Abstract—This work faces the problem of the HW/SW co-design of dedicated systems based on heterogeneous parallel architectures. In particular, it proposes an extension of a previous system-level design space exploration (DSE) approach able to suggest to the designer an HW/SW partitioning of the application specification and a mapping of the partitioned entities onto an automatically selected heterogeneous multi-processors architecture where each processor could be also homogeneous multi-core. The extended modeling strategy and the description of the adopted heuristics/metrics represent the core of the paper.

Design Space Exploration; HW/SW Co-Design; Heterogeneous Parallel Architectures; Electronic System-Level

I. INTRODUCTION
Systems based on heterogeneous multi-processor architectures (HMPS, Heterogeneous Multi-Processor Systems) have been recently exploited for a wide range of application domains, especially in the System-on-Chip (SoC) form factor (e.g. [1]-[4]). Such systems include several processors, memories, and a set of interconnections between them. By definition, the set of processors in the same architecture is heterogeneous. This implies that it is possible to have (introducing a slight variation of the classification proposed in [13]), at the same time:
- COTS (Common-Off-The-Shelf) general-purpose processors (e.g. ARM, MIPS, MicroBlaze, NiosII, etc.): they execute a fixed standard Instruction Set Architecture (ISA)
- COTS domain-oriented processors (e.g. DSP, Digital Signal Processor; GPU, Graphical Processing Unit; etc...): they execute a fixed domain-specific ISA
- Custom domain-oriented processors (the so called ASIP, Application Specific Instruction Processor): they execute a customizable domain-specific ISA
- COTS single-purpose processors (SPP, e.g. AES coder, JPEG coder, UART/SPI/I2C Controller, etc...): they execute a standard specific function (no ISA involved)
- Custom single-purpose processor (SPP, the actual ad-hoc developed digital HW component): they execute a custom specific function (no ISA involved)

Such processors could be then adopted in the form of (soft, hard or fuse) IP core or as discrete IC mainly depending on the final system form factor (e.g. on-chip, on-FPGA, on-board) and scope (final product or platform).

As stated in the title, this work focuses on dedicated systems. In the scope of this paper, a “dedicated system” is a digital electronic system with custom HW/SW architecture. It is specifically designed to satisfy a priori known application requirements. A dedicated system could be embedded in a more complex system or it could be subjected to real-time constraints.

When dedicated systems are also HMPS (D-HMPS), they are so complex that the HW/SW co-design methodology plays a major role in determining the success of a product. In fact, in the past years, a remarkable number of research works have focused on system-level co-design of HMPS (e.g. [5]-[12]). Each of them has proposed a different approach to the design space exploration but always relying on some fixed architectural elements or the designer’s experience. So, at the best of our knowledge, there does not exist a system-level co-design flow that addresses the problem of automatically suggest an HW/SW partitioning of the system functionalities specification, while also mapping the partitioned entities onto an automatically selected heterogeneous multi-processor architecture, where each processor could be also homogeneous multi-core.

According to this scenario, this work proposes an extension of the research results described in [17][7][19]. The final goal is the extension of an existing design space exploration (DSE) approach that, starting from the system functionalities specification and related requirements, would be able to suggest to the designer:
- an HW/SW partitioning of the given system functionalities specification;
- a D-HMPS architecture;
- a mapping of the partitioned entities onto the proposed architecture able to satisfy imposed requirements.

The proposed extension allows to consider, in the same co-design flow, heterogeneous multi-processor architectures where each processor could be also homogeneous multi-core, as better described in Section IV.

The paper is organized as follows. Section 2 presents the reference design flow while Section 3 highlights the main modeling issues, specification languages and related “internal models” of representation. The extended reference target HW architecture (described in Section 4) and the description of the proposed heuristics and metrics for design space exploration (Section 5) represent the core of the paper with an illustrative example that allows to clarify the main features of the whole approach. Finally, Section 6 draws out some conclusions and outlines the future work.
The reference system-level co-design flow (a slight extension of the one in [19]) is shown in Figure 1: it reports the main co-design activities that are briefly described in the following.

