Multi-Variant-based Design Space Exploration for Automotive Embedded Systems

Sebastian Graf, Michael Glaß, and Jürgen Teich
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany
Email: sebastian.graf, glass, teich}@cs.fau.de

Christoph Lauer
AUDI AG Ingolstadt, Germany
Email: christoph.lauer@audi.de

***Abstract*—This paper proposes a novel design method for modern automotive electrical and electronic (E/E) architecture component platforms. The addressed challenge is to derive an optimized component platform termed *Baukasten* where components, i. e., different manifestations of Electronic Control Units (ECUs), are reused across different car configurations, models, or even OEM companies. The proposed approach derives an efficient graph-based exploration model from defined functional variants. From this, a novel symbolic formulation of multi-variant resource allocation, task binding, and message routing serves as input for a state-of-the-art hybrid optimization technique to derive the individual architecture for each functional variant and the resulting Baukasten at once. For the first time, this enables a concurrent analysis and optimization of individual variants and the Baukasten. Given each manifestation of a component in the Baukasten induces production, storage, and maintenance overhead, we particularly investigate the trade-off between the number of different hardware variants and other established design objectives like monetary cost. We apply the proposed technique to a real-world automotive use case, i. e., a subsystem within the safety domain, to illustrate the advantages of the multi-variant-based design space exploration approach.**

I. Introduction

Modern automotive systems consist of a tremendous amount of applications that are implemented as a networked embedded system based on up to 100 Electronic Control Units (ECUs) and numerous sensors and actuators, all connected via several field buses. While the sheer problem complexity of the E/E architecture of a specific car has gained much attention (e. g., in system design [1], [2], new communication networks [3], or enhanced diagnosis capabilities [4]), existing design automation approaches have largely neglected the challenging task of definition and management of all different *manifestations* of the involved components within the overall series of different car models. Consider only the safety architecture as an example. The varying number of ignition circuits which depends on the number and kind of installed airbags already creates several manifestations of a single ECU. More than ten years ago, handling these degrees of freedom was not a main issue because each functionality was mapped to dedicated hardware components and, thus, functionality was integrated or configured in a car by just mounting the respective components. Today, model-driven software development together with a hardware-independent distributed implementation (cf. AUTOSAR [5]), powerful general purpose ECUs and high-bandwidth communication links enable the implementation of functionalities on

978-3-9815370-2-4/DATE14/©2014 EDAA

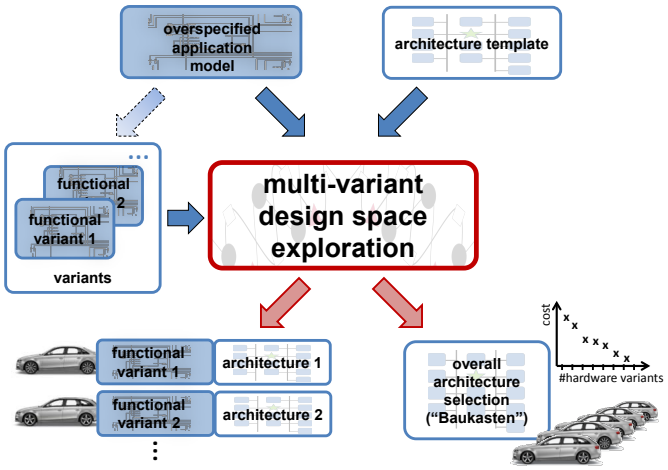

Figure 1. The proposed multi-variant-based design space exploration: Given are a classic architecture template as well as a novel *overspecified application model* from which designers constitute concrete *variants*. The novel combined optimization delivers (a) concrete architectures for each defined variant and (b) the resulting overall architecture, i. e., the *Baukasten*. This enables to evaluate and optimize both, the objectives (cost, timing, etc.) of individual variants and the Baukasten (overall cost, number of hardware variants, etc.) concurrently.

shared resources. Thus, most of the features and functions are no longer exclusively mappable to one specific hardware resource; also, a hardware resource is not only developed for the implementation of a single set of functions. As a measure against the ever-growing cost pressure, the automotive industry nowadays combines not only different features selectable by the customer for one car model, but even the architectures and features of different car models into one combined flexible E/E architecture; the *Baukasten* (engl. component platform). The resulting re-usability of components and subsystems helps to lower the number of different architecture manifestations and, thus, to lower the cost of developing, managing, maintaining, and storing different architectures. But, these cost savings are bought by trading-off highly-optimized specific architecture variants for variants that re-use the Baukasten components, inducing a possible over-dimensioning.

