AUTOSAR E/E System Design
VSx Tool Chain Overview

Kai Wu
Technical Marketing Engineer

March 2010
Mentor Graphics a Leading Technology Driver in Electronic Design Automation (EDA) since 1981

- Revenue of $790M in 2008
- Market share ~23% of worldwide EDA market
- Largest ECAD supplier to the automotive electronics industry
- One of the largest SW companies in the world (66th)
- 4,500 employees worldwide
- Acquired Volcano (VCT) in May 2005
- Member of AUTOSAR since 2004
Guiding principles for Mentor VSx tools

- Enable “front-loading” of development
- Enable shifting as much as possible of the validation effort to a virtual environment
- Use standard terminology and data exchange formats
  - AUTOSAR
  - Eclipse
  - EAST-ADL
- Cover the whole flow from requirements to SW and ECU implementation
- Enable customers to step-by-step adapt individual parts of a complete solution
Frontloading the E/E Development

Correctness by test

Correctness by design
BMW Example

Change from testing to correctness (impossible) to design to correctness!
Frontloading is the main subject of Electronic Design Automation!
Mentor is the company, which can take you there!
Products overview
Mentor VSx AUTOSAR SW development tools

![Diagram of Mentor VSx AUTOSAR SW development tools]

- **Volcano VSA**: SW architecture definition, HW architecture definition, SW to HW arch. mapping
- **Volcano VST**: ECU/BSW validation
- **Volcano VSI Com Designer**: Network design
- **Volcano VSB and BSW**: ECU design & config
- **BridgePoint**: Model driven design
- **Volcano VSI**: SW validation
- **Volcano VSI System level validation**: Behaviour (code)
- **Volcano VSA ECU/BSW validation**: Timing

**SWC desc**

**Behaviour (code)**

**System level validation**
VSx Toolchain

E/E Architecture & Authoring

System Design

Implementation (ECU Config + BSW)

Validation and Simulation

Testing and Conformance Testing

E/E Data-Backbone
Tools Overview

E/E Architecture & Authoring
- E/E Architecture Exploration
- System and ECU Design
- SWC, CSWC and deployment
- AUTOSAR and EAST-ADL based
- Concurrent engineering

System Design
- Network Design for (LIN, CAN, FlexRay)
- System and ECU Design
- SWC, CSWC and deployment
- RTE, Diagnostics (ODX), NVRAM

Implementation (ECU Config + BSW)
- ECU Configuration
- AUTOSAR BSW

Validation and Simulation
- Distributed simulation, debugging and validation of SWC
- xtUML and IDE based SW development

Testing and Conformance Testing
- TTCN-3 based testing of automotive SW (on PC and Target)
- AUTOSAR conformance testing

E/E Data-Backbone
- Data management
- E/E PLM
- Version, Release and Variant Management
VSA
Vehicle Systems Architect
Volcano VSA

- VSA is a system level design tool for vehicle SW, electronic and communication systems

- Currently VSA is focused on development and implementation level

- VSA is being extended into various areas:
  - High-level function design
  - E/E Architecture design support
  - Design Data Management
  - Variability management
VSA Technical Overview

**POINT TOOLS**
- Vehicle Function Description Editor
- ECU Resource Editor
- Topology Editor
- Signal/Group Editor
- Vehicle System Editor / Function Mapper

**GUI Layer**
- Cluster Builders
- Metrics Generator

**Functional Layer**
- CAN
- FR
- LIN

**Backplane**
- Eclipse Framework

**Basic Infrastructure**
- File Imp/Exp Interface
- Version Control Interface
- SQL Database Interface
- Customer Specific Interface
- ECU Extractor
- File Merger Tool

**Metrics**
- Version Control System e.g. SVN
- SQL DB
- Customer Specific Files/DB

**OPTIONAL**
- AUTOSAR
- VTP (fix,net)
- LDF, NCF
- Fibex
- DBC

**Editors**
- ECU
- Resource
- Topology
- Signal/Group
SWC definition

- Define ports, interfaces, runnables
- Graphical or tree-like views

Graphical SWC editor

AUTOSAR Editor
Compu-method definition

- Graphical editor to define relation between internal and physical values
  - Rational function
  - Linear
  - Piecewise linear
  - Constants
  - Texttable
  - ...

Software Components

Visualization

© 2010 Mentor Graphics Corp. Company Confidential
www.mentor.com
Runnable Editor

- Runnables
- Interrunnable variables
- RTE Events
- Data send/receive points
SW composition definition

- Graphical design of SW compositions
- Zoom in/out
- Outline view
- Delegation ports
VSA – Activities

Requirements Management

- Vehicle Function Def
- SW Composition
- Network design
- ECU scheduling
- Function allocation
- SW Composition
- SWC desc def
- Constraints and consistency (OEM design IP)
- Topology def
- SWC desc def
- Requirements Management

Model Driven Development

- SWC desc def
- Model Driven Development

VSA level electrical design (CHS)

- ECU scheduling
- Logical domain
- Physical domain
- Metrics
- Architecture evaluation
Topology Definition

- ECUs
- Networks
- Communication interfaces
- Communication Connectors
VSA – Activities

Requirements Management

Vehicle Function Def

100ms

Vehicle level electrical design (CHS)

Model Driven Development

ECUs

Networks

Peripherals

Topoogy def

Function allocation

Network design

ECU scheduling

Metrics

Architecture evaluation

Logical domain

Physical domain

SW Composition

SWC desc def

VSA

SW Composition

200ms

100ms

VSA

Constraint: if f.classification == "Safety"

for i in all inputs

assert (i.type == L SH)

Constraint: if SW map: unique ECU

assert (rel.ECU == x1.BCU)

Model Driven Development

ECUs

I/O

Carry-over

New design

Topology def

Constraints and consistency (OEM design IP)
SWC to ECU mapping

- Map SWC to ECUs
- Search by typing
ECU Resource Generator

- Generates ECU resource based on system design
- Allows user to define additional ports and pins
- Select external/internal property

![ECU Resource Generator Interface]

**Defining ECU HW Element**

The ECU Instance should be selected. Refine the name of the ECU and the related Sensors and Actuators.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Internal</th>
<th>External</th>
</tr>
</thead>
<tbody>
<tr>
<td>THRT_SW</td>
<td>SENSOR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ESC_WRNG_LP</td>
<td>SENSOR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ABS_WRNG_LP</td>
<td>SENSOR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>WHL_SPD_OUT</td>
<td>SENSOR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ESC_VLV1</td>
<td>SENSOR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ESC_VLV2</td>
<td>SENSOR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ESC_VLV3</td>
<td>SENSOR</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>
VSA – Activities

Requirements Management

- Vehicle Function Def
- SW Composition
- Network design
- ECU scheduling
- Function allocation
- Constraints and consistency (OEM design IP)
- Topology def
- Constraints / function
  - safety
  - Classification: "Safety"
  - Assert (type == LHS)

Model Driven Development

- SWC desc def
- Model driven development
- Carrying over
- New design
- Networks
- Physical domain
- Logical domain
- Metrics
- Architecture evaluation

Vehicle level electrical design (CHS)
System Signal to Data element mapping

- Define the relation between SWC data elements and system signals
Network Design
VSA - Com Designer (option to VSA)

- AUTOSAR based network design
  - ISignal to IPDU mapping
  - IPDU to frame mapping
  - consistency check

- The Com Designer options are separate products for each protocol type (CAN, LIN, FlexRay)
Manual FlexRay Scheduler
VSA – Activities

Requirements Management

Vehicle Function Def

SW Composition

Network design

Function allocation

ECU scheduling

Model Driven Development

Topography def

Networks

Peripherals

I/O

Carry-over

New design

Constraints and consistency (OEM design IP)

Constraint 1: Function == "Safety"
if f.classification == "Safety" {
    for i in all inputs
        assert (i.type == LNS);
}

Constraint 1: SW map: unique ECU
assert (el.ECU == el.BCU)

Metrics

Architecture evaluation

Vehicle level electrical design (CHS)

Logical domain

Physical domain

VSA
Script support

- Scripts can read/write from the VSA data model
- Implement custom consistency checks, reports, small features etc
- Can be used for almost any task, not only consistency checks

```javascript
/**
   * Get list of SNCs for all packages
   * @return {String, AtomicSoftwareComponentType, ARPackage[]} Returns a list of SNCs
   */
function getSnclist()
{
    var root = ModelUtils.getModelRoot();
    // Get list of ARPackage
    var packages = root.getTopLevelPackage();
    var sncList = new Array();
    var total=0;

    // Loop all packages
    for(var i = 0; i < packages.size(); i++) {
        var pack = packages.get(i);
        var packEl = pack.getElement();

        // Check each element in this package
        for(var j = 0; j < packEl.size(); j++) {
            for (var c = 0; c < sncTemplate.length; c++) {
                var element = packEl.get(j);
                if (element instanceof sncTemplate[c][1]) {
                    sncList[total] = [packElementBundle(pack,element), element, pack];
                    total++;
                }
            }
        }
    }
    return sncList;
}
```
VSA – Activities

Requirements Management

Vehicle Function Def

100ms

f

flasher

200ms

Constraints and consistency (OEM design IP)

Vehicle level electrical design (CHS)

Model Driven Development

Topography def

ECUs

SW Composition

SWC def

Network design

Network analysis

Peripherals

I/O

Carry-over

New design

Function allocation

Network design

Function allocation

ECU scheduling

Logical domain

Physical domain

Metrics

Architecture evaluation

VSA

© 2010 Mentor Graphics Corp. Company Confidential

www.mentor.com
Metrics Generator

- Defined metrics concept
- Implemented metrics generator
- Parts library
Configurable Report Engine

- User can configure content and layout
- Reads data from the VSA data model

Present data in various formats
- Table
- Diagram
- Pictures
- Text
- PDF, HTML, Word, Excel
Multi-user and other supporting features

- Multi-user
- Merge tool
- Timing model
VSA and AUTOSAR – roles (simplified)
VSA Merge Tool

- The VSA merge tool allows merging objects that has been updated by different sources
  - Enables iterative development in separate tools
  - Enables merging data modified by supplier or OEM respectively
VSA Merge Tool

- Displays difference of objects and allows merging selected elements and properties
VSx Timing Model

- Uses AUTOSAR and TIMMO (EAST ADL) timing model
- A full system timing model covering from sensor read to actuator effect
- Based on defining events and timing requirements between the events
VSx Timing Model

- Covers timing requirements of “Delay”, “Synchronization” and “Repetition” type
- Event chains can be defined to cover true “end to end” timing requirements for a function
ECU SW design and configuration - VSB
Volcano Vehicle Systems Builder (VSB)

Tool for ECU SW developers/integrators that need to:
- Transform parameters from Extract of System description into ECUC parameters for a specific set of BSW modules
- Configure BSW module ECUC parameters
- Create BSW module descriptions
- Integrate and build ECU software and BSW module software
- Get design help for specific set of BSW modules

VSB is containing several point tools, the most important are:
- BSW module generic configuration editor
- Designer Profile plug-ins (helps configuring a specific BSW module)
- RTE generator
- ECU project builder
VSB BSW Configuration Editor

- Two types of Configuration Editor
  - Generic
    - A generic tree editor for advanced users
    - Supports configuring OEM specific modules
  - Specialized (Designer Profile plug-ins)
    - Hides AUTOSAR complexity
    - Enhanced GUI
    - Targeting complex AUTOSAR modules:
      - Communication (CAN, LIN and FR)
      - NVRAM
      - OS, RTE, SchM
      - Diagnostics
      - I/O hardware abstraction
      - etc....

- Consistency and constraints
  - Automatic handling of AUTOSAR dependencies
  - Supports User defined constraints

- ECU builder plugin
  - Integrated make system to minimize build effort
Embedded Software or Basic Software - BSW
AUTOSAR Basic Software Overview

Currently at AUTOSAR version 3.0.2

- Standard AUTOSAR BSW
  - Reference MCU: NEC V850 / GreenHills compiler
  - Complete stack available now

- Optimized AUTOSAR BSW (Nano)
  - BSW stack footprint requirements: ROM < 130kB, RAM < 8kB
  - Reference MCU: S12XEP100 / Metrowerks CodeWarrior compiler
  - Release available Q1 2010
  - Major Nano configurations done = Minimized configuration effort

Design/configuration tools

- VSB – Configuration editor/design tool
- VSC – configuration generators
AUTOSAR BSW 3.0.2 stack

BSW is ported to target hardware and delivered as fully validated object libraries.
VSI
Vehicle Systems Integrator
Virtual Systems Integrator

- Multi-lingual multi-processor Model Driven Development environment
- Design verification and validation of embedded software in distributed systems
- Initial target: AUTOSAR
- Integrates C/C++, UML, Simulink, EDGE debug, etc
VSI Application Software Development

Sourcing functionality

Modeling

Models use code generated from BridgePoint UML, Simulink, etc. or handcoding

Algorithmic Modeling

Software Modeling

C code

Generate

VFB view

Virtual Functional Bus

Operating System

System Services

C code

Generate

C code

Generate

C code

Generate

C code

Generate
VSI Tool Suite

- Breakpoints
- Profiler, Code and Data Trace
- Software Component View SWC and Composition
- Runnable and Task
- IDE, Debugger
Benefits of an AUTOSAR System Simulator

- **Excellent Collaboration**
  - OEM and Tier1 communicate around executable models
  - Achieve early specifications
    - Required functionality is communicated unambiguously, before architectural decisions are made
  - Achieve early integration and test
    - Specifications with integrated functions and verification suites increase chances of first-time success

- **Verification is fast and accurate**
  - Simulates at a high-level of abstraction

- **Integrate functions from multiple tools**
  - Functionality is best expressed in multiple domain-specific modeling languages

- **Use native modeling tools**
  - Developers remain in familiar tools, preserving investments into existing tools and libraries
VST
Vehicle Systems Test
VST

- Unified test environment for all AR BSW modules
- Full compliance with AR conformance tests (TTCN3 based)
- Minimized testing implementation effort
- Automatically generates
  - Upper/Lower layers stubs
  - TTCN-3 triggers/responses records
  - Test reports
- Minimized test execution time
- Achieves easy integration between BSW modules from different sources
VST – Vehicle Systems Tester

- One single environment for Test Development, Execution and Reporting.
- A generic test tool, not limited to AUTOSAR SW testing
- Direct execution of the AUTOSAR CTS
- Test Execution either on Target or on PC
- Based on well known and open Standard Language (TTCN-3) for Portability
- Automatic code generation of test object wrappers and TTCN-3 templates
- Automatic report generation
Development Flow and Organizational coverage
Organization Coverage

- Project manager
- EDS designer
- Network designer
- Test engineer
- Component engineer
- System designer/Architect
- Function engineer

VSA Interaction layer

VSA
VSI
VST

© 2010 Mentor Graphics Corp. Company Confidential
www.mentor.com
Roadmap and direction
Mentor Graphics Automotive Network Design – Short Term Direction

- Establish VSx Tool Chain with initial products VSA, VSA COM FlexRay, VST, VSI (Now)

- BSW and VSB AR 3.0 (Now)
  - FlexRay AR 3.0 Now
  - Rest (LIN, CAN, OS, RTE etc.) Q1 2010

- Add manual Network Design LIN, CAN, FlexRay, Ethernet (H1 2010)

- Introduce design automation
  - Network Design synthesis (H2 2010)
    - Algorithms for LIN and CAN from VNA

- Add support for massive multi-user capabilities with central data-repository (H2 2010)
Product Definition

Feature List
Variant & Options
Requirements

Function Architecture

Functional building blocks
Functional network

Development

SWCs Compositions Runnables Network Design

Integration and

ECUs, Sensors, Actuators Topology Mappings SW configuration Basic Software

AUTOSAR

ADL

SWC

ECU

Engineering Data Backbone

Configuration / Change management

Consistency checks Multi

User access

Now

2010 H2

2011

Now
Summary

- Mentor Graphics Vehicle Systems Design Suite covers
  - Architecture design
  - Network Design
  - ECU configuration, design and test
  - VFB level simulation
  - Implementation in ECUs

- Overall goals
  - Enable optimisation of the E/E architecture
  - Enable (early) virtual verification of the system
  - "Correctness by design"
  - Digital Continuity from requirements to realisation