**Specification**
The entry point of the proposed co-design flow is the specification of the system functionalities, with related timing constraints, by means of an homogeneous specification language to avoid polarizing the design towards hardware or software.

**Functional Simulation**
The first step is the Functional Simulation [17][18] where the system functionalities are simulated to check their correctness with respect to typical input data sets. During this step, important data characterizing the dynamic behavior of the system are also collected: Profiling, Communication and Concurrency.

**Co-Analysis and Co-Estimation**
This step of the flow aims at extracting information about the system by statically analyzing the specification. This step is composed of the Co-Analysis [7][19] and Co-Estimation [17][21][22][23] activities. The former one provides a set of data expressing the Affinity of each system functionality towards a set of processor classes (actually this work considers only COTS GPP, COTS DSP and Custom SPP). The latter provides a set of estimations of the Timing required by each processor class in the set to execute each single statement composing the specification. Finally, another estimation is related to the Size: ROM/RAM bytes needed for SW implementations and equivalent gate (or similar metrics such as number of cells or LUTs) for HW ones.

**Load and Bandwidth Estimation**
Combining data provided by previous steps (timing and profiling) under a time-to-completion constraint allows the estimation of the Load [17] associated with the execution of each system functionality on a single COTS GPP system (i.e. the worst case). Finally, combining communication and timing data it is possible to estimate the Bandwidth needed to the different functionalities to communicate while fulfilling a time-to-completion constraint.

**Design Space Exploration**
Finally, the flow reaches the Design Space Exploration step that is constituted by two iterative tasks: HW/SW Partitioning and Architecture Definition [7], and Timing Co-Simulation [17][18]. All the data produced in previous steps are used to drive the process, together with additional information provided by the designer. The partitioning and architecture definition task explores the design space (it is based on a genetic algorithm as described in section V) looking for feasible mapping/architecture items suitable to satisfy imposed constraints. Then, the timing co-simulation task considers the suggested mapping/architecture items to actually check for timing constraints satisfaction.

**Algorithm-Level Flow**
When the mapping/architecture item proposed by the design space exploration is satisfactory, it is possible to implement the system. For this, the SW system functionalities will be typically transformed in C code while the HW system functionalities will be transformed in synthesizable HDL code or implemented by means of existing COTS component. Moreover, if proper models are available (i.e. Instruction Set Simulators or HDL descriptions for processors, and bit-level or transaction-level models for interconnections) it is also possible to build a virtual prototype of the system to check its properties at a lower level of details prior to proceed with its physical realization.

**III. Specification**
The entry point of the proposed co-design flow is the specification of the desired system functionalities. This aspect introduce some modeling issues: the adopted model of computation (MoC), related specification languages, and the “internal models” of representation to be used to allow a proper tool-chain to make automatic analysis and transformations.

In this work, the system functionalities specification is assumed to be based on the CSP model of computation (Communicating Sequential Processes [14][24]) and described by means of an homogeneous specification language suitable to support such a MoC (e.g. OCCAM, HandelC, SystemC, etc...). Once selected a specification language, in order to automatically analyze the specification, two internal models at different levels of abstraction are needed:

- a statement-level internal model, that strictly depends on the chosen specification language and on parsers used to analyze the specification at a level detailed enough to make reliable estimations and to compute metrics (i.e. the second step of the flow in Figure 1);
- a procedure-level internal model, possibly independent from specification languages, used to explore the design space.

Since the main internal model for the design space exploration is the procedure-level one, the approach presented in this work is based on the Procedural Interaction Graph (PING [7]). The Procedural Interaction Graph is a formalism that provides information about the relationships among procedures and the
exchanged data. Such a graph is suitable for representing a coarse-grain view of system functionalities taking into account communications, synchronizations and concurrency issues. By means of a PING it is possible to represent a CSP considering also a possible decomposition of each process. Figure 2 shows a CSP (with 6 processes and 6 channels) and a possible PING where processes have been decomposed by means of procedures.

