Building large-scale test systems

SCALEX

SCALEXIO, dSPACE's new hardware-in-the-loop (HIL) system, has been expanding since its launch in 2011. The latest version helps users build especially large-scale test systems, including failure simulation, and makes it possible to use complex environment models.

Multicore for Large Systems

As the amount of electronics in vehicles increases, so does the functional scope of the ECUs. The resulting, more complex ECUs have large amounts of I/O and numerous software components, and require powerful test systems. At the same time, the environment model is also usually extremely extensive: for example, if it has to represent the entire vehicle. The ECUs are tested together with the environment models in large-scale hardware-in-the-loop (HIL) simulations.
To guarantee that complex models can always be simulated in real time, the SCALEXIO® Processing Unit contains an Intel® Core™ i7 with four cores. One of the cores computes the system processes, and the other three are completely available for model and I/O calculation. The result is that SCALEXIO can perform the real-time simulation of very extensive, very complex models that a single core cannot cope with.

Two Ways of Distributing Models

The environment model has to be distributed flexibly for computation on several cores in parallel. Just as important are a streamlined workflow, clearly defined tasks for model developers, and short compile times. Optimal model management is needed to ensure well-designed communication between model parts and

IO Is Evolving

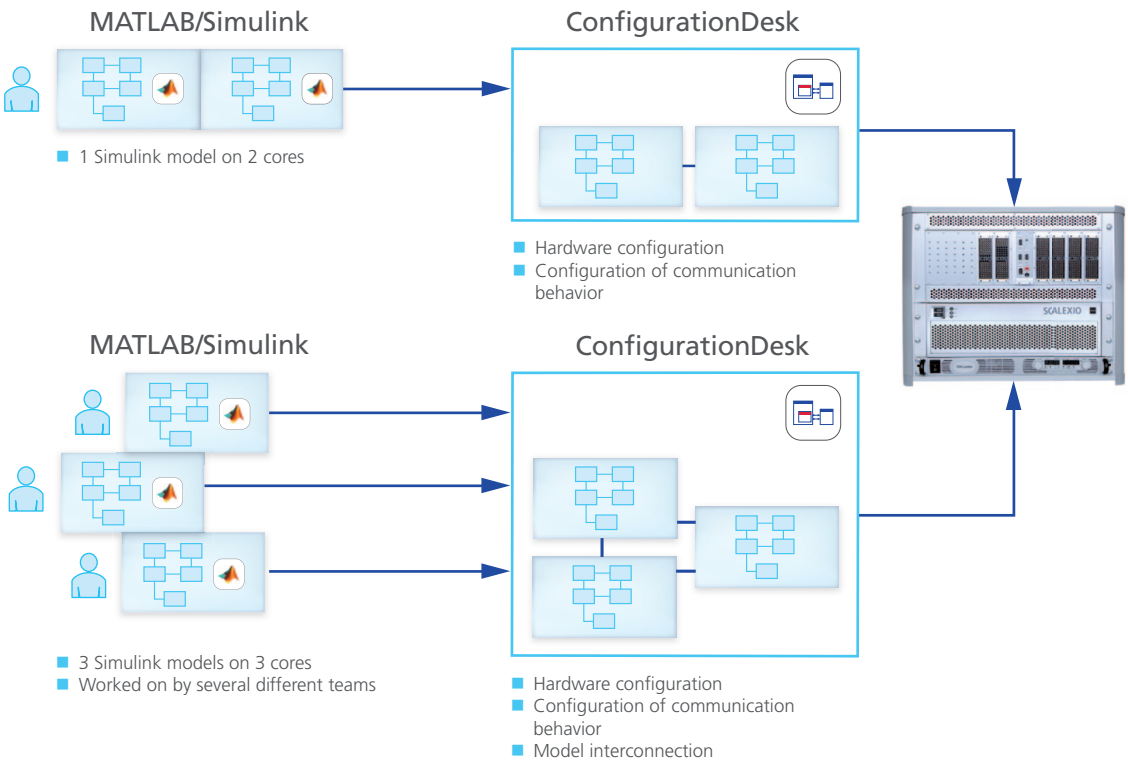

Figure 1: Two different workflows for model distribution.

well-organized connections between the model interfaces and the I/O boards.

There are two methods for creating models and distributing them on different processor cores (figure 1):

1. The model is developed as one overall system and then divided up for distribution to the cores. For this to be possible, the overall system must contain several "cuttable" model parts. Any arbitrary number of subsystems can be implemented in each of these model parts, and the subsystems can contain referenced models. dSPACE has developed a special MATLAB®/Simulink® block for assigning model parts directly to a processor core so that interprocessor communication (IPC) blocks are not needed. This makes the model easier to handle, and when changes are made to the assignment, only the model parts concerned need to be recompiled. Changes are made in the overall model itself, so offline simulation in Simulink is possible. Depending on the size of the model, long load and compile times might be necessary.
2. The Simulink model is developed as individual model parts instead of as one large overall system. Each model part holds processes that are coupled closely to one another and that must be computed in sequence, so they have to be placed on one and the same processor core. Each model is then assigned to exactly one core. Changes are made only in the individual submodels, so load and compile times are short. The advantage of model parts is that several development teams can work in parallel.

