Icy-Core Framework for Simulating Thermal Effects of Task Migration Algorithms on Multi- and Many-Core Architectures

Molka Becher, François Pacull
CEA, LETI, DACLE, LIALP
F-38054 Grenoble, France
{molka.becher, francois.pacull}@cea.fr

Saddek Bensalem
VERIMAG Lab, UMR 5104, UJF
F-38041 Grenoble, France
saddek.bensalem@imag.fr

***Abstract*—The design of complex many-core architectures represents a milestone for the next generation of mobile applications. But concentrating computing elements in a small area will raise the temperature of the chip very high. Consequently, the silicon is subject to overheating that leads to aging and damaging effects. Thermal-aware task migration algorithms could be a solution to this problem. In order to evaluate these algorithms, a test environment for analyzing dynamic thermal management solutions is required. In this paper, we present Icy-Core as a framework for simulating thermal effects of task migration algorithms on multi- and many-core platforms.**

Keywords*—*Many-Core Architectures; Thermal Management; Task Migration; Simulation.

I. INTRODUCTION

Currently, the trend for embedded system industry is to increase the number of computing elements in their platforms. This allows to design more powerful and flexible systems with a reduction of power consumption. However, this requires additional mechanisms to master power. One of the possible interesting aspects to study is the dynamic reconfiguration of the *application mapping* during its execution. By dynamic mapping of an application, we denote the reconfiguration of application task assignment to the platform cores at run-time, namely migration of tasks from one core to another.

In a first approach, we can consider two main reasons to trigger such a reconfiguration of the application:

Firstly, when we consider complex embedded systems (e.g., a smart-phone) which can run different type of applications or services, we need to dynamically manage the resources shared between the different programs according to unforeseen events. For instance, receiving a call while playing a game or displaying a video.

Secondly, due to the increase of the number of cores in a very reduced surface, the efficient management of thermal dissipation becomes more complex. As thermal stress can reduce the lifetime of the circuit or lead to its damage, dynamic thermal management of these platforms is necessary.

In general, to avoid this damage we use Dynamic Voltage and Frequency Scaling (DVFS), which acts on the frequency and voltage of cores in order to reduce the power consumption, and by consequence the temperature. This is part of the solution, however, this degrades performance in terms of execution speed since it makes some cores running slower. Dynamic reconfiguration, i.e., moving a task from one core to another, targets to keep the same level of performance. In fact, task migrations may come with overhead costs due to data transfer or frequent migrations. Therefore, there is a compromise to find between reducing migration costs and maximizing performance.

In this paper, we address only the thermal management of multi-core platforms. We present our framework, called Icy-Core, which has been designed to evaluate task migration algorithms on multi- and many-core platforms. The purpose of this paper is to discuss the framework capabilities. Thus, for the case study part, we use a naive task migration algorithm only to illustrate the framework features.

The paper is structured as follows. Section II is dedicated to related work. Section III presents the goal of Icy-Core framework, its overall architecture and its operating mode. In section IV, we present our case study through a simple example of task migration and the simulation time of Icy-Core. Section V introduces the advantages and limitations of Icy-Core. Finally, we conclude the paper in Section VI.

II. RELATED WORK

In the literature, there is a variety of simulation frameworks. Simulation features and environment structure are different from one framework to another depending on simulation objectives. Heterogeneous Architectures and Networks-on-chip Design and Simulation framework (HANDS) [1] [2], Structural Simulation Toolkit (SST) [3], Multicore Power, Area, and Timing framework (McPAT) [4], Polaris [5] roadmapping toolchain and Wattch [6] framework are used at early design stages while Bartolini et al. virtual platform [7] is designed for evaluation of control strategies at run-time.

