Experimental cases of fault injection test using virtual ECU and points of attention for the application

- Japan Virtual Microcontroller Initiative / vECU-MBD WG activity example introduction -

Japan Virtual Microcontroller Initiative
vECU-MBD WG

Virtual HILS TF Leader
Yoshihiro Miyazaki
Technology Development Div. Hitachi Automotive Systems, Ltd.

March 11, 2015
Automotive Functional Safety Conference 2015
@ Sola City Conference Center (Tokyo / Ochanomizu)
Today's presentation contents

1. Establishment background and activity summary of vECU-MBD WG

2. Activity example introduction: Fault injection

3. Conclusion

[Notes]
ECU: Electronic Control Unit
vECU: Virtual ECU
MBD: Model Based Development
WG: Working Group
1. Establishment background and activity summary of vECU-MBD WG

* Progress and subject of MBD
* Virtual microcontroller (MCU) and virtual ECU
* The goal image to be aimed
* Summary of vECU-MBD WG

2. Activity example introduction : Fault injection

3. Conclusion
Progress of the MBD utilization

Background of Automotive Product Development

- Development of high-functionality & high-quality product to respond product performance
- Shortening turn-around time for early release to the market

Increased load to S/W development engineer by electronic control of automotive part

- Engine and break etc., i.e. individual function is getting large & complex
- Assigning adjustment among functions is getting complex by network connection of functions

Utilize model for result prediction in later development process

Application of MBD for electronic control system is expanding
What is models in MBD?

Models enable simulation in virtual system

System Design Phase (Virtual System Design & Test)

- Clarification of input and output of a control target and a control device
- Clarification of each composition module
- Clarification of a total test and individual module tests

Optimize System Organization by Model Based Development

Models enable simulation in virtual system
Subject of a complex electronic control system test

In-Vehicle Test

Early detection of software errors is difficult, because test of application & platform & network is executed only after completion of real unit.
Solution ⇒ Virtual MCU and virtual ECU

Virtual microcontroller (MCU) (microcontroller model):
The model of a processor targeted for implementation. Peripheral circuits in
the microcontroller are included.

Virtual ECU (ECU model):
A model of ECU targeted for implementation. The model (virtual
microcontroller) of a processor targeted for implementation is included.

Target object code (binary code) which is the same as the product is simulated including
the base software with the OS.

System layer

Input model

MCU model

Output model

Unit layer

Virtual ECU

Parts layer

Virtual MCU

Target object code

Engine number of revolutions

Gear

AT gear

AT number of revolutions

Engine number of revolutions

Engine speed [rpm]

Virtual ECU (ECU model):
A model of ECU targeted for implementation. The model (virtual
microcontroller) of a processor targeted for implementation is included.
The goal image to be aimed

To apply virtual ECU in each development process phase → Shortening the right side of V-shape development process

- Requirement Design
- Spec. Design
- S/W Detail Design
- Implementation
- Mass Production
- In-Vehicle Verification
- Component Verification

Virtual System / Virtual Full-Vehicle Simulation

Virtual ECU / Virtual HILS

Virtual MCU
Application example: Virtual HILS (comparing to HILS)

**HILS (Hardware-in-the-loop simulation)**
- Structure: Real ECU + plant model
- Feature
  - Verify the same binary code as a product
  - Verify software of a real system
- Demerit
  Limitation of place, target environment connection, reproducibility, observability, and fault injection. etc.

