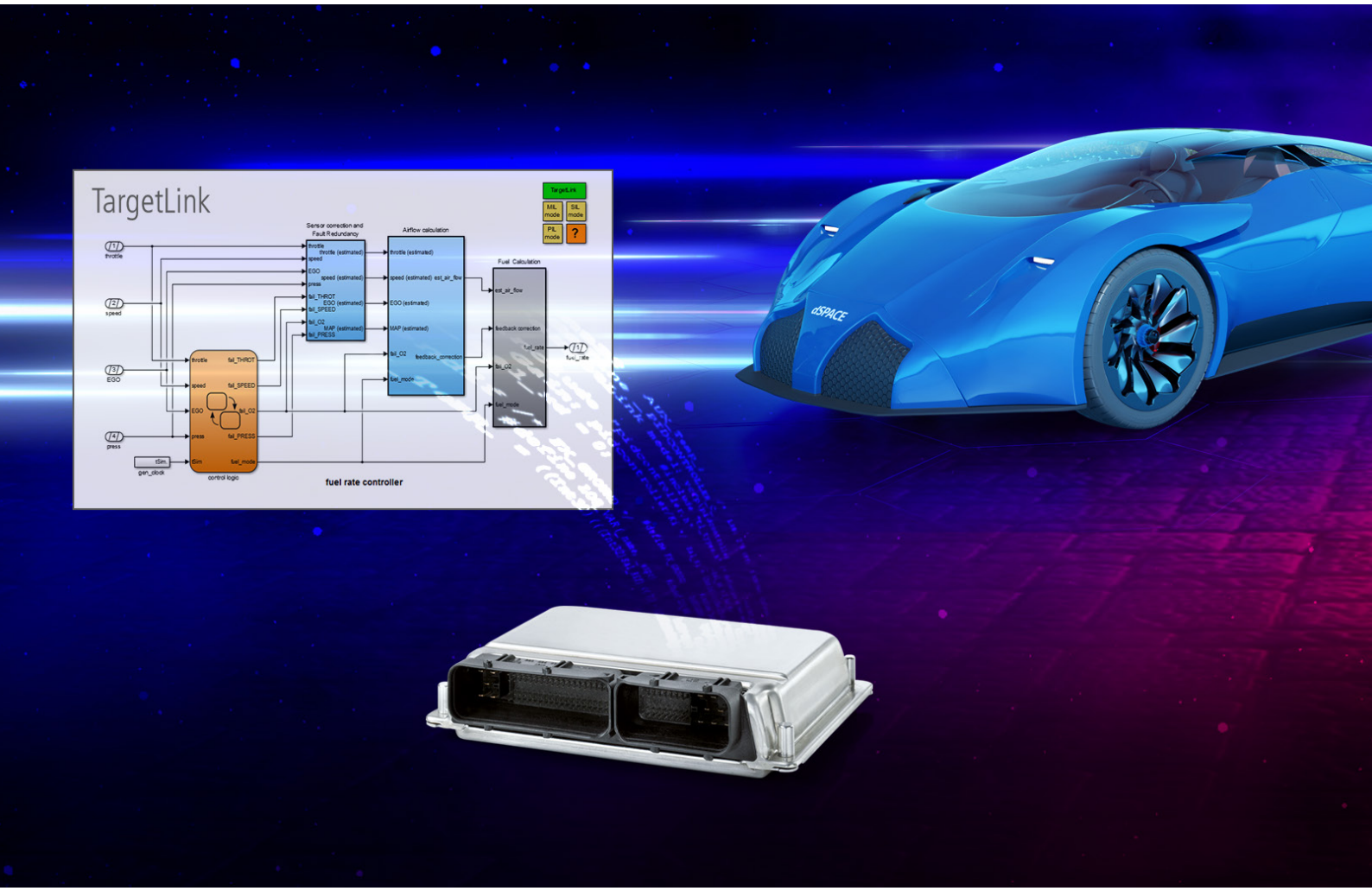


TargetLink

- High-quality production code generation directly from MathWorks® Simulink®/Stateflow®
- Powerful software design and testing features
- High-performance Classic and Adaptive AUTOSAR support
- Certified for IEC 61508, ISO 26262, and ISO 25119
- TargetLink Ecosystem – powerful tool chain for highly efficient model-based development

YOUR PARTNER IN SIMULATION AND VALIDATION

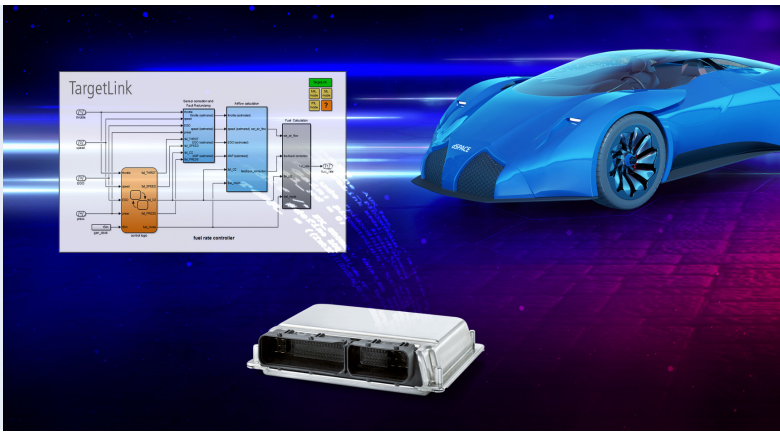
dSPACE

TargetLink

Production code generation for the highest demands

Highlights

- High-quality production code generation directly from MathWorks® Simulink®/Stateflow®
- Powerful software design and testing features
- High-performance Classic and Adaptive AUTOSAR support
- Certified for ISO 26262, ISO 25119, and IEC 61508
- TargetLink Ecosystem – powerful tool chain for highly efficient model-based development



Application Areas

Model-based design has become the established development method across many industries, and production code generation is the logical step for turning models into efficient, production-ready code. TargetLink generates production code (C code) straight from the Simulink/Stateflow graphical development environment. The C code generation options range from plain ANSI C code to optimized fixed- or floating-point code for AUTOSAR platforms. Versatile code configuration options ensure that the production code can handle processor constraints.

Key Benefits

With TargetLink, you can convert graphical models directly and deterministically into highest quality production code. Benefit from TargetLink's numerous strenghths, such as:

- Tool and code proven in practice – in countless production projects and millions of vehicles
- Best-in-class code – highly efficient, highly configurable and with fully traceable model/code dependency
- Powerful software design features, e.g., with the TargetLink Data Dictionary
- Convenient software verification due to the push-button model-in-the-loop (MIL)/software-in-the-loop (SIL)/processor-in-the-loop (PIL) simulation concept – verify your concepts and code early and avoid expensive ECU software errors
- Ideal for the automotive industry (including e-mobility, ADAS, and autonomous driving applications), and also for other industries
- Suitable for AUTOSAR and non-AUTOSAR projects
- Strong partner tools (TargetLink Ecosystem) for model design, validation, and verification
- Certified for use when developing safety-critical functions according to ISO 26262, ISO 25119, and IEC 61508 as well as derivative standards
- Tool integration with the simulation software VEOS and the system architecture software SystemDesk
- Perfect for on-target bypassing together with dSPACE's ECU interface software – closing the gap between function and production development

Main Features and Benefits

Feature	Description	Benefit
Code efficiency	■ Efficient fixed-point or floating-point production code directly from Simulink®/Stateflow® (with support for included MATLAB code, see optional module[1], p. 4)	■ Your specifications – models and diagrams – are directly translated into efficient C code
Code reliability	■ Consistent, deterministic translations of models into stress-tested C code	■ Errors such as typing errors, oversights, misunderstandings are avoided
Human readability	■ Concise, yet readable code	■ Code reviews are easy to perform
Automatic scaling	■ Intelligent scaling based on worst-case propagation of signal ranges and simulation-based scaling	■ Shortens the time-consuming and error-prone scaling process
Test mechanisms	■ Various test levels to test the production code against the specification (MIL, SIL, PIL)	■ Malfunctions are found at the earliest stage
Incremental code generation	■ Modular code generation, i.e., for specific subsystems	■ Faster code generation, preserving approved code
Model referencing support	■ Developing models on a modular/component basis	■ Distributed development by large teams is much easier, and large models can be handled more efficiently.
Multirate code	■ Full support of multirate systems with intertask communication	■ You can define tasks at block level
TargetLink Data Dictionary	■ Central container to handle variables, data structures, scaling formulas, tasks, functions	■ You can manage complex data to plan and structure your projects
Code generation straight from the Data Dictionary	■ Generation of code files and A2L files for Data Dictionary variables, independent of their use in TargetLink models	■ Simplified software integration and integration testing for code from multiple TargetLink models/subsystems and legacy variables
'Modeling Only' and ' Full-Featured' operation modes	■ The Modeling Only operation mode (not licence-protected) lets function developers prepare and design a model without generating production code for it.	■ Easy model exchange in a workgroup (Simulink® and TargetLink users)
Compliance with standards	■ Compliance with relevant standards such as ASAM-MCD 2MC (ASAP2), AUTOSAR, and MISRA	■ Quality and interoperability guaranteed
Classic AUTOSAR support	■ Support for modeling and code generation for AUTOSAR software components (SWC), and generation of SWC descriptions	■ TargetLink bridges the gap between model-based design and AUTOSAR-compliant software development
Adaptive AUTOSAR support	■ Support for modeling and code generation for Adaptive AUTOSAR-compliant C/C++ code (for supported features, please see p. 28)	■ Closely resemblant modeling styles for non-, Classic and Adaptive AUTOSAR ■ Simple and convenient Adaptive AUTOSAR code generation
Calibration data generation	■ Calibration data exported as ASAM-MCD 2MC (ASAP2) file for calibration tools	■ Automated and complete process with perfect consistency between model and calibration data
Documentation	■ Automatic model and code documentation	■ Your projects are transparent and trackable
AUTOSAR software component (SWC) container exchange	■ Exchanging AUTOSAR SWC containers between TargetLink and SystemDesk	■ Safe and convenient round trips for AUTOSAR software development ■ Access to SystemDesk simulation for proper software integration testing
Component-based development	■ Innovative interface concept for ports, measurement signals and calibration parameters	■ Easily increase software reuse across different projects
Connection to dSPACE VEOS	■ Export virtual ECUs from TargetLink and run them in VEOS in combination with other virtual ECUs and standard dSPACE tools like ControlDesk	■ Easy testing and experimenting with TargetLink code
FMI support	■ Export of Functional Mock-up Units (FMUs) from TargetLink models based on the Functional Mock-up Interface (FMI) standard	■ Execution of TargetLink-generated code with offline and real-time simulators from third parties and dSPACE
On-Target Bypassing	■ Integrating new functions directly into existing ECUs (using TargetLink and dSPACE ECU interface software, see p. 35)	■ Seamless transition from prototyping to series production ■ No need to perform complex software integration

[1] *dSPACE offers the TargetLink Module for MATLAB® Code subject to a license from The MathWorks, Inc.*

Order Information

Classification	Type	Order Number
TargetLink Base Suite	Base Suite	■ TBS
Other modules	Target Simulation Module (PIL, for all supported processors)	■ TSM
	TargetLink AUTOSAR Module	■ TAS
	TargetLink Adaptive AUTOSAR Module	■ TAAS
	TargetLink Data Dictionary Manager (included in TargetLink Base Suite)	■ DSDD_MANAGER
	TargetLink Module for MATLAB® Code[1)]	■ TMMLC

[1)] dSPACE offers the TargetLink Module for MATLAB® Code subject to a license from The MathWorks, Inc.

Relevant Software

Software		
Included	Data dictionary	■ TargetLink Data Dictionary
	Designing controllers directly with TargetLink including free model exchange	■ Using the new Modeling only operation mode (as of TargetLink 5.0, this option replaces the former TargetLink Blockset stand-alone; the Modeling only mode is not licence-protected)
Required	Integrated development environment	■ MathWorks MATLAB®/Simulink®/Stateflow® ■ Compiler for host simulation included in MATLAB
	Operating system	■ www.dspace.com/go/os_compatibility
Optional	Compilers for processor-in-the-loop tests	■ Target-specific compiler for processor-in-the-loop tests with Target Simulation Module (p. 6, p. 7)
	Tool chain for On-Target Bypassing (p. 20, p. 35)	■ ECU Interface Base Package
		■ dSPACE Internal Bypassing Service
		■ Target-specific compiler (third-party product: HighTec compiler)
	Running virtual ECUs (p. 33)	■ VEOS
	ConfigurationDesk for scenarios that use dSPACE real-time systems[2)] and TargetLink	■ See p. 30
	Modeling system architecture and generating virtual ECUs	■ SystemDesk
	Comparing MathWorks® Simulink®, Stateflow®, and dSPACE TargetLink models	■ Model Compare (p. 40)
	Comparison of two versions of AUTOSAR artifacts	■ dSPACE AUTOSAR Compare (p. 46)

[2)] Currently: SCALEXIO and MicroAutoBox III

TargetLink Product Support Center

The TargetLink Product Support Center is the primary online resource for TargetLink users, providing information about releases, compatibility, application notes, additional utilities, TargetLink Known Problem Reports, etc. The address is *www.dspace.com/tlpsc*

NEW: TargetLink 5.2

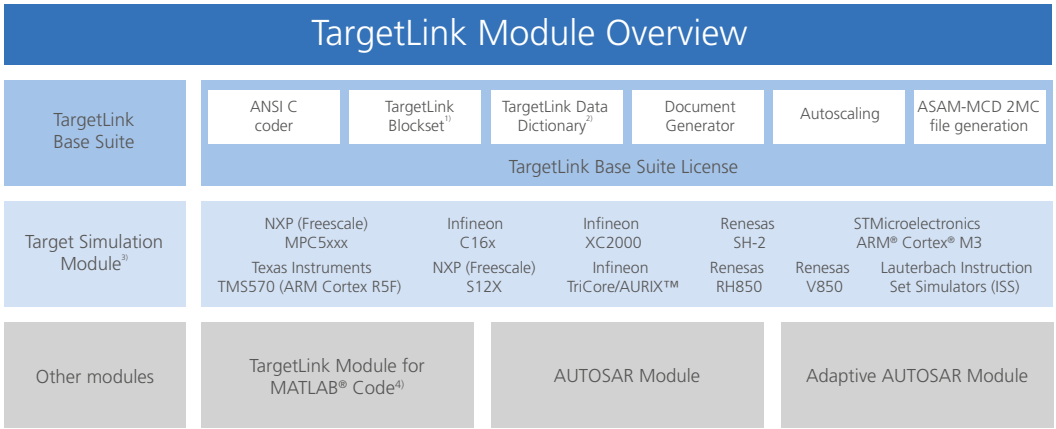
Feature Area	Improvement
Data formats	■ Dynamic access to array-of-bus signals from within the function model, allowing field elements in object lists (e.g., for radar and lidar sensors) and in REST API applications to be addressed directly. ■ Support of 64-bit data types at ports and custom code blocks to work with large values (e.g., as used in ADAS/AD applications).
Adaptive AUTOSAR	■ Access to Adaptive AUTOSAR error handling.
Classic AUTOSAR	■ Generation of frame models for AUTOSAR SWC compositions, which is ideal for distributed development.

Usability	■ Data Dictionary Manager
	  ■ Error pane lets you jump to specific objects
	  ■ Tabs now have a close button.
	  ■ Improved sorting capabilities.
	■ Exceptions-based TargetLink Data Dictionary API for cleaner tool chain code.
	■ Automatic upgrade of Data Dictionary files for smooth automation.

Module Overview

TargetLink Modules

TargetLink is available as a base suite plus additional modules, so that you can adapt it to your needs.



1) Usable in Modeling Only operation mode without license.
2) The Data Dictionary Manager is also available as a stand-alone license, e.g., for use with the stand-alone blockset.
3) Selection of major microcontroller families supported. For a complete list, please refer to www.dspace.com/go/tlpil
4) dSPACE offers the TargetLink Module for MATLAB® Code subject to a license from The MathWorks, Inc.

TargetLink Base Suite

- Highly efficient ANSI C code generation from Simulink/ Stateflow
- For all microcontrollers with ANSI C compiler
- Fixed-point code, floating-point code or a mixture of both
- TargetLink Data Dictionary (p. 18)
- TargetLink Blockset (p. 10, p. 16)
- Autoscaling (p. 11, p. 13-14)
- Code coverage analysis (p. 24)
- Modular development and code generation (p. 20)

Target Simulation Module (optional)

- Test your generated code in PIL simulation on the target microcontroller (for supported processors and evaluation boards see p. 7)

TargetLink AUTOSAR Module (optional)

- Support for the development of Classic AUTOSAR-compliant software (p. 26)

TargetLink Adaptive AUTOSAR Module (optional)

- Support for the development of Adaptive-AUTOSAR-compliant software (p. 28)

Supported Processors and Evaluation Boards

For processor-in-the-loop simulation, TargetLink supports the most common processors for embedded applications, especially in the automotive field.

Processor Family	Compiler Supported by Target Simulation Module	Evaluation Boards Supported by TargetLink
NXP (Freescale) MPC5xxx	Green Hills, NXP (Freescale) CodeWarrior, GNU and Wind River compilers	NXP (Freescale) MPC5748GEVB
NXP (Freescale) S12X	Cosmic and NXP (Freescale) CodeWarrior compilers	NXP (Freescale) EVB9S12XEP100
Infineon C16x	Tasking compilers	i+ME eCAN C167 CR
Infineon TriCore	Tasking and GNU[1)] compilers	Infineon TriBoard TC1766, Infineon TriBoard TC1767, Infineon TriBoard TC1796 and Infineon TriBoard TC275 Infineon TriBoard TC397 Lauterbach Instruction Set Simulator
Infineon XC2000	Tasking compilers	Infineon SK-EB XC2287
NVIDIA Tegra X2	GNU C Compiler	NVIDIA Jetson TX2
Renesas RH850	Green Hills compilers	Renesas YRH850F1L
Renesas RL78	IAR Systems Embedded Workbench, Renesas CubeSuite+	Renesas TBRL78F14F10PP
Renesas SH-2	Renesas compilers	Renesas SDK7058 and SDK72513
Renesas V850	Green Hills compilers	Renesas AB_050_Fx4_70F4012
ST Microelectronics SPC58xx	Green Hills; Wind River Diab	ST Microelectronics SPC58NG84EVB ST Microelectronics SPC58NN84EVB
STMicroelectronics ARM Cortex M3	Keil compilers	Emerge-Engineering MEDKit on ARM Lauterbach Instruction Set Simulator
Texas Instruments ARM Cortex R5F	Texas Instruments Code Composer	Studio Texas Instruments LAUNCHXL2570LC43

[1)] *Only for TC275.*

Some of the evaluation boards need to be modified (loader, external RAM, etc.). Please order them through dSPACE to ensure a correct board setup.

For more information on software compatibility with target compilers and evaluation boards, please refer to: *www.dspace.com/go/tlpil*

TargetLink Engineering Services

Our engineering portfolio includes special TargetLink customer services, for example:

- Hands-on support during introduction of TargetLink
- Support during evaluations and pilot projects
- Customer-specific TargetLink training
- Integrating TargetLink into your development processes
- Model analysis and advice
- Support for developing Classic- and Adaptive-AUTOSAR-compliant software and integrating Classic AUTOSAR-software components
- Support for integrating the generated code in the ECU's software environment
- Tool chain development and maintenance
- Process consulting

High-Quality Production Code Generation

Fundamentals of High-Quality Production Code

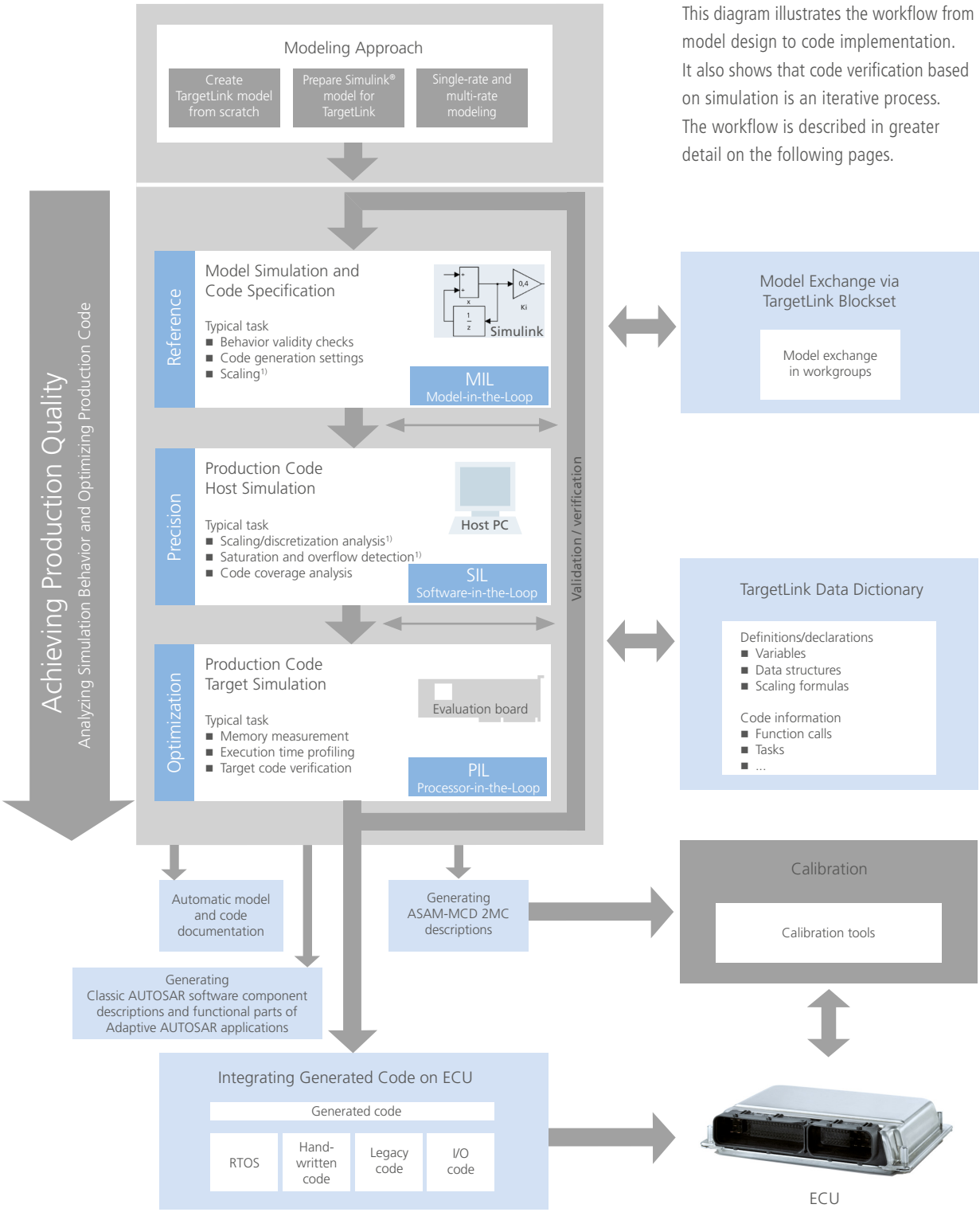
Why Code Quality Counts

Compared to code used in rapid control prototyping, where developers need to have a lot of freedom when testing new concepts, production code for electronic controls in vehicles, planes, and for any other real-use industrial application needs to fulfill very strict requirements in terms of quality. For example, developers have to be sure that the code is highly optimized, that it can be easily integrated with other software, that it is readable in the present and in the future to ensure maintainability, that all connections between requirements, model and code are clear and comprehensible, and that the code fulfills the relevant standards. In short: topmost reliability, efficiency, security, maintainability, traceability, and – concerning the generation process – reproduceability (i.e., the same model leads to the same code) is required.