Existing automatic *design space exploration* (DSE) approaches already struggle with various degrees of freedom like assimilable computation resources (different processors, ASICs, FPGAs, etc.) and communication controllers for numerous field bus systems when performing system synthesis. But, they only target one variant (e. g. for maximum functionality) and gather one optimized architecture. This not only fails to assist Baukasten developers, it also neglects the possible

re-use and synergy effects for components across various car models neglecting for example that some redundant parts in an ECU may provide a broad applicability and avoid numerous additional ECU versions to be maintained.

As a remedy, the work at hand proposes an approach for automatic *multi-variant-based design space exploration* as depicted in Fig. 1. The key feature of this technique is that it (a) starts with one consistent overall application model, cf. [6], from which designers can seamlessly define functional variants, (b) defines a 0-1 Integer Linear Program (ILP) formulation that performs system synthesis for all defined variants concurrently employing a state-of-the-art hybrid optimization technique, and (c) delivers both concrete architectures for each defined variant and the resulting overall Baukasten. This holistic approach with its *variant layer* particularly enables to automatically analyze and optimize the trade-off of the quality of individual variants and the respective overall Baukasten w.r.t. the previously outlined aspects of performance numbers and arising costs in a multi-objective fashion. To the best of our knowledge, the presented approach is the first tackling this kind of optimization problem as it regularly appears in the design phase of new car series within the automotive domain.

The rest of the paper is structured as follows: Section II discusses related work while Sec. III introduces required fundamentals for the employed overall system model and design space exploration. In Sec. IV, the proposed multi-variant-based DSE model, the 0-1 ILP encoding for the concurrent synthesis of all variants and the Baukasten is introduced. Section V proposes how to evaluate the Baukasten and the individual variant's architectures and how to use the results as design objectives. Section VI presents a real-world use case of a sub-domain of the safety architecture that was optimized and evaluated using our presented multi-variant approach. Section VII concludes the paper.

II. Related Work

In recent years, especially in the automotive domain, architectural design decisions gained importance to the overall system design. As the overall search space is very large, automatic design space exploration approaches as presented in [7], [1] become familiar, but are not yet established in general. Moreover, several tools and approaches for platform-based system-level design like MESCAL [8], Metropolis [9], SCE [10], Sesame [11], and SystemCoDesigner [12] have been proposed. But, most of these methodologies for *system-level design*, typically targeting SoC or MPSoC platforms, try to decouple the functional description from the hardware implementation and, thus, to find one optimized architecture for a specific functional input. But nowadays, as already pointed out in the introduction, typically multiple different functional variants have to be implemented on the same hardware architecture and, thus, this modeling and optimization always has to consider the overall maximum functionality to be implemented on the system hardware architecture to ensure the correct usage. This neglects the architecture design and optimization to handle multiple but alternative functional variants. Here, our approach integrates the possibility to model multiple functional variants explicitly and integrates them in a design space exploration approach to optimize a systems architecture for multiple variants.

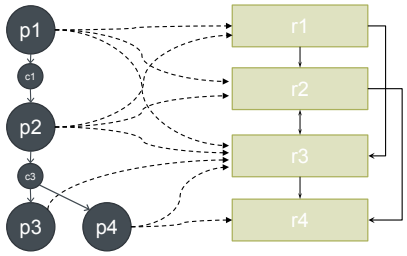

Figure 2. Classic model of the design space exploration. The functional model (left side of the figure) and the architecture template (right side of the figure), both represented by graph-based models, are linked by mapping edges.

Some approaches already try to tackle the handling of multiple different functional variants in case of multi-mode systems (see [13]) or different scenarios (as in [14]). They tackle the problem of further optimizing the architectures extracted by a DSE to better fit the real-world behavior of the overall functionality. But typically, the result of the DSE is to extract only one specific architecture implementing all possible variants of the functionality. This neglects to hold multiple architecture variants in a Baukasten to implement only subsets of the functionality and, thus, to add hardware variants as an additional layer of freedom to the designer. To reach this goal, our work proposes a *multi-variant-based design space exploration* with the additional capability to extract a hardware Baukasten implementing a set of subsets of functionality in case of pre-defined functional variants. This allows the architectural designer to trade-off between the number of different hardware variants (e. g. the size of the Baukasten) and common objectives as already known from DSE approaches.

