Design Space Exploration for Mixed-Criticality Embedded Systems considering Hypervisor-based SW Partitions

Vittoriano Muttillo
University of L'Aquila, L'Aquila, Italy
vittoriano.muttillo@graduate.univaq.it

Giacomo Valente
University of L'Aquila, L'Aquila, Italy
giacomo.valente@univaq.it

Luigi Pomante
University of L'Aquila, L'Aquila, Italy
luigi.pomante@univaq.it

Abstract—**This work faces the role of Design Space Exploration for embedded systems based on heterogeneous parallel architectures and subject to mixed-criticality system requirements, while considering the exploitation of hypervisor-based SW partitions to better manage isolation. In particular, it presents an evolutionary partitioning and mapping approach integrated into a reference Electronic System Level HW/SW Co-Design framework to propose and early validate design solutions by means of HW/SW Co-Simulations.**

Index Terms—**HW/SW Co-Design, Heterogeneous Parallel Systems, Design Space Exploration, Mixed-Criticality, Hypervisor**

I. Introduction

In recent years, there has been a growing trend for switching from single-processor/core to (heterogeneous) multi-processor/cores (i.e. parallel) platforms to execute embedded applications with different levels of criticality (i.e. *Mixed-Criticality Embedded Systems*, MCESs). In case of single-processor/core MCES, it is crucial to ensure temporal isolation among tasks, meanwhile for parallel MCES, different embedded applications run in parallel on different processors/cores competing to access shared resources by using different communication and synchronization mechanisms. The main problem in the management of a MCES is to ensure that low criticality embedded applications do not interfere with high criticality ones. This type of systems can be found in many domains such as aerospace [1] and automotive industry [2]. Critical and non-critical applications can be further divided by identifying different criticality classes. The goal is always to allow these applications to interact and coexist on the same platform, but a proper management of such mixed criticality (MC) systems becomes a very complex task that poses several challenges [3]. The basis for integrating various critical applications are the isolation mechanisms that allow to enforce strict temporal and spatial separation [4]. As an example, according to this approach, embedded applications with different levels of criticality can be allocated on different partitions by exploting hypervisors (HPVs) technologies and virtualized environments, which can be verified and validated in isolation, or can be allocate on different HW components. The identification of the best solution considering heterogeneous scenarios is not always possible, so sub-optimal and heuristics methods are needed to support MCESs designers. In such a context, the purpose of this work is to present a *Design Space Exploration* (DSE) step to support the development of heterogeneous parallel MCES considering HPV (and SW partitions) technologies.

II. DSE for Safety Critical Applications

In the last few years, a growing trend in the embedded systems domain is to run multiple applications with different levels of criticality on a shared hardware platform, where the criticality of an application is an indication of the required level of safety and security (i.e. assurance). In such a context, the most critical development steps are related to the *System Specification* and the *Design Space Exploration* activities [5] and the main differences among the various works in the literature are mainly related to the different amount of information and actions that explicitly rely on the designer experience. In this scenario, AUTOFOCUS3 [6] proposes a model-based development process at different levels of abstraction introducing safety-oriented constraints associated to computing components. CONTREP [7] is a framework supporting UML/MARTE based modeling, analysis and design of MC embedded systems, and considers safety constraints into the different design activities, integrating external tool like *Multicube Explorer* [8] for the DSE step. DeSyDe [9] provides a DSE tool for bare-metal applications, finding implementations for a set of tasks on a shared multi-processor platform starting from synchronous dataflow graphs (SDFGs), used to describe the application, and a predictable model for target platform, introducing MC requirement at scheduling level. Considering HPV-based SW partitions technologies, a lot of HPVs have been developed to match certification requirements, avoiding interferences among partitions into a self-contained environment. In particular, *PikeOS* [10] is a real-time operating system able to provide paravirtualization services to manage platform and to support different software services and partition layer. *XtratuM* [11] is a bare metal hypervisor supporting paravirtualization for embedded systems to meet safety critical real-time requirements. *Wind River Hypervisor* [12] integrating a real-time embedded, type 1 hypervisor into the core of VxWorks and *Wind River Linux*.