**Virtual HILS**
- Structure: Virtual ECU + plant model
- Feature
  - Verify the same binary code as a product
  - Verify software almost equivalent as a real system
  - Real machine-less (Real machine is unnecessary and test can be done before preparation of the real machine)
  - Demerits of HILS (left) are solved. (place, target environment connection, reproducibility and observability, fault injection, etc.)
  - Remote usage or parallel execution can be done on cloud

---

**HILS**
- Vehicle model
- HILS device
- ECU (Real machine)
- Software (binary code)

**Virtual HILS**
- Plant model (vehicle model)
- Controller model (virtual ECU)
- Software (binary code)
- MCU model (virtual MCU)
- Simulator
- Virtual ECU Simulator

---

Goal
Problems in the utilization of Virtual ECU

Wall of guarantee of modeling behavior
Wall of modeling precision
Wall the modeling skill
Wall of tool environment
Wall of confidential

Construction of the model supply chain beyond the domain of car makers, ECU makers, semiconductor makers and simulation tool makers is necessary.

- To prepare MCU model / ECU model timely with low-cost and high-quality
- To establish co-simulation technology of multiple models and tools, etc.
Start of the cooperation activity in Japan

The activity through vertically integrated industry domains (Car – ECU – semiconductor – development tool) is necessary for realization of model based design using virtual ECU in order to keep superiority in global competition.

Activity of vECU-MBD WG was started by voluntary members from April, 2010
Outline of vECU-MBD WG

◆ Feature: Cooperation activity which crosses vertically through industry
◆ Objectives: To promote application of MBD environment using virtual ECU
◆ Participating members:
  Car makers, ECU suppliers, semiconductor vendors, tool vendors, research institutions and universities
◆ Activity history: April 2010～present
◆ Main achievement:
  (1) Use cases, Glossary, and User support guide to consider introduction
  (2) Experimental example (system integration and evaluation)
◆ Important activity theme of 2013FY～:
  (1) multiple ECU co-simulation, (2) fault injection test

[member]
(Total 28 organizations) [in no particular order] (September, 2014)
Activity Roadmap

Considering importance and difficulty, three activity phases have been planned. TF activities started from 2011.

2013FY~: Model supply chain TF were closed because its purpose were almost achieved. Virtual HILS TF were newly established.

<table>
<thead>
<tr>
<th>Phase 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Supply Chain TF</td>
<td>Virtual HILS TF</td>
<td>(TBD): Cloud supporting, co-operation in business field supporting</td>
</tr>
<tr>
<td>Microcontroller Model TF</td>
<td>Standardization in business field, Supporting model and tool</td>
<td>Standardization in business field, Supporting model and tool</td>
</tr>
<tr>
<td>- Standardization and mechanism for model distribution&lt;br&gt;- Model development of use case&lt;br&gt;- Definition of development process and model supply chain business process&lt;br&gt;- Fault injection&lt;br&gt;- Interface model connecting multiple ECUs&lt;br&gt;- Co-simulation of multiple ECUs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Main activity results in past

### 2011FY
- Use-cases of the virtual ECU <enlightenment>
  - Experimental example: Virtual microcontroller was cooperated with power windowing system offered by JMAAB

### 2012FY
- User support guide & Glossary <enlightenment & standardization>
- CAN I/F behavior model <standardization>
- Experimental example: Fault injection @ MCU model
- Proposal and trial of SILS with peripheral model

### 2013FY～～14FY early half
- Experimental example: Multiple ECUs (CAN connection)
- Experimental example: Fault injection @ ECU input/output circuit model

### Information publication
- 2012/5: ISIT 12th Car-Electronics Research Workshop
- 2012/11: Public HP open
- 2012/11: ET2012xEDSF2012 panel session
- 2013/3: Glossary uploaded on HP
- 2013/5: ISIT Car-Electronics Research Workshop
- 2013/6: User guide uploaded on HP
- 2013/9: IFAC-AAC2013
- 2013/9: Public HP open (English version)
- 2013/12: User guide uploaded on HP (English version)
- 2014/1: ISIT Car-Electronics Research Workshop
- 2014/7: ISIT Car-Electronics Research Workshop
- 2014/10: User guide (Rev2.0) uploaded on HP

### Works shown in public (example)

**User Support Guide to Consider Introduction of Virtual ECU**

1. **purpose:** This book summarizes the information as a guide which seems to be useful in considering of introduction & use of virtual ECU