HANDS is a modular tool to simulate performance, power, temperature and reliability metrics for exploring network-on-chip interconnections. McPAT framework combines power, area and timing modeling to performance simulation to help architects evaluate the future of many-core architectures. Polaris, a system-level framework, focuses on the

 ISBN: 978-1-61208-319-3

choice of network-on-chip interconnection, which preserves performance and physical constraints. Wattch and SST are architecture-level frameworks. The former is designed to evaluate architecture power consumption and optimize its power dissipation while the latter simulates power, area and temperature for exploring network-on-chip design. Bartolini et al. virtual platform evaluates the control strategies of many-core systems by simulation of power, temperature and reliability management. There is also a commercial tool called AceThermalModeler [8], which is paired to Aceplorer [8], a power simulator, with the intention of getting a thermal simulation.

Compared to these frameworks, Icy-Core has a different objective which is to simulate thermal effects of task migration algorithms. Thus, for the thermal simulators as stand-alone tools, we considered HotSpot [9] and 3D Interlayer Cooling Emulator (3D-ICE). HotSpot is a popular simulator that generates an equivalent circuit of a targeted 2D or 3D integrated circuit based on the thermal resistances and capacitances. As for 3D-ICE, it was developed on the same compact thermal model of HotSpot simulator. So, it uses the same format of inputs / outputs and the same model for thermal problem construction. 3D-ICE offers features for accessing the thermal profile details during the simulation. Moreover, it is easily encapsulable. This is the main reason of our choice.

III. Icy-Core Framework

As presented in the section before, several simulation frameworks exist for exploring many-core and network-on-chip design. However, none provides an integrated tool able to evaluate task migration algorithms according to a thermal constraint. We propose here to simulate application power consumption and targeted platform *thermal profile*. The thermal profile of a platform is the set of temperatures at a given time of the smaller units –*thermal cells*– in the chip depending on the granularity of the discretization specified. When the thermal profile is depicted by a graph, it's called *heat map*. Having an idea of the platform thermal profile helps us to make decisions of task migration and to evaluate later the decisions made. This is the idea of Icy-Core framework.

A. Icy-Core Goal

The purpose of Icy-Core is to assess task migration algorithms by drawing up the thermal profile of an application mapping upon a targeted platform. As the application mapping gives the location of tasks on cores, we can compute core temperatures and detect overheating that could damage cores. Then, we can apply on the initial mapping the required changes to avoid these undesired effects.

The main contribution of this work is the design of a modular framework which allows us to monitor and compare thermal behavior of task migration algorithms.

B. Icy-Core Architecture

Fig. 2 illustrates the overall architecture of our proposed tool. Icy-Core is composed of five modules. There is two existing components that we modified: the *Platform Simulator* and the *Thermal Simulator*. The *Task Migration Algorithm* is the module we want to test. We specifically developed the *Power Profile Generator* and the *Heat Map Visualization* modules. The role of each module is detailed here after.

1) Platform Simulator: For our study, we have chosen as target platform the ST-microelectronics Heterogeneous lOw poweR Many-core (STHORM) described in Fig. 1. A platform simulator provides performance estimation about the cores and the network interconnections.

