Virtual ECUs for Developing Automotive Transmission Software

Dr. Thomas Liebezeit[1], Jakob Bräuer[1], Roland Serway[1], Dr. Andreas Junghanns[2]
[1]IAV GmbH, Carnotstraße 1, 10587 Berlin
[2]QTronic GmbH, Alt-Moabit 92, 10559 Berlin
thomas.liebezeit@iav.de, andreas.junghanns@qtronic.de

1 Introduction

The development of controller software for transmissions for production use is characterized by high demands on algorithm and code quality. Extensive validation measures are required to achieve these demands. In addition to independent validation by a test team, developers are required to test functional behavior of their modules in the system context.

For a long time now, IAV has used dSPACE-HiL systems and test vehicles for this purpose, as complex open-loop and closed-loop control features can only be verified with the help of a controlled system or model thereof with regard to system characteristics, going beyond the module test capabilities. However, both approaches only offer limited possibilities for troubleshooting and analysis, especially because real code debugging is impossible in such environments.

For this reason, a software-in-the-loop system has been commissioned using QTronic's Silver, giving every developer the possibility for source-code debugging on their development computer in the system context (with a plant model). To this end, a virtual control unit in SiL was created, co process identified debugging of control software for potential

- **Reuse where possible**
  Existing work results should be reused to prevent unnecessary extra work.
- **Consistency**
  It should be possible to re-use the data files consistently in X-in-the-loop and in the vehicle. For example, it must be possible to integrate the SiL in the existing build process as an additional option and to incorporate it in version management.
- **Minimum possible extra workload for SiL**
  As a general rule, the necessary additional workload for the SiL system should be kept to the necessary minimum. Improving the debugging possibilities may not result in increased workload for configuration, familiarization or adaptation.
- **Full SiL control**
  It should be possible for IAV to provide all necessary know-how for setting up and updating the SiL.

This results in the following points relevant to concrete SiL implementation:

- Support of standard formats and techniques such as:
  - Parameter and mesurement signal definition (ASAP2)
  - Parameter files (PAR)
  - CAN bus definition and use (DBC)
  - Recording and replaying of measured signals (MDF)
- Support for C-code and Simulink models (reusing plant models from the HiL domain)
- Simple tool handling
- Flexible licenses (dongle or network license) according to current usage requirements
- Easy configuration for adapting to specific needs

3 Silver

Silver is a product of QTronic GmbH and has been used for some time now at IAV in various transmission development projects. Silver provides a simulation environment for software-in-the-loop simulations. It supports many automotive standards (A2L, DCM, PAR, XCP, HEX, MDF, DAT, ...).

However, up to now it has been used in IAV transmission projects with another focus. For example, Silver is used for virtual integration. This entails integrating the software functions developed by IAV in the customer code that has not been revealed to IAV and then checking for its correct behavior in the overall system context.

The positive experience gained with Silver so far, together with an additional evaluation in terms of the specific criteria defined above led to the selection of this particular SiL implementation.

4 SiL setup

4.1 Function Software

The function software is mostly hand-written C-code. Parts of the higher levels of the

here this would bypass the lower levels in the function software so that it would then no

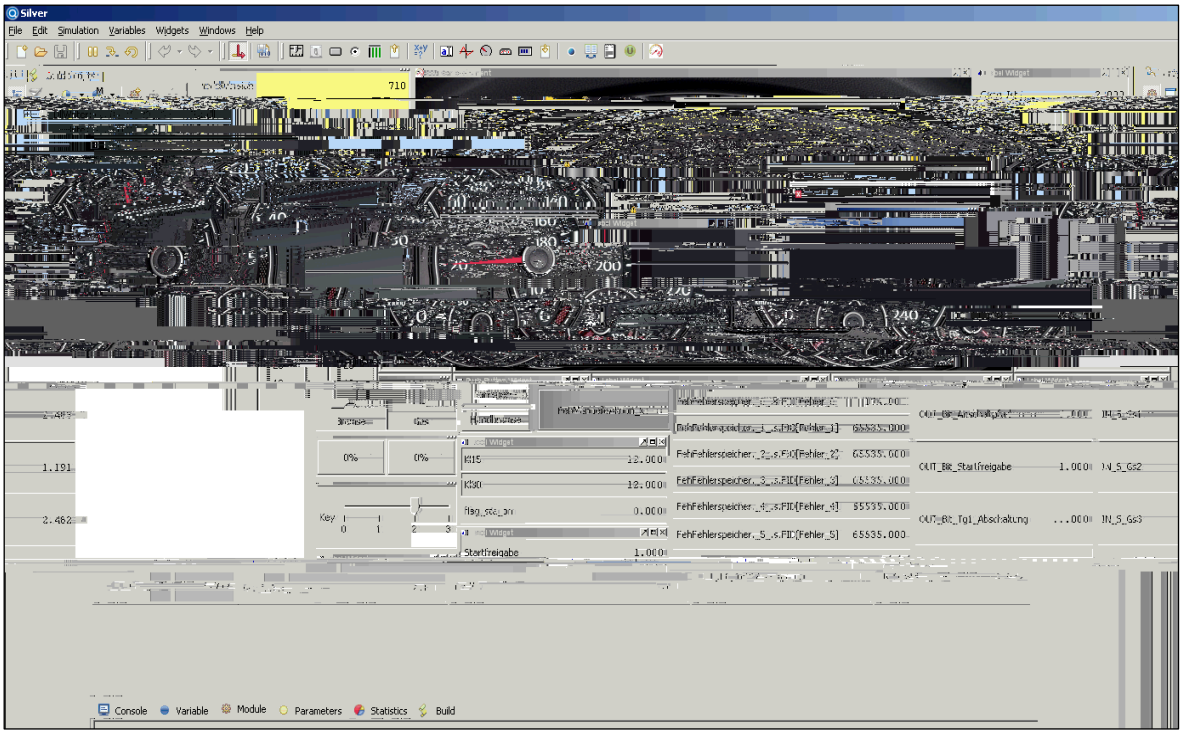

Figure 3: Silver GUI

The current configuration contains the following parts:

- Flashing the parameters from PAR file to the softw

code lines requiring analysis. The SiL system can be started once Visual Studio is connected with the Silver process. Using the GUI, a relevant driving situation can now be started, leading to the code with the breakpoint. On reaching the breakpoint, the whole system stops, i.e. the function software, virtual control unit and environment model are stopped.

Visual Studio then permits debugging and step-by-step execution of the code. At the same time, all common features are available, such as read and write access to any variable. This procedure permits swift, direct debugging without additional definition of measured variables or similar.

Debugging can also be used in the following scenarios:

- **Situations that are difficult to produce**
  Manual debugging (e.g. changing internal status variables) generates a specific driving situation for the function software. Correct processing of the corresponding code parts can be tracked in debugging.
  This procedure is suitable particularly for checking rarely achieved boundary conditions.
- **Timing errors**
  It is easy to specifically adjust and analyze timing errors by means of debugging.
-

adaptations that are to be completed promptly. Fast reaction is important to warrant high availability of the SiL for function developers.
Consequent reuse of existing work and integration in the existing development and build process keeps the additional workload to a minimum.

- **Added value from debugging**
  For the function developers, debugging puts a new quality into the analysis of the run-time behavior of function software. It can be seen that the analysis times are drastically reduced by faster change-analysis-change cycles.
- **User acceptance**
  Debugging the function software in the SiL system is well on the way to asserting itself like the HiL systems years ago. At the moment it constitutes a supplement to established work methods.