2. **table of contents:**
   1. Purpose of Introduction & Use of Virtual ECU Simulator
   2. Positioning on Development Process
   3. Present Condition Example (Use-case)
   4. Timing Accuracy
   5. Synchronization of Two or More Simulators
   6. Interface between Models
   7. User Interface (UI)
   8. Fault Injection
   9. Performance Evaluation
   10. Points of Attention in Peripheral Model Design
Experimental example (system construction and test)

Virtual MCU cooperated with JMAAB power window system

Sample model of executable ECU spec. in power window system

Target Object Code

Virtual MCU

Replace Microcontroller Behavior Area

PWM drive circuit model
H-bridge circuit model
DC motor model
Voltage/Current I/F Circuit model
A/D converter model
Current sensing model
Power window mechanical model
Rotational pulse calculation model
Pulse interval counter model
Rotational pulse I/F circuit model

Virtual MCU Tool: Experimented on 2 tools
- Synopsys / Virtualizer
- Gaio Technology / No.1 System Simulator

Simulation result

Motor current
Window speed

Sample model of executable ECU spec. in power window system

Top level model

2011/5/18

All Rights Reserved by JMAAB

15
Activity scene of the 18th vECU-MBD WG meeting  June 20, 2014 @ Semiconductor Technology Academic Research Center (STARC)
Homepage URL

Japanese version:  http://www.vecu-mbd.org/

Activity contents and the works can be referred on the above URL.
1. Establishment background and activity summary of vECU-MBD WG

2. Activity example introduction : Fault injection
   - Functional safety standard and virtual ECU
   - Difficulties of traditional fault injection test on real machine and aim of virtual fault injection test
   - Issues for application of virtual fault injection test
   - Proposal of standardization of test specification
   - Experimental example
   - Estimation of effectiveness

3. Conclusion
Functional safety standard ISO26262 and virtual ECU

Test environment of a fault injection test

Part 4 (System) - Section 8  8.4.2 Hardware-software integration and testing

Table 8 — Effectiveness of a safety mechanism's diagnostic coverage at the hardware-software level

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Fault injection test(a)</td>
<td>+</td>
</tr>
<tr>
<td>1b Error guessing test(b)</td>
<td>+</td>
</tr>
</tbody>
</table>

Note: +: Recommended, ++: Highly recommended

Part 5 (Hardware) - Section 10  Hardware integration and testing

Table 11 — Hardware integration tests to verify the completeness and correctness of the safety mechanisms implementation with respect to the hardware safety requirements

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Functional testing(a)</td>
<td>++</td>
</tr>
<tr>
<td>2 Fault injection testing(b)</td>
<td>+</td>
</tr>
<tr>
<td>3 Electrical testing(c)</td>
<td>++</td>
</tr>
</tbody>
</table>

Note: +: Recommended, ++: Highly recommended

b) ~ Model-based fault injection (e.g. fault injection done at the gate-level netlist level) is also applicable, especially when fault injection testing is very difficult to do at the hardware product level.

⇒ Practical use is possible suitably in virtual ECU

Fault injection test using virtual ECU ⇒ Effective for verification of safety
Difficulties of fault injection test using real ECU

Difficulties of traditional way (Fault injection test using real ECU)
1. No-flexibility of fault injection point, timing, etc.
2. Difficulty of waveform observation of various points
3. Delay of verification start
4. Long time and many man-hours for verification work

Fault injection test environment using real machine (real ECU)

- Verification methods: short circuit & wiring disconnection of ECU
  -> limitation of reproducible methodologies

Solution ⇒ Fault injection test using virtual ECU
Aim of fault injection test using virtual ECU

To use virtual environment which can inject any faults and has reproducibility, observability, and completeness can derive the system verification with high-quality and high-efficiency.