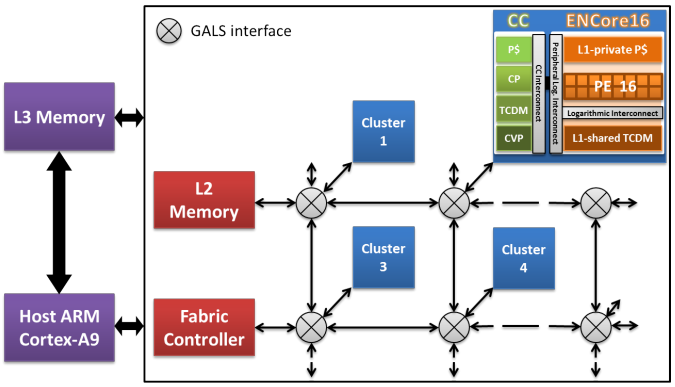

Fig. 1. STHORM Architecture

STHORM, formerly called P2012 [10], is a many-core accelerator. The platform is composed of a *fabric controller* that controls the system and four clusters connected by an *Asynchronous Network-on-Chip (ANoC)*. The routers of this *NoC* provide a *Globally Asynchronous Locally Synchronous (GALS)* scheme. There are up to sixteen *Processing Elements (PE 16)* per cluster. Each cluster is composed of a *Cluster Controller (CC)* and a *multi-Core computing ENgine (ENCore)*. The *ENCore* contains the cores (PE 16), a *L1-private Program cache (P$)* and a local *Tightly-Coupled Data Memory (TCDM)* shared by all the cluster processors. The *CC* consists of a *Cluster Processor (CP)*, a *Program cache (P$)*, a *local Tightly-Coupled Data Memory (TCDM)* and a *Clock Variability and Power (CVP)* module. The latter is in charge of dynamic frequency scaling.

The interest of this platform is firstly the number of cores (up to 64) and secondly its hierarchical and non-uniform architecture. STHORM platform offers different simulation environments at different accuracy levels. From the fastest to the slowest simulator we have:

- The Gepop-Posix environment for X86 functional simulation that provides fast results but less accuracy;
- The Gepop-ISS environment for first-level time approximation that is an Instruction-Set-Simulator on STxP70 code;
- The Gepop-ARM-ISS environment that is an Instruction-Set-Simulator on ARM processors.

In our case, we use the Gepop-ISS environment, which provides the core power consumption according to the power model [11] of the platform.

 ISBN: 978-1-61208-319-3

2) Thermal Simulator: We have chosen 3D-ICE [12] [13], developed by EPFL, as our Thermal Simulator. Generally, a thermal simulator is responsible for computing the thermal profile of the application mapping on the targeted platform.

3D-ICE is a simulation platform able to analyze the temperature trace in three dimensional integrated circuits. This simulator is fast enough to allow quick feedback to the user. This is important for the global reactivity of our tool.

In our case, we use the software thermal library of 3D-ICE and we encapsulate it in such a way we can call it remotely to obtain the current heat map of the platform.

In addition, the *spatial thermal distribution* is modeled in 3D-ICE by a 3D resistance and capacitance network. Spatial thermal distribution means that heat moves in three directions depending on distribution of the platform materials, cooling mechanism (i.e., heat sink) and power consumption of the integrated circuit elements.

3) Power Profile Generator: A *power profile* corresponds to the power dissipation values of each functional block of a chip. The power profile of processing elements is needed by the Thermal Simulator to compute the thermal profile of the platform.

For the Icy-Core initialization phase (see Section III-C1), we obtain the power dissipation values of each core from the platform performance traces. These power dissipation values corresponds to the *core power profile*. The Platform Simulator takes too much time to run all the application and to compute performance traces. In addition, it does not allow us to modify the mapping of the application during the execution. Yet, to iterate the dynamic reconfiguration process (see Section III-C2), we need to modify the *task mapping*, i.e., which task runs on which core, during the execution of the application according to the thermal stress. Here comes the role of the Power Profile Generator module which computes the core power profile according to the *task power profile* and the task mapping. The task power profile corresponds to the power consumption values of each task of the application while the task mapping is the output of the Task Migration Algorithm. We compute the task power profile according to the information extracted from the Platform Simulator and the initial task mapping.

To sum up, the Power Profile Generator aggregates the task power profile and the task mapping to supply the Thermal Simulator with the core power profile. Moreover, the Power Profile Generator stores the different task mappings to cores for traceability.

4) Task Migration Algorithm: This module is a placeholder for the Task Migration Algorithm we want to benchmark. It takes as input the outputs of the Thermal Simulator and computes accordingly which tasks need to be migrated from one core to another. Thus, it takes the mapping decision for the dynamic reconfiguration of the application on the targeted platform given the current context. The task migration algorithm may follow different strategies from very simple to sophisticated ones.

In the literature, dynamic thermal management has been addressed through different strategies: combination of techniques to stop rising the chip temperature [14], thermal predictive scheme [15], distributed thermal balancing [16] with agent-based thermal management [17]. These environments are applied to simplified architectural models either because the number of cores is limited or because the architecture itself is very regular. They seem to be very far from the architecture of current multi-core trends. However, we will test these algorithms on our environment to understand their strengths and weaknesses and we will use them as baseline for our own algorithms.

5) Heat Map Visualization: The Thermal Simulator produces the simulation outputs as a sequence of numerical values that are not easily readable by human. To get more visibility on the simulation outputs, we developed the Heat Map Visualization module. It is used to plot heat map graphs corresponding to the distribution of the temperatures in the chip at a given time.

C. Icy-Core Operating Mode

As depicted in Fig. 2, our framework Icy-Core is decomposed into three parts: *Initialization*, *Main loop* and *Visualization* which use the five modules described above. The *Initialization* part corresponds to the Platform Simulator module. The *Main Loop* connects three modules: Power Profile Generator, Thermal Simulator and Task Migration Algorithm. The *Visualization* part corresponds to the Heat Map Visualization module.

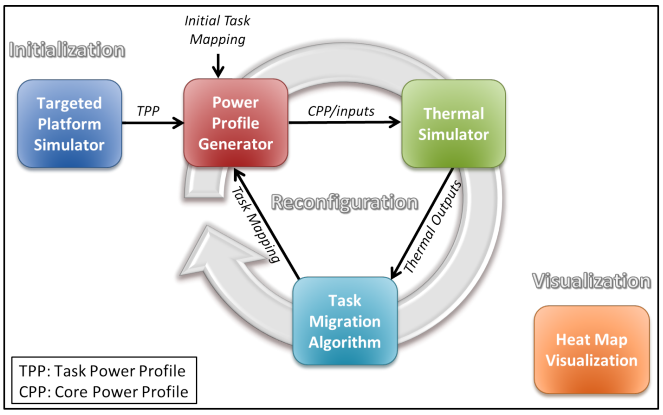

Fig. 2. Icy-Core Overall Architecture

1) Initialization: This phase purpose is to compute the application power consumption through the Platform Simulator. Thus, we run the application on the simulator to obtain performance estimation traces. From these traces, we obtain the average power consumption for each core during the execution of the application. In the implementation of the application, we specify which task runs on which core to have an idea of the initial task mapping. Knowing the initial task mapping and the execution time of each task, we compute the average power consumption of each task called the *task power profile*. This is the bootstrapping phase run only once at the beginning.

 ISBN: 978-1-61208-319-3

2) Main Loop: Dynamic Reconfiguration: This loop is composed of three sub-steps:

- Computation of the power profile of each core: It is the corresponding power consumption of a core deduced from the task power profile and the the task mapping. If the reconfiguration process is executed for the first time, the initial task mapping is taken into account. Otherwise, the new task mapping generated by the Task Migration Algorithm is used.
- Thermal simulation based on 3D-ICE simulator: It takes the power profile of cores and delivers the platform thermal profile.
- Execution of the Task Migration Algorithm: It detects from the *hot spots* (cores with thermal issues) which cores need to be managed. The objective of the algorithm is to find a solution of task migration such that the thermal issues decrease or disappear. The output is a new task mapping of the application on STHORM.

This loop runs in a regular time interval which is a parameter to define at the beginning of the simulation.

3) Visualization: At each iteration, the user may have a visual feedback of what is currently happening on the platform. The visualization phase is a separated module that can be called at each iteration to plot a heat map graph from the thermal outputs given by 3D-ICE. It represents the temperature of each thermal cell of the chip described by the Cartesian coordinates.

