Dataflow-based Adaptation Framework with Coarse-Grained Reconfigurable Accelerators*

Claudio Rubattu

University of Sassari, 07100 Sassari, Italy
claudio.rubattu@uniss.it
Univ Rennes, INSA Rennes, IETR, UMR CNRS 6164, 35708 Rennes, France
claudio.rubattu@insa-rennes.fr

Abstract. Today, the demand of adaptive systems is constantly growing, especially in hard-constrained contexts such as Cyber-Physical Systems. However, the efficient management of such platforms requires dealing with several issues such as the real-time execution, energy saving and dynamic context changes. Such strict requirements imply a high flexibility of the application and of the architecture on which it is executed. Runtime managers offer the possibility to dynamically schedule and map an application on the available software processing units. However, hardware acceleration may also be necessary for computationally-intensive workloads that depend on the running functionality, additionally complicating runtime management. Coarse-Grained Reconfigurable (CGR) accelerators have the ability to switch among different domain-specific functionalities with a small overhead. To support energy and time adaptivity in heterogeneous systems, and to exploit multi-core architectures and CGR accelerators, this work proposes the combination of the SPIDER software runtime manager and the dataflow-to-hardware MDC design suite for CGR accelerators.

Keywords: Coarse-Grained Reconfiguration · MDC · SPIDER · FPGA · Dataflow MoC · HW/SW Co-design · Runtime Manager · Datapath Merging

1 Context and Motivation

In the last few years, besides the concepts of embedded and interconnected systems, Cyber-Physical Systems (CPS) have become known and studied with interest by the scientific community. These systems are capable of monitoring and controlling physical elements and consider heterogeneous components that interact with each other in different modalities depending on the context in which they operate. Design and maintenance of such systems are extremely complex because of their multidisciplinary nature, the elaborate requirements, the heterogeneity of their components and the continuous communication between physical

* This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732105.

and computing layers. Adaptation according to uncertain events and to changing functional and non-functional requirements is the most important challenge for the developers. In this context, the Cross-layer modEl-based fRamework for multi-oBjective dEsign of Reconfigurable systems in unceRtain hybRid envirOnments (CERBERO) H2020[1] european project aims at developing a design environment for CPS based on two main elements: i) a cross-layer and model-based approach to describe, optimize and analyze the system according to different views; and ii) an extended adaptivity of the calculation to the system state as well as to its environment, adaptivity provided by an autonomous reconfiguration engine. This work is mainly focused on the second CERBERO element, and, in particular, on the study of the ability of the system to dynamically reconfigure itself according to its state and to its environment.

Compared to all these approaches considered in Section 2, the proposed framework is focused on providing a combination of the state-of-the-art functionalities within the CERBERO project. To do that, in Section 3, an open-source adaptive management system is proposed, portable over several embedded software systems and heterogeneous CGR hardware accelerators. This type of CGR units are hardware blocks that accelerate a given set of application functionalities switching among them without significant reconfiguration time overheads. Moreover, CGR accelerators can be implemented on both ASIC or FPGA platforms. In order to combine these Processing Elements (PEs) with multi-core architectures, the integration activity between the SPIDER[2] dataflow-based runtime manager (see Section 2.1) and the MDC[3] dataflow-to-hardware suite (see Section 2.2) has been conducted. Both tools are based on the dataflow Model of Computation (MoC) that can be used to separate temporal and functional problems in hardware design. Moreover, modularity of the dataflow representations favors a natural splitting of the computation into different blocks, making it possible to automatically map them onto heterogeneous PEs. Thus, the proposed idea is to leverage on a software-hardware design flow to develop and manage dataflow-based autonomous reconfigurable systems, as requested by the CERBERO model-based approach to system design. Challenges and steps related to the implementation of the integration framework are presented in Section 4.