In this context, this work proposes a DSE approach that is able to consider MC issues into the development of heterogeneous parallel MCES, that exploit HPV technologies. The main differences among the previous works are related to the introduction of HPV-based SW partitions that allow to find cheaper (in terms of area/chip cost) and robust solutions (in term of timing, communication and concurrency performance), increasing the space of feasible final implementations, because none of the previously reported tools consider HPV solutions in the DSE step. A work that considers HPV and a methodology to identify a set of HPV-based SW partitions, and to allocate applications to partitions is [13], but it considers only a fixed multi-core architecture, managing allocation and binding of application among HPV partitions in a homogeneous multiprocessor platform. So, at the best of our knowledge, there are few works that introduce mixed-criticality issues directly into a HW/SW co-design flow, and there is a lack with respect to consider HPV technologies in a early design stage.

III. HW/SW Co-Design Framework

In the context of MCES, this work adopts a specific design flow (Hepsycode: *HW/SW Co-Design of Heterogeneous Parallel Dedicated Systems*) [14], shown in Fig. 1, based on an existing *ESL HW/SW Co-Design Methodology* [15], and extends it with the capability to manage MC requirements. The ***System Description*** step defines three reference models: *Application Model, Partition Model* and *Platform Model.*

*1) **Application Model:*** The first model exploits a behavioral modeling language, named HML (*HEPSY Modeling Language*), [16] based on the Communicating Sequential Processes (CSP) Model of Computation (MoC) [17] and SystemC [18]. By means of HML it is possible to specify the *System Behavior Model* (SBM), i.e. an executable model of the system behavior, a set of *Non Functional Constraints* (NFC) and a set of *Reference Inputs* (RI) to be used for simulation-based activities. $SBM = \{PS, CH\}$ is a CSP-based executable model of the system behavior that explicitly defines also a model of communication among processes (PS) using unidirectional point-to-point CSP-like blocking channels (CH) for data exchange. $PS = \{ps_1, .., ps_n\}$ is a set of n concurrent processes that communicate exclusively by means of channels and use only local variables. Each process has a criticality level $C(ps_i)$: 0 (lower) to *max* (higher) imposed by the designer depending on the safety standard related to the specific application domain [3]. $CH = \{ch_1, .., ch_c\}$ is a set of c logical channels where each channel is characterized by source and destination processes, and some details about transferred data (i.e. *size*, *type*). NFC is composed of *Timing Constraints* (TC), *Architectural Constraints* (AC) and *Scheduling Directives* (SD). In this work, the only TC expressed by the designer is the *Time-to-Completion* (TTC) one: it is the time available to complete the SBM execution from the first input trigger to the complete output generation. AC is related to the considered *Target Form Factor* (TFF), i.e. *System On-chip* (SoC: ASIC or FPGA) or *System On-Board* (SoB: PCB), and to the considered *Target Template Architecture* (TTA). Finally, SD specifies the available scheduling policies. At the moment, they are only *First-Came First-Served* and *Fixed Priority* [19].

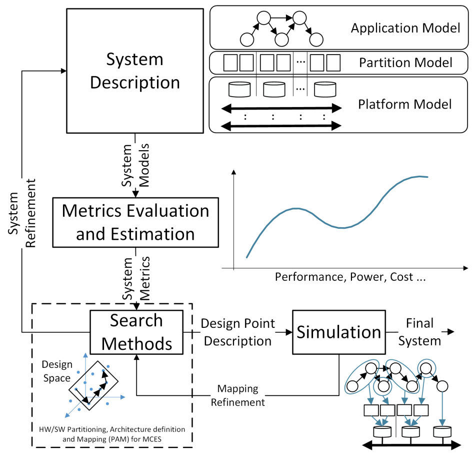


Fig. 1. Hepsycode Methodology.

*2) **Partition Model:*** The second model represents the HPV-based SW partitions layer. $PT = \{pt_1, .., pt_q\} \in \mathbb{N}^q_{\geq 0}$ is a set of q HPV partitions (represented by scalar partition number identifiers). The main hypothesis is that only General Purpose Processors (GPPs) are able to manage and use virtualized technologies. If $pt_i = 0$, then there is no virtualized component on the specific processor. If $pt_i \neq 0$, the GPP processor exploits HPV technologies in order to manage MC processes.