III. Fundamentals

Our DSE approach follows the Y-Chart approach as proposed, e. g., in [15]. Its basic model separates the functionality from the architecture and allows to perform system synthesis, i. e., resource allocation, task binding, and message routing.

The basic system model, see [7], termed *specification* is a graph-based model comprising an *application graph* G_T modeling the system's functionality, an *architecture graph* G_R representing the overall architecture template, and *mapping edges* E_M between functionality and architecture. The application is modeled as a bipartite graph $G_T = \{T, E_T\}$: Each vertex $t \in T = P \cup C$ is either a functional task $p \in P$ or a message $c \in C$. Each directed edge $e \in E_T$ connects a function in P with a message in C and vice versa. Whereas a function can have multiple preceding and succeeding messages, each message has exactly one sender but may have multiple receivers, implementing a dataflow with multicast communication semantics. The architecture design space is spanned by an overspecified template of possible hardware components and represented by a directed graph $G_R(R, E_R)$. Each vertex $r \in R$ represents a hardware resource like an ECU, sensor, actuator, or a communication controller or field bus. The set of directed edges $E_R \subseteq R \times R$ represents commu-

nication connections between adjacent resources and, in this case links between hardware components. The mapping edges $E_M \subseteq P \times R$ specify on which resources each function may be implemented, i. e., each mapping edge $m = (p, r) \in E_M$ represents a possible implementation of task p on resource r.

Given such a model, a hybrid optimization approach is proposed in [16] where the specification and system synthesis are formulated as a 0-1 ILP, efficiently solvable by a Pseudo-Boolean (PB) solver. A meta-heuristic optimization algorithm, e. g., an evolutionary algorithm, is then employing the solver and performing a multi-objective optimization by influencing the solver's search strategy, gathering different system implementations, respectively. In the following section, we introduce our novel multi-variant-based design space exploration that extends the introduced basic system model to a multi-variant specification. Moreover, although following the principles of [16], we formulate a new 0-1 ILP that solves the system synthesis problem for all specified variants to derive optimized variant implementations and an optimized Baukasten concurrently.

IV. Multi-Variant-based Design Space Exploration

In the following, we present the proposed system model, capable of considering multiple functional variants within one single system specification. Afterwards, we introduce an efficient 0-1 ILP encoding to be employed in a state-of-the-art hybrid optimization technique, cf. [16], [7], now capable of concurrently synthesizing all defined variants within one single DSE.

A. Multi-Variant Model

Our first proposed extension for a multi-variant-based DSE is to extend the graph-based specification to

- integrate multiple different application models, each representing a functional variant $v \in V$, forming
- an overspecified application model that combines all specified variants and represents the overall functional model.

This results in an extension of a specification to a *multi-variant specification* that is a tupel $(\mathcal{G}_T, G_o, G_R, E_M)$. $\mathcal{G}_T = \{G_T^1, G_T^2, G_T^3, ..., G_T^{|V|}\}$ contains a set of pre-defined application graphs $G_T^v = (T^v = (P^v \cup C^v), E_T^v) \in \mathcal{G}_v$ for each variant v. $G_o = \{T_o, E_o\}$ with $T_o = P_o \cup C_o$ is the *overspecified application model*, given by the union (overall hull) $T_o = \bigcup_{v \in V} T^v$ and $E_o = \bigcup_{v \in V} E_T^v$ of all variants' application graphs $G_T^v \in \mathcal{G}_T$. Note that E_M and G_R are not specific to a variant v, thus, requiring no changes to the basic system model.

B. Model Encoding / System Synthesis

The main contribution of our work is the seamless integration of the proposed *multi-variant specification* in a design space exploration to realize a concurrent optimization of multiple variants and obtain a Baukasten which is to select a set of subsets of resources $R_B \subseteq R$ that will be used in the architecture of one or more variants. The task of system synthesis within *multi-variant-based design space exploration* now comprises the following tasks:

- Select a subset of resources $R_B \subseteq R$ that defines the overall Baukasten and is used to define alternating components of the Baukasten.
- Select a subset $R^v \in R_B$ of the Baukasten resources R_B for each variant $v \in V$, forming the architecture for variant v.
- For each variant v, bind each task $p \in P^v$ to exactly one resource of the variant's architecture R^v.
- For each variant v, route all messages $c \in C^v$ according to the computed binding and the topology of the variant's architecture R^v.

