Establishing Freedom From Interference and detecting Data Access Violations, Control Flow Violations within Automotive Safety Software

Authors: Akhila Shamsunder, Swasati Baishya, Sandeep Chandrashekar, Karan Mundhra
# Table of contents

1. Introduction .............................................. 3
2. Freedom From Interference (FFI) ...................... 5
3. Control Flow Analysis ................................... 9
4. Data Access Analysis .................................... 12
5. Conclusion ................................................ 18
6. References & Acknowledgement ........................ 20
<table>
<thead>
<tr>
<th></th>
<th>Table of contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Freedom From Interference (FFI)</td>
</tr>
<tr>
<td>3</td>
<td>Control Flow Analysis</td>
</tr>
<tr>
<td>4</td>
<td>Data Access Analysis</td>
</tr>
<tr>
<td>5</td>
<td>Conclusion</td>
</tr>
<tr>
<td>6</td>
<td>References &amp; Acknowledgement</td>
</tr>
</tbody>
</table>
Introduction

Automotive

Security System
- Adaptive Front Control
- Adaptive Cruise Control
- Airbag Deployment
- Electronic Stability Control
- Electronic Throttle Control
- Engine Control
- Night Vision
- Interior Lightning
- Antilock Braking

Automatic Braking System
- Lane Departure Warning
- Regenerative Braking
- Windshield Wiper Control
- Lane correction

Data Communication
- Head up Display
- Electric Power Steering
- Instrument Cluster
- Tire Pressure Control
- Navigation System

Battery Management
- Active Suspension
- Seat Position Control

And many more....
# Table of contents

<table>
<thead>
<tr>
<th></th>
<th>Introduction</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Freedom From Interference (FFI)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Control Flow Analysis</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Data Access Analysis</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Conclusion</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>References &amp; Acknowledgement</td>
<td>20</td>
</tr>
</tbody>
</table>
Freedom From Interference

- Freedom From Interference is:
  - Absence of cascading failures leading to undefined behavior in the system
- One of the primary causes leading to such failures is unintended access to memory regions, primarily global variables and hardware registers

The safety objective of a system is to avoid such cascading failures, where unintended access to a memory location by a software element leads to the failure of another software element.
Example of FFI

As per design

As per implementation
A method to for FFI Verification

Architecture and Design Phase
- Architecture: Hardware and Software interfaces Identified
- Design: HSI accesses Identified

Source Code/Implementation

Unit Testing Framework
- Feedback to Design or code for further analysis to incorporate mitigation measures
- Expected HSI accesses as per the design

FFI Framework and Tool
- Log From unit Test
- Compare
- Reference From Design

Report

Example of the “HSI” Report

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Memory / Address accessed</th>
<th>HSI verification Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Func 1</td>
<td>0xAB123456</td>
<td>Pass</td>
</tr>
<tr>
<td>Func 2</td>
<td>0xAB1234EE</td>
<td>Pass</td>
</tr>
<tr>
<td>Func 3</td>
<td>0xAB1234EF</td>
<td>Fail</td>
</tr>
<tr>
<td></td>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>---</td>
<td>---------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>Introduction</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Freedom From Interference (FFI)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Control Flow Analysis</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Data Access Analysis</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Conclusion</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>References &amp; Acknowledgement</td>
<td>20</td>
</tr>
</tbody>
</table>
Control Flow Analysis

- Control flow violations can occur due to unintended execution of instructions or function calls. Such violations can lead to unintended behavior in the system.
- Control Flow Analysis is performed as a safety measure to overcome control flow violations. By definition, it is a static code analysis method to determine the control flow of a program. It can occur due to unintended execution of instructions or function calls leading to unintended behavior in the system.

Control flow violations can be identified by safety measures like Range and Plausibility check, Boundary value and Equivalence class checks, Requirement based testing, Inspection reviews, Control flow analysis.
A method for Control Flow Analysis
# Table of contents

<table>
<thead>
<tr>
<th></th>
<th>Introduction</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Freedom From Interference (FFI)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Control Flow Analysis</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Data Access Analysis</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Conclusion</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>References &amp; Acknowledgement</td>
<td>20</td>
</tr>
</tbody>
</table>
Data Flow Violations can occur due to incorrect handling of shared data.

Data Access Analysis is performed as a safety measure to overcome data flow violations. By definition, it is a process of analyzing memory regions accessed by software programs in a multi-thread, multi-core, and pre-emptive execution environment.

Data flow violations can be identified by safety measures like requirement-based testing, unit verification to cover equivalence class and boundary value analysis, static analysis, inspection reviews, safety analysis and Data Access analysis.
Example for Data Flow Violation: Shared memory access (Concurrent Tasks)

Core0
Task A

Write access

Shared memory
Inconsistent data

Write access

Core1
Task B

Read access

Task C

Not acceptable!!
Example for Data Flow Violation: Deadlock Scenario

Core 0

API 1

Acquire_Lock (S1);
Statement 1;
Statement 2;
Acquire_Lock (S2);
Statement 3;
Release(S1);
Release(S2);

Core 1

API 2

Acquire_Lock (S2);
Statement 4;
Statement 5;
Acquire_Lock(S1);
Statement 3;
Release(S2);
Release(S1);

Not acceptable!!
A method for Data Access Analysis

Architecture Phase

Design Phase

Source Code

DAA Tool

DAA Analysis

DAA Report

Requirements

Feedback to Design or code for further analysis to incorporate mitigation measures
Areas to be focussed during Data Access Analysis

- Concurrency Issues
- Reentrancy
- Global Data and Critical Section
- Hardware Resources
# Table of contents

1. Introduction .................................................. 3
2. Freedom From Interference (FFI) ...................... 5
3. Control Flow Analysis ........................................ 9
4. Data Access Analysis ......................................... 12
5. Conclusion ..................................................... 18
6. References & Acknowledgement ...................... 20
Conclusion

- FFI Verification, Data Access Analysis and Control Flow Analysis:
  - Can also be achieved in different methods than what is proposed in the paper. For example, logging a hardware trace.
  - Aids in identifying issues at an early stage of development, thus ensuring a safe and reliable system.
  - Probability of bugs detected at a later stage is reduced.
  - Proper synchronization between different hardware and software resources is also analyzed.

- The framework has been successful in identifying the unintended access of hardware resources that are due to erroneous design or code. CFA, DAA, and FFI verification proposed in this paper have at an early stage helped in identifying violations and thus improving the quality of the product.
# Table of contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Freedom From Interference (FFI)</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Control Flow Analysis</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Data Access Analysis</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Conclusion</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>References &amp; Acknowledgement</td>
<td>20</td>
</tr>
</tbody>
</table>
References & Acknowledgments

- ISO 26262, Road vehicles - Functional safety, ISO copyright office, 2011-11-15
- Freedom from Interference for AUTOSAR-based ECUs: a partitioned AUTOSAR
- Infineon TriCore Architecture Manual

We (authors of the paper) thank the entire automotive software team at Infineon for continuous support in improving and deploying the methods described in the paper.
Part of your life. Part of tomorrow.