Graphical Configuration with ConfigurationDesk

If the model is created as an overall system, model communication and I/O board communication are configured with dSPACE ConfigurationDesk®. The same applies when model parts are created, except that model connections also have to be created. The model components and the corresponding communication files are imported from Simulink and displayed graphically in ConfigurationDesk. This procedure is identical for both model distribution methods. The communication files make it easy to transfer information and implement subsequent modifications.

ConfigurationDesk also lets users create configuration sets and use them to generate different, compiler-specific build versions for each model part. This maximizes the efficiency

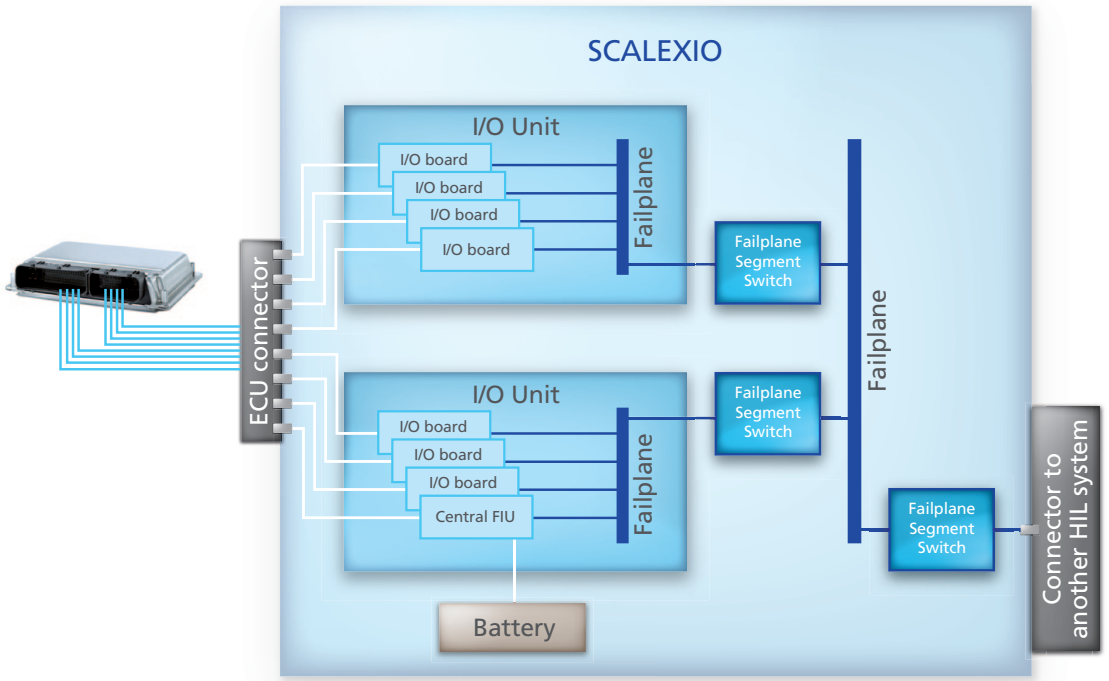

Figure 2: Infrastructure of the FIU in large systems.

of the generated code for varying conditions such as different memory sizes.

Better Signal Quality During Failure Simulation

Large SCALEXIO systems with a high number of I/O channels also need optimized solutions for failure simulation (figure 2). A Failure Insertion Unit (FIU) is used to insert electrical failures on the I/O pins of an ECU. To use the FIU for failures on all ECU signals, all the signals can be routed to the central FIU components via so-called failplanes. It does not matter if the routing runs through several cabinets in a large simulator, because the failplanes in the individual cabinets can be connected to one another and extended whenever more I/O channels are added. The failplanes are subdivided into segments that can be switched in separately. This minimizes cabling – long cables can cause signal corruption – and guarantees high signal quality. When a failure has to be inserted, the failplane segment switches activate only the segments that are actually needed for signal routing. Segmentation is performed after each I/O unit and for each cabinet individually. Extending the system is therefore not a problem, and extensions never impair its quality. The failplane segment switch is addressed directly via SCALEXIO without the user having to interact.

Free Choice of Power Supply

With a SCALEXIO system, the battery simulation can be addressed straight from the simulation model. The power supply unit is controlled by the DS2907 Battery Simulation Controller, which provides the current and voltage values. Up to two different power supply units can be used for each DS2907 in a SCALEXIO system.
This provides a way to implement electrical systems with different voltages to test whether the ECU behaves correctly. ■