AVOID MISTAKES

Engine calibration is typically performed by an OEM, while the ECU code is owned by the supplier of the ECUTherefore, the OEM is typically unable to set up a ECU simulation based on the original C code of the ECU. Instead, to set up a simulation, time consuming and error prone reverse engineering is needed to develop an "equivalent model" of the ECU function of interest. To improve this situation, we have integrated a chip simulator into the virtual ECU tool Silver. This is used to simulate hex files compiled for TriCore targets directly on PC. Simulation requires:

- a hex file that contains programme code and parameters of the simulated functions
- start addresses of the functions to be simulated
- an ASAP2/a2l file that defines the conversion rules for the involved inputs, outputs, and characteristics, as well as corresponding addresses; the start addresses of functions can e.g. be extracted from a map file generated together with the hex file.

Silver uses the a2l file to automatically convert scaled integer values to physical values and vice versa during simulation. A TriCore simulation can also be exported as SFunction (mexw32 file) for use in Matlab/Simulink. On a standard PC, hex simulation runs with about 40 MIPS. If only simulating selected functions of an ECU, this is fast enough to run a simulation much faster than real-time. In this article, we also report how such simulations are used today to support the development of gasoline engines at Daimler.

VIRTUAL ECUS IN THE DEVELOPMENT PROCESS

Simulation has great potential to improve the development process for ECUs. Simulation helps to move development tasks to PC, where they often can be performed faster, cheaper or better in some respect. Examples that illustrate this point:

- On a PC, an engineer can easily "freeze time", i. e. stop simulation and inspect the call stack and all variables of a virtual (i.e. simulated) ECU without band width limitation and repeat a simulation deterministically as often as needed. In contrast, real ECU as used in HiL set-exchange [4]. However, sometimes C code is not available for implementing a virtual ECU. There are two main sources for such a situation:

- Protection of intellectual property – all or major parts of the ECU have been developed by a supplier and the OEM interested in building a virtual ECU (e.g. to support calibration, a task typically performed by an OEM) has therefore no access to the C code.
- Target-specific C code – C code is available but the C code uses inline assembler or other target or compiler specific constructs, which prevents compilation for other targets, such as the Windows x86 platform.

To deal with such situations, we have recently integrated a chip simulator into the virtual ECU tool Silver. This way, a virtual ECU can be build based on a hex file compiled for the target processor of the ECU. No access to C code is needed in this case. Instead of compiling C code for the Windows x86 platform, the chip simulator takes the binary compiled for the target processor and simulates the execution of the instructions by the target processor

E[redacted]

on Windows PC.

CHIP SIMUL

We have now partially replaced these hand-coded models with SFunctions generated automatically by Silver from a given hex file. The generated SFunctions proofed to run as fast as their hand coded counterparts.

CONCLUSIONS

As demonstrated above, an ECU hex file compiled for some target processor can be executed by the virtual ECU tool Silver on Windows PC, either open-loop driven by measurements or in closed-loop with a vehicle model. Depending on the application, selected ECU functions are simulated, or nearly the entire ECU.

This kind of simulation opens new possibilities to move development tasks from road, test rig or HiL to PCs, where they can be processed faster, cheaper or better in some respect, without requiring access to the underlying C code. Daimler currently uses this innovative simulation approach to support controls development for gasoline engines. Other applications, such as online calibration on PC via XCP seem to be doable as well.

REFERENCES

[1] Junghanns, A.; Serway, R.; Liebezeit, T.; Bonin, M: Building Virtual ECUs Quickly and Economically, ATZelektronik 03/2012, June 2012
[2] Brückmann, H.; Strenkert, J; Keller, U; Wiesner, B.; Junghanns A.: Model-based Development of a Dual-Clutch Transmission using Rapid Prototyping and SiL. International VDI Congress Transmissions in Vehicles 2009, Friedrichshafen, Germany, 30.06.-01-07.2009. http://qtronic.de/doc/DCT_2009.pdf
[3] Röpke , R.(ed.): Design of Experiments (DoE) in Engine Development – Innovative Development Methods for Vehicle Engines. Expert Verlag, 2011
[4] Blochwitz, T.; Otter M. et. al.: Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models. 9th International Modelica Conference, Munich, 2012
[5] Tatar, M.; Schaich, R.; Breitinger, T.: Automated test of the AMG Speedshift DCT control software. 9th International CTI Symposium Innovative Automotive Trans-missions, Berlin, 2010. http://qtronic.de/doc/TestWeaver_CTI_2010_paper.pdf

EDITOR IN CHARGE
Dr. Johannes Liebl, Wolfgang Siebenpfeiffer

EDITOR IN CHIEF
Dr. Alexander Heintzel
phone +49 611 7878-342 · fax +49 611 7878-462
alexander.heintzel@springer.com

VICE-EDITOR IN CHIEF
Markus Schöttle
phone +49 611 7878-257 · fax +49 611 7878-462
markus.schoettle@springer.com