2 Background

Several tools capable of handling runtime adaptivity have been proposed in literature. Most approaches concentrate on software management and rely on existing software frameworks [1, 5–7, 9, 14, 15, 20, 21]. Hardware runtime adaptivity is managed in [18], in which data routing in a specific HoneyComb processor array hardware has been considered, and in [2], in which, besides the software runtime handling, hardware tasks can be implemented in FPGA devices. However, among the proposed frameworks only a few are open-source and effectively

[1] http://www.cerbero-h2020.eu
[2] https://github.com/preesm/spider
[3] http://sites.unica.it/rpct/

available [2, 5, 6, 15]. Among these, [5] and [15] are High Performance Computing (HPC) management systems that place themselves over large-scale facilities composed of multiple CPUs and GPUs. In [6], the framework does not consider hardware acceleration, which is today compulsory in most High Performance Embedded Computing (HPEC) systems, such as embedded video processing systems, embedded deep learning, telecommunication and computer vision systems [19]. Although the framework presented in [2] can handle hardware tasks, these are specific to Intel FPGA devices.

As mentioned in Section 1, the proposed framework considers a combination of the features provided by the state-of-the-art tools and consists of two existing tools: the SPIDER runtime manager and the MDC suite.

2.1 SPIDER

The Synchronous Parameterized and Interfaced Dataflow Embedded Runtime (SPIDER) is a runtime manager for applications described through Parameterized and Interfaced Synchronous Dataflow (PiSDF) MoC and executed on heterogeneous multi-core architectures [8]. When compared to DPN, PiSDF increases processing and communication predictability, serving as input information for multicore and multisystem partitioning, at the cost of some expressiveness, i.e. some unpredictable application behaviors cannot be modeled with PiSDF. The SPIDER runtime is currently available for ARM/Linux-based architectures, Intel x86 platforms, Keystone II architectures from Texas Instruments, and MPPA256 from Kalray. In order to ensure independence between application and platform levels, the SPIDER runtime structure consists of the following layers:

- the *Application Layer*: that is composed of dataflow actors and PiSDF specifications describing a stream processing application;
- the *Runtime Layer*: the core of the runtime manager consisting of a master part called global runtime (GRT), that handles scheduling and mapping of the application, and slave elements named local runtimes (LRT), that execute the processing of the actors depending on the current scheduling decided by the GRT;
- the *Hardware Specific Layer*: this layer is a platform-dependent component designed to manage the inter-core communication and synchronization.

Figure 1 shows the execution scheme of SPIDER. The GRT (master) schedules the application (1) and sends the execution order based on the mapping decisions (2). The LRTs (slaves) deal with the execution of the actors present in the dedicated job queue (3). Jobs are data structures containing the information about synchronization, data and code of the actors to properly perform one instance of an actor in a specific slave. The LRT can be implemented over general- or special-purpose processors, as well as accelerators. During code execution, LRTs exchange data tokens through a pool of data FIFOs (4). Once the processing of the actor is completed, the LRTs send new parameter values (if any) to the GRT (5). Indeed, a parameter value can be set dynamically by a configuration actor mapped in a LRT, influencing the algorithm execution. Moreover, the GRT receives also the execution traces (the actor start and end times based on the same timing reference) by the LRTs (6).

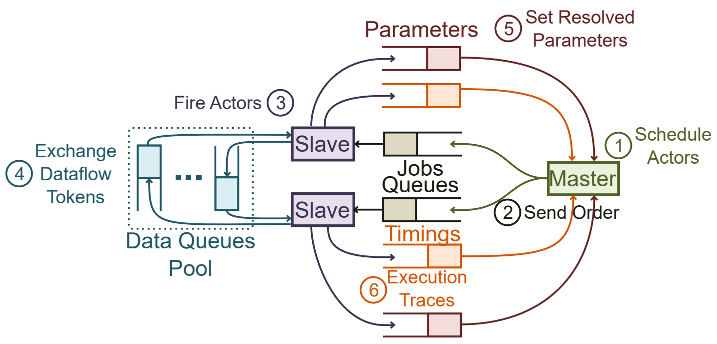

Fig. 1. The SPIDER runtime internal scheme.