■ Advantage
  1. Possibility fault injection: Any point & any timing
  2. Easy observation of waveform: Any point
  3. Early detection & short verification time without actual product
  4. Verification available without destructive measurement

Fault injection test environment using virtual ECU

ECU + Plant Modeling on PC

LAN

System model

Virtual ECU

vHILS

Virtual test environment

Monitor MCU, peripheral, plant information

PC/Server

Object code

MCU model

H/W model

H/W model

Plant model

Monitor MCU, peripheral, plant information
Analysis of prior examples

Prior example 1
- Failure model
  To cover the functional output with the failure to express the output abnormality
- Motor control system

Elemental technologies for virtual fault injection test are already available.
- Fault injection model in MCU
- Co-simulation between MCU model and plant model
- Parallel test using cloud computing, etc.

However, individual handmade mechanisms shall be added by users in prior examples. The following is effective for utilization expansion of virtual fault injection test.
  1) Standardization of test specification, 2) Standard repertoire in model or tool, etc.

Prior example 2

Analysis Result of prior examples

- Elemental technologies for virtual fault injection test are already available.
  - Fault injection model in MCU
  - Co-simulation between MCU model and plant model
  - Parallel test using cloud computing, etc.
- However, individual handmade mechanisms shall be added by users in prior examples.
  The following is effective for utilization expansion of virtual fault injection test.
  1) Standardization of test specification, 2) Standard repertoire in model or tool, etc.
Issues for application expansion of virtual fault injection test

Current status
The fault injection / modeling methodologies is not standardized, and it is not efficient.
⇒ Proposal of standardization specifications (plan) is desirable

Goal image
- Fault level, mode & Injection point
- Target-independent fault injection methodologies
- Easy judgment of the results
⇒ Proposal of the standardization specifications (plan)

Requirements to fault injection method

<table>
<thead>
<tr>
<th>Feasibility</th>
<th>Assumed faults can be simulated easily</th>
</tr>
</thead>
<tbody>
<tr>
<td>Versatility</td>
<td>Keep verification target (software or function block, etc.)</td>
</tr>
<tr>
<td>Validity</td>
<td>Easy judgment of the results</td>
</tr>
</tbody>
</table>

Analysis: Faults assumed in virtual ECU
- Propose fault level / injection point
- Propose fault mode

Change of the function block is not desirable
⇒ Only I/F expansion of fault model

System
ECU / MCU / circuit parts

Simulation result
OK / NG
Proposal of standardization for virtual fault injection test specifications

vECU-MBD WG activity: Proposal of standardization for virtual fault injection test specifications

(1) Fault injection conditions (level / point / mode / timing) and fault injection method
In virtual environment, fault injection conditions (= test cases) can be expanded without any limit.
(injection points x fault modes x timing, etc.)
Standardization of fault injection conditions to be considered is proposed.
Also, it is desirable that fault injection method can be standardized as possible.

(2) Result judgment conditions
Standardization of description specifications of result judgment conditions is proposed.
Automatic judgment is desirable.

Aim of proposal of standardization:
(1) Standard repertoire in model or tool for fault injection test can be feasible.
(2) Tool can support additional function for automatic test of exhaustive fault injection and automatic judgment.
(3) Man-hours, cost and schedule for environment integration and execution of virtual fault injection test can be improved dramatically.

Standardization of fault injection test requirement specification
Standard repertoire in model or tool for fault injection test
Automatic test of exhaustive fault injection
Automatic judgment of FMEA

Notes: FMEA: Fault Mode and Effect Analysis
Proposal example 1: Definition of fault level and Injection point

■ Fault level

<table>
<thead>
<tr>
<th>Chip level</th>
<th>ECU / MCU / ASIC level</th>
<th>System level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stuck low</td>
<td>Power &amp; clock off</td>
<td>Parity error</td>
</tr>
<tr>
<td>Stuck high</td>
<td>Incorrect R/W at ROM/RAM</td>
<td>Frame error</td>
</tr>
<tr>
<td>Drift</td>
<td>Incorrect Output from I/O</td>
<td>Overflow</td>
</tr>
<tr>
<td>Oscillation, etc.</td>
<td>Miss interrupt timing etc.</td>
<td>Timing error, etc.</td>
</tr>
</tbody>
</table>