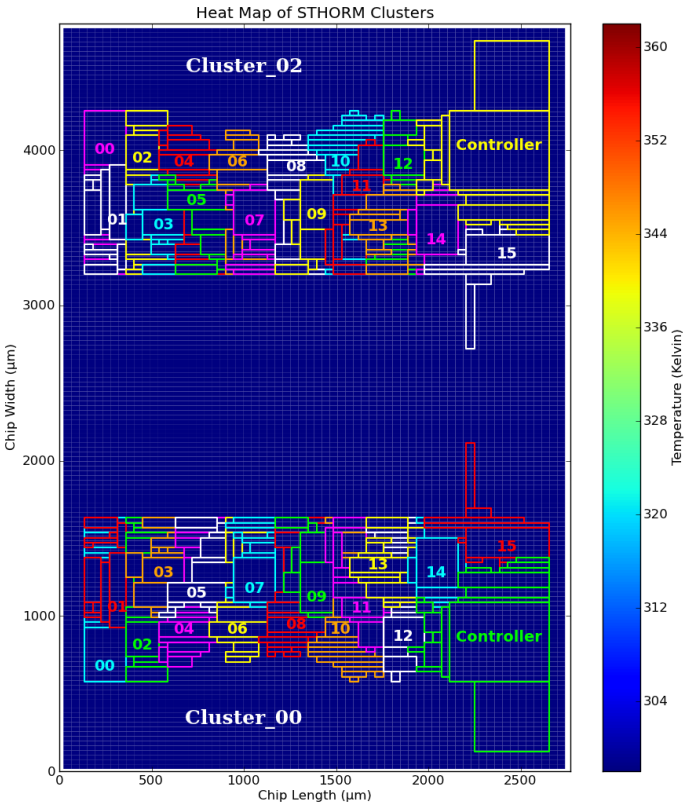

Fig. 3. Thermal Visualization of Two Clusters of STHORM When Idle

IV. Case Study

In order to illustrate how our system works, we consider to benchmark a naive task migration algorithm in the context of STHORM platform.

To simplify the case study illustration, we use only two clusters of STHORM: $Cluster\_00$ and $Cluster\_02$. Each cluster is composed of 16 cores and a cluster controller. Fig. 3 presents the thermal visualization of the two clusters when no task is running. The rectangles with same border color represent a functional block of STHORM. The initial temperature of the chip is 298K (24.85 °C).

By examining Fig. 3 more carefully, we can see that a core is decomposed into several sub-parts that are located at different places of a cluster. In our case, we manage only the core temperatures and we do not take account of the memories and the interconnections.

A. Transient Temperature Simulations

The task migration algorithm we have used as a toy example for the sake of simplicity is: At each iteration the hottest core above a given threshold is considered. We evaluate the tasks that are assigned to the core in order to determine the *greediest task*. The greediest task is the task that consumes more power than other tasks assigned to the core. Then, we migrate the greediest task to the coolest core of the platform.

As this algorithm sorts the core temperatures from the highest temperature to the lowest one, it has a linearithmic time complexity of $O(n\ log(n))$ in the average case. This algorithm is very simple, but it helps in the understanding of the framework.

The step by step execution is given from Fig. 5 to Fig. 7. At each step, we execute the main loop of our framework as shown in Fig. 4. It basically consists in pipelining the output of the previous module in the loop into the next one.

```
loop
  powerProfile ← PowerProfileGenerator(taskMapping)
  thermalProfile ← ThermalSimulator(powerProfile)
  heatMap ← Visualization(thermalProfile)
  hotspotList ← TaskMigrationAlgo(thermalProfile)
  if hotspotList not empty then
    taskMapping ←
        TaskMigrationAlgo(taskMapping, hotspotList)
  else
    exit loop
  end if
end loop
```

Fig. 4. Icy-Core Main Loop

As input, we suppose we have 20 tasks running on the platform. Step I of Fig. 5 is the result of the initial task mapping deduced from the Platform Simulator. According to this mapping, our algorithm detects two hot spots in the cluster number two: $Core2\_1$ and $Core2\_3$. Their respective temperatures 353.6K (80.45 °C) and 351.6K (78.45 °C) exceed the threshold fixed at 350K (76.85 °C).