IV. TARGET HW ARCHITECTURE

The target HW architecture selected for the proposed extended methodology is an heterogeneous multi-processor one. In particular, each processor could be also homogeneous multi-core. The memory is supposed to be shared between cores in the same processor and distributed between processors. The whole architecture is so composed of proper interconnections of some instances of different Basic Block (BB, Figure 3). In particular, each BB is composed of three main elements: a set of Processing Units (PU), a Local Memory (LM) and an External Communication Unit (ECU).

The elements in the same BB share the LM and they are interconnected by an Internal Interconnection Link (IIL, typically a shared bus mastered by the PUs while the ECU is a memory-mapped SPP slave). It is worth noting that having more PUs in single BB allows to model shared-memory multi-core architecture while considering more than one BBs allow to model also distributed-memory multi-processor ones.

In the actual methodology, PUs could belong only to one of three different processor classes (i.e. COTS GPP, COTS DSP and Custom SPP) where GPP and DSP are characterized by the cost ($\€$) and the maximum load $L_{\text{MAX}}$ while SPP is characterized by the cost ($\€$) and the max number of equivalent gates $G_{\text{eq,MAX}}$ (in the case of a fixed family of reconfigurable logic the last metric should be changed with the max number of available cells or LUT). LM is the local memory directly addressable by the PUs belonging to the same BB and it is characterized by a cost ($\€$) and max size for data ($KBD_{\text{MAX}}$) and max size for code and parameters ($KBC_{\text{MAX}}$). ECU is characterized by the set of External Interconnection Links (EIL) that it is able to manage. Moreover, each EIL is characterized by the following parameters:

- the max available bandwidth ($BW_{\text{MAX}}$);
- the min/max number of BBs that should/could use an EIL instance ($N_{\text{min}}$ and $N_{\text{MAX}}$);
- the max number of allowed concurrent communications ($CC_{\text{MAX}}$);
- the cost ($\€$).

Finally, to consider the optimal number of PUs in a single BB, the Internal Interconnection Link is characterized by:

- the max available bandwidth ($BW_{\text{MAX}}$);
- the max number of PUs that could be connected to it ($N_{\text{MAX}}$).

So, given some instances of BBs and interconnecting them by means of some instances of EILs it is possible to define a feasible dedicated heterogeneous parallel architecture on which the system functionalities can be mapped. Such an architecture could be so represented by means of a hierarchical architecture graph [16] (as well as the BB in Figure 3).
V. DESIGN SPACE EXPLORATION

This section explains in detail the design space exploration approach proposed for the “HW/SW Partitioning and Architecture Definition” task of Figure 1, with the adopted heuristic, metrics and cost functions. The final goal is the automatic identification of an HW/SW partitioning of the system functionalities, an heterogeneous parallel architecture, and a mapping of the partitioned entities onto it, able to optimize the adopted cost function. The proposed approach is decomposed into two sequential phases, as shown in Figure 5.

A. First Phase

The first phase is mainly related to computation issues (the approach in [7] has been extended to consider the new target HW architecture). The internal-model representing the specification, annotated during the Co-Analysis&Co-Estimation step (Figure 1), is provided as input to the PAM1_v2 (i.e. Partitioning, Architecture Definition and Mapping Phase I) tool. In particular, each PING procedure is annotated with several metrics:

- load imposed by each procedure to a single COTS GPP under a time-to-completion constraint: \( l \);
- bandwidth needed to communicate with other procedures while fulfilling a time-to-completion constraint: \( b \);
- size for HW and SW implementations: \( s \) (KBD and KBC bytes, and Geq/cells/LUT);
- affinity of each procedure towards a set of processor classes (actually only COTS GPP, COTS DSP and Custom SPP): \( a \).

