Design For ReConfigurability: an Electronic System Level Methodology to exploit Reconfigurable Platforms

***Abstract*—The presented Ph.D. project deals with the definition of an Electronic System-Level methodology, and the development of related prototype tools, for the exploitation of FPGA platforms which offer the Dynamic Partial Reconfiguration feature. The final goal of the project is to provide a Design for ReConfigurability approach able to suggest, at design-time, a set of system configurations that allow to satisfy functional requirements and to trade-off in different way non-functional ones. Such configurations are then exploited, at run-time, by a system manager that is also able to take into account the reconfiguration cost to fully evaluate pro and cons of such a process. While the proposed Ph.D project is still under development, its current status is described highlighting the planned works.**

***Index Terms*—Electronic System-Level, Design Methodologies, Dynamic Partial Reconfiguration, FPGA.**

I. Introduction

Nowadays, the usage of reconfigurable platforms that allow modifying both the HW and SW configurations is becoming even more relevant in the (real-time) embedded systems domain. In particular, the Field Programmable Gate Arrays (FPGAs) have become very attractive, due to their reconfigurable nature, often supported by the Dynamic Partial Reconfiguration (DPR) feature. In fact, thanks to the DPR, it is possible reconfiguring at run-time the HW architecture of a platform without blocking the entire system, or affecting the tasks running on the same platform but not involved in the reconfiguration itself. For instance, it is possible to accelerate a SW task by loading a configuration on which such a task is implemented in HW. However, the DPR process produces an impact [8] [9] (e.g., in term of timing and energy overhead) on the system that uses it. This impact, if not well considered, could nullify the advantages of the DPR itself [1]. Hence, before adopting the DPR, it is important to characterize its cost in terms of reconfiguration time and energy. In such a context, the presented Ph.D. project deals with the definition of an Electronic System-Level (ESL) methodology for the exploitation of FPGA-based & DPR-enabled platforms, and the development of related prototype tools. The topic is named as *Design for ReConfigurability (DFRC)* and it is mainly targeting the (real-time) embedded systems domain. The principal goals of DFRC are twofold:

- at design-time, to identify the configurations that allow developing a system able to satisfy the functional requirements while trading-off non-functional ones (e.g., timing performances vs. energy/power);
- at run-time, to select the best configuration to be used in a given moment, depending on internal/external status and events, and to apply it by exploiting the DPR feature of the target platform while also considering its cost.

More specifically, at design-time, an ESL HW/SW co-design methodology is used to define the HW/SW system architecture and to map onto it the set of system tasks by implicitly inducing an HW/SW partitioning of such tasks. In particular, the Ph.D. project, named *DFRC Project*, will extend an existing ESL HW/SW co-design methodology [12] to consider also energy/power requirements. In such a way, it will be possible to identify several configurations able to trade-off in a different way timing/energy metrics (i.e., from max timing performance to max energy savings, considering one or more intermediate configurations). Then, this DFRC Project will develop a run-time system manager, with the goal of considering the status and events internal and/or external to the system, and of reconfiguring the latter by selecting a new configuration among the ones identified at design-time. For example: a low-power configuration can be selected when a low-battery status is detected; a high-performance configuration can be considered if, due to an unexpected change in the workload (e.g., an unexpected set of sporadic events from the external world), given timing constraints are going to be no more satisfied; in a real-time scenario, a SW task can be accelerated in HW to keep schedulability of a dynamically changing task set [6]. However, all the decisions about the reconfiguration process should take into account the DPR cost. For this, the run-time manager will be designed to be aware of such a cost thanks to the modeling and the characterization of the whole DPR process. Such modeling and characterization activities are also a relevant part of the DFRC Project and represent the main outcomes of the first Ph.D. year. The next two sections will provide, respectively, more details about the first year activities and the work planned for the last two years.

II. DPR modeling and characterization

The DPR modeling and characterization activities have been performed through an in-depth study on the FPGAs produced by two of the most valuable vendors: Xilinx [5] and Intel [4]. Each platform, in order to allow the reconfiguration, contains a dedicated reconfiguration area that will host a configuration (called *Bitsream*, BS hereinafter) opportunely prepared at design-time. Based on this knowledge, the first outcomes of the DFRC Project ([1] [2]) have defined a general DPR compliant platform by also identifying the reconfiguration path followed to transfer the BS from a generic external memory to the

FPGA reconfiguration area. However, the BS transfer requires a certain amount of time depending on both BS size and available bandwidth; as proved in [9] the reconfiguration bandwidth can vary up to 21x and consequently also the reconfiguration time. Therefore, if aiming to develop a run-time system manager able to exploits the DPR, an approach to calculate in advance the reconfiguration time is essential to evaluate the DPR cost before applying the reconfiguration process. This manager, by using a custom HW monitoring sub-systems [13], observes the status of the system and, in case of timing performance losses, evaluates whether a DPR can help on recovering them by loading a new configuration. More in general, the manager evaluates whether the DPR adoption could be profitable or not for the system at a specific time. Starting from this generalization, a DPR profitability definition, from the DPR time cost point of view, has been proposed in [2] together with an approach to calculate in advance the DPR time (t_{DPR}^{CAL}). This approach, unlike other works in literature (i.e., [7] [10] [14]), is based on a deterministic and target-independent static timing analysis that accounts for all the reconfiguration path elements. It allows calculating in advance the t_{DPR}^{CAL} with an accuracy of 30% with respect to the real DPR time ($acc = |t_{DPR}^{CAL} - t_{DPR}^{ACT}|/t_{DPR}^{ACT}$). Moreover, such an approach is potentially applicable to all the actual DPR-compliant platforms [4] [5] since, differently from [11], it considers also the architectural elements recently introduced by the FPGA vendors to reduce the DPR impact (e.g., *Direct Memory Access* (DMA) controller in Xilinx Zynq-7000 [15]).