As outlined, this novel synthesis problem is formulated symbolically as a 0-1 ILP using the following binary variables, all given in boldface:

- $\boldsymbol{r}$ - indicating whether a resource $r \in R$ is included in the Baukasten (1) or not (0).
- $\boldsymbol{r}_v$ - indicating whether a resource $r \in R$ is included in the architecture R^v of variant v (1) or not (0).
- $\boldsymbol{p}_r$ - one variable for each mapping of a task $p \in P_o$ to a resource $r \in R$, indicating the task is mapped to resource r (1) or not (0).
- $\boldsymbol{p}_{r,v}$ - one variable per mapping of a task $p \in P^v$ of each variant $v \in V$ to a resource $r \in R^v$, indicating the task in this variant is mapped to resource r (1) or not (0).
- $\boldsymbol{c}_r$ - a set of variables for each message $c \in C_o$ and the resources $r \in R$, indicating whether the message might be routed over resource r (1) or not (0).
- $\boldsymbol{c}_{r,v}$ - a set of variables for each message $c \in C^v$ and the resources $r \in R^v$, indicating whether the message is routed over the resource r in variant v (1) or not (0).
- $\boldsymbol{c}_{r,h,v}$ - additional variables for each variant $v \in V$, indicating on which communication step $h \in \boldsymbol{N}$ a message $c \in C^v$ is routed over resource $r \in R$.

In the following, we present the constraints of the 0-1 ILP according to the two concurrent synthesis steps of generating the overall Baukasten and determining the allocation, binding, and routing for each variant.

1) Baukasten Constraints: The generation of the overall Baukasten is given follows:

$\forall p \in P_o$:

$$\sum_{r \in R} \boldsymbol{p}_r \geq 1 \tag{1a}$$

$\forall p \in P_o, r \in R$:

$$\boldsymbol{r} - \boldsymbol{p}_r \geq 0 \tag{1b}$$

$\forall c \in C_o, r \in R$:

$$\boldsymbol{r} - \boldsymbol{c}_r \geq 0 \tag{1c}$$

$\forall r \in R$:

$$-\boldsymbol{r} + \sum_{c \in C_o} \boldsymbol{c}_r + \sum_{p \in P_o} \boldsymbol{p}_r \geq 0 \tag{1d}$$

For the overall Baukasten, Eq. (1a) ensures that each task p of the overspecified application model has at least one active mapping. Equation (1b) guarantees that a mapping p_r for task p can only be active if the corresponding resource r is

included in the Baukasten allocation. Equation (1c) ensures that messages are only routed over allocated resources. To remove unused resources, Eq. (1d) forces each active resource r to be used by at least one active mapping p_r or routing c_r.

2) Variant-Specific Constraints: The addition of the following individual variant-specific constraints induces that each variant $v \in V$ is uniquely implemented. For the sake of simplicity, we first introduce per-variant and then Baukasten-related variant constraints. The following linear constraints are generated for each variant $v \in V$:[1]

$\forall p \in P^v$:

$$\sum_{r \in R} \boldsymbol{p}_{r,v} = 1 \tag{2a}$$

$\forall p \in P^v, \forall r \in R$:

$$\boldsymbol{r}_v + \overline{\boldsymbol{p}_{r,v}} \geq 1 \tag{2b}$$

$\forall p \in P^v, \forall r \in R, \forall c \in C^v : (c,p) \in E_T^v$:

$$\boldsymbol{c}_{r,v} + \overline{\boldsymbol{p}_{r,v}} \geq 1 \tag{2c}$$

$\forall c \in C^v, \forall r \in R$:

$$\boldsymbol{r}_v + \overline{\boldsymbol{c}_{r,v}} \geq 1 \tag{2d}$$

$$\sum_{h=\{0,..,n-1\}} \boldsymbol{c}_{r,h,v} - \boldsymbol{c}_{r,v} \geq 0 \tag{2e}$$

$$\sum_{h=\{0,..,n-1\}} \boldsymbol{c}_{r,h,v} \leq 1 \tag{2f}$$

$\forall c \in C^v, \forall r \in R, h = \{0,..,n-1\}$:

$$\boldsymbol{c}_{r,v} - \boldsymbol{c}_{r,h,v} \geq 0 \tag{2g}$$

$$\sum_{\tilde{r} \in R, (\tilde{r},r) \in E_R^v} \boldsymbol{c}_{\tilde{r},h,v} - \boldsymbol{c}_{r,h+1,v} \geq 0 \tag{2h}$$

$$\sum_{\tilde{r} \in R, (r,\tilde{r}) \in E_R^v} \boldsymbol{c}_{\tilde{r},h+1,v} + \overline{\boldsymbol{c}_{r,h,v}} \geq 1 \tag{2i}$$