So, starting from an annotated PING (Figure 6 shows an annotated version of the one in Figure 2) the first phase goal is to determine number and type of BBs/PUs and a mapping of PING procedures onto them trying to:

- minimize the cost of the set of BBs
- The max number of instances allowed for each kind of PUs and the max number of PUs allowed in a single BB could be provided as an architectural constraints by the designer
- keep the load of each PU near but under its LMAX
- minimize communications between different BBs
- exploit the affinity between PUs and the procedures
- keep the used size near but under KBDMAX, KBCMAX (for GPP/DSP) or Geq_MAX/CellMAX/LUTMAX (for SPP)
- exploit the explicit parallelism expressed in the PING
- keep each IIL bandwidth under but near its BWMAX
- keep the number of PUs inside a single BB using the same IIL under but near its NMAAX

by minimizing a cost function by means of a genetic approach [15] where each individual of the population represents a possible mapping/architecture item. Such a cost function is composed of several terms related to the following system-level metrics [7] that are evaluated for each individual during the genetic evolution (a value of 0 indicates the best situation):

- Affinity Index (Ia): [0, 1]
  - Affinity between the features of the procedures and the processors on which they have been mapped to
- Load Indexes (I\( l \)): [0, 1]
- Communication Index (Ic): [0, 1]
- Physical Cost Index (Iw): [0, 1]
- Cost of the solution
- Size Indexes (IKBD, IGeq): [0, 1]
  - Memory (GPP/DSP) or resources (SPP) utilization
- Concurrency (Parallelism) Index (I\( b \)): [0, 1]
  - Exploitation of the concurrency expressed in the CSP
- Internal Saturation Index (\( I_{\text{Sat}} \)): [0, 1]
  - Respect of the max bandwidth offered by each IIL
- Internal Exploitation Index (\( I_{\text{Exp}} \)): [0, 1]
  - Respect of the max number of PUs inside each BB that can use an IIL instance

By combining the previous indexes a linear cost function has been built to compare different mapping/architecture item:

\[
CF = \sum w_l \cdot l + \sum w_c \cdot c + \sum w_e \cdot e + \sum w_a \cdot a + w_{\text{Ia}} \cdot I_a + w_{\text{Ik}} \cdot I_{\text{KBD}}
+ w_{\text{Geq}} \cdot I_{\text{Geq}} + w_{\text{IP}} \cdot I_{\text{Parallel}} + w_{\text{IB}} \cdot I_{\text{BW}} + w_{\text{IE}} \cdot I_{\text{IE}}
\]
where the weights $w$ (belonging to the interval $[0,1]$) are used to identify individuals that better tradeoffs different parameters. The structure of an individual is represented by an entry for each PING procedure: each procedure is associated with a type of processor, a processor instance number, and a core instance number. Figure 8 shows a possible individual instance with its corresponding mapping/architecture for the PING of Figure 6. In general, the mapping could be formally expressed as done in [16]. The initial population is randomly generated and during its evolution the algorithm minimizes the cost function following the classical rules of genetic algorithms. In particular, each crossover operation generates two new individuals combining two existing ones. The mutation operation changes randomly type and/or instance number of a processor (or a core) associated with a randomly selected procedure. During the evolution, the individuals that score the worst values tend to be replaced by better ones. Several parameters in the algorithm (e.g. population size, number of generations, mutation probability, etc…) allow a wide exploration of the design space, with the goal to avoid local minima. It is worth noting that, to avoid complexity explosion while coping with large system specifications, it is possible to properly take into account: unfeasible solutions will get an upper bound could be freely fixed by the designer to penalize unfeasible interconnections/topology item) are used to identify individuals that better tradeoffs different parameters. The structure of an individual of the population is represented by a triangular matrix with number of rows and columns equal to the number of BBs. Each couple of communicating BBs is associated with an instance of an EIL. For example, the individual of Figure 7 represents an architecture with 4 BBs that communicates by means of an instance of IL2 while an instance of IL1 directly communicates issues as in [19]. We obtain form the previous indexes a linear cost function has been built to compare different interconnections/topology item: 

\[
CF = w_8 \cdot I_B + w_7 \cdot I_E + w_6 \cdot I_C + w_5 \cdot I_C + w_6 \cdot I_C + w_7 \cdot I_F
\]

where the weights $w$ (belonging to the interval $[0,1]$) except for $w_7$ which upper bound could be freely fixed by the designer to penalize unfeasible interconnections/topology item) are used to identify individuals that better tradeoffs different parameters. The starting point of the second phase is the so called BBs Interaction Graph (BING, [19]), i.e. an internal model used to represent the partial system obtained at the end of the first phase. Such a model is provided as input to the PAM2 (i.e. Partitioning, Architecture Definition and Mapping Phase 2) tool (Figure 5). So, starting from a BING the goal is to determine number and type of EILs between BBs in order to:

