Lessons learned: Experience of a SMP2 compliant Hardware-In-the-Loop simulation framework

A. Weihusen, W. Bothmer, A. Bittkau, G. Robbers

OHB System AG
Universitätsallee 27-29
28359 Bremen, Germany
Email: andreas.weihusen@ohb.de

INTRODUCTION

The simulation of satellite systems plays an increasing role in the support of several engineering and operational activities during the lifecycle of a satellite programme. In order to reduce the development effort, costs and risks for software-based simulators in satellite programmes, it is beneficial to maximise the reuse of simulation software items for different use cases. OHB has adopted the SMP2 (Simulation Modelling Portability 2) Standard [1] to facilitate such a model reuse among the different simulator facilities of a project, as well as from project to project. SMP2 is widely used in the European space sector and is constantly evolving, as shown by the recent publication of the SMP standard [2]. Because simulation technologies also rapidly advance in other industrial domains, like avionics and automotive, it is important to identify any opportunity to leverage those technological advancements also in the space sector. For simulation in combination with hardware, the FMI standard [7] has been established, which originates in the automotive sector but also seems applicable in the space domain. In the following, OHB's experience in using SMP2 and FMI together is described. A similar approach has been investigated in parallel in the "SIMULUS Next Generation" project [3].

Over the last years, OHB developed its own SMP2 compliant simulation runtime *Rufos*. This was first presented at SESP 2015 [4] and has been extended afterwards to the major component of OHB's 'Software Base Simulator' [5]. This *Base Simulator* approach was first used within the *SARah* satellite mission for the development of three different simulator facilities:

- A *Software Validation Facility* (SVF), used to develop and validate on-board software,
- an *Assembly Integration & Verification Simulator* (AIVS), used to emulate non-available hardware in an EGSE environment,
- and a *Training, Operations and Maintenance Simulator* (TOMS), used to validate flight control procedures, train the flight control team and support operations.

The SARah SVF and TOMS simulators are delivered as pure software applications with software interfaces to the corresponding monitoring and control facilities, while the AIVS must also provide hardware interfaces to the EGSE environment, such as electrical inputs/outputs and a MIL-STD-1553B bus interface. Moreover, the AIVS must provide hard real-time simulation capabilities to support hardware-in-the-loop testing.

Fig. 1 illustrates the different designs of SVF and AIVS. The TOMS design is similar to the SVF design.

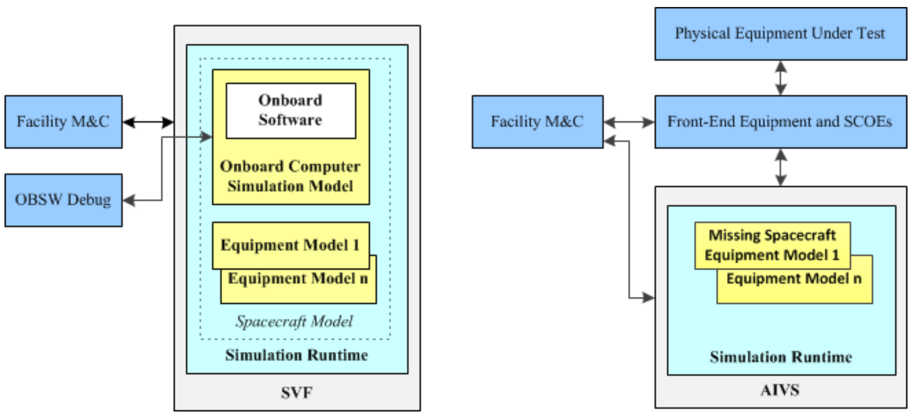


Fig. 1. Comparison of SVF and AIVS design

SIMULATOR IMPLEMENTATION

The implementation of the SARah AIVS can be divided into three main steps: the selection of an appropriate hardware platform, the adaptation of the simulation runtime to the selected hardware platform and the implementation of appropriate interfaces between the software models and the hardware I/O cards. Each of these steps will be described in the following.