*3) **Platform Model:*** The third model defines the basic HW components available to build the final HW platform based on the selected TTA. The final HW platform is composed of several basic HW components. These components are collected into a *Technologies Library* (TL). $TL = \{PU, MU, EIL\}$ can be considered as a generic "database" that provides the characterization of the available processor technologies, where $PU = \{pu_1, .., pu_p\}$ is a set of *p Processing Units*, $MU = \{mu_1, .., mu_m\}$ is a set of *m Memory Units* and $EIL = \{il_1, .., il_l\}$ is a set of *External Interconnection Links*. The designer uses TL to build a set of b *Basic Blocks* $BB = \{bb_1, .., bb_b\}$, available during DSE step to automatically define the HW architecture. A generic BB is composed of a set of PU, a set of MU, an *Internal Interconnection Link* (IIL) component and a *Communication Unit* (CU). IIL is the shared link between PU, MU and CU (e.g. AMBA internal bus). CU represents the set of EIL that can be managed by a BB. BB internal architecture is dependent on TFF and TTA. This work focuses on *Heterogeneous Multi-Processor System with Distributed Memory* [20] where each BB element is composed of only 1 PU element (possibly heterogeneous among the different BBs), some local MU elements and 1 CU element. The ***Metrics Evaluation and Estimation*** step provides several metrics related to the BB involved in the design flow. This step aims at extracting as much information as possible about the system by analyzing the SBM (*Application Model*), while considering the available BBs (*Platform Model*) and the use of HPV technologies (*Partition Model*). This step is composed of *Co-Analysis* and *Co-Estimation* activities to evaluate/estimate several metrics related to the BB involved in the design flow.

Co-Analysis performs evaluation of *Affinity* [21], *Concurrency* and *Communication* metrics [22]. *Co-Estimation* performs a *Static Estimation* of *Timing* [23] and *Size*, and a *Dynamic Estimation* of *Load* [22] and *Bandwidth*.

After these steps, the reference co-design flow reaches the **Design Space Exploration** step. Starting mainly from *Application Model*, *Partition Model* and *Platform Model*, it includes two iterative activities: *Search Methods*, that performs *HW/SW Partioning, Architecture definition and Mapping* (PAM), by using a genetic algorithm that allows to explore the design space looking for feasible mapping/architecture items suitable to satisfy imposed constraints; *Timing Co-Simulation*, that considers suggested architecture/mapping items to actually check for timing constraints satisfaction (Fig. 1).

IV. Mixed-Criticality Evolutionary Approach

The proposed DSE is based on a *Genetic Algorithm* (GA) [24] used to optimize a multi-objective cost function that quantifies the quality of each individual of the GA population, as listed below:

$$\min_i CF_i^j = \min_i f_i^j(X_{i,1}^j,\ X_{i,2}^j,\ ..\ ,\ X_{i,k}^j) \quad \forall i = 1\ ..\ P,\ j = 1\ ..\ I \tag{1}$$

$$CF_i^j = \sum_k \omega_k \cdot X_{i,k}^j = \omega_{NTCC} \cdot X_{NTCC_i}^j + \omega_L \cdot X_{L_i}^j + \omega_C \cdot X_{C_i}^j + \omega_{CRIT} \cdot X_{CRIT_i}^j \tag{2}$$

CF_i^j is the cost function evaluated at iteration *j* for each individual *i*, *I* is the maximum number of iterations of the search algorithm and *P* is the size of population at iteration *j*. $X_{i,k}^j$ represents the value of metric *k* for each individual IND_i, while ω_k is the weight associated to each metric. The rest of this paragraph defines the metrics (called indexes) and the methods used to evaluate them at each iteration (j apex in the $X_{i,k}^j$ equations has been removed for this purpose). In this context, the instance of an individual IND_i is defined as a matrix where the column index represents processes and the value represents BB instances and PT elements.

