mind that this management method has an impact on performance. We do not use DVFS as a method for thermal management since we use it to represent the different workload of a thread. Hence a high workload thread will be simulated by running the core at a high voltage and frequency setting so that it heats at a high rate. We use thread migration as our main mechanism for thermal management and assess the following strategies.

- static - threads are not migrated (used for comparison),
- rotation - threads are rotated in a fixed sequential pattern around the cores,
- temperature - the thread on the hottest core is moved to the coolest core, the next hottest to the next coolest, and so on,
- power1 - the thread that is consuming the highest power is moved to the core that was running the thread that consumed the least power and so on,
- power2 - the thread that is consuming the highest power is moved to the coolest core, the thread consuming the least power to the hottest core and so on.

III. Models

A. Modelling the thermal characteristics of a processor

In order to validate the specification of a suitable RTM we use the Modelica continuous domain modelling tool Dymola to create a model of the thermal characteristics of a quad core processor.

Fig.1a shows the Dymola model of the thermal characteristics of a single core. Instantaneous power is calculated as $P = aFV^2 + bV$, where F is the frequency and V is voltage. The bV term is a static leakage current which is always present unless the device is completely switched off. The aFV^2 term is the dynamic switching power which may be clock gated as part of the thermal management strategy. Instantaneous power then generates heat energy which accumulates in a heat capacitance, dissipates via external cooling to a fixed ambient temperature and flows to other cores. A temperature monitor is included in each core and instantaneous power is also monitored. Four instances of the single core model are then used to create a quad core processor, Fig.1b, with heat flow between each pair of cores depending on topology.

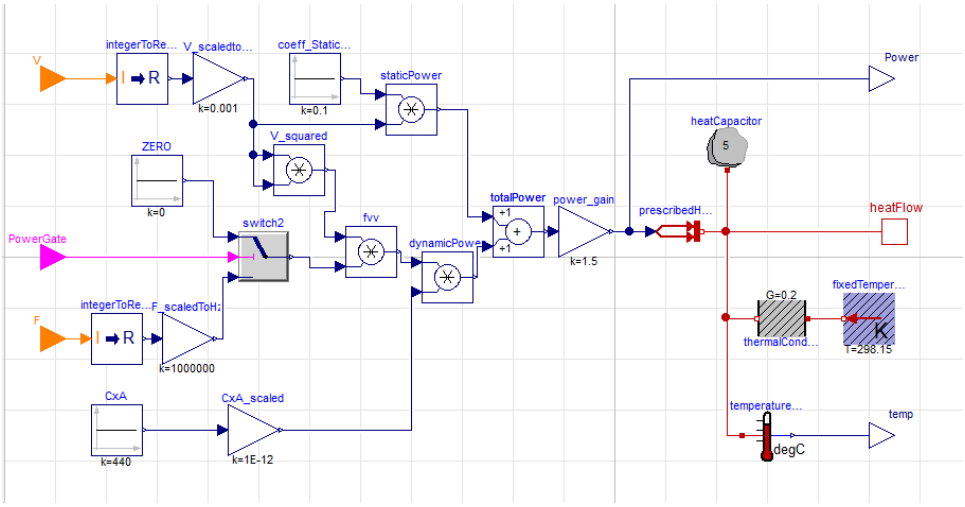
Once the environment model of thermal characteristics was tested within Dymola it was exported as an FMU which allows it to be run as a simulation outside of the Dymola tool. We then imported the FMU into the Rodin co-simulation environment, linked its I/O with our RTM Event-B model of the system and co-simulated the combined models to compare different