Hardware Platform

The chosen hardware platform of the SARah AIVS is the SCALEXIO system from dSPACE. This system provides a highly flexible HIL simulation platform, which has been customized to the specific needs of the SARah EGSE team.

The SARah AIVS system is composed of a processing unit, I/O boards and electronics. The processing unit is the computing core. It is based on an industrial PC with an Intel XEON processor. The SCALEXIO system uses the real-time operating system QNX. The I/O boards and electronics were configured and built according to the hardware interface requirements of the SARah AIVS. The following hardware interface types are provided:

- High voltage high power command (HV-HPC) receiver
- Analogue signal monitor (ASM) source
- ASM receiver
- Bi-level discrete monitor (BDM) source
- Bi-level switch monitor (BSM) source
- Variable electrical load
- RS-422 UART
- MIL-STD-1553B

Accompanying the hardware is a software suite from dSPACE that contains specific tools to support the development, configuration and operation of the SCALEXIO simulation. These tools are installed and executed on an external PC that is connected to the SCALEXIO system via LAN.

Although the SCALEXIO system does not support SMP2 natively, OHB determined through analysis that developing SMP2 support is possible by encapsulating the simulation components in a *Functional Mock-up Unit* (FMU), which exchanges data with the SCALEXIO I/O model via *Functional Mock-Up Interface* (FMI) variables as defined in [7].

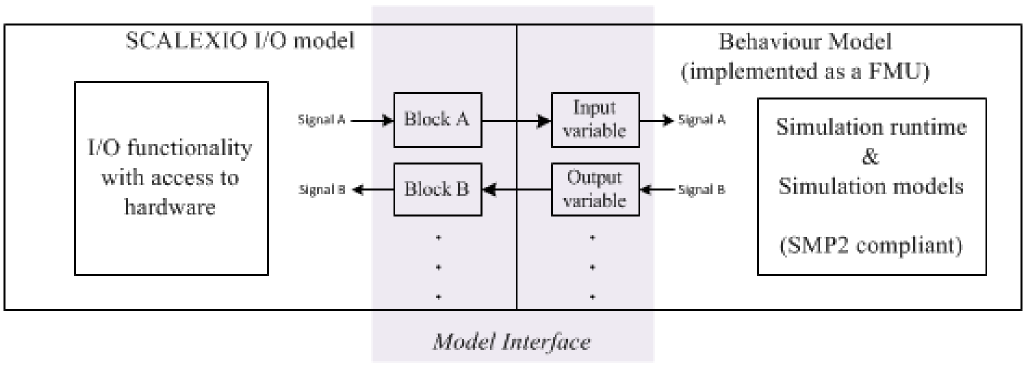
Generally, a simulation is executed as a real-time application on the SCALEXIO processing unit. This application is built from a 'real-time model', which consists of two parts: the 'I/O model' and the 'behaviour model'. The interface between the two parts is called the 'model interface' and can be seen in Fig. 2.

The **I/O model** is implemented in the software *ConfigurationDesk*, which is part of the software suite provided by dSPACE. The model defines the functions for measuring and generating I/O signals with access to the real-time hardware. For example, an analogue input can be created in the I/O model by linking the voltage measured at a specified channel of a specified board to a named variable of type Float64 in the model interface.

The I/O model has been prepared and delivered by dSPACE according to the specification of the SARah AIVS.

The **behaviour model** had to be implemented by OHB. It contains the algorithm of the controlled system. In case of the SARah AIVS, the behaviour model includes the SMP2 simulation environment and the equipment models. The implementation of the model can be done in three ways on the SCALEXIO system: as MATLAB/Simulink model, as V-ECU (virtual Engine Control Unit) or as FMU (Functional Mock-up Unit). As SMP2 is implemented in C++, and the FMU interface is implemented in C, an FMU is the most suitable method to implement the behaviour model for the AIVS.

The SCALEXIO software stack ensures that data between the I/O model and behaviour model is interchanged in accordance with the model interface.