A. Processes Communication Index

The ***Processes Communication Index*** is based on the *Communication Matrix* $CM \in \mathbb{R}^{n \times n}$, calculated in the Co-Analysis step. CM is expressed by the number of bits sent/received over each channel. So, for each individual IND_i, it is possible to define a *Processes Communication Selection Matrix*, $S^{cm}(\bar{x}) \in \mathbb{R}^{n \times n}$, as listed below:

$$S^{cm(i)} = \begin{cases} s_{j,k}^{cm(i)} = 1, & if\ ps_j \in pu_x \wedge ps_k \in pu_y \wedge pu_x \neq pu_y \\ s_{j,k}^{cm(i)} = 0.25, & if\ ps_j \in pt_x \wedge ps_k \in pt_y \wedge pt_x \neq pt_y \\ s_{j,k}^{cm(i)} = 0, & \text{otherwise} \end{cases} \tag{3}$$

So, for each individual IND_i, the *Inter Cluster Communication Cost*, $ICCC_i \in \mathbb{R}^{n \times n}$, represents the cost associated to process communication if processes are allocated on different processors or HPV-based SW partitions:

$$ICCC_i = CM \cdot S^{cm(i)} \tag{4}$$

Starting from ICCC matrix, the *Normalized Total Communication Cost* index is:

$$X_{NTCC_i} = \frac{\sum_{j=1}^{n}\sum_{k=1}^{n} iccc_{j,k}^i}{max_{NTCC}},\ maxNTCC = \sum_{j=1}^{n}\sum_{k=1}^{n} cm_{j,k} \tag{5}$$

B. Load Index

The ***Load Index*** is based on the *Load Matrix* $L = \{[l_1, .., l_n] : l_j = [l_{1,j}, .., l_{s,j}]^\intercal\} \in \mathbb{R}^{s \times n}$, calculated in the Co-Estimation step, where each matrix element represents the load that each process ps_j would impose to each *s non-SPP* processor pu_k (i.e. SW processor, used in at least one BB) to satisfy TTC. L is estimated by allocating all the *n* processes to a single-instance of each SW processor pu_k and performing a simulation for each one into the Co-Estimation step [22]. Three parameters are computed: FRT_k *(Free Running Time)*, i.e. the total simulated time on processor pu_k; $t_{k,j}$, the simulated time for each process ps_j on processor pu_k; $N_{k,j}$, the number of executions of each process ps_j on processor pu_k. Starting from these estimated parameters, it is possible to define the *Free Running Load Matrix* $FRL \in \mathbb{R}^{s \times n}$, where $frl_{k,j} = (t_{k,j} \cdot N_{k,j})/FRT_k$. $FRT_k/N_{k,j}$ is the average period of each processes ps_j on processor pu_k. By imposing that the simulated time shall be equal to *TTC*, it is possible to evaluate the Load $l_{k,j}$ that processes ps_j would impose to the SW processor pu_k to satisfy *TTC* itself. In fact, if $TTC \leq (FRT_k + OH_k)$, by defining x_k such that:

$$TTC = x_k \cdot (FRT_k + OH_k) \ \ with\ 0 \leq x_k \leq 1 \tag{6}$$

where OH_k is the overhead introduced by a given scheduling policy, the value of estimated load $l_{k,j}$ that a process imposes to processor pu_k to satisfy *TTC* is equal to:

$$l_{k,j} = \frac{(t_{k,j} \cdot N_{k,j})}{TTC} = \frac{(t_{k,j} \cdot N_{k,j})}{FRT_k} \cdot \frac{FRT_k}{TTC} = frl_{k,j} \cdot \frac{FRT_k}{TTC} = \frac{frl_{k,j}}{x_k} \cdot \frac{FRT_k}{(FRT_k + OH_k)} \tag{7}$$

From a DSE perspective, by considering the sum of the loads $l_{k,j}$ of all the processes allocated to a GPP/ASP pu_k, it is possible to check if the total imposed load is acceptable. Considering [25], the load upper bound is on the order of $\simeq$ 70%. In this work, the introduction of HPV-based SW partitions adds a second level scheduling, introducing hierarchical scheduling issues, so the new load upper bound became $\simeq$ 36% [26]. So, it is possible to define the *Load Index* as:

$$\begin{gathered} X_{L_i} = 1 - \mathrm{tr}\left[L \cdot ALL^i\right] = 1 - \frac{\sum_{k=1}^{s}\sum_{j=1}^{n} l_{k,j} \cdot all_{k,j}^i}{s}, ALL^i = \begin{cases} all_{k,j}^i = 1 & if\ ps_j\ \in\ pu_k \\ all_{k,j}^i = 0 & otherwise \end{cases} \\ L = \begin{cases} l_{k,j} = \frac{frl_{k,j}}{x_k} \cdot \frac{FRT_k}{(FRT_k + OH_k)} & if\ TTC\ \leq (FRT_k + OH_k) \\ l_{k,j} = frl_{k,j} & otherwise \end{cases} \end{gathered} \tag{8}$$