TargetLink for High-Quality Production Code

The production code generator dSPACE TargetLink is an extremely powerful software system which ensures that your code always achieves superior quality – and it does so while letting you take your individual requirements and the relevant standards into account. Despite TargetLink's comprehensive features, the typical steps in generating your production code are easy to follow (p. 9, p. 10), with many options to fit the process to your individual environment. Typical steps include the control design, the iterative validation and verification (based on simulation), and the code integration on the ECU. Furthermore, TargetLink interacts smoothly with other tools from dSPACE and with tools from other vendors, which together form the TargetLink Ecosystem (p. 38).

Typical Steps in Generating Production Code



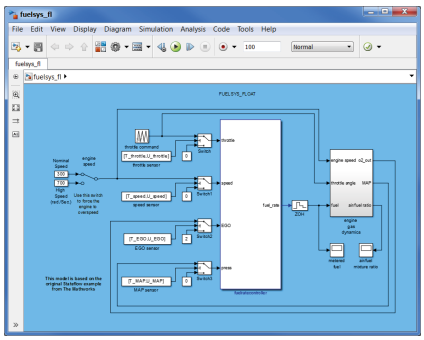
This diagram illustrates the workflow from model design to code implementation. It also shows that code verification based on simulation is an iterative process. The workflow is described in greater detail on the following pages.

1) *Valid only for fixed point software.*

Workflow

Control Design and Function Prototyping

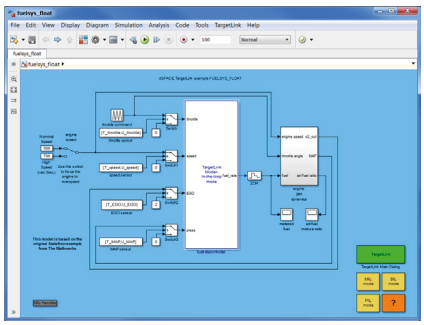
Control design starts with creating a control model in the integrated design environment MATLAB/Simulink/Stateflow. Before production code generation with TargetLink, you can use dSPACE prototyping systems to carry out convenient function prototyping and validation of your new ECU control algorithms. When the functionality of an existing production ECU needs to be extended, it is possible to efficiently merge function prototyping and production software development with the dSPACE on-target bypassing tool chain (p. 20, p. 35), which also includes TargetLink. Furthermore, it is possible to validate production code generated with TargetLink on SCALEXIO and MicroAutoBox III real-time hardware (p. 31).

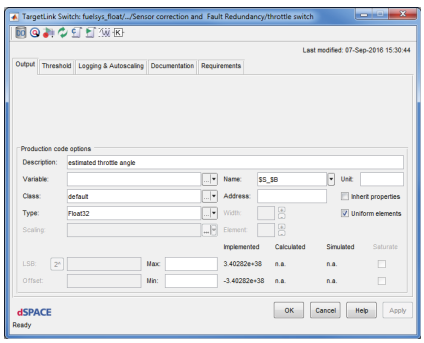


Simulink model with control algorithm.

Using the TargetLink Block Library

To implement the control algorithms in C code, TargetLink provides a dedicated block library. TargetLink blocks contain additional data for code generation, such as the scaling information for fixed-point variables, variable classes, variable names, etc. A utility automatically replaces the Simulink controller model with blocks from the TargetLink block library. The process is reversible without any data losses. If you use the Modeling Only operation mode (p. 18) during control design, conversion is not required.

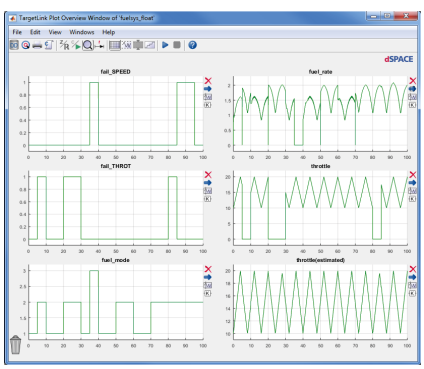




TargetLink model (above) with TargetLink blocks to select the simulation mode and relevant production code options which can be specified directly in the TargetLink block dialogs (below).

Model-in-the-Loop Simulation on Host PC

To verify the implemented control algorithms, TargetLink provides different simulation modes (p. 22). The model-in-the-loop simulation (floating-point) serves as a reference for subsequent steps and provides the minimum and maximum values for variables as a basis for subsequent fixed-point scaling if required.



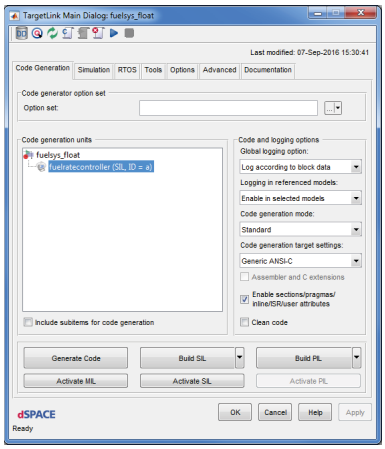
Simulation results in a plot overview.

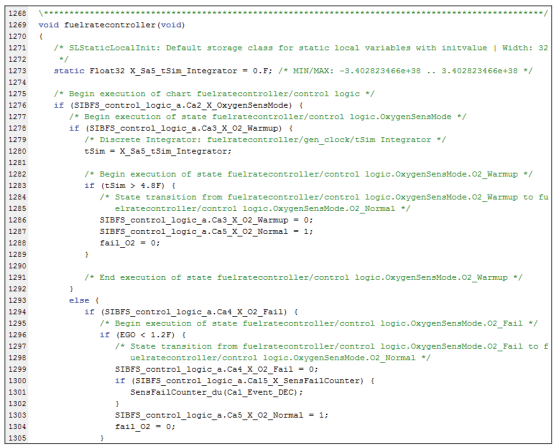
Automatic or Manual Scaling

If you want to generate fixed-point code, the scaling has to be specified. You can use manual scaling, simulation-based autoscaling or worst-case autoscaling. You can choose from a broad range of scaling options (p. 13-14) for each individual TargetLink block.

Code Generation

The TargetLink Base Suite generates highly efficient ANSI C code for a controller model at the click of a button.



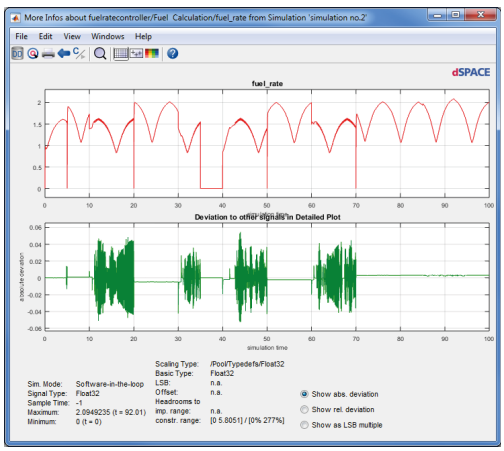


```
1268  \*************************************************************************/
1269  void fuelratecontroller(void)
1270  {
1271    /* SLStaticLocalInit: Default storage class for static local variables with initvalue | Width: 32
1272     */
1273    static Float32 X_Sa5_tSim_Integrator = 0.F; /* MIN/MAX: -3.402823466e+38 .. 3.402823466e+38 */
1274
1275    /* Begin execution of chart fuelratecontroller/control logic */
1276    if (SIBFS_control_logic_a.Ca2_X_OxygenSensMode) {
1277      /* Begin execution of state fuelratecontroller/control logic.OxygenSensMode */
1278      if (SIBFS_control_logic_a.Ca3_X_O2_Warmup) {
1279        /* Discrete Integrator: fuelratecontroller/gen_clock/tSim Integrator */
1280        tSim = X_Sa5_tSim_Integrator;
1281
1282        /* Begin execution of state fuelratecontroller/control logic.OxygenSensMode.O2_Warmup */
1283        if (tSim > 4.8F) {
1284          /* State transition from fuelratecontroller/control logic.OxygenSensMode.O2_Warmup to fu
1285             elratecontroller/control logic.OxygenSensMode.O2_Normal */
1286          SIBFS_control_logic_a.Ca3_X_O2_Warmup = 0;
1287          SIBFS_control_logic_a.Ca5_X_O2_Normal = 1;
1288          fail_O2 = 0;
1289        }
1290
1291        /* End execution of state fuelratecontroller/control logic.OxygenSensMode.O2_Warmup */
1292      }
1293      else {
1294        if (SIBFS_control_logic_a.Ca4_X_O2_Fail) {
1295          /* Begin execution of state fuelratecontroller/control logic.OxygenSensMode.O2_Fail */
1296          if (EGO < 1.2F) {
1297            /* State transition from fuelratecontroller/control logic.OxygenSensMode.O2_Fail to f
1298               uelratecontroller/control logic.OxygenSensMode.O2_Normal */
1299            SIBFS_control_logic_a.Ca4_X_O2_Fail = 0;
1300            if (SIBFS_control_logic_a.Ca15_X_SensFailCounter) {
1301              SensFailCounter_du(Ca1_Event_DEC);
1302            }
1303            SIBFS_control_logic_a.Ca5_X_O2_Normal = 1;
1304            fail_O2 = 0;
1305          }
```

TargetLink Main Dialog and an example of generated code.