2.2 MDC

The Multi-Dataflow Composer (MDC) is a toolset for the design and development of CGR systems based on the Dataflow Process Network (DPN) MoC [11]. This design suite is composed of two main sub-components:

- the *Multi-Dataflow Generator (MDG)*: a dataflow-based model-to-model compiler that, given an input set of dataflow networks describing the functionalities to be executed in hardware, generates a high-level multi-dataflow specification of the system leveraging on datapath merging techniques [17];
- the *Platform Composer (PC)*: a dataflow-to-hardware synthesizer that, given the mentioned multi-dataflow specification and the hardware description of the dataflow actors and the protocol used for communication among them, generates a CGR hardware accelerator.

MDC offers also other features that optimize the generated systems and favor their integration in real environments, such as:

- a structural profiler [12] that, taking into account the low level feedback coming from a priori synthesis of the generated platform, is capable of identifying the optimal multi-dataflow configuration depending on a set of metrics (area, power, frequency);
- a power manager [3, 4] that automatically sets power saving strategies, such as clock- and power-gating at system modelling level;
- a rapid prototyper [16] embedding the generated CGR systems onto ready-to-use platform-dependent IPs (for Xilinx devices).

3 CGR Adaptation Framework

SPIDER and MDC show complementary characteristics that motivate their integration. SPIDER provides software scheduling and memory management at runtime for multi-core architectures. However, SPIDER supported processing elements do not include reconfigurable hardware blocks and adaptivity is based on an a priori knowledge of several metrics (latency, throughput and memory utilization) evaluated with respect to changes in software parameters. On the contrary, MDC provides a model-to-model compiler capable of merging several dataflow applications, as well as a dataflow-to-hardware synthesizer that implements CGR systems. Moreover, MDC profiles CGR system configurations, providing different metrics (area, power, frequency) and includes a power manager that offers clock- and power-gating techniques.

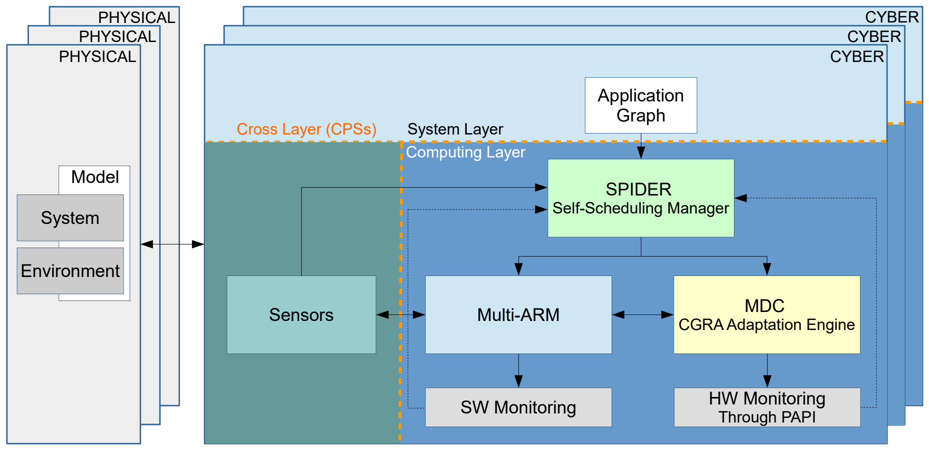

Fig. 2. Adaptation scheme for heterogeneous hardware/software reconfigurable systems.

The proposed framework aims at exploiting the features of SPIDER and MDC in order to respectively manage software and hardware reconfigurability at runtime. Tool integration is also based on the use of dataflow models with similar properties in order to favour the estimation of metrics such as latency, throughput and energy. The main idea behind these metrics is to use CGR blocks as slave processing elements in the target system, and re-schedule these processing elements from a host processor at runtime using models of the instantaneous hardware behavior. The adaptation architecture is illustrated in Figure 2. An application graph, conforming to a PiSDF MoC, is dynamically scheduled by SPIDER. Depending on the scheduling, a hardware system composed of ARM cores and CGR accelerators (implemented in Xilinx or Intel modern SoC FPGAs) performs the computation. Software and hardware monitoring provides feedback to SPIDER about the current execution of the tasks. Regarding the monitoring, this feature will be provided through the integration of Papify[4], an event-based performance monitoring tool [10], and MDC[5]. In addition, reconfiguration/re-scheduling can also be triggered by sensors in order to adapt the computing layer to the environment changes or system needs.