As $Core2\_1$ is warmer than $Core2\_3$, the task mapping

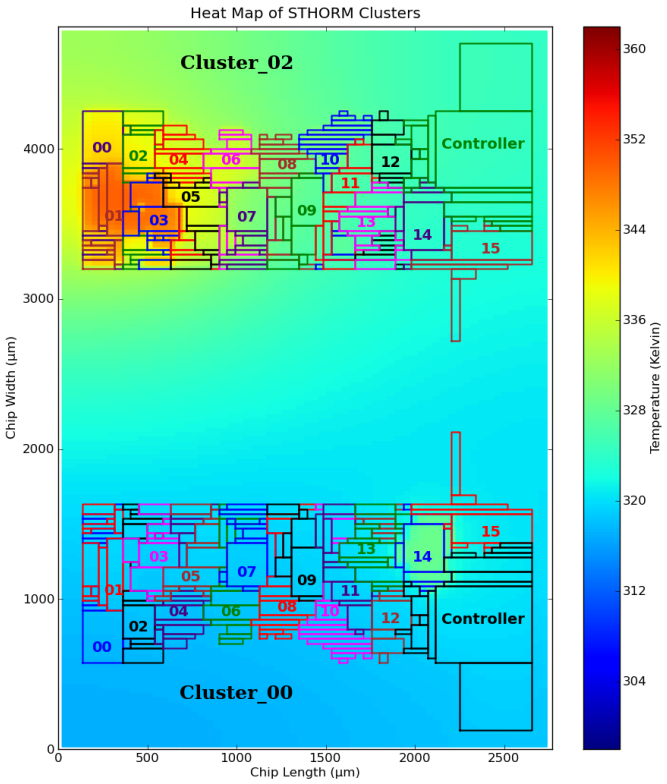

Fig. 5. Step I: Detection of $Core2\_1$ and $Core2\_3$ as hot spots

algorithm takes the decision to handle $Core2\_1$ first.

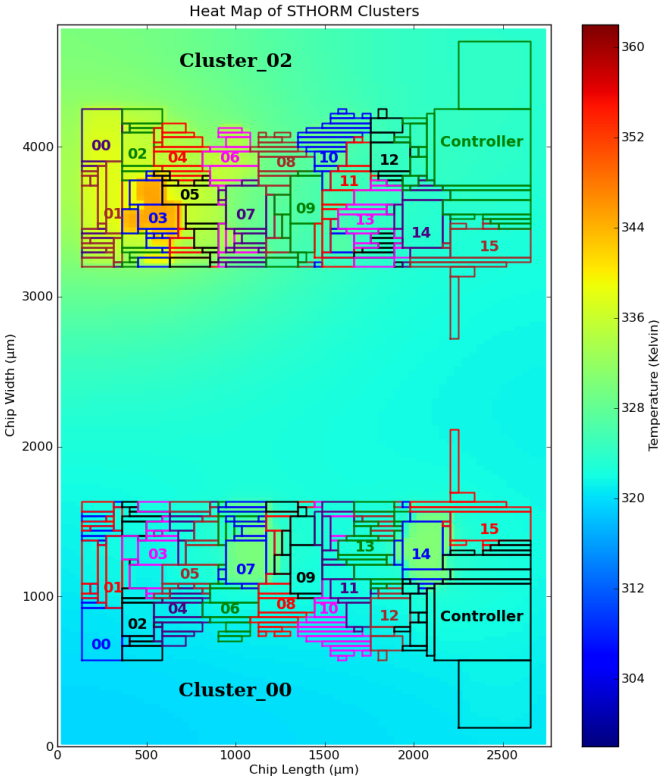

Fig. 6. Step II: Task migration from $Core2\_1$ to $Core0\_7$

Thus, the greediest task executed on $Core2\_1$ is migrated to the coolest core, in our example $Core0\_7$. This migration corresponds to Step II and it is represented by Fig. 6. When several destination cores having the lowest temperature coexist, one is chosen randomly.