$\forall c \in C^v, \forall r \in R, p \in P^v, (p,r) \in E_M, (p,c) \in E_T^v$:

$$\boldsymbol{c}_{r,0,v} - \boldsymbol{p}_{r,v} = 0 \tag{2j}$$

$\forall c \in C^v, \forall r \in R, p \in P^v, (p,r) \notin E_M, (p,c) \in E_T^v$:

$$\overline{\boldsymbol{c}_{r,0,v}} \geq 1 \tag{2k}$$

$\forall c \in C^v$:

$$\sum_{\forall r \in R, p \in P^v, (p,r) \in E_M, (p,c) \in E_T^v} \boldsymbol{c}_{r,0,v} = 1 \tag{2l}$$

Equation (2a) ensures that each task $p \in P^v$ is mapped to exactly one resource, whereas Eq. (2b) ensures that a resource is included in the variant allocation if a mapping to the resource is active. Equation (2c) enforces that each incoming message is routed to the resource its receiving tasks $p \in P^v$ is mapped to. Equation (2d) ensures that a message can only be routed on resources included in the architecture of the variant. Equation (2e) forces a message to be active only on the route, i. e., it is a hop and Eq. (2f) eliminates cycles in the route. If a resource is a hop of a route, the message on this resource is activated, ensured by Eq. (2g). Equations (2h) and (2i) make sure the contiguous hops on a route are adjacent resources. Equation (2j) ensures that only a sending resource is allowed to be the first hop of a message routing. Finally, Eq. (2k) prevents the first hop to be located on a non-sending resource and Eq. (2l) ensures each message has exactly one sender.

The constraints are completed by linking individual variants and Baukasten together, by means of the following constraints:

$\forall p \in P_o, r \in R$:

$$|V| \cdot \boldsymbol{p}_r - \sum_{v \in V : p \in P^v} \boldsymbol{p}_{r,v} \geq 0 \tag{3a}$$

$$\boldsymbol{p}_r - \sum_{v \in V : p \in P^v} \boldsymbol{p}_{r,v} \leq 0 \tag{3b}$$

$\forall r \in R$:

$$|V| \cdot \boldsymbol{r} - \sum_{v \in V} \boldsymbol{r}_v \geq 0 \tag{3c}$$

$$\boldsymbol{r} - \sum_{v \in V} \boldsymbol{r}_v \leq 0 \tag{3d}$$

$\forall c \in C_o, r \in R$:

$$|V| \cdot \boldsymbol{c}_r - \sum_{v \in V, c \in C^v} \boldsymbol{c}_{r,v} \geq 0 \tag{3e}$$

$$\boldsymbol{c}_r - \sum_{v \in V, c \in C^v} \boldsymbol{c}_{r,v} \leq 0 \tag{3f}$$

$\forall c \in C_o, \forall r \in R, \forall v \in V, c \in C^v$:

$$\boldsymbol{c}_r - \boldsymbol{c}_{r,v} \geq 0 \tag{3g}$$

First, it is ensured that each mapping (Eq. (3a)), resource (Eq. (3c)), and communication (Eq. (3e)) is activated in the overall Baukasten R_B if it is activated in at least one defined variant $v \in V$. To prevent the Baukasten from containing unused components, Equations (3b), (3d), (3f), and (3g) ensure that mappings, resources, and communications are deactivated in the Baukasten if they are not activated in at least one variant. These constraints achieve consistency between Baukasten and the individual architectures of each defined variant at all times.

