AVS: A Test Suite for Automatically Generated Code

Ekkehard Pofahl
Ford Motor Company

Torsten Sauer
Continental Automotive Systems

Oliver Busa
TÜV Rheinland Industrie Service GmbH
Objectives:

• **Demonstrate**: That C code corresponds to graphical model as generated from design in Simulink® and Stateflow®

• **Validate / Verify (V&V)**: The quality and robustness through dynamic testing of the code generation down the tool chain (code generator / compiler / linker / target)

• **Generate**: Meaningful test cases for automatic execution in a constantly growing test suite

• **Support**: Maintaining quality levels through Regression testing
AVS: Automotive Code Validation Suite

Project History

- Joint project of Ford / ContiAutomotive / TUV Rheinland
- Originated from a Compiler Validation Project
- Work started in early 2002
- There have been three major versions since 2002
- Supports different code generator / compiler / linker tool chains, e.g., dSPACE TargetLink and The MathWorks Real-Time Workshop® Embedded Coder
- Meaning of the acronym AVS changed during the project
Automatic Code Generation Tool Chain

Executable model

Code generator

C function

Compiler / linker

Binary on target microcontroller
Motivation for the AVS Project

• The advance of high level software design tools into the world of safety critical applications
• The prevailing use of graphical modeling and simulation, as in MATLAB® and Simulink®, among control engineers
• The facilitating of shorter production cycles through reduced complexity and approved reuse on validated design tools
• The maintenance and continuous improvement of product quality and robustness
• The tremendous risks involved in cases of faulty software in a mass produced vehicle
• The existing quality means (like e.g., FMEA) are not always applicable for software and its certification
AVS (Automotive Code Validation Suite) Concept

Simulink Model

Stimuli

C - Function

Target ECU

Code Generator

Compiler

Analysis o.k. / failure

Simulation results

Target results
Key Properties of AVS

- Validation by means of an **automated** test-suite
- Test cases are **generic** and **derived** from the actual control algorithms
- Suite contains different test cases
  - **positive**: result o.k. expected
  - **negative**: expected to provoke error message
  - "**fault**": documents a faulty behaviour
- Easy changeability to fit different scenarios (different code generators, compilers and optimization settings).
- Provides easy way to use private test cases to protect intellectual property (IP)
- Communication to tool vendors and automotive industry to avoid double work
Processor in the Loop (PIL) with Evaluation Board
Target in the Loop (TIL) Setup with Production ECU
AVS Target in the Loop (TIL) Setup
Architecture of AVS compatible Test Cases

• TargetLink Test Logik

• Real-Time Workshop® Embedded Coder Test Logik
Embedded Coder Test Case

Description of Test Logic

(1) Subsystem under test
(2) Stimuli
(3) Results
### AVS User Interface

![AVS User Interface](image)

<table>
<thead>
<tr>
<th>Testcases</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVS</td>
<td></td>
</tr>
<tr>
<td>Basic Calculate Functions</td>
<td>Discrete Transfer Function with a Step as input</td>
</tr>
<tr>
<td>Basic Logic Functions</td>
<td>Discrete Transfer Function with Sine Wave input</td>
</tr>
<tr>
<td>Discrete Calculate Functions</td>
<td>PI Controller with a Step as input</td>
</tr>
<tr>
<td>Expanded Calculate Functions</td>
<td>Testcase with several basic as well as discrete calculations</td>
</tr>
<tr>
<td>Floating Point Scaling</td>
<td>Testcase with several basic as well as discrete calculations</td>
</tr>
<tr>
<td>Lock Up Table Functions</td>
<td>Continental Teves Testcase “Filters” with one Band limiting</td>
</tr>
<tr>
<td>Negative Testcase</td>
<td>Continental Teves Testcase “Speed” with one sine wave</td>
</tr>
<tr>
<td>Optimization Methods</td>
<td>Continental Teves Testcase “Speed” with one sine wave</td>
</tr>
<tr>
<td>TLU000000</td>
<td>Provides a faulty test case</td>
</tr>
<tr>
<td>TLU000001</td>
<td>Simple R construction in Statflow and calculate functions ...</td>
</tr>
<tr>
<td>TLU000011</td>
<td>Simple usage of Gain with observable variables</td>
</tr>
<tr>
<td>TLU0000171</td>
<td>Two Look Up Tables combined with multiplication block a ...</td>
</tr>
<tr>
<td>TLU000027a</td>
<td>Testcase with several basic calculations and Statflow in ...</td>
</tr>
<tr>
<td>TLU000037</td>
<td>Typical MinMax Block usage</td>
</tr>
<tr>
<td>Stateflow</td>
<td></td>
</tr>
<tr>
<td>TargetLink Bugs</td>
<td></td>
</tr>
</tbody>
</table>
AVS Result Evaluation