■ Injection point

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Pin of MCU &amp; ASIC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pin of Electronic parts on ECU</td>
</tr>
<tr>
<td></td>
<td>(Resistor, Capacitor, Transistor, etc.)</td>
</tr>
<tr>
<td>Communication</td>
<td>ROM/RAM/Register in MCU &amp; ASIC</td>
</tr>
<tr>
<td>network</td>
<td></td>
</tr>
<tr>
<td>Inter communication</td>
<td>Inter communication (UART etc.)</td>
</tr>
<tr>
<td></td>
<td>Outer communication (CAN, LIN etc.)</td>
</tr>
</tbody>
</table>

The fault level and injection point have been defined.
Proposal example 2: Selection plan of fault injection point and mode

◆ Selection plan of the fault injection test cases (fault injection level / point / mode)

<table>
<thead>
<tr>
<th>Place (level, point)</th>
<th>Fault mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Inside of ECU: Connector terminals</td>
<td>Stuck at H/L of each terminal</td>
</tr>
<tr>
<td>b. Inside of ECU: Pins of MCU or ASIC</td>
<td>Short-circuit of adjacent terminals</td>
</tr>
<tr>
<td>c. Inside of ECU: Pins of electronic parts (resistor, capacitor, transistor)</td>
<td>Resistor</td>
</tr>
<tr>
<td></td>
<td>Open</td>
</tr>
<tr>
<td></td>
<td>In case of an analog circuit, middle value fault mode should be added.</td>
</tr>
<tr>
<td>d. Inside of ECU: Pins of MCU or ASIC</td>
<td>Capacitor</td>
</tr>
<tr>
<td></td>
<td>Open / short</td>
</tr>
<tr>
<td></td>
<td>In case of an analog circuit, middle value fault mode should be added.</td>
</tr>
<tr>
<td>e. Inside of ECU: Pins of electronic parts (resistor, capacitor, transistor)</td>
<td>Transistor</td>
</tr>
<tr>
<td></td>
<td>Open of each pin</td>
</tr>
<tr>
<td></td>
<td>Short-circuit of adjacent pins</td>
</tr>
<tr>
<td>f. Between ECUs: Communication network</td>
<td></td>
</tr>
<tr>
<td>g. Inside of MCU or ASIC: I/F part of the top layer function blocks</td>
<td></td>
</tr>
<tr>
<td>h. Inside of MCU or ASIC: The error detection by the error detection</td>
<td></td>
</tr>
</tbody>
</table>

Selection policy of error injection conditions (fault injection level / point / mode)
(1) To cover the fault injection conditions (fault injection level / point / mode) which are conventionally covered by the conventional test on real machine.
(2) To add the fault injection conditions (fault injection level / point / mode) which are hard to be carried out in the conventional test on real machine. They are selected due to the following viewpoints.
   a) Middle value fault mode, b) I/F part of the top layer function blocks in microcontroller or ASIC, c) Timing, etc.

⇒ The considered result was reflected on the WG’s user guide (rev.2) and it has been uploaded on HP
Experimental example 1: Fail-safe verification of memory fault

To verify behavior of fail-safe function of the control system at inner MCU fault
~Co-simulation of two commercial tools (Virtualizer or No.1 System Simulator + MATLAB/Simulink)~

■ Overview
Fail-safe function verification of the overcurrent detection to occur at the time of MCU inside memory fault

[Software specification]
Motor overcurrent : Over 3A
If the motor overcurrent continues 200ms, it is determined abnormally, and a motor stops.