Solving the introduced constraints via a standard PB solver directly delivers an overall Baukasten allocation and respective architectures for each variant as follows: The overall Baukasten allocation R_B is an induced subgraph of G_R, given by the activated variables $\boldsymbol{r} = 1$. For each variant $v \in V$, R^v is an induced subgraph of G_R by the activated variables $\boldsymbol{r}_v = 1$ while the task binding and message routing is directly deduced from the activated variables $\boldsymbol{p}_{r,v} = 1$ and $\boldsymbol{c}_{r,h,v} = 1$. Whereas, in case of a design space exploration of a single ECU, each unique allocation of resources for variants that might be reused in other variants is equivalent to one component in the Baukasten of the E/E architecture, the task binding and message routing might be specific per variant.

[1] n is set to the architecture network diameter or to a maximum number of hops that a message is allowed to pass on its route.

V. Multi-Variant Evaluation

The success of a multi-objective DSE also depends on the proper definition of the relevant design objectives. Thus, to extract proper objectives for the multi-variant-based DSE, we have to consider the evaluation of the overall Baukasten, the evaluation of the individual variants' architectures, as well as a system-wide view, bringing both aspects together.

A. Baukasten Evaluation

The main goal of the Baukasten evaluation is to rate the whole multi-variant implementation. Thus, we propose to use the overall Baukasten allocation, as well as each variant allocation to extract at least the following two objectives:

a) Rating of the Baukasten allocation: One objective is extracted from the overall Baukasten allocation and is derived from each included element. As in the automotive domain monetary cost is one very important issue and, thus, typically should be minimized, we suggest to minimize the cost o_b of the overall Baukasten by the following objective function with $\text{cost}(r)$ supplying the cost of one resource r:

$$o_b = \sum_{\forall r \in R_B} \text{cost}(r) \tag{4}$$

b) Rating of the hardware variants: The next proposed objective o_v represents the number of different allocations within all variants V and, thus, gives the number of different hardware variants required to implement all defined functional variants $v \in V$. Moreover, a smaller value is an indicator for more re-usage of hardware variants. It is determined by:

$$\mathcal{R}^v = \bigcup_{v \in V} R^v \tag{5}$$

$$o_v = |\mathcal{R}^v| \tag{6}$$

Here, the set union operation returns $\mathcal{R}^v$ that comprises all unique allocations required to implement all defined variants $v \in V$. Each unique allocation corresponds to one individual hardware variant that is included (and, hence, produced, stored, and maintained) in the Baukasten. It holds that $o_v = [1, |V|]$, i. e., the minimum is one hardware variant that is capable of implementing all functional variants and at maximum one specific hardware variant for each functional variant.

Additional to these two criteria, as o_v is typically very course grained, we suggest to use one guide objective o_g to further enhance the optimization process to solutions with a small number of hardware variants. As this typically results in an optimization contrary to o_b as additional unused resources have to be added to merge two allocations to one hardware variant, we use the overall difference from all variants' allocations to the overall Baukasten allocation as objective.

$$o_g = \sum_{\forall v \in V} (|R_B| - |R^v|) \tag{7}$$

B. Individual variant evaluation

Of course it is still possible to evaluate each single variant $v \in V$ individually and, thus, reuse existing evaluators like cost, load, or timing evaluators for each variant. But, this results in a very large number of objectives and, thus, aggravates the overall optimization process. We propose to use a condense evaluator that is able to combine the result $o_i^v \geq 0$ of an individual evaluation of each variant $v \in V$, to a single optimization criterion o_i. Additional to this, e. g., to represent different installation rates, a weighting factor $w_v \in W$ for each variant v is included.

$$o_i = \sum_{\forall v \in V} (w_v \cdot o_i^v) \tag{8}$$

For our real-world use case, each individual objective o_i^v (as monetary cost of the architecture) is equally involved in the combined objective, thus, the weighting factor is assumed to be 1 for all $w_v \in W$. In case of other circumstances, this can be extended to consider maintenance cost or management overhead, but is not further discussed in this paper.

VI. Experimental Results

The proposed multi-variant-based design space exploration methodology is applied to a real-world case study from the automotive domain, in particular, the design of a new safety architecture for future car series. The existing degrees of freedom with respect to architectural decisions mostly stem from different processors and on-board peripherals for airbag ECUs, different ASIC and FPGA components for the connection of external sensors and actuators, as well as additional smart camera systems connected to the processors via field buses. The resulting architecture template consists of about 120 different interconnected resources. Without loss of generality, we assume the whole architecture to be assembled in one monolithic ECU and, thus, each hardware variant is directly related to one manifestation of the airbag ECU.