Verification on Host PC via Software-in-the-Loop Simulation

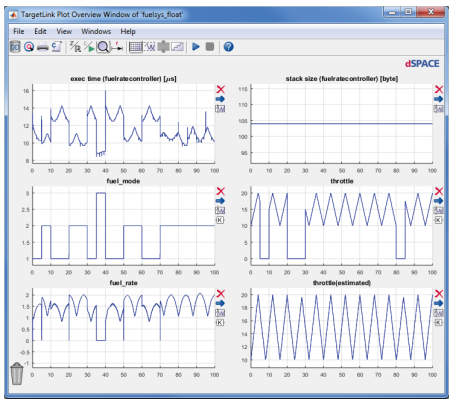
By means of a software-in-the-loop simulation on a host PC, you can compare the behavior of the generated code with the reference data obtained in a model-in-the-loop simulation. TargetLink offers a graphical user interface, where you can select the signal histories of blocks for detailed analysis.



Results from MIL and SIL simulation.

Verification on Target Processor via Processor-in-the-Loop Simulation

Using the optional Target Simulation Module (p. 6, p. 7), you can execute a processor-in-the-loop simulation to verify the generated code on an evaluation board equipped with the same target processor as the final ECU. Successful verification of a processor-in-the-loop simulation with a model-in-the-loop simulation and a software-in-the-loop simulation ensures the software quality of the generated code. TargetLink also provides information on the code size, the required RAM/ROM, and the stack consumption as it evolves over time. The execution time can be displayed as well.

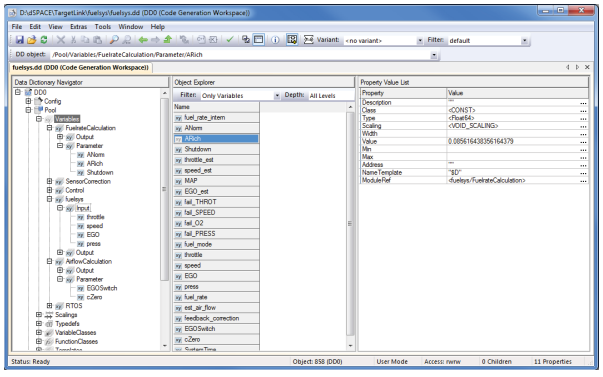



```
13
14  ****************************************************************
15  *****  Summary for module: fuelratecontroller
16  ****************************************************************
17      Functions                                          [bytes]
18  ----------------------------------------------------------------
19      Tab2DIntp2I1T1_a                                      288
20      Tab2DS17I2T4169_a                                     372
21      TabIdxS49T1_a                                         132
22      Ca26_Running_ex                                       128
23      FuelingMode_du                                        544
24      SensFailCounter_du                                    348
25      fuelratecontroller                                   2404
26
27
28  ****************************************************************
29  *****  Model summary                                 [bytes]
30  ****************************************************************
31      RAM                                 =       2882
32      ROM without libraries               =       7065
33      ROM with libraries                  =       7065
34
```

Results from PIL simulation (left) and code summary (right).

Data Dictionary

The file-based TargetLink Data Dictionary (p. 18), which is included in the TargetLink Base Suite, allows you to reuse code properties and clearly separate them from models.



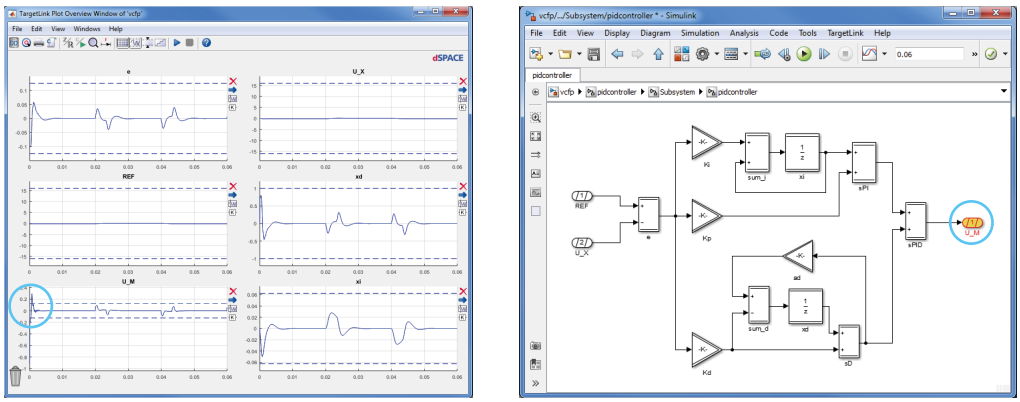
TargetLink Data Dictionary

Excellent Code Properties

Highly Efficient Production Code

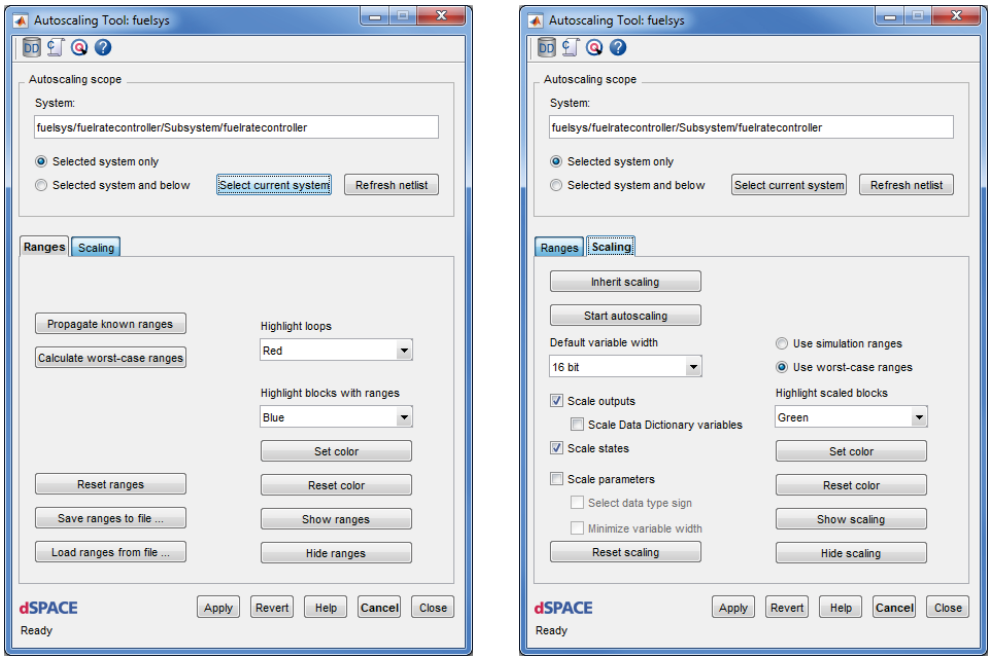
Fixed-Point and Floating-Point Code

Regardless of the application type (fixed-point or floating-point), TargetLink generates highly efficient production code for your embedded system. Multiple scaling properties give ample choices to fine-tune fixed-point code to the conflicting requirements of low execution time, high computational precision, and overflow avoidance. If an overflow occurs, TargetLink's overflow detection shows the exact location in the block diagram. The problem can be corrected right away.



Plot overview window (left) indicating an overflow of an outport block.