C. Cost Index

The ***Cost Index*** is a metric related to the cost $C = [c_1, .., c_b]$ associated to each bb_k considered in the specific IND_i (considering PU, MU and CU):

$$\begin{gathered} X_{C_i} = 1 - C \cdot ALL^i = 1 - \frac{\sum_{k=1}^{b} c_k \cdot all_k^i}{maxC}, maxC = size(BB) \cdot max(c_k),\ \forall k = 1..b \\ ALL^i = \begin{cases} all_k^i = 1 & if\ \exists\ ps_j\ :\ ps_j\ \in\ pu_k,\ \forall j = 1..n \\ all_k^i = 0 & otherwise \end{cases} \in \mathbb{R}^b \end{gathered} \tag{9}$$

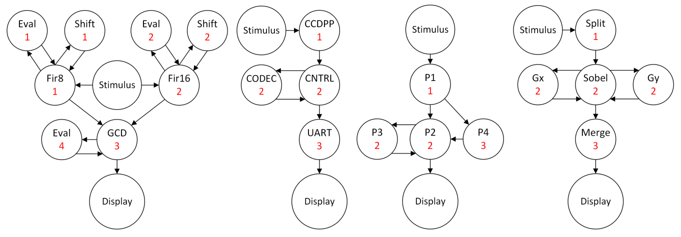
D. Criticality Index

The metric specifically introduced in [27] [28] and extended in this work to considers HPV-based SW partitions is the ***Criticality Index***, related to the criticality level associated to each process ps_j. In particular, defined the array $CRIT =$

$\{[crit_1, .., crit_n]\ :\ crit_j$ is the criticality level associated to process $ps_j\}$, then it is possible to define the *Criticality Index*:

$$X_{CRIT_i} = \begin{cases} 1 & if\ |crit_j - crit_k| > 0\ \wedge\ ps_j \in pu_x\ \wedge\ ps_k \in pu_y\ \wedge\ pu_x = pu_y \\ 1 & if\ |crit_j - crit_k| > 0\ \wedge\ ps_j \in pt_j \in pu_x \wedge\ ps_k \in pt_k \in pu_y\ \wedge\ pt_j = pt_k\ \wedge\ pu_x = pu_y \\ 0 & otherwise \end{cases} \tag{10}$$

The goal behind this metric is to avoid having processes with different criticality levels on the same (shared) partition/processor/core resource. If no HPV-based SW partitions are allowed, limiting the processes allocation taking into account MC has generally two main effects on the design space of feasible solution: to increase the minimum cost and to decrease the maximum execution time, because the number of BBs instances will not be less than the number of criticality levels (Fig. 3). The introduction of HPV-based SW partitions has two main effects on the same design space: to decrease the minimum cost and to increase the maximum execution time respect to the MC scenario without HPV partitons, because it is possible to use a number of BBs instances less than the number of criticality levels, increasing the number of feasible solution respect to criticality requirements (as shown in Fig. 3).



(a) Fir-Fir-GCD (b) Dig Cam (c) Simple CSP (d) Sobel Image

Fig. 2. CSP Use Cases Application Example.

