Model based development of Cruise Control for Mercedes-Benz Trucks

M. Wünsche, J. Elser
15.06.2004

Agenda

- Introduction
  - functional and technical overview
- Project description
  - motivation and development process
- C-Code analysis
  - analysing and documentation methods
- Control-module
  - Structure, metrics and co-operation methods
- Auto-Code generation
  - experiences and results
- Testing
  - SIL, PIL and HIL test methods
World wide application

- Cruise Control is used in several commercial vehicles e.g.
  - Heavy duty trucks
  - Delivery Trucks
  - Coaches
  - working machines

- Cruise Control is used in several market specific vehicles in
  - Europe
  - Turkey
  - Brazil
  - future: NAFTA

Functional Overview

- **Combined Cruise Control**
  - Drive Mode
    - only drive torque allowed
  - Brake Mode
    - only brake torque allowed
    - permanent open outlet valve
    - butterfly valve
    - drive line retarder, e.g. eddy current

- **Adaptive Mode**
  - separate controller
  - constant distance regulation

- **Road Speed Limiter**
  - limits drive torque
  - activates Brake Cruise Control
Controller Overview

CC-Lever
Acceleration Pedal
CAN - Signals

Governor Limiter

driver set value
governor desired value

desired speed
Hysteresis

Brake mode

Display

act. vehicle speed

Speed Governor

act. vehicle acceleration

Distance Governor

a

a'

Distance Governor

Engine

T

T'

Drive - Brake Switch

Advantages
- reuse of reliable functionality
- reuse of known scaling
- model based restructuring
- simulation based function verification
- one step fixed point code generation

World wide application needs functional extension
- reengineering of Cruise Control was necessary
- Project of model based software development

Auto code generation
(Real-Time Workshop, Embedded Coder)

m-script based build process

hand written frame code

Project description
C - Code analysis

- Analysis steps
  - Study of documentation
  - Consulting the developers
  - Automatically C-code analysis
  - Modelling of complicated code parts

- Analysis results
  - Structure charts
  - Function and variable database

- Requirements
  - Use cases
  - Simulation models

---

Reengineering Tool
“Analyzer”

- Automatic C-Code analysis tool
- Developed in R/T Department of DaimlerChrysler in an European research project
- Outputs
  - Structure charts
  - Function trees
  - Variable lists
  - Relation charts

SQL like access
Functions list
List of related variables
Cruise Control Model

- Model consists of:
  - 16 libraries
  - 4MB mdl-files
  - 2 MB Simulink
  - 2 MB Stateflow
  - 3,000 blocks
  - 140 inputs
  - 40 outputs
  - 340 parameters

- Co-operation:
  - Model structure is built up in CVS
  - Several programmers simultaneously

Co-operation methods

- Interface blocks:
  - Encapsulate modules
  - Data type and scaling of each signal will be checked
Tool chain for auto-code generation

- One supplier tool chain
  - Simulink / Stateflow
  - Fixed-Point blockset
  - Stateflow Coder
  - Real-Time Workshop Embedded Coder
- no auto scaling used in the project
- Fixed point blockset
  - most of the Simulink blocks can be used for float and fixed point simulation
- Embedded Coder
  - adjustment of settings was sufficient to receive desired code appearance
  - user defined storage classes improve code efficiency

Main steps for code optimisation

- **RAM: 410 → 210 Byte**
  - Top level function-call scheduler (1)
  - Using bit-fields in exported internal signals (2)
  - Using bit-fields in generated data structures (3)
  - Elimination of intermediate interface variables (4)
- **STACK: 220 → 80 Byte**
  - Some subsystems realised as atomic functions
  - Elimination of the most 32bit operations
  - Redesign of some atomic subsystems
Hand written vs. Auto code

- RAM effort of Auto-Code is 16% less than optimised hand written code!

Test step overview

- Vehicle Test
- Auto-Code Evaluation
- SIL Simulation
- Module Test
SIL - Simulation

Interactive tests, test-database with predefined tests and free combination of both

Auto-Code Evaluation

Deviation has to be zero!
HIL Tests

- Real-Time vehicle simulation
- Real-Time sensor simulation
- Stimulation by drive cycles
- Measuring by CAN-ID’s
- Real vehicle-CAN
- Software-module
- real controller hardware and software

Project schedule

- Design phase
  - C-Code analysis
  - Requirements documentation
  - Structure development
  - Structure verification
  - Modelling of basic functions
- Winter test drive
  - basic functionality
  - ABS / ASR functionality
- Extending phase
  - Modelling of additional functions
  - HIL - testing
  - Auto-Code evaluation
- Summer test drive
  - Final inspection of the whole functionality

Truck Product Creation (4P)
## Conclusions

### Results
- Project needs only 18 month until release
  - including analysis, restructuring, modelling, and testing
- SIL based function development
  - high state of maturity before vehicle tests start
  - higher test efficiency
  - desktop debugging instead of debugging in vehicle
- Code generation
  - Embedded Coder meets our demands
  - code efficiency and readability like hand written code
- Project aims could be reached in time!

### Experiences
- Well defined model structure
  - is essential base for all work, especially in stateflow parts
  - allows multiple use of the feature
  - is a stable base of further enhancements
  - gives easy overlook of complex functions
- Model reviews
  - efficient method of model optimising
  - shows possible incorrect parts
- Code evaluation
  - manual code reviews not possible because the whole code will be renewed each time
  - automatic code evaluation is necessary

Truck Product Creation (4P)