4 Open Issues and Research Plan

The challenges that this research plan intends to address are summarized as follows:

i) management of hardware/software adaptivity in systems for which low energy and time overheads are required;
ii) performance predictability related to the different configuration of the system;
iii) hardware/software fault robustness in architecture composed of heterogeneous processing elements, such as ARM cores and CGR accelerators.

In order to guarantee fast reconfiguration and the correct execution of the application, CGR acceleration and hardware/software monitoring will be considered respectively. In addition, a strategy based on modeling of architecture and chosen applications will be used to achieve energy and latency predictability, as similarly proposed in [13].

This research plan is part of the activities of the H2020 CERBERO european project, which will be assessed in three different use cases among which: planetary exploration (lead by: Thales Alenia Space), smart marine vehicles for ocean monitoring (lead by: AmbieSense), and a smart travelling system for electric vehicles (lead by: TNO in cooperation with Centro Ricerche Fiat). The proposed flow will be used to test applications deriving from the scenarios of the Thales Alenia Space and AmbieSense use cases. To build the proposed flow, the following three steps have been envisioned:

- *Step 1.* Integrate MDC and SPIDER by combining software and hardware adaptation according to variable application parameters;
- *Step 2.* Verify this approach with respect to relevant CERBERO key performance indicators (as latency, throughput and energy);
- *Step 3.* Derive a proof of concept of the proposed approach in the context of CERBERO project use case scenarios.

The work done so far has been focused on the state-of-the-art adaptation tools and open issues, the study of the tools identified for the integration, and the strategies to address the problems described above.

Further developments of the proposed flow will be necessary to improve the decision-making strategy that SPIDER considers for the adaptability of the system. The runtime manager inputs should embed a more detailed description of

[4] https://github.com/Papify
[5] This activity is planned within the CERBERO project and is currently ongoing.

the application to be executed and the model of the architecture on which it has to be performed. Nevertheless, besides managing the known scenarios during the development phase, it would be necessary to take into account also the statistical data related to the experience over a certain period of time, in order to prevent unexpected events at design time. This information should be based on the user interrupts, the external environment and the hardware and software monitors.