B. Modelling the RTM

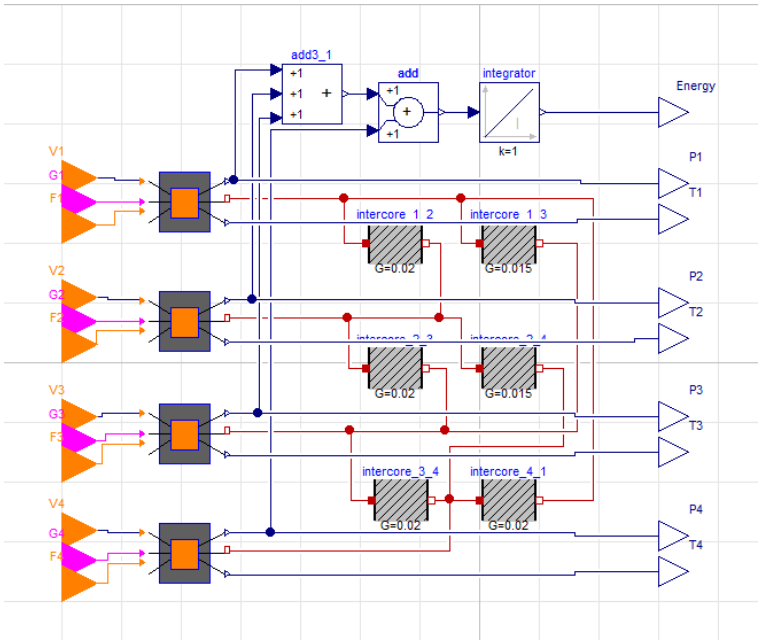
The Event-B model of the RTM thermal management functions consists of a cycle of events as depicted in the state-machine diagram of Fig.2. This diagram is drawn using the iUML-B tool [14] and is used to automatically generate parts of the Event-B model. The Event-B model at this level abstracts away from many details. For example, thread allocation is represented as a variable relation between sets and the impact of thread migration is not accounted for. Such details could be introduced in further refinements of the model if desired.

After reading the temperatures and power consumption of the cores (*ReadTemps*), the RTM checks whether any cores have reached their over-temperature limit and, if so, sets them as power-gated (*StopGo*). The RTM then migrates threads between cores according to the selected thermal management strategy (*Migrate*) before calculating the voltage and frequency setting for each core depending on the workload parameter associated with its allocated thread (*CalcVF*). The DVFS and power gating settings are output to the Modelica FMU (*SetControls*) and then the FMU is run to simulate execution for the scheduling interval (*RunThreads*).

The *StopGo* transition uses a lamda function, $\lambda c \cdot c \in CORE | bool(temp(c) > OT(c))$, where OT is the temperature limit for each core, to calculate a new value for the variable *gated* which is a function from the set of cores to boolean values. The *migrate* transition calculates a new value for the variable *alloc* which is a function from the set of cores to the set of threads. For the rotation strategy the new value for *alloc* is calculated using the forward composition $Rotate; alloc$ where $Rotate$ is a fixed bijection from the set of cores to the next core in the rotation. For the temperature based strategy, the new value of $alloc$ is given by $temp\_order; Invert; temp\_order^{-1}; alloc$ where $Invert$ is a fixed bijection that reverses the order of the indices $1..N$ and $temp\_order$ is a local variable function over cores, satisfying $\forall c1, c2 \cdot\ temp\_order(c1) > temp\_order(c2) \Rightarrow temp(c1) \geq temp(c2)$. The calculation for the first power based strategy is similar to the temperature based strategy except that $power$ replaces $temp$ in the definition of $order$.



(a) Single core



(b) Quad core

Fig. 1: Dymola model of thermal characteristics of a CPU

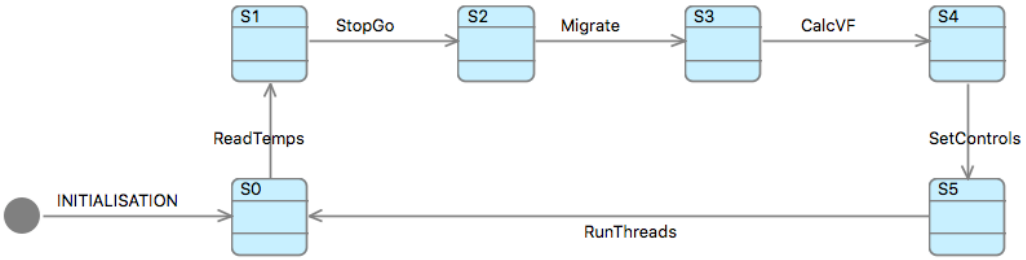
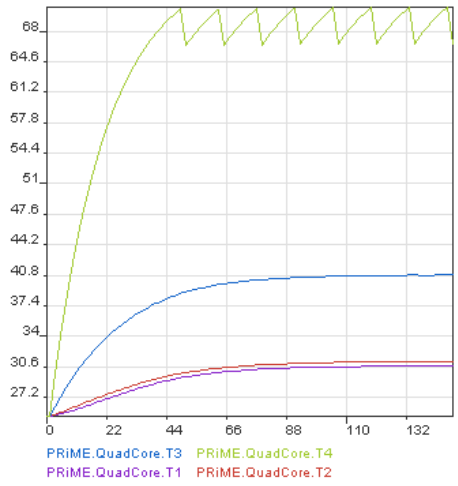
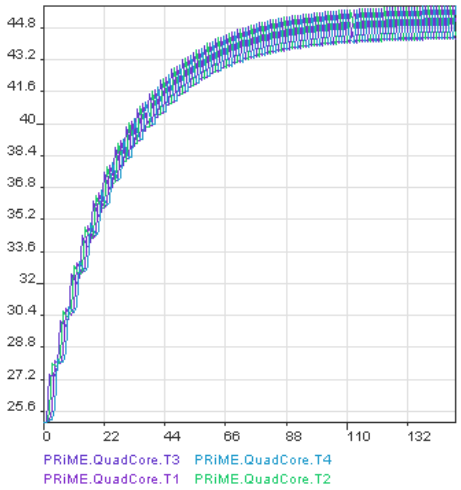