SCALEXIO I/O model
Behaviour Model (implemented as a FMU)
I/O functionality with access to hardware
Signal A
Block A
Input variable
Signal A
Signal B
Block B
Output variable
Signal B
Simulation runtime & Simulation models (SMP2 compliant)
Model Interface

Fig. 2. SCALEXIO real-time model components

Adapting Rufos

The initial development of OHB's SMP2 compliant simulation runtime Rufos focused on its use for purely software-based simulators, which are delivered as desktop applications and without any specific support for use in HIL simulators. At that time, the only HIL simulation platform known to support SMP2 was EuroSim, which was developed by a consortium of Airbus D&S Netherlands B.V., Altran Netherlands B.V. and the Netherlands Aerospace Centre NLR. The fact that Rufos and the required models have already been used successfully in the SARah SVF pushed the decision to adapt these components to the SARah AIVS hardware platform as well.

Since both operating system (Linux for SVF, QNX for AIVS) are POSIX-compatible, only a few adaptations concerning the toolchain were required to port Rufos and its dependencies to the 32-bit QNX 6.5.0 real-time operating system of the SCALEXIO processing unit. This work included:

- updating and patching the QNX toolchain for C++11 support
- patching and cross-compiling boost and python libraries (Rufos dependencies)
- cross-compiling Rufos

Additional changes were made to ensure real-time performance, such as:

- adding a memory pool so that dynamic memory can be allocated with constant execution time
- removing system calls with non-bounded execution time
- removing writing of log files to disk

Afterwards, Rufos was encapsulated in a FMU to provide the behaviour model for the SCALEXIO environment. A FMU denotes a model that implements the Functional Mock-up Interface (FMI), a free, tool-independent standard that defines an interface for the coupling of simulation tools [7].

In accordance with the FMI standard, the Rufos FMU implements an initialisation function, a step function that is to be executed periodically, and getters and setters for input and output variables, which form the interface to the I/O model. The initialisation function initialises the SMP2 simulation runtime and models (without a MMI). In the step function, the simulation time is advanced equal to the elapsed Zulu time (the real clock time based on the computer's clock) since the function was last executed and the simulator events that have been scheduled for the elapsed simulation time are executed.

The SCALEXIO software stack manages the periodic execution of the step function and synchronises the values of the FMU input and output variables with the I/O model between every step. Fig. 2 illustrates the relationship between the Rufos FMU and the I/O model.

Software/Hardware Interfaces for SMP2 Models

SMP2 compliant simulation models of the satellite units were first developed for use in the SARah SVF. The electrical harness (i.e. electrical lines and communication buses) is simulated by corresponding line models, which are SMP2 compliant as well. The interfaces between the equipment models and the line models are implemented as 'OHB Simulation Interfaces'. These are standardised software-based interfaces that define how the electrical interfaces are to be simulated. They are similar in scope to those defined by the 'ISIS Training, Operation and Maintenance Interface

Specification' [8] and comparable to SystemIF Ports of Spacecraft Simulation Reference Architecture [9]. Equipment models include a reference to a corresponding OHB Simulation Interface for each electrical interface that is being simulated. An example of such a connection in the SVF is illustrated in the upper part of Fig. 3.

The unit simulation models can be reused in the SARah AIVS without modification or rework because the OHB Simulation Interfaces hide the implementation of the electrical interfaces from the equipment model. The equipment models depend only on the C++ standard libraries, SMP2 interfaces (provided by Rufos) and OHB Simulation Interfaces. They have been designed with SVF/AIVS interoperability from the start.

The major adaptation for AIVS was the development of new SMP2 compliant AIVS line models that facilitate the software/hardware interface. An AIVS line model implements an OHB Simulation Interface at one end, so that it can connect to an equipment model. The other end of the AIVS line model is connected to the I/O model (via FMU input and output variables), joining the signal chain to the physical equipment. An exception is the AIVS M1553 line model, which is connected directly to the SCALEXIO M1553 hardware card instead of the I/O model. This line model uses the C++ API of the M1553 hardware driver directly and implements hardware interrupt handlers to provide the higher responsiveness required for MIL-STD-1553B communications.