- maximize cost of the set of EILs
- The max number of instances for each EIL can be specified by the designer as an architectural constraints
- keep the bandwidth of each EIL under but near BWMAX
- keep the number of BB using each EIL under but near NMAX
- keeping feasibility by respecting ECUs characterization
- by minimizing a cost function by means of a genetic approach where each individual of the population represents a possible interconnections/topology item. Such a cost function is composed of several terms related to the following system-level metrics that are evaluated for each individual during the genetic evolution (a value of 0 indicates the best situation):

- **Saturation Index (I_B)**: $[0, 1]$
  - Respect of the max bandwidth offered by the EIL
- **Exploitation Index (I_E)**: $[0, 1]$
  - Respect of the min/max number of BBs that can use an EIL instance
- **Physical Cost Index (I_P)**: $[0, 1]$
  - Cost of the solution
- **Concurrent Communications Index (I_C)**: $[0, 1]$
  - Respect of the max number of concurrent communications allowed by the EIL

\[
\begin{array}{c|c|c|c|c|c}
\hline
\text{BB1} & \text{IL2.1} & \text{IL2.1} & \text{IL2.1} & \text{IL2.1} & \text{IL2.1} \\
\hline
\text{BB2} & 0 & 0 & 0 & 0 & 0 \\
\hline
\text{BB3} & 0 & 0 & 0 & 0 & 0 \\
\hline
\text{BB4} & 0 & 0 & 0 & 0 & 0 \\
\hline
\end{array}
\]

Figure 7. An example of individual and corresponding architecture graph

- Feasibility Index ($I_F$): $[0, 1]$
  - A pair of ECUs should be able to manage at least a common IL in order to allow the related BBs to directly communicate: such an index indicates how much of the actual communications are unfeasible

By combining the previous indexes a linear cost function has been built to compare different interconnections/topology item:

\[
CF = w_8 \cdot I_B + w_7 \cdot I_E + w_6 \cdot I_C + w_5 \cdot I_C + w_6 \cdot I_C + w_7 \cdot I_F
\]

In order to perform the exploration of the design space the initial population is randomly generated, while during its evolution the algorithm follows the classical rules of genetic algorithms. Unfortunately, such an approach could give rise to unfeasible solutions so the feasibility of the children has to be checked [19] and properly took into account: unfeasible solutions will get an higher cost function and so they will be probably removed from the population. However, there is always the possibility of a mutation process giving rise to better individuals that otherwise could be difficult to obtain (i.e. avoiding local minimum).

C. Illustrative Example

In order to clarify the main features of the whole approach with respect to the proposed extension an illustrative example is reported in the following (more details could be found in [7] and [19]). Let be the CSP specification the one shown in Figure 2 and the related annotated PING the one shown in Figure 6. Let be the annotated values of $a$, $l$, $s$, and $b$ such that the PAM1_v2 tool would provide as output of the first phase the mapping/architecture shown in Figure 8. Now, let be all the communications issues as in [19]. We obtain form PAM2 the suggested mapping/architecture shown in Figure 9.

VI. CONCLUSIONS

This paper has coped with the problem of hw/sw co-design of dedicated digital systems based on heterogeneous multi-processor architectures. In particular, it has proposed an extension of a
previous system-level design space exploration approach able to suggest to the designer an HW/SW partitioning of the application specification and a mapping of the partitioned entities onto an automatically selected heterogeneous multi-processors/multi-core architecture. The approach is currently under validation by means of proper tools by checking the consistency of the adopted metrics and heuristics. The preliminary experimental results are encouraging and justify further research efforts in this direction.

VII. ACKNOWLEDGEMENTS

This work has been partially supported by the ARTEMIS projects PRESTO (GA n. 269362) and CRAFTERS (GA n. 295371).

REFERENCES