System Design for Flexibility

Christian Haubelt, Jürgen Teich
University of Paderborn
Paderborn, Germany
{haubelt, teich}@date.upb.de

Kai Richter, Rolf Ernst
Technical University of Braunschweig
Braunschweig, Germany
{richter|ernst}@ida.ing.tu-bs.de
Outline

• Hierarchical specification of embedded systems

• Hierarchical system synthesis

• Hierarchical design space exploration
  – Computing flexibility
  – Reduction of the search space

• Conclusions
Introduction

- Designing a system to best meet a set of requirements on cost, power, speed, . . . , is challenging
- Although these problems are hard to solve, they have been formalized [Blickle, Teich, Thiele 1998]
- In adaptive or platform-based system design an additional objective arises ⇒ Flexibility
- Flexibility quantitatively describes the richness of functionality which a given architecture is able to implement
Specification of a Digital TV Decoder

\[ \gamma_D^1 \rightarrow P_D^1 \]
\[ \gamma_D^2 \rightarrow P_D^2 \]
\[ \gamma_D^3 \rightarrow P_D^3 \]
\[ \gamma_U^1 \rightarrow P_U^1 \]
\[ \gamma_U^2 \rightarrow P_U^2 \]

- \( P_A \) Authentification Process
- \( P_C \) Controller Process
- \( I_D \) Interface for Decryption Processes
- \( I_U \) Interface for Uncompression Processes
- \( \gamma_{Di} \) Decryption Algorithms
- \( \gamma_{Ui} \) Uncompression Algorithms

Christian Haubelt
System Design for Flexibility
DATE’02
• Possible architectures are also modeled by means of hierarchical graphs.

• Interfaces which are refineable by clusters represent:
  1. alternative design decisions: compile time decisions
  2. reconfigurable architectures: runtime decisions
Specification of Embedded Systems (1)

• Problem Graph:
  – Vertices represent processes
  – Edges model dependence relations

• Architecture Graph:
  – Resources are represented by vertices
  – Interconnections are specified by the edges
  – Resources are viewed as potentially allocatable components

• Mapping Edges:
  – indicate user-defined constraints
  – map leaves of the problem graph to leaves of the architecture
Specification of Embedded Systems (2)
Timed Activation

To avoid infeasible solutions propose hierarchical activation rules:

1. The activation of an interface at time $t$ implies the activation of exactly one associated cluster at the same time.

2. The activation of a cluster $\gamma$ at time $t$ activates all embedded vertices and edges in $\gamma$.

3. Each activated edge $e$ has to start and end at an activated vertex. This must hold for all times $t \in T$.

4. Due to (eventually implied) timing constraints, the activation of all top-level vertices and interfaces in the problem graph $G_P$ is required.
A timed implementation consists of:

- **Timed allocation** $\alpha(t)$
  $\alpha(t)$ is the subset of all activated vertices and edges of the problem and architecture graph at time $t$

- **Timed binding** $\beta(t)$
  $\beta(t)$ is the subset of all activated mapping edges at time $t$
Computing Flexibility

\[ f = 0 \cdot (1) + 1 \cdot (2 - 0) + 1 \cdot (1 + 1 - 1) = 3 \]
\[ f = 1 \cdot (1) + 1 \cdot (1 - 0) + 1 \cdot (2 + 1 - 1) = 4 \]
Our goal: To find all Pareto-optimal solutions!
Exploration Algorithm

EXPLORE
IN: specification graph $G_S$
OUT: Pareto-optimal set $\mathcal{O}$
BEGIN
$A = G_S.\text{possibleResourceAllocations}()$
$f_{\text{max}} = G_S.\text{computeMaximumFlexibility}()$
FOR each candidate $a \in A$ DO
IF $a.\text{computeMaximumFlexibility}() > f_{\text{cur}}$ THEN
$i = G_S.\text{computeImplementation}(a)$
IF $i.\text{isFeasibleImplementation}()$ THEN
IF $i.\text{meetsAllConstraints}()$ THEN
IF $i.\text{flexibility}() > f_{\text{cur}}$ THEN
$\mathcal{O} = \mathcal{O} \cup i$
$f_{\text{cur}} = i.\text{flexibility}()$
ENDIF
ENDIF
ENDIF
ENDIF
ENDFOR
END
\[ A = G_S.\text{possibleResourceAllocations}() \]

\[ A = \{\{\mu P\}, \{\mu P, C_2\}, \{\mu P, \text{ASIC}\}, \{\mu P, C_2, \text{ASIC}\}\} \]
Activatable Clusters (1)

Activatable clusters in the problem Graph:

\( \gamma_{D1}, \gamma_{U1} \Rightarrow f = 1 \)
Activatable clusters in the problem Graph:

\[ \gamma_{D1}, \gamma_{D2}, \gamma_{U1}, \gamma_{U2} \Rightarrow f = 3 \]
• Search space consists of $2^{|V_P \times V_A|} = 2^{39}$ points

• Exploiting hierarchy: $2^{25} \Rightarrow$ Reduction 99.99%

• Possible Resource Allocation: $2^{14} \Rightarrow$ Reduction 99.95%

• Flexibility estimation: 1050 \Rightarrow Reduction 93.59%
Conclusions and Future Work

• Flexibility/cost-design space exploration
  – Computing flexibility
  – Reduction of the search space

• Case-study has shown that a typical search space can be reduced to 0.0000002% of its original size

Future Work

• Timing analysis
• Hierarchical design space exploration