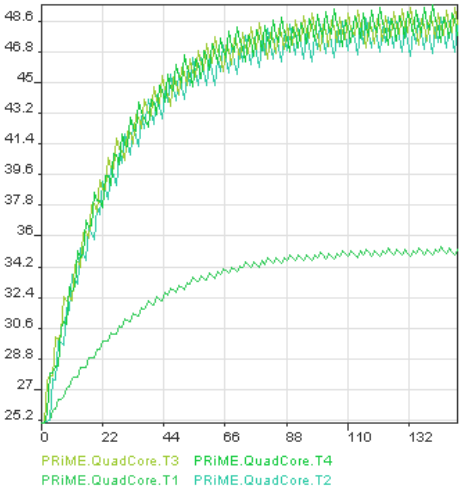
Fig. 2: Cyclic behaviour of RTM Event-B Model



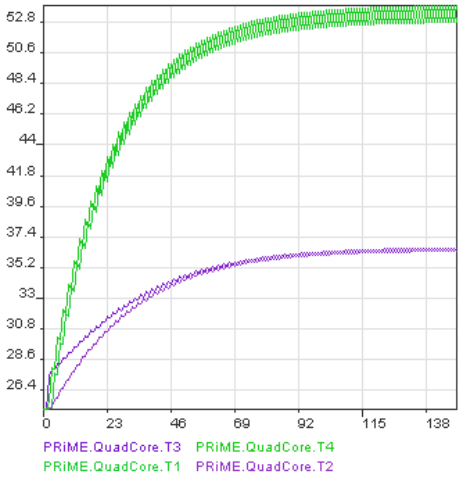
(a) Static



(b) Rotation



(c) Temperature



(d) Power

Fig. 3: Simulation results - core temperatures for different thermal management strategies

IV. Results

To test the method we set up an initial test case consisting of 4 threads of different workloads which result in the four cores being run at different DVFS settings. We chose representative mV/MHz VF points as follows: 800,200; 1000,500; 2000,1000; 3000,2000.

Fig.3a shows the control case where no thread migration is attempted and the four core temperatures rise corresponding to the different DVFS settings with the highest worked core needing to be power gated to stay below the temperature limit of 70°C. Notice how the exponential nature of cooling/heating means that the core is still being run for a high proportion of the time. The temperatures reflect the v^2 term in the model with significant temperature saving by moving from 3V to 2V and less from 2V to 1V.

The rotation strategy, Fig.3b, gives a good equalisation of temperatures between the cores for our simple test case where the workload of a particular thread is constant. The cores stabilise at 45°C.

The temperature-based strategy, Fig.3c, is less effective than the simple rotation strategy. This is because the ordered swapping strategy results in a resonance that under utilises one of the cores and hence the remaining 3 cores run hotter at 48°C than they did with simple rotation. The heat capacitance of a core results in a lag of its temperature which means that the thread on the hottest core is not the highest workload. The strategy settles into a repeating sequence of thread swaps

that, due to the heat capacitance, never allocates the highest workload thread to core 1 even though it is the coolest core.

The first power-based strategy, Fig.3d, fares even worse with two cores being under utilised and two cores running at $53°C$. This is because the strategy takes no account of temperature and reflects the fact that constantly running middling workloads uses less energy than oscillating between extreme high and low workloads. In fact, we made a mistake; this was not the intended strategy. The strategy should have been to order the cores by temperature and then assign the least power-hungry thread to the hottest core and so on. After correcting the power-based strategy it gives similar results to the rotation strategy with the cores stabilising at just under $45°C$.

Our test example is unrealistically static in that the thread workloads do not change over time. However, it is a valid example because such situations could occur for some period of time during the running of an application. The advantage of using a modelling approach to explore DTM strategies is that we have tighter control over setting up different scenarios in order to explore the intricacies of phenomena associated with using different strategies for a particular workload/thread group pattern. The knowledge we gain from modelling helps us understand and interpret empirical experiments when we run our RTM implementation with sample applications on real hardware platforms.