References

1. Assayad, I., Girault, A.: Adaptive Mapping for Multiple Applications on Parallel Architectures. In: Sabir, E., García Armada, A., Ghogho, M., Debbah, M. (eds.) Ubiquitous Networking. pp. 584–595 (2017)
2. Bragg, G.M., Leech, C.R., Balsamo, D., Davis, J.J., Wachter, E.W., Merrett, G., Constantinides, G.A., Al-Hashimi, B.: An application- and platform-agnostic control and monitoring framework for multicore systems. In: 3rd International Conference on Pervasive and Embedded Computing (2018)
3. Fanni, T., Sau, C., Meloni, P., Raffo, L., Palumbo, F.: Power and clock gating modelling in coarse grained reconfigurable systems. In: Proceedings of the ACM International Conference on Computing Frontiers, CF'16. pp. 384–391 (2016)
4. Fanni, T., Sau, C., Raffo, L., Palumbo, F.: Automated Power Gating Methodology for Dataflow-based Reconfigurable Systems. In: Proceedings of the 12th ACM International Conference on Computing Frontiers, CF'15. pp. 61:1–61:6 (2015)
5. Gautier, T., Lima, J.V.F., Maillard, N., Raffin, B.: XKaapi: A Runtime System for Data-Flow Task Programming on Heterogeneous Architectures. In: 2013 IEEE 27th International Symposium on Parallel and Distributed Processing. pp. 1299–1308 (2013)
6. Gerostathopoulos, I., Bures, T., Hnetynka, P., Keznikl, J., Kit, M., Plasil, F., Plouzeau, N.: Self-adaptation in software-intensive cyberphysical systems: From system goals to architecture configurations. Journal of Systems and Software **122**, 378–397 (2016)
7. Han, M., Park, J., Baek, W.: CHRT: A criticality- and heterogeneity-aware runtime system for task-parallel applications. In: Design, Automation Test in Europe Conference Exhibition (DATE), 2017. pp. 942–945 (2017)
8. Heulot, J., Pelcat, M., Desnos, K., Nezan, J.F., Aridhi, S.: SPIDER: A Synchronous Parameterized and Interfaced Dataflow-based RTOS for multicore DSPS. In: 2014 6th European Embedded Design in Education and Research Conference (EDERC). pp. 167–171 (2014)
9. Hormati, A.H., Choi, Y., Kudlur, M., Rabbah, R., Mudge, T., Mahlke, S.: Flextream: Adaptive Compilation of Streaming Applications for Heterogeneous Architectures. In: 2009 18th International Conference on Parallel Architectures and Compilation Techniques. pp. 214–223 (2009)
10. Madroñal, D., Morvan, A., Lazcano, R., Salvador, R., Desnos, K., Juárez, E., Sanz, C.: Automatic Instrumentation of Dataflow Applications using PAPI. In: Proceedings of the 15th ACM International Conference on Computing Frontiers, CF'18. pp. 232–235 (2018)
11. Palumbo, F., Fanni, T., Sau, C., Meloni, P.: Power-Awarness in Coarse-Grained Reconfigurable Multi-Functional Architectures: A Dataflow Based Strategy. J. Signal Process. Syst. **87**(1), 81–106 (2017)

12. Palumbo, F., Sau, C., Raffo, L.: Coarse-grained reconfiguration: dataflow-based power management. IET Computers & Digital Techniques **9**(1), 36–48 (2015)
13. Pelcat, M., Mercat, A., Desnos, K., Maggiani, L., Liu, Y., Heulot, J., Nezan, J., Hamidouche, W., Mnard, D., Bhattacharyya, S.S.: Reproducible Evaluation of System Efficiency with a Model of Architecture: From Theory to Practice. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems pp. 1–14 (2018)
14. Quan, W., Pimentel, A.D.: A hierarchical run-time adaptive resource allocation framework for large-scale MPSoC systems. Design Automation for Embedded Systems **20**(4), 311–339 (2016)
15. Robson, M.P., Buch, R., Kale, L.V.: Runtime Coordinated Heterogeneous Tasks in Charm++. In: 2016 Second International Workshop on Extreme Scale Programming Models and Middleware (ESPM2). pp. 40–43 (2016)
16. Sau, C., Fanni, L., Meloni, P., Raffo, L., Palumbo, F.: Reconfigurable coprocessors synthesis in the MPEG-RVC domain. In: International Conference on ReConFigurable Computing and FPGAs, ReConFig 2015. pp. 1–8 (2015)
17. Souza, C.C.d., Lima, A.M., Araujo, G., Moreano, N.B.: The Datapath Merging Problem in Reconfigurable Systems: Complexity, Dual Bounds and Heuristic Evaluation. J. Exp. Algorithmics **10** (2005)
18. Thomas, A., Becker, J.: Dynamic Adaptive Runtime Routing Techniques in Multigrain Reconfigurable Hardware Architectures. In: Becker, J., Platzner, M., Vernalde, S. (eds.) Field Programmable Logic and Application. pp. 115–124 (2004)
19. Wolf, M.: High-Performance Embedded Computing, Second Edition: Applications in Cyber-Physical Systems and Mobile Computing. 2nd edn. (2014)
20. Yun, J., Park, J., Baek, W.: HARS: A heterogeneity-aware runtime system for self-adaptive multithreaded applications. In: 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC). pp. 1–6 (2015)
21. Zhang, F., Cao, J., Khan, S.U., Li, K., Hwang, K.: A task-level adaptive MapReduce framework for real-time streaming data in healthcare applications. Future Generation Computer Systems **43-44**, 149–160 (2015)