To shorten the time-consuming and error-prone manual scaling process, TargetLink provides an Autoscaling Tool. It supports you in finding suitable ranges and calculating appropriate scaling parameters. A scaling task that took days and weeks in the past can now be done in minutes and hours.



Dialogs to carry out automatic scaling.

Scaling Variables – A Closer Look

As a scaling method, TargetLink offers the two-coefficient linear scaling, which is widely used in embedded control applications. The properties for specifying fixed-point scalings in TargetLink are:

- Data type
- Power-of-two scaling factor or arbitrary scaling factor
- Offset value
- Constraint values
- Bit safety margins
- Saturation options

While fixed-point scaling can be done manually by a software engineer, in most instances it is left to TargetLink's autoscaling tools.

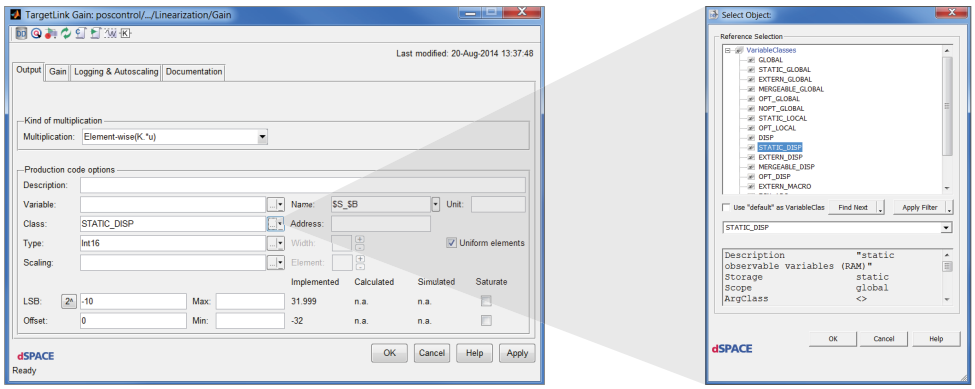
Two autoscaling procedures are available:

- Simulation-based autoscaling – benefit from maximum computational precision
- Worst-case autoscaling – no plant model required and overflows automatically prevented

Moreover, TargetLink also supports scaling with scaling formulas and type definitions.

Highly Configurable Code

TargetLink code is easily readable and includes helpful comments. Comprehensive configuration options give you full control over variables, functions, and file naming, as well as the flexibility to partition the code into functions and files to keep the structure logical and manageable.



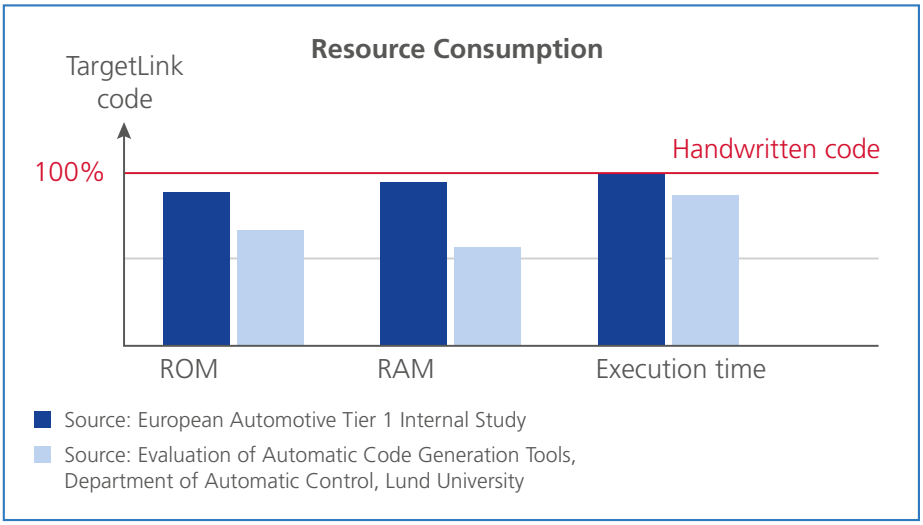
Variable class specification for a Gain block.

For the integration of external code, TargetLink offers a wide variety of specification options, e.g., on the block diagram level for easily interfacing with external code, such as device drivers, or with any other routine written in C or assembler.

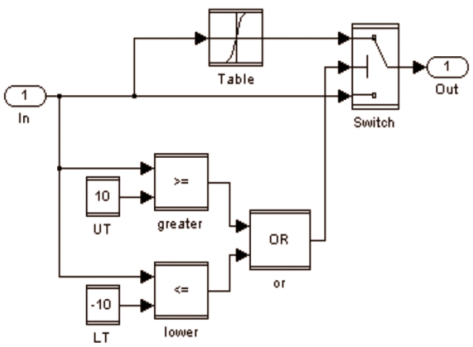
Company-specific programming style guides can be easily applied via TargetLink´s flexible code output formatting, e.g., by using XML configuration files or XSL style sheets.

Powerful Code Optimization

TargetLink generates highly optimized ANSI C code that is just as efficient as optimized handwritten code. This is achieved by various optimization techniques, such as standard optimization techniques that are also used by modern C compilers.



TargetLink's interblock optimization gives the generated code a human touch, because it combines code in a very similar way to what a skilled software engineer would do.



Non-optimized Code:

```
bool1 = (In >= 10);
bool2 = (In <= -10);
bool3 = bool1 || bool2;

tmp1 = table_lookup(Table, In);
tmp2 = in;

if (bool3)
   out = tmp1;
else
   out = tmp2;
```

TargetLink Code:

```
if ((in >= 10 ) || (in <= -10 ))
   out = table_lookup(Table, in);
else
   out = in;
```

Example of interblock optimization.

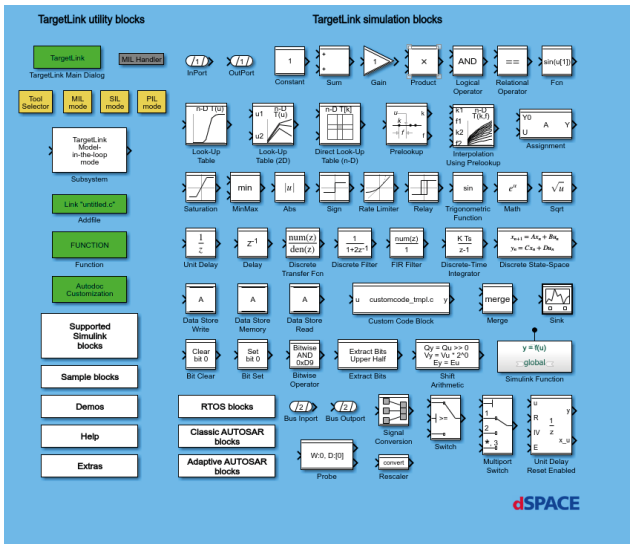
For more complex blocks, TargetLink uses code from an internal code pattern library during the code generation process, ensuring that the code for complex blocks is also highly efficient.

Control and Software Design

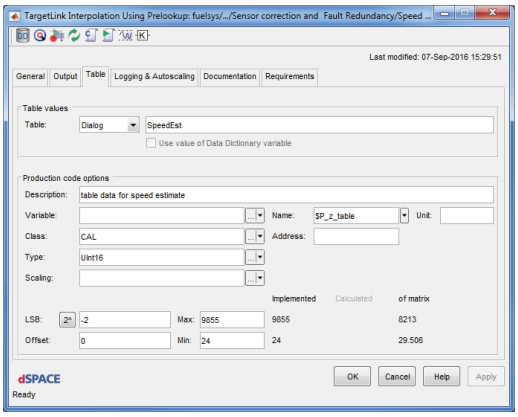
Designing Controller Models Directly with TargetLink

TargetLink Block Library: Extended, Implementation-Specific Functionality

The Simulink® block library is very powerful in simulation tasks, providing all necessary specification features. When it comes to code generation, however, more information is needed for each block. For example, the blocks need additional capabilities for fixed-point simulation. That is why TargetLink comes with the TargetLink Block Library which offers a block for each supported Simulink block. The TargetLink blocks significantly enhance the functionality of the supported Simulink blocks and have an extended dialog that allows you to enter the implementation-specific information necessary for code generation. Each block also provides a means of data logging and overflow detection. A special routine automatically prepares Simulink models for code generation with TargetLink by enhancing the Simulink blocks to TargetLink blocks.



