Generating hardware descriptions from automotive function models for an FPGA-based body controller: A case study

Authors: Matthias Traub (Daimler AG)  
Oliver Sander (Karlsruhe Institute of Technology)  
Arik Rathner (Karlsruhe Institute of Technology)  
Prof. Jürgen Becker (Karlsruhe Institute of Technology)
# Outline

## 1. Introduction

- Motivation
- Evolution up to day
- Application example

## 2. Case study

- Computation platforms for automotive systems
- System concept
- System generation
- Design flow
- Conclusion
1 Introduction

1.1 Motivation

A lot of innovations inside a car are driven by electric and electronic systems

<table>
<thead>
<tr>
<th>State of the art</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ The automotive E/E-architecture is a complex technical system</td>
</tr>
<tr>
<td>■ The development of E/E-functions is distributed between the OEM and the suppliers</td>
</tr>
<tr>
<td>■ More and more E/E-functions are developed model-based</td>
</tr>
<tr>
<td>■ The need of computing power increases steadily</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ Mastering the increasing complexity of E/E-systems</td>
</tr>
<tr>
<td>■ Evaluation of new technologies</td>
</tr>
<tr>
<td>■ To be ready for future requirements</td>
</tr>
</tbody>
</table>
1 Introduction
1.2 Evolution up today

Increasing complexity
1 Introduction

1.3 Increasing complexity*

*By a car of the luxury-class
1 Introduction
1.4 Central ECUs

Actual trend
- Integration of applications in central ECUs

Limits are
- The computing power,
- the packaging inside car, and
- the number of output pins
## 1 Introduction

### 1.5 Application example

<table>
<thead>
<tr>
<th>Opening and closing of the windows</th>
<th>Opening and closing of the windows</th>
<th>Opening and closing of the windows</th>
<th>Opening and closing of the windows</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>mechanically</strong> by using the crank</td>
<td><strong>electrically</strong> by using discrete components</td>
<td><strong>electric/electronically</strong> by using discrete and digital components. Anti-pinch protection</td>
<td><strong>electric/electronically</strong> by using discrete and digital components. Anti-pinch protection</td>
</tr>
</tbody>
</table>

⇒ The distribution of functions leads to more data traffic
Outline

1. Introduction
   - Motivation
   - Evolution up to day
   - Application example

2. Case study
   - Computation platforms for automotive systems
   - System concept
   - System generation
   - Design flow
   - Conclusion
2 Case study

2.1 Computing platforms

Due to the increasing performance requirements of automotive applications, new hardware architectures/technologies for embedded systems are necessary, for example multi-core processor or Field Programmable Gate Array (FPGA).

Idea

Evaluation of the FPGA-Technology for automotive applications

Main advantages of FPGAs

- Parallel execution of application functions
- High performance and flexibility
- Resource-efficient system partitioning
## 2 Case study

### 2.2 Cooperation project

<table>
<thead>
<tr>
<th>Main focus of the project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluation of the FPGA technology for embedded systems for automotive applications</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Topics of interest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic reconfiguration of applications</td>
</tr>
<tr>
<td>Body controllers</td>
</tr>
<tr>
<td>Gateway systems</td>
</tr>
<tr>
<td>Design flow</td>
</tr>
<tr>
<td>Automatic code generation of function models (C and VHDL)</td>
</tr>
</tbody>
</table>

Does the FPGA-technology fulfill the automotive requirements?
2 Case study
2.2 System concept

Key topics are:

- Integration of interior functions on a central ECU (Body-Controller)
- CAN-LIN-Gateway
- LIN-Master on the Body-Controller
- Seamless design flow
- Hardware-software-partitioning of model-based developed functions
- Standardized LIN-Slaves for sensors and actuators (from Siemens VDO)
2 Case study
2.4 Block diagram of the Body-Controller

Application modules realized in hardware (see next slide)

Generic interface for all modules, which are connected to the GNoC

Gateway-Network-on-Chip
- Packet-based communication structure
- Broadcast, unicast
- Bus topology

Softcore Processor
- 32-Bit Microblaze® (Xilinx®)
- 32 kB RAM
- RS232 for Debugging
- Timers
- Interrupt controller
- External Flash 1MB

Message RAM
- Message routing
- Message timing
- Timeout handler
- Default values

Routing Engine
- Signal routing
- Message generation

Bus-Interfaces
- CAN, LIN
- USB for Debugging
2 Case study

2.5 Application modules

Example:
Necessary resources for the convertible top function (implemented on a Spartan3® FPGA from Xilinx®): 1162 Slices
## 2.6 Library concept

### Application library
- Function A
- Function B
- Function C
- ...

### Basic software library
- Operating system
- Network manage.
- ...

### Hardware library
- Processor A
- Processor B
- Timer
- Controller
- Message RAM
- NoC
- NoC-IF

### Interface library
- CAN
- FlexRay
- Ethernet
- LIN
- USB
- MOST

### Software modules
- Automatic code generation
- Routing tables

---

### Interface
- CAN
- LIN
- USB

### FPGA
- Hardware module
- Softcore-Processor
- Message RAM
- Routing Engine

---

© Daimler AG; GR/EEA; Matthias Traub

Mathworks Automotive Conference 2008
2 Case study
2.6 Library concept

Hardware/Software codesign for automotive applications
2 Case study
2.7 Design flow

- Application models
- Configuration tool for FPGA Gateway
- Generation tool
- Stateflow®
- Simulink®
- Automatic code generation
- Realtime Workshop Embedded Coder
- Simulink HDL-Coder
- C-Code
- VHDL-Code
- Verification and Test
- Software DEV
- Processor type, RAM size
- Breadboard assembly
- Hardware DEV
- Testbench (ModelSim)
- Established design flow
2 Case study
2.8 System Integration

- LIN-Slaves for mirror and switch control
- LIN-Slaves for front window control
- LIN-Slaves for rear window control
- I/O-control for the roof control
- FPGA Body controller
2 Case study

2.9 Conclusion

Lessons learned

- The HDL coder
  - generates good results from our automotive application models,
  - offers the advantage of FPGA-independent code generation
  - and closes a gap in the designflow for FPGA-based automotive applications

The FPGA technology

- offers the possibility to speed-up of time critical functions, e.g. gateway systems,
- provides more flexibility during the design phase
- and enables a resource-efficient system partitioning

Open issues

- The missing integration of the FPGA technology in AUTOSAR
- and some open points of support in the standard tool flow for OEMs and suppliers
Thank you for your attention

Questions?