To add an equipment model to the AIVS, the only additional work is to configure the signal chain for each of its electrical interfaces. Each simulated electrical interface is linked to an instance of an AIVS line model via an OHB simulation interface. The AIVS line model in turn is linked to the I/O model via FMU input and output variables. The I/O model defines a link to the SCALEXIO hardware. An example of this is illustrated in the lower part of Fig. 3. This results in output signals from an equipment model that control SCALEXIO hardware outputs, and inputs from the SCALEXIO hardware that are passed as inputs signals to the equipment model. Fig. 3 shows a comparison of the signal chains in SVF and AIVS for an example model.

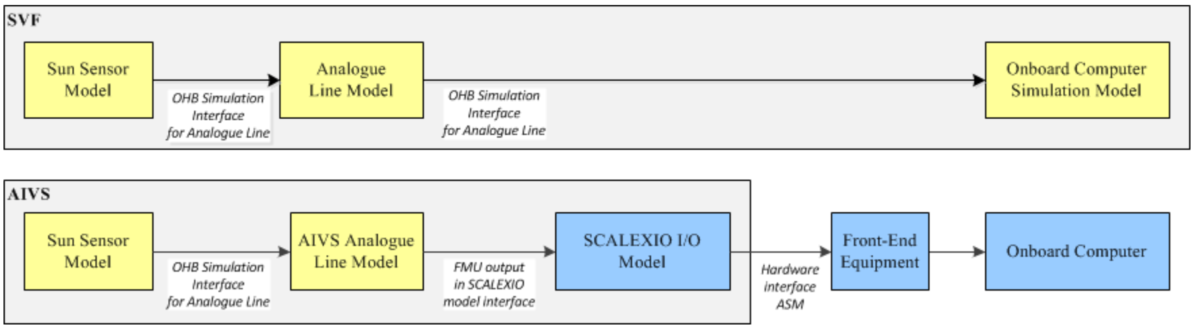

SVF
Sun Sensor Model
OHB Simulation Interface for Analogue Line
Analogue Line Model
OHB Simulation Interface for Analogue Line
Onboard Computer Simulation Model
AIVS
Sun Sensor Model
OHB Simulation Interface for Analogue Line
AIVS Analogue Line Model
FMU output in SCALEXIO model interface
SCALEXIO I/O Model
Hardware interface ASM
Front-End Equipment
Onboard Computer

Fig. 3. Comparison of signal chains in SVF and AIVS

RESULTS

The development of the SARah AIVS has been completed and the simulator is now used by the EGSE team for functional testing at subsystem and satellite level, simulating unavailable physical equipment in the engineering model (EM) of the SARah satellite. The AIVS' behaviour model includes instances of ten different equipment models, simulating AOCS components as well as payload components. The simulation models were reused from the SARah SVF without further modifications.

The AIVS is connected to the EM by supporting the following electrical interfaces:

- HV-HPC receiver, for pulse widths ≥ 50 milliseconds
- ASM source and receiver
- BDM and BSM source
- Variable electrical load
- MIL-STD-1553B
- RS-422 UART

The simulation runs in steps at 1000 Hz, meaning that simulation events are executed within one millisecond of their scheduled time. Operation and monitoring of the AIVS are performed through PUS packets carried over the EDEN protocol, in the same way as other SCOEs (Special Check-Out Equipment).

The SARah AIVS passed its interface tests and is now regularly used in HIL campaigns. According to the good experience with the initial simulator version, the EGSE team even requested to add additional equipment models to the AIVS that had not originally been foreseen; the integration of such models now typically takes around one week.

LESSONS LEARNED

The following experiences were gained with the development of the SARah AIVS:

The usage of the FMI standard for exchanging data between the behaviour model and the I/O model has proven as an efficient way to integrate the SVF simulation models into the AIVS. This approach granted the real-time capability of the simulator within the HIL setup.