The TargetLink Block Library

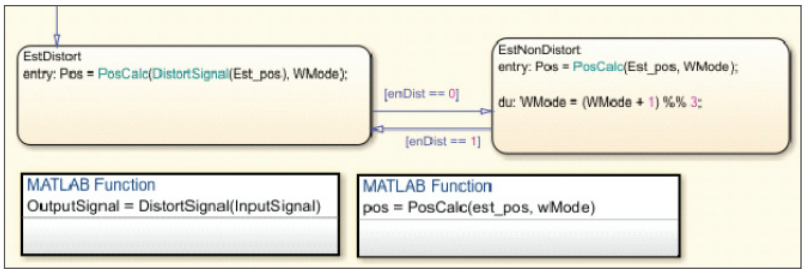


TargetLink block dialog for entering implementation-specific information such as the data type, variable name, scaling data, etc..

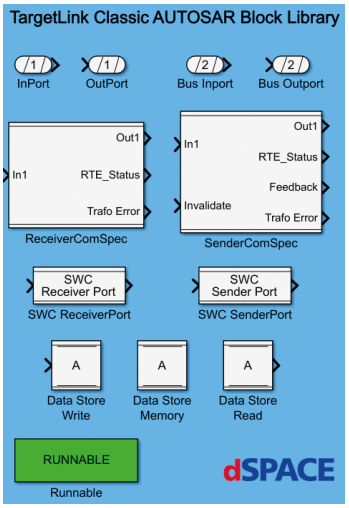
The comprehensive features of the TargetLink Block Library include:

- **Supported Simulink blocks** – several Simulink blocks can be used in a TargetLink subsystem as they are. Options include, for example, blocks for combining data sets to matrices and for specifying matrix signals in your model.
- **Stateflow® support** – TargetLink fully supports the MathWorks Stateflow toolbox.
- **MATLAB® support (separate module[1])** – TargetLink supports code generation from certain MATLAB code for selected modeling constructs. The TargetLink library contains links to key supported Simulink blocks, e.g., the MATLAB Function block.
- **TargetLink Utility blocks** – these provide access to specific features of TargetLink or further specify the code generation process for the model.
- **TargetLink Simulation blocks** – they mostly correspond to the standard Simulink blocks but include additional features necessary for production code development.
- **AUTOSAR blocks (separate modules)** – model-based design for both Classic and Adaptive AUTOSAR ECUs (p. 26, p. 28) is supported by blocks for certain communication elements.

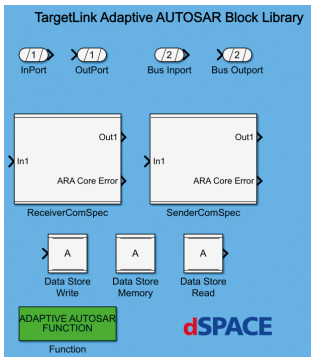
[1] *dSPACE offers the TargetLink Module for MATLAB® Code subject to a license from The MathWorks, Inc.*



Code generation from MATLAB code in Simulink/Stateflow models (requires the TargetLink Module for MATLAB® Code, see p. 4 and p. 16).



The TargetLink Classic AUTOSAR Block Library.



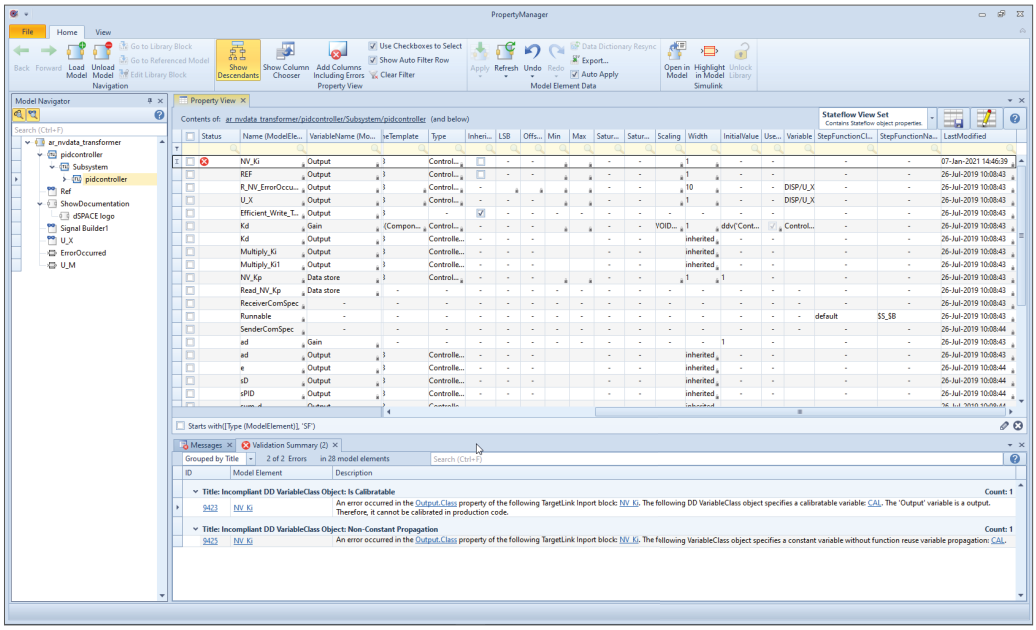
The TargetLink Adaptive AUTOSAR Block Library.

Property Manager

If you need to change the properties of a large model, TargetLink's Property Manager is a quicker alternative to manually changing the properties in the block dialogs. It displays the properties of TargetLink blocks and Stateflow objects in the model, and lets you view, filter, and modify several properties simultaneously.

Key Features:

- Intuitive graphical user interface (GUI)
- Quick access views
- Efficient filters
- Integrated validation and error visualization



The Property Manager for handling models with numerous blocks.

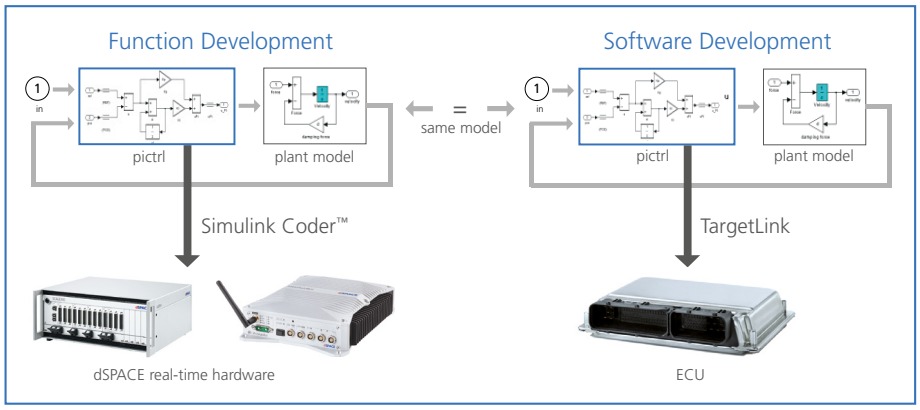
'Modeling Only' and 'Full-Featured' Operation Modes

As of TargetLink 5.0, TargetLink lets you switch between two operation modes: Modeling Only and Full-Featured. The Modeling Only operation mode (not licence-protected) lets function developers prepare and design a model without generating production code for it. In the course of development, this model can be exchanged with software developers using the Full-Featured operation mode. Users of the Full-Featured operation mode can generate production code for the model. As a result, different groups of developers can work on the same model but use different operation modes. Therefore, using both operation modes lets you ideally connect the prototyping and the production code generation development phases, and development iterations are thus easier to perform and less prone to error.

If you use On-Target Bypassing, you can benefit from the highly efficient TargetLink production code as early as the prototyping phase. For more information, please see p. 20.