The functionality is explicitly modeled as different functional variants $v \in V$ as given in Table I.[2] It consists of several features related to the safety domain that are additionally selectable in different variants or series of cars, respectively. To gather the best-case in monetary cost, we performed an individual design space exploration for each variant and, thus, explored specific manifestations of the architecture.

For the multi-variant-based design space exploration, we used the objectives o_b, o_v, o_g, and o_i (with equal weighted monetary cost), see Section V, to optimize the architecture and the Baukasten for all 8 predefined functional variants. The best-case for overall cost is to implement all variants on their individually optimized architecture and, thus, to maintain 8 different hardware variants. For our example, this gives the theoretical minimum cost of 435.64 as the sum of all individual per-variant best-case monetary cost. As depicted in Fig. 3 (x_8), our approach delivers this solution and, thus, has at least the same capability as individual per-variant DSE

[2]For reasons of secrecy, we cannot give specific details on the individual functional features and real costs.

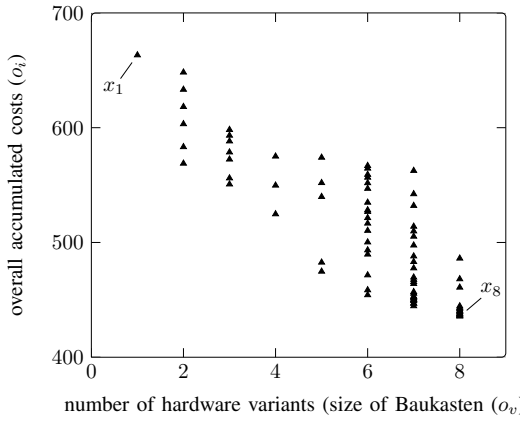

Figure 3. Pareto-plot of the results of the use case as a two-dimensional projection. Our approach extracted solutions with all possible numbers of different hardware variants to implement the defined 8 functional variants and finds solutions that reach the theoretical best-case calculated from individual optimization processes (e. g. x_1, x_8).

approaches. The other extrema is spending only one architecture for all defined variants, i. e., each variant is implemented on exactly the same hardware setup. In our use case, the architecture of variant $v7$ is able to implement all functional features and, thus, is the only hardware variant that is able to implement all functional variants. Thus, the theoretical minimum for a single hardware variant solution cause an overall monetary cost of 663.28. Also, as depicted by x_1 in Fig. 3, our approach finds this point as well, offering at least the same capability as existing DSE approaches that extract only one overall architecture. Beyond this, we deliver trade-offs between the number of different hardware variants and other optimization criteria like monetary cost and, thus, to ponder between maintenance and management overhead for providing different numbers of hardware variants and general design objectives even early in the design process.

VII. Conclusion

In this paper, we have shown how to efficiently perform a multi-variant-based design space exploration by formulating the problem of finding proper E/E architecture component platforms. Due to the symbolic encoding, it is possible to efficiently integrate an additional variant layer that handles multiple input variants in case of functionality as well as multiple output variants in case of different hardware variants, i. e., components in the Baukasten of the E/E architecture. Additional to this, fitted objective functions to evaluate multi-variant implementations are introduced. The proposed methodology was applied to a real-world use case and shows the capabilities of the newly introduced variant layer. It could be seen that our approach integrates the features of classical DSE approaches in case of single variant optimizations as well as capabilities of extracting overall architectures. Furthermore, we contribute to extend a DSE approaches to integrate the number of different hardware variants as an additional layer of freedom. In the future, we want to extend our use case to represent the whole safety subdomain and, thus, to holistically handle multiple different Baukasten ECUs in parallel.

Table I
Functional variants of the case-study with their functional features and the lowest monetary cost for a dedicated variant architecture.

	functional features								best-case
variant	f1	f2	f3	f4	f5	f6	f7	f8	monetary cost
v1	x								29.3
v2	x	x							35.0
v3	x		x						39.4
v4	x	x		x					45.11
v5	x		x		x			x	72.81
v6	x	x		x		x		x	78.51
v7	x	x	x	x	x	x	x	x	82.91
v8	x							x	52.6

References

[1] R. Moritz, T. Ulrich, and L. Thiele, "Evolutionary exploration of e/e-architectures in automotive design," in *Proc. of the International Conference on Operations Research*, 2011, pp. 361–366.