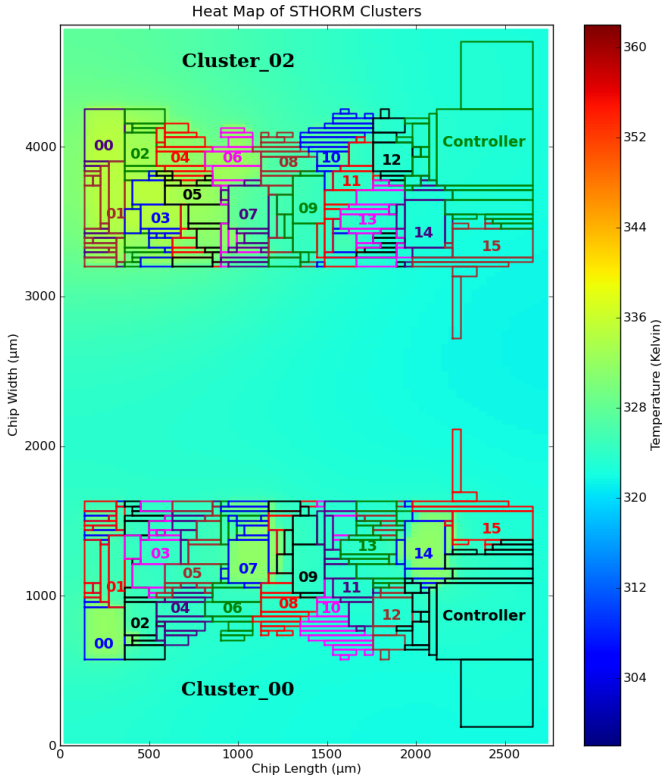

Fig. 7. Step III: Task migration from $Core2\_3$ to $Core0\_0$ – No core is above the threshold, the system is thermically balanced

In spite of this migration, $Core2\_3$ is still considered as hot spot (see Fig. 6). Thus, in Step III, the task running on $Core2\_3$ is migrated to $Core0\_0$. Following this decision, the system thermal profile is well balanced as there is no more core temperature above the threshold (see Fig. 7).

B. Icy-Core Simulation Time

Icy-Core has been tested by the initial task migration algorithm to determine the simulation time overhead. Table I shows the total CPU time for each module of Icy-Core and the total time scale of the loop.

TABLE I
SIMULATION TIME OVERHEAD

Module	Simulation Time (Seconds)
Heat Map Visualization	2.268
Power Profile Generator (PPG)	0.032
Thermal Simulator (TS)	0.102
Task Migration Algorithm (TMA)	0.028
Icy-Core Loop (PPG + TS + TMA)	0.162

 ISBN: 978-1-61208-319-3

The Icy-Core loop has been customized to take effect each $200ms$.

V. Icy-Core Advantages and Limitations

To dynamically manage the chip temperature, an input of the current thermal state is needed to react when hot spots occurs. However, currently, the temperature of the chip and its localized elements is not provided by the platform sensors. Thus, this framework is required to have an idea of the platform thermal profile during the execution of the application. This helps us to take decisions of migration according to the thermal stress. In addition, it allows us to change the task mapping of the application during the execution. Currently, this is not possible in the platform simulator as the current execution model is run-to-completion.

Thanks to Icy-Core modular structure, we can test various task migration algorithms against the thermal effect. These task migration algorithms can later be compared according to different criteria, for instance, the task migration overhead, the maximum temperature reached, etc. We can so determine a task migration algorithm suited to our target platform. Icy-Core does not consider all the hardware constraints, however, it allows us to save the high cost of a chip.

VI. Conclusion

This paper presents a framework implementation to assess task migration algorithms on a targeted many-core platform. Differently from the existing simulation frameworks, the purpose of this framework is to ease the analysis of dynamic reconfiguration solution in order to face thermal issues. The proposed framework represents a combination of two existing modules: i) a Platform Simulator (Gepop-ISS), ii) a Thermal Simulator (3D-ICE), two specially built modules: i) a Power Profile Generator, ii) a Heat Map Visualization module and the Task Migration Algorithm we want to test.

The advantage of the Icy-Core architecture is its generic model. Thus, it is used to test various dynamic reconfiguration algorithms against the same constraint like thermal effect. This helps us to compare the reaction of these algorithms according to the same constraint. In addition, it allows us to apply different constraints than thermal one in order to manage different problems. For instance, to achieve resource management purposes, e.g., management of the platform memory, the thermal simulator module can be replaced by another module that estimates memory utilization.