V. Experimental Results

This section presents some experimental results related to the MC-aware DSE step with HPV-based SW partitions. Table I shows the parameters settings. In the context of this

TABLE I
DSE Parameter Settings

Parameters	Constraints	Values
BBs	$\leq$ 10	2 8051, 2 DSPIC, 2 LEON3, 2 Spartan3an, 2 Virtex-7
App. (Processes)	$\leq$ 8	CSP processes
App. (Channels)	$\leq$ 15	CSP channels
HPV-SW Partition	$\leq$ 4	HPV-based SW Partitions
GA Selection	1	Random
GA Crossover (C)	1	One-Point
C probability (pc)	1	0.3
GA Mutation (M)	1	Random
M probability (pm)	1	0.1
Survival Selection (S)	1	Fitness-Based
S probability (ps)	1	0.15
Search Iteration (I)	30	-
Initial Population (P)	100	# of Starting individuals
Max Population (P)	$\leq$ 1000	Max # of Final individuals

validation, the available BBs are: bb_1 16 MHz 8-bit ***8051*** CISC core with 128 byte of Internal RAM, 64KB of internal ROM (cost 10); bb_2 16 MHz 16-bit ***PIC24*** core with 14KB of internal ROM and 1KB of internal RAM (cost 20); bb_3 150 MHz 32-bit ***LEON3*** soft-processor with 2*4 KB L1 caches, RAM size of 4096 KB and a ROM of 2048 KB (cost 100); bb_4 50 MHz ***Spartan3an*** (cost 400); bb_5 250 MHz ***Virtex-7*** (cost 900). Considering AC, the maximum number of instances

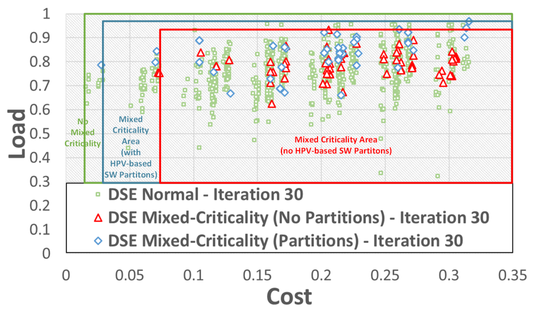


Fig. 3. Fir-Fir-GCD Pareto Set from the DSE respect to Load and Cost.

for each bb_i is 2, the maximum number of instances of bb_i considered into the DSE is equal to the number of processes ($\leq$ 8) and bb_i are supposed to communicate by means of a shared bus. The maximum number of SW HPV partitions that could be used by GPP bb_i are equal to 4.

The CSPs related to the reference use cases are shown in Fig. 2. In particular: a) *Fir-Fir-GCD* is a synthetic application that takes in input two values (triggered by *Stimulus*), makes two filtering actions (*Fir8* and *Fir16*) and then makes the greatest common divisor (GCD) and displays the result [29]; b) *Dig Cam* is an academic use case that represents a simple digital camera [30]; c) *Simple CSP* is a synthetic application that implements a simple data flow between stimulus trigger and final display [19]; d) *Sobel Image* is an application that performs the sobel filter on sample input image [31]; The red number under the name of each process represents its criticality level (values have been assigned in a synthetic way, mainly depending on number of channels and interactions among different processes). Fig. 3 shows some results from DSE step. Considering Load and Cost metrics, it is possible to note that the results without considering HPV-base SW partitions are slightly thickened respect to results with them. This is due to the possibility to find solutions and number of basic HW components less than the number of criticality levels. It is also possible to note that the pareto points follow a specific pattern (they are grouped into sub-sets that appears independent among each others). This behaviour could encourage the use of clustering methods into the GA steps in order to drive and find different solutions, increasing diversity into individual generation/mutation activities. Table II presents the feasible reduction percentage respect to the DSE without MC constraints and the comparison between MC with and without HPV-based SW partitions (in term of % of feasible pareto solution). From Table II it is possible to say that the introduction of HPV-based SW partitions into the Co-Design flow increases the number of feasible possible solutions (and

TABLE II
DSE PERCENTAGE REDUCTION OF FEASIBLE SOLUTION

Use Cases	AVG Red. No Part. - Nor.	AVG Red. Part. - Nor.	AVG No Part. - Part.
Fir-Fir-GCD	79%	68%	11%
Digital Camera	25%	14%	12%
Simple CSP	82%	70%	12%
Sobel Image	51%	39%	13%
AVG	59.25%	32.75%	12%

TABLE III
DSE MINIMUM (RELATIVE) COST ANALYSIS

Use Cases	Min No Part.	Min Part.	% Reduction No Part. - Part.
Fir-Fir-GCD	660	250	62.1%
Digital Camera	420	40	90.5%
Simple CSP	530	160	69.8%
Sobel Image	960	140	85.4%
AVG	642.5	147.5	76.95%

increases the number of possible different individuals into the GA algorithm) in a quantity $\simeq 12\%$ respect to no HPV-based SW partitions scenarios. Furthermore, it increases the number of feasible solutions respect to no MC scenarios in term of $\simeq 26\%$. Respect to final individual (relative) cost, Table III shows the minimum (Min) relative cost, found on the 30^{th} iteration, that demonstrates how the HPV-based SW partitions introduction improves detection of cheaper final HW/SW solutions from $\simeq 60\%$ to $\simeq 90\%$, meanwhile also in average (AVG) and maximum (Max) relative cost, MC DSE with HPV-based SW partitions seems better than no partitions ones.

VI. CONCLUSION AND FUTURE WORK

This work has proposed a criticality-driven DSE approach for mixed-criticality heterogeneous parallel embedded systems, considering HPV-based SW partitions into a HW/SW co-design flow. By introducing the *Criticality Index* into an evolutionary algorithm, the DSE is able to suggest solutions that fulfill constraints avoiding allocating tasks with different levels of criticality on the same shared resource. Results show that mixed-criticality solutions are typically more expensive, and this work helps to partition processes into a heterogeneous parallel platform. The introduction of HPV-based SW partitions into the DSE step improves the number of feasible solutions (increasing diversity and avoiding to remain in a local minimum point), and allows to find cheaper solutions for given mixed-criticality requirements. Future works will focus on the GA, by experimenting different selection, mutation and crossover techniques, comparing results with other meta-heuristic algorithms, also considering simulation time reduction to provide fast early co-design activities.

ACKNOWLEDGMENT

This work has been partially supported by the ECSEL RIA 2016 MegaM@Rt2 and AQUAS projects.