■ Image of the fault injection
Injected fault: Change memory data

Fault injection model

Fault injection model example

Time | Object | Command | Address | Trouble level
-----|--------|---------|---------|----------------
400   | MEM    | Change  | 0x****** | 0x0000

Simulation execution screen

Window position
Window speed
Motor drive
Current stop
Overcurrent detection
Switch input
Comparison signal
UP Switch Monitor
UP Switch Signal

Note: Normal behavior waveform
Experimental example 2: Fail-safe verification of H/W circuit fault

To verify behavior of fail-safe function of the control system at ECU circuit fault
\sim Co-simulation of three commercial tools (Virtualizer or No.1 System Simulator + Saber + MATLAB/Simulink) \sim

[Overview]
Fail-safe function verification of the input un-match detection to occur at the time of the H/W circuit fault

[Original software specification]
If the comparison result of two input signals is unmatched, it is determined abnormally, and a motor stops.

[Verification result]
Up switch becomes dominant. The cause is ECU software bug. ⇒The bug was detected.

Image of the fault injection
The fault is simulated by the resistor value change.
- Short-circuit of pull-up resistor
- Short-circuit of capacitor

Resistor value change 10kΩ⇒1μΩ (short fault)

Test environment:
Circuit : SaberRD 2013.12 (Synopsys)
MCU: Virtualizer H-2013.06SP1 (Synopsys)
Control system: Matlab R2012a (MathWorks)
Run environment: Windows7(64bit)
Points of attention in applying fault injection test

Application method of virtual fault injection test was discussed in vECU-MBD WG activity.

- Frequent question by users:
  Users want to calculate the voltage at the time of the short circuit when adjacent pins of the microcontroller short-circuited because the voltage is unknown. Can the virtual fault injection test support such use cases?

- Answer (The WG discussion result):
  It is desirable to use individually the circuit simulator (SPICE, etc.) which is suitable for such use case.

  The virtual fault injection test using virtual ECU should be used for exhaustive verification that the electronic control system does not have any problem in handling assumed error phenomenon.

  E.g. When adjacent pins of the MCU becomes short-circuited, is the voltage value low-level or high-level?
  There can be two cases ~ low-level or high-level. And then, it should be verified that the electronic control system does not violate the safety goals in both of two cases.

<table>
<thead>
<tr>
<th>Use purpose</th>
<th>Suitable simulator</th>
<th>Judging method</th>
</tr>
</thead>
<tbody>
<tr>
<td>To verify that the electronic control system do not violate the safety</td>
<td>Co-simulation to apply virtual ECU (virtual ECU = virtual microcontroller + circuit simulator)</td>
<td>It is judged that either case does not have any problem (does not violate security goal) even if</td>
</tr>
<tr>
<td>goals for all of the assumed error phenomenon</td>
<td></td>
<td>the voltage at the time of the short circuit is recognized to be H or L.</td>
</tr>
<tr>
<td>To calculate voltage value when adjacent pins of the microcontroller</td>
<td>Individual use of the circuit simulator</td>
<td>It is judged that the voltage at the time of the short circuit does not exceed the expected range or</td>
</tr>
<tr>
<td>short-circuited</td>
<td></td>
<td>the permission level</td>
</tr>
</tbody>
</table>

⇒ The considered result was reflected on the WG’s user guide (rev.2) and it has been uploaded on HP
Estimation of effectiveness (example)

- Parts count in the whole system

**Total parts count in the power window system = 50 pieces**

Example:
- Single combination of fault injection point / modes
- For the analog parts, the middle value (drift) fault assumes six points as the following
  
  * Drift change: Four points of the standard value x2, x3, x(1/2), x(1/3)
  * Robustness: Two points of around threshold x1.1, x0.95, etc.