The reuse of SMP2 compliant simulation components decreased the development time of the behaviour model considerably. As described before, only some adaptation steps were required to port the simulation runtime Rufos to the real-time OS QNX, while the simulation models could be integrated without modifications. It turned out, that some of the configuration scripts initially violated the real-time conditions, but this could be fixed by dividing the scripting functions into smaller steps that are assigned to dedicated time-slots. The general AIVS approach has proven as applicable.

The SCALEXIO system is a complete solution, consisting of a real-time capable processing unit, HW I/O interfaces and the corresponding drivers for the target OS. In addition, the system was delivered in a configuration customized to the needs of OHB with the appropriate I/O model already integrated. Accordingly, OHB did not need to undertake any further effort in configuring the HW / SW interface and could focus on the integration of the simulation runtime and the simulation models into the behaviour model.

To guarantee a stable performance, the user access to the SCALEXIO system has been restricted. This made the development of the behaviour model more challenging, because the standard tooling for development, debugging and performance analysis could not be used in the usual way. Thus, the model was composed on the standard development environment on Linux, subsequently compiled with the QNX toolchain and finally integrated to the real-time application and installed on the SCALEXIO system by the *ConfigurationDesk* software. The simulation is controlled via specific SCOE TCs that are send from the Central Checkout System (CCS) using the EDEN protocol.

The SARah AIVS was used initially in a test laboratory, where it was accessible via LAN. In this setup, it was possible to perform the software uploads, configurations and tests remotely. Later on, the AIVS moved to the integration hall for integrating it in the satellite’s EM. With this step, remote access was no longer possible. Since then software uploads, configurations and tests have to be done manually inside the integration hall, which increases the effort considerably. Moreover, the time slots for the simulator developers decreased, because the system is intensively used by the EGSE team. However, this situation is considered a general challenge when using an AIVS for co-simulation in an EGSE environment and must be taken into account in the planning.

CONCLUSIONS

The development and application of the SARah AIVS have shown that the combination of the two standards SMP2 and FMI in a real-time capable simulator is possible and works well in practice. SMP2 enables the reuse of SVF software items like runtime environment and equipment models in an AIVS, while FMI adds the interfaces to the hardware drivers of the I/O cards. The reuse of software components in purely software-based simulators (SVF, TOMS) as well as in HIL simulators (AIVS) without modification is an important rationalization step that helps to reduce development efforts and risks, thereby reducing costs while maintaining quality.

A promising continuation of this activity would be an exchange of experience with the SIMULUS NG study [3] and the identification of topics that can be explored further on in the scope of RATIO-SIM or related research activities.

REFERENCES

[1] “SMP 2.0 Handbook” EGOS-SIM-GEN-TN-0099, issue 1.2, 28.10.2005

[2] “Space Engineering – Simulation modelling platform” ECSS-E-ST-40-07C-DIR1, provided for public review, 23.10.2018

[3] P. Steele, V. Reggestad, “Evolution of the Operation Simulator Infrastructure at ESOC: SIMULUS Next Generation”, Proceedings of SESP 2017

[4] P. Froehner, A. Gamarra, M. Gehre, A. Weihusen, F. Hoffmann, D. Della Ratta, „MTG SVF: An excellent opportunity for assessing the SMP2 compatibility“, Proceedings of SESP 2015, March 2015

[5] A. Weihusen et al., “OHB’s Software Base Simulator: Efficient Development of Software-Based Simulators by Re-Use of Generic Components”, Proceedings of SESP 2017

[6] A. Trung, M. Gehre, P. Froehner, D. Della Ratta, N. Lambl, D. Lammers et al, “Developing a SMP2 compliant Hardware-In-the-Loop simulation framework”, Proceedings of SESP 2017

[7] MODELISAR consortium, Modelica Association Project “FMI - Functional Mock-up Interface for Model Exchange and Co-Simulation”, version 2.0, July 2014

[8] ISIS AIV Project Team, “ISIS Training, Operation and Maintenance (TOMS) Interface Specification”, Issue 4, September 2011.

[9] Steinle T., Eisenmann H., “Spacecraft Simulation Reference Architecture – High level Architecture and Interface Requirements”, Issue 1.3, October 2010.