Production Code Generation for Engine Control System

June 15th, 2004

Tetsuji Katayama
Akira Ohata

TOYOTA MOTOR CORPORATION

Yoshitaka Uematsu
DENSO CORPORATION

Contents

- MBD (Model Based Development)
- ACG (Auto-Code Generation) and the effect
- Required features for ACG
- Summary
MBD (Model Based Development)

Virtual World

Engine Performance Specification

Validation

Combination

Controller model

Engine Model

SILS

Control Software Specification

Real World

Engine

Controller

Validation

Combination

Rapid prototyping ECU

HILS

Engine

Actuators

Sensors

Hardware

Software
ACG (Auto-Code Generation)

ACG Mechanism

RTW files

RTW files

To analyze algorithm information from Simulink® diagrams.

To transform algorithm information into C codes according to TLC file

TLC: Target Language Compiler

Defining coding style

RTW : Real-Time Workshop®

Generated codes

This information is exclusive property of TOYOTA MOTOR CORPORATION and DENSO CORPORATION. Without their consent, it shall not be reprinted or given to third parties.
Development Flow with ACG

TOYOTA

System Developments

ACG

Specification dev.

Spec. review

Ordering ECU

Receiving Check

Distributing ECU

DENSO

Understanding Spec.

Embedded software design

Embedded codes dev.

Verification

ECU Shipping Check

Differences between Toyota’s and DENSO’s

Toyota and DENSO had each original ACG in the past.

TOYOTA ACG

(1) Based on RTW®
   – TLC customization
   – Feedback to RTW-EC

(2) Fixed-point MPU
   – Auto-Scaling

(3) System development-oriented
   – Ease of simulation
   – Easy operation
   – Seamless linkage with other development tools and data

DENSO ACG

(1) Based on RTW-EC®
   – Pre/Post processing
   – TLC customization

(2) FPU

(3) Software development-oriented
   – Conformance to coding rules
   – Well defined Software structure
   – Linkage with other software development tools
**Purpose of Integration**

To improve efficient development environment for both of advanced and production phase

1. **Reliability**
   - Production quality control overall processes
   - Reuse ACG codes as development bases
   - Readability (Existing process should be adopted.)

2. **Development efficiency**
   - Seamless linkage with other tools
   - Efficient simulation environment
   - Minimum manipulation

3. **Easy to use**
   - Customized GUI
   - Easy introduction

**Required features for ACG**
**Evaluation factors for ACG**

1. Code quality
2. ACG reliability
3. Efficiency (Memory size and execution time)
4. Replaceable to existing software modules
5. Usability
6. Productivity
7. Ease of maintenance

**Code Quality**

- **Conformance to in-house coding rules**
  
  *Premise: Generated codes are put into the existing development process.*

  - **Readability**: Reliability is built up during whole development process including code reviews.
  - **Variable and constant names definitions**: The names have means which can support understanding logic.
  - **Safety code style standard**: This can avoid embedded code safety problems and misunderstanding codes.
ACG Reliability

- Code review should be done to ensure the reliability

Premise 1: Customizations are needed but the portion should be minimized.

Premise 2: The reliabilities of RTW-EC® and the customized portion hasn’t been sufficiently established.

- The cooperated work with TMW: To identify the causes of the bugs in generated codes.

Efficiency

- The increases must be within 1.2 times of hand codes.
Replaceable feature

- Generated codes should be built into the base configuration without any modifications.

<table>
<thead>
<tr>
<th>Simulink® Model</th>
<th>Legacy code</th>
<th>Legacy code</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Simulink Model" /></td>
<td>void Function(a) {}</td>
<td>void Function(a) {}</td>
</tr>
<tr>
<td>Interface code</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Application Framework</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Platform</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECU</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Usability

- ACG should be easy to use, have the interfaces with other tools and be operated with the minimum number of operations.

Premise: ACG should be used in the existing process and with existing tools.

- Developments of the efficient required interfaces:
- Reusing the existing databases:
- Developments of efficient GUI:
- Efficient simulation environment: Simulations with Simulink® diagrams and generated codes are often used for validation and verification.
Effect of ACG on Productivity

(1) System development at Toyota

without ACG: a month → with ACG: a day

Generated code verification

(2) Embedded code development at DENSO

20% of development time have been reduced.
(Improvement of verification/validation process will increase the productivity remarkably.)

Effect on productivity (detail in DENSO)

The time of coding has been cut down mostly.

Improvement of verification/validation process will increase the productivity remarkably.
Summary

- A seamless ACG environment from advanced to production developments has been developed.
- The average of the memory increase is within 1.2 times.
- Remained works
  - Reliability
  - Efficiency of specification reviews
  - Verification and validation of generated codes
  - Configuration and project managements
  - Integration of Simulink® and Stateflow®

Actual Example

![Actual Example Image]