REFERENCES

[1] P. J. Prisaznuk, “Integrated Modular Avionics,” In Proc. of IEEE National Aerospace and Electronics Conference (NAECON), 1992.
[2] International Organization for Standardization (ISO): 46: 2011 Road VehiclesFunctional Safety, 2011.
[3] S. Baruah, H. Li, and L. Stougie, “Towards the Design of Certifiable Mixed-criticality Systems,” 16th IEEE RealTime and Embedded Technology and Applications Symposium (RTAS), 2010, pp. 13-22.
[4] R. Pellizzoni, P. Meredith, M. Y. Nam, M. Sun, M. Caccamo and L. Sha, “Handling mixed-criticality in SoC-based real-time embedded systems, In Proc. ACM international conference on Embedded software, 2009.
[5] J. Teich, ”Hardware/Software Codesign: The Past, the Present, and Predicting the Future,” In Proc. IEEE, vol. 100, no. Special Centennial Issue, 2012, pp. 1411-1430.
[6] S. Voss, J. Eder, F. Hlzl, ”Design Space Exploration and its Visualization in AUTOFOCUS3” Soft. Eng. (Workshops), 2014, pp. 57-66.
[7] F. Herrera, P. Pablo, and V. Eugenio, ”A model-based, single-source approach to DSE and synthesis of mixed-criticality systems” Proc. Int. Work. on Soft. and Comp. for Embedded Systems. ACM, 2015.
[8] V. Zaccaria et al., ”Multicube Explorer: An Open Source Framework for Design Space Exploration of Chip Multi-Processors” Int. Conf. on Arch. of Comp. Syst. 2010, Hannover, Germany, 2010, pp. 1-7.
[9] K. Rosvall et al., ”Throughput Propagation in Constraint-Based Design Space Exploration for Mixed-Criticality Systems” In Proc. Work. on Rapid Sim. and Perf. Eval.: Methods and Tools (RAPIDO ’17). ACM, New York, NY, USA, 2017.
[10] PikeOS Hypervisor, https://www.sysgo.com/products/pikeos-hypervisor/
[11] M. Masmano, I. Ripoll, A. Crespo and J. J. Metge, Xtratum: a hypervisor for safety critical embedded systems, In Proc. Real-Time Linux Workshop, 2009.
[12] Wind River Virtualization, https://www.windriver.com
[13] S. Trujillo, A. Crespo and A. Alonso, ”MultiPARTES: Multicore Virtualization for Mixed-Criticality Systems,” 2013 Eur. Conf. on Dig. Syst. Des., Los Alamitos, CA, 2013, pp. 260-265.
[14] Hepsycode, www.hepsycode.com
[15] L. Pomante, System-level design space exploration for dedicated heterogeneous multi-processor systems. Conference on Application-specific Systems, Architectures and Processors (ASAP), pp. 79-86, 2011.
[16] D. Di Pompeo, E. Incerto, V. Muttillo, L. Pomante, and G. Valente. An Efficient Performance-Driven Approach for HW/SW Co-Design. In Proc. 8th ACM/SPEC on International Conference on Performance Engineering (ICPE ’17). pages 323-326, ACM, 2017
[17] C. A. R. Hoare. Communicating sequential processes. Springer, New York, NY, 413-443. 1978
[18] SystemC, http://www.accellera.org
[19] V. Muttillo, G. Valente, D. Ciambrone, V. Stoico, and L. Pomante. HEPSYCODE-RT: a Real-Time Extension for an ESL HW/SW Co-Design Methodology. In Proc. 10th Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools (RAPIDO ’18), 2018.
[20] L. Pomante. HW/SW Co-Design of Dedicated Heterogeneous Parallel Systems: an Extended Design Space Exploration Approach. IET Computers & Digital Techniques, vol. 7, no. 6, pp. 246-254, Nov. 2013.
[21] L. Pomante, D. Sciuto, F. Salice, W. Fornaciari, and C. Brandolese. Affinity-Driven System Design Exploration for Heterogeneous Multiprocessor SoC. In IEEE Transactions on Computers, 55(5):508-519, 2006
[22] L. Pomante, G. Valente, V. Muttillo, HEPSIM: an ESL HW/SW Co-Simulator/Analysis Tool for Heterogeneous Parallel Embedded Systems, In 6th EUROMICRO/IEEE Workshop on Embedded and Cyber-Physical Systems (ECYPS 2018), 2018.
[23] V. Muttillo, G. Valente, L. Pomante, V. Stoico, F. DAntonio, and F. Salice, CC4CS: an Off-the-Shelf Unifying Statement-Level Performance Metric for HW/SW Technologies, In Companion of the 2018 ACM/SPEC International Conference on Performance Engineering (ICPE ’18), 2018, pp. 119-122.
[24] Abdullah Konak et al., Multi-objective optimization using genetic algorithms: A tutorial, Rel. Eng. & Syst. Saf., 2006, pp. 992-1007
[25] C.L. Liu, J.W. Layland. Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. Jour. ACM, pp. 37-53, 1973.

[26] S. Saewong, R. Rajkumar, J.P. Lehoczky, and M.H. Klein. Analysis of hierarchical fixed- priority scheduling. In Proc. of the 14th Euromicro Conference on Real-Time Systems (ECRTS), pages 173181, 2002

[27] V. Muttillo, G. Valente and L. Pomante. Criticality-aware Design Space Exploration for Mixed-Criticality Embedded Systems. In Proc. 9th ACM/SPEC on International Conference on Performance Engineering (ICPE '18). 2018.

[28] V. Muttillo, G. Valente and L. Pomante. Criticality-driven Design Space Exploration for Mixed-Criticality Heterogeneous Parallel Embedded Systems. In Proc. 9th Workshop on Parallel Programming and Run-Time Management Techniques for Many-core Architectures and 7th Workshop on Design Tools and Architectures for Multicore Embedded Computing Platforms(PARMA-DITAM '18). 2018.

[29] L. Pomante and P. Serri, SystemC-based HW/SW Co-Design of Heterogeneous Multiprocessor Dedicated Systems, International Journal of Information Systems, Volume 1, July 30, 2014.

[30] F. Vahid and T. Givargis, "Embedded system design: A Unified Hardware/Software Approach," Department of Computer Science and Engineering University of California, 1999.

[31] K. Rosvall and I. Sander, "A constraint-based design space exploration framework for real-time applications on MPSoCs," 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, 2014, pp. 1-6.