V. Conclusion

We have demonstrated the use of a continuous domain model of the thermal characteristics of a multi-core CPU which we co-simulate with an Event-B model of an RTM to provide an efficient tool with which to explore the comparative benefits of different thermal management strategies. We have illustrated how the results may give insight into potentially unexpected phenomena. The advantages of running simulations of mixed-domain models are that such phenomena can be detected and explored more easily and reliably. Tests are quick to set up and run, allowing different scenarios to be explored efficiently. Running real applications on real hardware may obscure such effects due to the difficulty of setting up and controlling the test case.

On the other hand, we need to verify that our model simulations are realistic by comparing them with empirical measurements. To do this we plan to replicate our tests on real hardware to improve confidence in the models. In future work we plan to expand the models with more cores and to make the test scenarios more sophisticated by, for example, allowing power coefficients to be varied to represent different execution characteristics of workloads. We need to incorporate into our assessment the performance that is actually achieved by a strategy so that the effects of stop-go power gating do not result in falsely positive impressions. Performance considerations also facilitate exploration of DVFS strategies which exploit abundant resources in order to operate at lower frequencies. We plan to explore the different thermal management strategies more in order to gain a better understanding of the characteristics of thread groups that might be used to determine the best DTM strategy to use in a particular case. As we have demonstrated, the selection of strategies can be counter-intuitive in some scenarios.

Acknowledgement

This work is funded by the EPSRC PRiME Project, www.prime-project.org.

Data Availability

All models and relevant data can be downloaded from http://eprints.soton.ac.uk/398615/.

References

[1] J-R. Abrial. *Modeling in Event-B: System and Software Engineering*. Cambridge University Press, 2010.

[2] J-R. Abrial, M. Butler, S. Hallerstede, T.S. Hoang, F. Mehta, and L. Voisin. Rodin: An open toolset for modelling and reasoning in Event-B. *Software Tools for Technology Transfer*, 12(6):447–466, November 2010.

[3] P. Chaparro, J. Gonzles, G. Magklis, Q. Cai, and A. Gonzlez. Understanding the thermal implications of multi-core architectures. *IEEE Transactions on Parallel and Distributed Systems*, 18(8):1055–1065, Aug 2007.

[4] Anup Das, Bashir Al-Hashimi, and Geoff Merrett. Adaptive and hierarchical run-time manager for energy-aware thermal management of embedded systems. *ACM Transactions on Embedded Computing Systems*, pages 1–25, September 2015.

[5] Dassault Systemes. Catia Systems Engineering Dymola. http://www.3ds.com/products-services/catia/products/dymola. (accessed Jan, 2016).

[6] Asieh Salehi Fathabadi, Luis Alfonso Maeda-Nunez, Michael Butler, Bashir Al-Hashimi, and Geoff Merrett. Towards automatic code generation of run-time power management for embedded systems using formal methods. In *9th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC-15)*, September 2015.

[7] Asieh Salehi Fathabadi, Colin Snook, and Michael Butler. Applying an integrated modelling process to run-time management of many-core systems. In *11th International Conference on Integrated Formal Methods (iFM)*, September 2014.

[8] FMI Steering Committee. Functional Mock-up Interface. https://www.fmi-standard.org. (accessed Jan, 2016).

[9] T.S. Hoang. An introduction to the Event-B modelling method. In *Industrial Deployment of System Engineering Methods*, pages 211–236. Springer-Verlag, 2013.

[10] M. Leuschel and M. Butler. ProB: An automated analysis toolset for the B method. *Software Tools for Technology Transfer (STTT)*, 10(2):185–203, 2008.

[11] PRiME project partners. PRiME: Power-efficient, Reliable, Many-core Embedded systems. http://www.prime-project.org/, May 2016.

[12] V. Savicks, M. Butler, and J. Colley. Co-simulating Event-B and continuous models via FMI. In *Proceedings of the 2014 Summer Simulation Multiconference*, SummerSim '14, pages 37:1–37:8, San Diego, CA, USA, 2014. Society for Computer Simulation International.

[13] C. Snook. Modelling control process and control mode with synchronising orthogonal statemachines. In *B2011, Limerick*, 2011.

[14] C. Snook. iUML-B statemachines. In *Proceedings of the Rodin Workshop 2014*, Toulouse, France, 2014. http://eprints.soton.ac.uk/365301/.