AVS Test Record

21.10.2005

<table>
<thead>
<tr>
<th>Current Status</th>
<th>Testcase</th>
<th>Current compare of Simulation results to Target results</th>
<th>Previous stored reference compare status</th>
<th>Simulation LOG</th>
<th>Testcase Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>⚠️</td>
<td>TL000000</td>
<td>Wrong number of simulation and target result values.</td>
<td>No reference file</td>
<td>No Error Logfile</td>
<td>TL000000</td>
</tr>
<tr>
<td>⚫</td>
<td>TL00017d</td>
<td>passed</td>
<td>No reference file</td>
<td>No Error Logfile</td>
<td>TL00017d</td>
</tr>
<tr>
<td>⚫</td>
<td>TL00018</td>
<td>passed</td>
<td>No reference file</td>
<td>No Error Logfile</td>
<td>TL00018</td>
</tr>
<tr>
<td>⚫</td>
<td>TL00019</td>
<td>passed</td>
<td>No reference file</td>
<td>No Error Logfile</td>
<td>TL00019</td>
</tr>
<tr>
<td>⚫</td>
<td>TL00032</td>
<td>failed</td>
<td>No reference file</td>
<td></td>
<td>TL00032</td>
</tr>
<tr>
<td>⚫</td>
<td>TL00032a</td>
<td>passed</td>
<td>No reference file</td>
<td>No Error Logfile</td>
<td>TL00032a</td>
</tr>
<tr>
<td>⚫</td>
<td>TL00034</td>
<td>passed</td>
<td>No reference file</td>
<td>No Error Logfile</td>
<td>TL00034</td>
</tr>
</tbody>
</table>

21.10.2005 15:25

Legend:

- Current compare of Simulation results to Target results:
  Shows the status of a previous stored comparison result.
- Simulation Log:
  Contains a list of different results which were found during comparison. A link to the list is only available if different simulation and target result were determined.

- In Oct 2005, a code generation tool chain including Real-Time Workshop Embedded Coder and the TI TMS 470 C compiler successfully passed AVS v3.0
- Successful validations with varying tool chains and versions.
AVS for safety-related Applications, Software Safety

• Validation by means of an AVS: additional quality assurance for organizations using production code generation for safety-related applications

• Supports requirements according to IEC 61508 and upcoming ISO 26262
Optimization Level Verification (OLV)

- Almost all tools (code generators, intermediate tools, compiler, assemblers, linkers) allow optimization settings.
- Typical range for optimization levels is between 0 (no optimization) up some n (e.g., 6, full optimization).
- Every optimization step means additional operations between source model and final binary code.
- Every optimization can possibly change the behavior of the final ECU with regard to overall performance and timing constraints.
- OLV property of AVS either proofs “optimization works”, or gives the maximum level, where optimization has no negative impact on the final ECU.
Possible Interaction of AVS Partners

Different parties involved
- Car manufacturers (OEM)
- Tier 1 supplier
- TÜV
- Tool manufacturer
- Service provider
Affected Areas : Where AVS can be used

- CASE tools & methods
- Safety critical devices
- Controls
- Functional architecture
- Electric/Electronic (EE) architecture
- Vehicle integration
- Testing and regression testing
Automotive Code Validation Suite

Summary

• AVS validates the entire tool chain, including the microcontroller and target ECU
• Processor in the Loop (PIL) Setup, Target in the Loop (TIL) Setup
• Joint venture Ford / ContiAutomotive / TUV Rheinland
• Active participation in German AVS working group (Audi, Ford, Daimler AG, BMW, Volkswagen, ContiAutomotive, Bosch, Wabco, Siemens-VDO, Getrag-Ford, ZF Friedrichshafen), comparable to AUTOSAR aim:
  "Cooperate on standards, compete on implementation"
• Open for new partners
• 3 major revisions showed high quality of investigated tool chains
• Optimization Level Validation (OLV) saves target memory and execution time
• New features and areas of improvement for AVS are defined
AVS (Automotive Code Validation Suite) Concept

- **Simulink Model**
- **Code Generator**
- **Compiler**
- **C - Function**
- **Target ECU**
- **Stimuli**
- **Simulation results**
- **Analysis o.k. / failure**
- **Target results**
REFERENCES


• “Validation of the MathWorks code generator Real-Time Workshop® Embedded Coder with the Autocode Validation Suite (AVS) v3.0.” Report No. 968/EL 211.02/05, TÜV Rheinland Group, 2005

• Information concerning the V-model : http://www.v-modell.iabg.de/

• AUTOSAR (AUTomotive Open System ARchitecture) http://www.autosar.org/
CONTACTS

Ekkehard Pofahl, Ford Motor Company
epofahl@ford.com

Torsten Sauer, Continental Automotive Systems
Torsten.Sauer@contiautomotive.com

Oliver Busa, TUV Rheinland Group
Oliver.Busa@de.tuv.com