We are currently investigating improvements made on the STHORM simulator power profile in the frame of management of low power consumption. This will favorably impact the accuracy of our tool and will help us in designing an efficient task migration algorithm for the targeted platform, which is our ultimate objective.

References

[1] D. Zoni, S. Corbetta, and W. Fornaciari, "Hands: Heterogeneous architectures and networks-on-chip design and simulation," in Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), 2012, pp. 261–266.

[2] S. Corbetta, D. Zoni, and W. Fornaciari, "A temperature and reliability oriented simulation framework for multi-core architectures," in IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2012, pp. 51–56.

[3] M. Hsieh, A. Rodrigues, R. Riesen, K. Thompson, and W. Song, "A framework for architecture-level power, area, and thermal simulation and its application to network-on-chip design exploration," SIGMETRICS Perform. Eval. Rev., vol. 38, no. 4, pp. 63–68, Mar. 2011.

[4] L. Sheng, J.-H. Ahn, R. Strong, J. Brockman, D. Tullsen, and N. Jouppi, "Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures," in 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-42), 2009, pp. 469–480.

[5] V. Soteriou, N. Eisley, H. Wang, B. Li, and L.-S. Peh, "Polaris: A system-level roadmap for on-chip interconnection networks," in International Conference on Computer Design (ICCD), 2006, pp. 134–141.

[6] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in Proceedings of the 27th International Symposium on Computer Architecture, 2000, pp. 83–94.

[7] A. Bartolini, M. Cacciari, A. Tilli, L. Benini, and M. Gries, "A virtual platform environment for exploring power, thermal and reliability management control strategies in high-performance multicores," in Proceedings of the 20th Great Lakes Symposium on VLSI (GLSVLSI). ACM, 2010, pp. 311–316.

[8] "AceThermalModeler data sheet," Docea Power, European Headquarters, Moirans, France.

[9] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: a compact thermal modeling methodology for early-stage VLSI design," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 5, pp. 501–513, may 2006.

[10] L. Benini, E. Flamand, D. Fuin, and D. Melpignano, "P2012: Building an ecosystem for a scalable, modular and high-efficiency embedded computing accelerator," in Design, Automation Test in Europe Conference Exhibition (DATE), mar. 2012, pp. 983–987.

[11] T. Ducroux, G. Haugou, V. Risson, and P. Vivet, "Fast and accurate power annotated simulation: Application to a many-core architecture," in 23th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS), 2013.

[12] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza, "3D-ICE: Fast compact transient thermal modeling for 3D ICs with inter-tier liquid cooling," in IEEE/ACM International Conference on Computer-Aided Design (ICCAD), nov. 2010, pp. 463–470.

[13] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza, "Compact transient thermal model for 3D ICs with liquid cooling via enhanced heat transfer cavity geometries," in 16th International Workshop on Thermal Investigations of ICs and Systems (THERMINIC), oct. 2010, pp. 1–6.

[14] J. Choi, C.-Y. Cher, H. Franke, H. Hamann, A. Weger, and P. Bose, "Thermal-aware task scheduling at the system software level," in ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), aug. 2007, pp. 213–218.

[15] I. Yeo, C. C. Liu, and E. J. Kim, "Predictive dynamic thermal management for multicore systems," in 45th ACM/IEEE Design Automation Conference (DAC), jun. 2008, pp. 734–739.

[16] Y. Ge, P. Malani, and Q. Qiu, "Distributed task migration for thermal management in many-core systems," in 47th ACM/IEEE Design Automation Conference (DAC), jun. 2010, pp. 579–584.

[17] M. A. Al Faruque, J. Jahn, T. Ebi, and J. Henkel, "Runtime thermal management using software agents for multi- and many-core architectures," IEEE Design Test of Computers, vol. 27, no. 6, pp. 58–68, nov.-dec. 2010.

 ISBN: 978-1-61208-319-3