[2] V. Rupanov, C. Buckl, L. Fiege, M. Armbruster, A. Knoll, and G. Spiegelberg, "Early Safety Evaluation of Design Decisions in E/E Architecture According to ISO 26262," in *Proc. of the 3rd International Symposium on Architecting Critical Systems*, 2012, pp. 1–10.

[3] M. Glaß, S. Graf, F. Reimann, and J. Teich, "Design and Evaluation of Future Ethernet AVB-based ECU Networks," in *Embedded Systems Development: From Functional Models to Implementations*, A. Sangiovanni-Vincentelli, H. Zeng, M. Di Natale, and P. Marwedel, Eds. Springer, 2014, pp. 205–220.

[4] M. Eberl, M. Glaß, J. Teich, and U. Abelein, "Considering Diagnosis Functionality during Automatic System-Level Design of Automotive Networks," in *Proc. of DAC*, 2012, pp. 205–213.

[5] S. Fürst, J. Mössinger, S. Bunzel, T. Weber, F. Kirschke-Biller, P. Heitkämper, G. Kinkelin, K. Nishikawa, and K. Lange, "AUTOSAR–A Worldwide Standard is on the Road," in *Proc. of 14th International VDI Congress Electronic Systems for Vehicles, Baden-Baden*, 2009.

[6] S. Graf, M. Glaß, D. Wintermann, J. Teich, and C. Lauer, "IVaM: Implicit Variant Modeling and Management for Automotive Embedded Systems," in *Proc. of CODES+ISSS*, 2013, pp. 1–10.

[7] M. Lukasiewycz, M. Streubühr, M. Glaß, C. Haubelt, and J. Teich, "Combined System Synthesis and Communication Architecture Exploration for MPSoCs," in *Proc. of DATE*, 2009, pp. 472–477.

[8] A. Mihal, C. Kulkarni, M. Moskewicz, M. Tsai, N. Shah, S. Weber, Y. Jin, K. Keutzer, C. Sauer, K. Vissers, and S. Malik, "Developing Architectural Platforms: A Disciplined Approach," *IEEE Design & Test*, vol. 19, pp. 6–16, 2002.

[9] H. Zeng, A. Davare, A. Sangiovanni-Vincentelli, S. Sonalkar, S. Kanajan, and C. Pinello, "Design Space Exploration of Automotive Platforms in Metropolis," in *Society of Automotive Engineers Congress*, 2006.

[10] R. Dömer, A. Gerstlauer, J. Peng, D. Shin, L. Cai, H. Yu, S. Abdi, and D. D. Gajski, "System-on-chip environment: a SpecC-based framework for heterogeneous MPSoC design," *EURASIP J. Embedded Syst.*, vol. 2008, pp. 5:1–5:13, Jan. 2008.

[11] A. D. Pimentel, C. Erbas, and S. Polstra, "A Systematic Approach to Exploring Embedded System Architectures at Multiple Abstraction Levels," *IEEE Transactions on Computers*, vol. 55, pp. 99–112, 2006.

[12] J. Keinert, M. Streubühr, T. Schlichter, J. Falk, J. Gladigau, C. Haubelt, J. Teich, and M. Meredith, "SYSTEMCODESIGNER - An Automatic ESL Synthesis Approach by Design Space Exploration and Behavioral Synthesis for Streaming Applications," *TODAES '09*, vol. 14, no. 1, pp. 1–23, 2009.

[13] S. Wildermann, F. Reimann, J. Teich, and Z. Salcic, "Operational Mode Exploration for Reconfigurable Systems with Multiple Applications," in *Proc. of FPT*, 2011, pp. 1–8.

[14] P. van Stralen and A. Pimentel, "Scenario-based design space exploration of MPSoCs," in *Proc. of ICCD*, 2010, pp. 305–312.

[15] F. Balarin, M. Chiodo, P. Giusto, H. Hsieh, A. Jurecska, L. Lavagno, C. Passerone, A. Sangiovanni-Vincentelli, E. Sentovich, K. Suzuki *et al.*, *Hardware-software co-design of embedded systems: the POLIS approach*. Kluwer Academic Publishers, 1997.

[16] M. Lukasiewycz, M. Glaß, C. Haubelt, and J. Teich, "Efficient Symbolic Multi-Objective Design Space Exploration," in *Proc. of ASPDAC*, 2008, pp. 691–696.