After that, in order to evaluate how the dedicated DMA can reduce the DPR impact, an investigation on this element has been conducted pointing out that the DMA usage reduces the system overheads by increasing the reconfiguration speed-up [7]. Indeed, the dedicated DMA has been introduced by the vendors to transfer, in place of a processor, the BS along the reconfiguration path. In this way, the processor, after seeding to the dedicated DMA a DPR request (i.e., a request to transfer the BS), is released from the DPR process and can handle other tasks running on the same platform. Unfortunately, the issue related to how *many DPR requests can be efficiently managed by the dedicated DMA* (hereinafter, *DPR scalability*) is not clearly addressed in the literature. Therefore, another outcome of the presented DFRC Project ([3]) has conducted on a Zynq-7000 SoC an experimentation about the DPR process scalability. The results of this experimentation prove that the DPR process is scalable, but it is limited by the architecture of the dedicated DMA. Specifically, the work in [3] has found that a DMA architectural feature, not indicated in the platform datasheet [15], imposes an upper bound on the number of DPR requests that can be handled at the same time. This finding has been validated both theoretically, by exploiting the approach defined to calculate the aforementioned reconfiguration time properly adapted to consider with the discovered DMA feature, and experimentally, by developing a workaround that allows overcoming the DMA limitation.

III. Planned works

In the next years, the plan is to continue working on DFRC, also addressing the aspects of energy and power in addition to time. Specifically, for the run-time domain, the DFRC Project will continue developing the system by considering also the DPR characterization with respect, other than the time, also to energy/power. For what concerns the design-time domain, the DFRC Project will continue with an in-depth study about the ESL HW-SW co-design domain extending an existing methodology [12]; with such extension, the considered methodology will be able to provide to the designer the system configurations able to trade-off performance and energy/cost metrics.

References

[1] Removed for blind review.

[2] Removed for blind review.

[3] Removed for blind review.

[4] Intel FPGAs and Programmable Devices. https://www.intel.it/content/www/it/it/products/programmable.html. (Accessed: Oct. 3,2019).

[5] Xilinx - Adaptable. Intelligent. https://www.xilinx.com. (Accessed: Oct. 3,2019).

[6] A. Biondi, A. Balsini, M. Pagani, E. Rossi, M. Marinoni, and G. Buttazzo, "A framework for supporting real-time applications on dynamic reconfigurable fpgas," in *IEEE Real-Time Systems Symposium (RTSS)*, Nov. 2016, pp. 1–12.

[7] C. Claus, F. H. Muller, J. Zeppenfeld, and W. Stechele, "A new framework to accelerate virtex-ii pro dynamic partial self-reconfiguration," in *2007 IEEE International Parallel and Distributed Processing Symposium*, March 2007, pp. 1–7.

[8] C. Claus, B. Zhang, W. Stechele, L. Braun, M. Hubner, and J. Becker, "A multi-platform controller allowing for maximum dynamic partial reconfiguration throughput," in *International Conference on Field Programmable Logic and Applications*, Sept. 2008, pp. 535–538.

[9] M. Damschen, L. Bauer, and J. Henkel, "Corq: Enabling runtime reconfiguration under wcet guarantees for real-time systems," *IEEE Embedded Systems Letters*, vol. 9, no. 3, pp. 77–80, Sept., 2017.

[10] M. Liu, W. Kuehn, Z. Lu, and A. Jantsch, "Run-time partial reconfiguration speed investigation and architectural design space exploration," in *2009 International Conference on Field Programmable Logic and Applications*, Aug 2009, pp. 498–502.

[11] K. Papadimitriou, A. Dollas, and S. Hauck, "Performance of partial reconfiguration in fpga systems: A survey and a cost model," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 4, no. 4, pp. 36:1–36:24, Dec., 2011.

[12] L. Pomante, V. Muttillo, M. Santic, and P. Serri, "Systemc-based electronic system-level design space exploration environment for dedicated heterogeneous multi-processor systems," *Microprocessors and Microsystems*, vol. 72, p. 102898, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0141933119301000

[13] G. Valente, V. Muttillo, L. Pomante, F. Federici, M. Faccio, A. Moro, S. Ferri, and C. Tieri, "A flexible profiling sub-system for reconfigurable logic architectures," in *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, Feb 2016, pp. 373–376.

[14] K. Vipin and S. A. Fahmy, "Zycap: Efficient partial reconfiguration management on the xilinx zynq," *IEEE Embedded Systems Letters*, vol. 6, no. 3, pp. 41–44, 2014.

[15] Xilinx, *Zynq-7000 SoC Technical Reference Manual (UG585)*, 2018, Accessed: Oct. 3,2019. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf