Decoupling Test Cases from Real and Virtual Test Systems with ASAM HIL API

Dr. Rainer Rasche, dSPACE GmbH
Dr. Dietmar Neumerkel, Daimler AG

Workshop der ASIM/GI-Fachgruppen Simulation technischer Systeme (STS) und Grundlagen und Methoden in Modellbildung und Simulation (GMMS), Wolfenbüttel 23./24.02.2012
Original equipment manufacturers (OEMs) and component suppliers have enormously increased the efficiency of electronic control unit (ECU) testing by using hardware-in-the-loop (HIL) simulation. The advantages are obvious: HIL tests are reproducible and reusable. With test automation, they can even run 24 hours a day, 7 days a week.

By means of HIL technology, function tests can be shifted to earlier development stages to increase the maturity of new software and/or electronics components. Without simulation-based test automation, expensive and time-consuming test drive cycles are performed directly in a vehicle or on conventional test benches. Test automation software provides broadly based access to the test system, e.g., to the real-time simulation model or to the electrical error simulation unit.

The goal of the ASAM HIL API standardization efforts is to allow for more reuse in test cases and to decouple test automation software from test hardware. This has been done by defining different access ports, e.g. Model Access (MAPort) and Electrical Error Simulation Access (EESPort), for a HIL system in the HIL API Project 1.0, released in 2009.

This presentation shows how test cases are decoupled from different test systems of different vendors. Furthermore, by means of the ASAM HIL API standard, test development can be shifted to earlier development stages thanks to the virtualization of expensive and limited HIL hardware. Thus, engineers can develop test cases in Virtual ECU Testing environments to initially check the consistency of the ECU software and its interfaces, task scheduling, etc.

Some ports are not HIL-specific. The MAPort, for example, can also be used to adapt simulation tools. This allows engineers to develop test cases in very early stages and in different domains in order to reuse them in later stages at a real HIL Simulator using HIL API. The Functional Mock-up Interfaces (FMI) initiative is a cooperation within the current HIL API Project 2.0. As a result of the ITEA2-funded project Modelisar, standardized interfaces for model exchange and cosimulation of subsystems from different domains have been developed. These “functional mock-up interfaces” will support simulation system setup at all stages of function software development (MIL, SIL, HIL, etc.). The plan is to release a subset of ASAM HIL API 2.0 mainly dealing with the MAPort and simulator control as “Functional Mock-up Interface for Applications”. This means that tests written in those early simulation environments can be directly reused in real HIL environments at a later stage.

Dr. Rainer Rasche, dSPACE GmbH
Dr. Dietmar Neumerkel, Daimler AG
Introduction: HIL Simulation

Testautomation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

Summary
The Control Loop

Plant

Actuator signals

Controller: ECU(s)

Sensor signals
Reality and Models

Reality → Physical Principles → Simulation Model

- Environment
- Engine
- Drivetrain
- Vehicle Dynamics
Hardware-in-the-Loop (HIL) Simulation

Advantages

- Early testing without real vehicle prototypes
- Modifying test parameters easily
- Avoiding dangerous situations
- Avoiding abrasion, resource consumption
- Automated testing possible
Overview HIL Simulation

- Processor board(s) run plant models in real-time, e.g., vehicle dynamics
- Connected to ECU(s) via I/O and signal conditioning
- Real actuator loads
- Power supply replaces vehicle battery
- Real sensor loads
- Failure simulation on ECU pins
- Connected to bus systems e.g. CAN, LIN, FlexRay
Automated ECU Testing

- Repeating tests precisely and automatically as often as required
- Access to all relevant test interfaces
- State-of-the-art: Convenient PC-based test development and execution

Interfaces to HIL test bench

- Real-time simulation
- Reality
Today, interfaces to HIL Test benches are often proprietary.
Interfaces to HIL Test Bench: Today’s Situation

Exchange of **test automation software** from different vendors and different **test systems** is **not** possible.

Proprietary interfaces
Introduction: HIL Simulation

Test Automation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

Summary
Interfaces to HIL Test Bench: Advantages of ASAM HIL API 1.0

Exchange of test automation software from different vendors and different test systems is possible.
Existing HIL API 1.0 Ports

- **MAPort**
  Model Access port provides access to the simulation model read and write parameters, capture and generate signals.

- **DiagPort**
  Diagnostic port communicates with a diagnostic system, reads data via diagnostic services from an ECU.

- **ECUPort (ECUMPort, ECUCPort)**
  The ECUM port allows capturing and reading of measurement variables. The ECUC port is used for calibration.

- **EESPort**
  Electrical Error Simulation port controls electrical error simulation hardware. It allows the setup of different types of errors (e.g. short cuts).
New Trends: Virtualization and Frontloading

Test hardware requires a high invest

- Needed for productive ECU testing „24 hours a day“
- Frontloading of test development
Automated ECU Testing on Different Platforms

Real-time Simulation

Real ECU
- Test Automation Software
- Plant Model
- ECU
- Reality
- Real-time Simulation

V-ECU
- Test Automation Software

Offline-Simulation

SiL or MiL
- Test Automation Software
- Plant Model
- ECU
- Reality
- PC-based Simulation

V-ECU
- Test Automation Software
Motivation for HIL API 2.0

Real-time Simulation
Real ECU

V-ECU

Test Automation Software

Port specific access

Plant Model

ECU

Reality

Real-time Simulation

Offline-Simulation
SiL or MiL

Test Automation Software

Limited access to connected offline simulators

Simulator-/tool-specific identifiers

V-ECU

Test Automation Software

PC-based Simulation
Agenda

Introduction: HIL Simulation

Testautomation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

Summary
Current Focus of HIL API 2.0

Framework HIL API 2.0

- Mapping decouples test cases and HIL API 1.0 ports
- Simulator control allows standardized initialization of ports, start, stop simulation etc.

Network Port:
Connects to the bus systems CAN, LIN, and FlexRay
1. **FMI for Model Exchange**
   - Simulation Tool
   - User Interface
   - Internal Model
   - Solver

2. **FMI for Co-Simulation**
   - Simulation Tool 1
     - User Interface
     - Internal Model
     - Solver
   - Simulation Tool 2
     - User Interface
     - Internal Model
     - Solver

3. **FMI for Applications**
   - Simulation Tool
     - User Interface
     - Internal Model
     - Solver
     - ASAM HIL API

4. **FMI for PLM (Product Lifecycle Management)**
   - Simulation Tool
     - User Interface
     - Internal Model
     - Solver
   - PLM system

---

**Functional Mock-up Interfaces**

Access to Connected Offline Simulators
Physical system is simulated by domain-specific tools

Test automation is connected to master simulator via ASAM HIL API / FMI for Applications

- Master gives access to all subsystem parameters and signals
- Test automation starts / stops complete simulation system
Agenda

Introduction: HIL Simulation

Testautomation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

Summary
Breaking Distance at Different Velocities and Mue Values

Test Results

<table>
<thead>
<tr>
<th>Ids</th>
<th>Variant</th>
<th>Mue</th>
<th>Breaking Distance &lt;= 70 m</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>v80</td>
<td>0.7</td>
<td>45.4194598174</td>
<td>ok</td>
</tr>
<tr>
<td>2</td>
<td>v80</td>
<td>0.8</td>
<td>39.9757830826</td>
<td>ok</td>
</tr>
<tr>
<td>3</td>
<td>v80</td>
<td>0.9</td>
<td>35.6929258878</td>
<td>ok</td>
</tr>
<tr>
<td>4</td>
<td>v80</td>
<td>1.0</td>
<td>32.338947688</td>
<td>ok</td>
</tr>
<tr>
<td>5</td>
<td>v90</td>
<td>0.7</td>
<td>55.9964128682</td>
<td>ok</td>
</tr>
<tr>
<td>6</td>
<td>v90</td>
<td>0.8</td>
<td>49.3076149999</td>
<td>ok</td>
</tr>
<tr>
<td>7</td>
<td>v90</td>
<td>0.9</td>
<td>44.9714312986</td>
<td>ok</td>
</tr>
<tr>
<td>8</td>
<td>v90</td>
<td>1.0</td>
<td>39.8734252759</td>
<td>ok</td>
</tr>
<tr>
<td>9</td>
<td>v100</td>
<td>0.7</td>
<td>70.3048077963</td>
<td>failed</td>
</tr>
<tr>
<td>10</td>
<td>v100</td>
<td>0.8</td>
<td>61.9330931503</td>
<td>ok</td>
</tr>
<tr>
<td>11</td>
<td>v100</td>
<td>0.9</td>
<td>55.3798247755</td>
<td>ok</td>
</tr>
<tr>
<td>12</td>
<td>v100</td>
<td>1.0</td>
<td>50.1041226752</td>
<td>ok</td>
</tr>
<tr>
<td>13</td>
<td>v110</td>
<td>0.7</td>
<td>83.6175271017</td>
<td>failed</td>
</tr>
<tr>
<td>14</td>
<td>v110</td>
<td>0.8</td>
<td>73.6794544687</td>
<td>failed</td>
</tr>
<tr>
<td>15</td>
<td>v110</td>
<td>0.9</td>
<td>65.9023631879</td>
<td>ok</td>
</tr>
<tr>
<td>16</td>
<td>v110</td>
<td>1.0</td>
<td>59.6391719084</td>
<td>ok</td>
</tr>
</tbody>
</table>

Max. Breaking Distance: 70 km/h
Breaking Distance at Different Velocities and Mue Values

<table>
<thead>
<tr>
<th>V</th>
<th>Mue 0.7</th>
<th>Mue 0.8</th>
<th>Mue 0.9</th>
<th>Mue 1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td><strong>83.6</strong></td>
<td><strong>73.7</strong></td>
<td>65.9</td>
<td>59.6</td>
</tr>
</tbody>
</table>

ABSBrakeTest v110 Mue: 0.8
Result
EM_CurrentSegment: 4 -- (19.943, 29.931)

<table>
<thead>
<tr>
<th>Idx</th>
<th>Variant</th>
<th>Mue</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>v80</td>
<td>0.7</td>
<td>45.4194598174</td>
</tr>
<tr>
<td>2</td>
<td>v80</td>
<td>0.8</td>
<td>39.9757830826</td>
</tr>
<tr>
<td>3</td>
<td>v80</td>
<td>0.9</td>
<td>35.6929258878</td>
</tr>
<tr>
<td>4</td>
<td>v80</td>
<td>1.0</td>
<td>32.3389476788</td>
</tr>
<tr>
<td>5</td>
<td>v90</td>
<td>0.7</td>
<td>55.9964128682</td>
</tr>
<tr>
<td>6</td>
<td>v90</td>
<td>0.8</td>
<td>49.3076145999</td>
</tr>
<tr>
<td>7</td>
<td>v90</td>
<td>0.9</td>
<td>44.0714312985</td>
</tr>
<tr>
<td>8</td>
<td>v90</td>
<td>1.0</td>
<td>39.8734252759</td>
</tr>
<tr>
<td>9</td>
<td>v100</td>
<td>0.7</td>
<td>70.3048877983</td>
</tr>
<tr>
<td>10</td>
<td>v100</td>
<td>0.8</td>
<td>61.9350931503</td>
</tr>
<tr>
<td>11</td>
<td>v100</td>
<td>0.9</td>
<td>55.3798247765</td>
</tr>
<tr>
<td>12</td>
<td>v100</td>
<td>1.0</td>
<td>50.1041226752</td>
</tr>
<tr>
<td>13</td>
<td>v110</td>
<td>0.7</td>
<td>83.6175271017</td>
</tr>
<tr>
<td>14</td>
<td>v110</td>
<td>0.8</td>
<td>73.6794544687</td>
</tr>
<tr>
<td>15</td>
<td>v110</td>
<td>0.9</td>
<td>65.9023631879</td>
</tr>
<tr>
<td>16</td>
<td>v110</td>
<td>1.0</td>
<td>59.6391719084</td>
</tr>
</tbody>
</table>
Agenda

Introduction: HIL Simulation

Testautomation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

Summary
Today, test cases often directly depend on the used test hardware

**ASAM HIL API 1.0**
Decouples test cases from real and virtual test systems using ports

**ASAM HIL API 2.0**
Mapping decouples test cases and HIL API 1.0 ports
Standardized simulator control to initialize ports
Network port supports CAN, LIN, FlexRay

Easy test case exchange between coupled offline simulators in early stages and productive HIL test benches

Better know-how transfer from one test bench to the other

Reduced training costs for employees

**From end users perspective:**
This allows the ‘best’ test software combined with the ‘best’ test hardware.
Thank you very much for your attention.