<table>
<thead>
<tr>
<th>number (number of the terminals)</th>
<th>MCU</th>
<th>Resistor</th>
<th>Capacitor</th>
<th>Diode</th>
<th>OP AMP</th>
<th>CMOS logic</th>
<th>MOS-FET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up switch input detect circuit</td>
<td>1(2)</td>
<td>4(8)</td>
<td>2(4)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Down switch input detect circuit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Current detection, current voltage I/F circuit</td>
<td>1(1)</td>
<td>6(12)</td>
<td>1(2)</td>
<td>4(8)</td>
<td>2(6)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Rotary pulse signal I/F circuit</td>
<td>1(1)</td>
<td>4(8)</td>
<td>2(4)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PWM drive, H bridge circuit</td>
<td>1(3)</td>
<td>5(10)</td>
<td>-</td>
<td>4(8)</td>
<td>-</td>
<td>11(33)</td>
<td>4(12)</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>1(7)</td>
<td>19(38)</td>
<td>5(10)</td>
<td>8(16)</td>
<td>2(6)</td>
<td>11(33)</td>
<td>4(12)</td>
</tr>
</tbody>
</table>

- Virtual FMEA verification (partially extracted)

<table>
<thead>
<tr>
<th>ASIL</th>
<th>Object</th>
<th>Part name</th>
<th>Function</th>
<th>Error</th>
<th>Fault mode</th>
<th>Effect degree</th>
<th>Fatality degree</th>
<th>Occurrence frequency</th>
<th>Detection degree</th>
<th>Fatality degree</th>
<th>Occurrence frequency</th>
<th>Detection degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASIL X</td>
<td>Input detect circuit</td>
<td>Resistor -n</td>
<td>level detection</td>
<td>Wrong detection</td>
<td>stuck at H The window continues going up without driver's will</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>stuck at L The window does not rise</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

**Effectiveness of virtual fault injection test**

1) Expansion of test coverage
2) Efficiency up of verification work

Reduction of test man-hours [example]
real(traditional): approx. 1 month ⇒ virtual: approx. 1 day
1Hr x 500 cases / 20 computer nodes = 25Hr
1. Establishment background and activity summary of vECU-MBD WG

2. Activity example introduction: Fault injection

3. Conclusion
◆ vECU-MBD WG  
- Purpose of the virtual ECU utilization: Shortening the right side of V-shape development process  
- Cooperation activity in Japan which crosses vertically through technologies of car makers, parts makers, semiconductor makers, tool makers, and research organizations  
⇒ To promote application expansion of virtual ECU

◆ Functional safety standard and virtual ECU  
- In functional safety standard ISO26262, fault injection test is required as safety evidence. It recommends application of virtual test environment.  
- Fault injection test using virtual ECU ⇒ Effective for verification of safety

◆ Activity example: Fault injection test using virtual ECU  
- Fault level / injection points / mode have been defined.  
  In addition, selection plan of the fault injection test cases have been proposed.  
  (The considered result was reflected on the WG's user guide (rev.2))  
- Experimental evaluation: Fail-safe verification without change of the target object code  
  Co-simulation with three commercial tools (MCU+ECU circuit+plant)  
⇒ Effectiveness of virtual fault injection test has been shown.
Future expectation

- Tool vendors:
  To support virtual fault injection test function as standard repertoire

- Semiconductor vendors:
  To help technically (if necessary)

\[ \Rightarrow \]

- Car makers, ECU suppliers:
  To expand application of virtual fault injection test

Application of virtual fault injection test
\[ \Rightarrow \] easy

Verification of safety
\[ \Rightarrow \] high quality and high efficiency
References

- ISO26262 Road vehicles - Functional safety -
- JMAAB exhibition document http://jmaab.mathworks.jp/
- Oho , A Virtual Development of Automotive ECU and Cloud Environment, the 10th Car-Electronics Research Workshop, Jan., 2012
- Shimada, Yoshino, Saito: Trial of Model Based Development using Virtual ECU, ,The 13th Car-Electronics Research Workshop, May 2013
Thank you for your attention.