'TargetLink Operation Modes

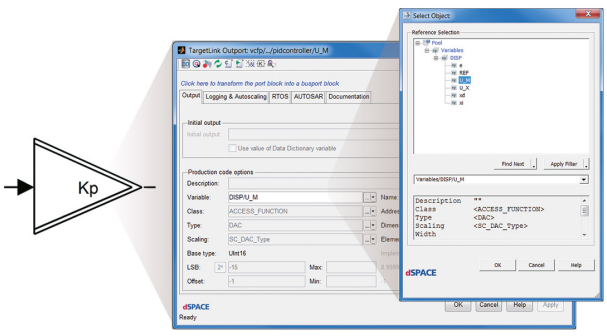
Modeling Only	Full-Featured
■ Model exchange in a workgroup (Simulink® and TargetLink users) ■ Inspecting block properties related to production code generation ■ Simulating in double-precision with Simulink® (MIL) ■ Real-time simulation on dSPACE prototyping hardware ■ Configuration of HTML document generation	■ Production code generation ■ Production code simulation on a PC, i.e., software-in-the-loop simulation (SIL) ■ Production code simulation on a target, i.e., processor-in-the-loop simulation (PIL) ■ Data logging in all modes, letting you compare data ■ Autoscaling of variables ■ Generation of HTML documents ■ Export of relevant files of a software project ■ Property Manager ■ Preparation of Simulink® systems for TargetLink (model preparation)



Function development (including rapid control prototyping) in the TargetLink Modeling Only operation mode and software development in the Full-Featured operation mode.

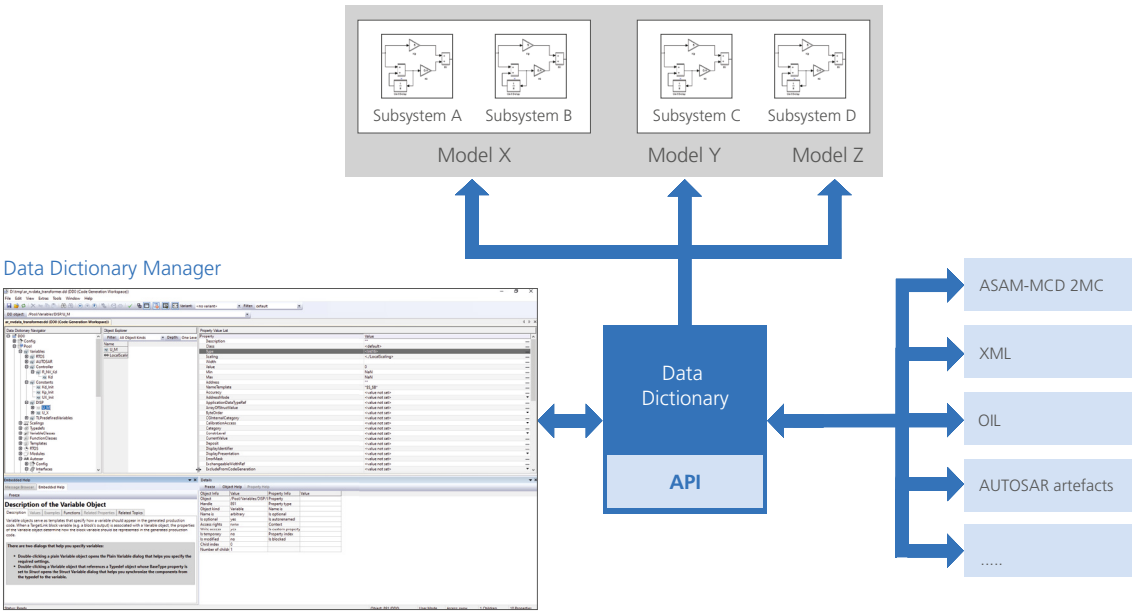
Software Design with the TargetLink Data Dictionary

The file-based TargetLink Data Dictionary is a central data container holding the relevant information for a model's design, code generation, and implementation on an ECU. Data dictionary objects can be referenced from TargetLink models. You can define and manage variables and their properties, and you can specify structured data types and use them for variable declarations. Scaling formulas can be entered and used to uniformly scale fixed-point signals and parameters in the model. You can import and export standardized or proprietary data and share the data with the calibration system.



To specify the properties of the code to be generated, Data Dictionary objects are referenced from block diagrams of the model, in this case for a specific variable.

The TargetLink Data Dictionary is perfect for defining and handling project-related code specifics, even for workgroups. It provides access to additional information such as specifics on C modules, function calls, tasks, variable classes, data variants and so forth. The data is presented in a well-organized tree and can also be accessed via an application programming interface (API). The Data Dictionary also supports common import and export formats, so that existing and proven definitions, for example, a calibration file, can be used as templates.



With the TargetLink Data Dictionary, the data and the model are kept separate.

Further features extend the functionalities of the TargetLink Data Dictionary:

- The **Data Dictionary Manager** provides the user interface to the Data Dictionary for convenient administration of the data. Elements such as type definitions, variables, and scaling formulas are organized clearly. Typical user interface functions are available, like copying and pasting objects, loading and saving individual data branches, and searching the entire dictionary. The tool can be customized by using the MATLAB code plugin mechanism for menus, context menus, and properties. Additional panes can be used for outputs generated from user scripts.
- Various **import/export formats** are supported, including variables and Simulink data objects from/to MATLAB workspace and files, XML (Extensible Markup Language), ASAM MCD-2 MC (Standardized Description Data, formerly ASAP2), and AUTOSAR artefacts.
- The TargetLink Data Dictionary MATLAB **API** gives you full access to the TargetLink Data Dictionary via MATLAB. Open API interfaces make it easy to integrate the TargetLink Data Dictionary into your company's environment.

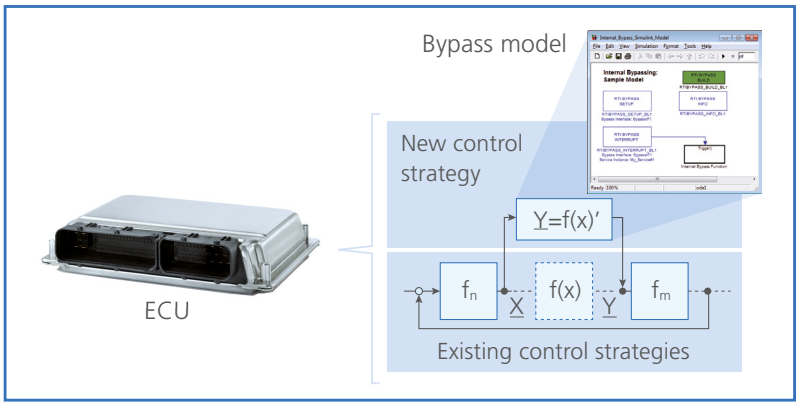
Modular Development and Code Generation

TargetLink lets you easily handle large models and software integration with dedicated mechanisms.

Mechanism	Description
Incremental code generation	Incremental code generation is available for Simulink subsystems which contain a TargetLink function block. Each member of a development team can work on a subsystem individually and just generate code for that. This way, code for individual subsystems in a model can be tested, reviewed, and frozen while development in other subsystems continues. TargetLink performs the necessary consistency checks when building the overall application from all the parts. The overall code generation time can be significantly reduced by generating code only for subsystems that have been modified.
Model referencing support	TargetLink supports model referencing. Functionalities can be flexibly partitioned into several model files, versioned separately, and developed on a modular basis. Parts of models can be simulated separately and implemented individually by means of incremental code generation.
Simplified software integration	TargetLink code from separate (incrementally generated) subsystems and (referenced) models can be easily integrated and tested by means of code generation from the Data Dictionary. Variables with global relevance that are shared by separate models and subsystems are specified and generated from the Data Dictionary.
Code generation straight from the TargetLink Data Dictionary	The Data Dictionary supports the generation of both code and A2L files for Data Dictionary variables independently of their use in TargetLink models. Variables with global relevance that are shared between different models, subsystems and referenced models should be specified in the Data Dictionary and implemented from there. Generating code files with definitions and declarations, and also A2L files for variables from the Data Dictionary, makes software integration and software integration testing easy. Moreover, all the calibration parameters for an entire ECU project can be created in one file.
Diff&Merge mechanisms via TargetLink Data Dictionary	The TargetLink Data Dictionary has the mechanisms needed to compare different versions and display changes, e.g., old/new. You can then trace modifications back to the model to analyze their effects on it. Diff&Merge mechanisms let you update interface definitions simultaneously to ensure consistency when changes are made.

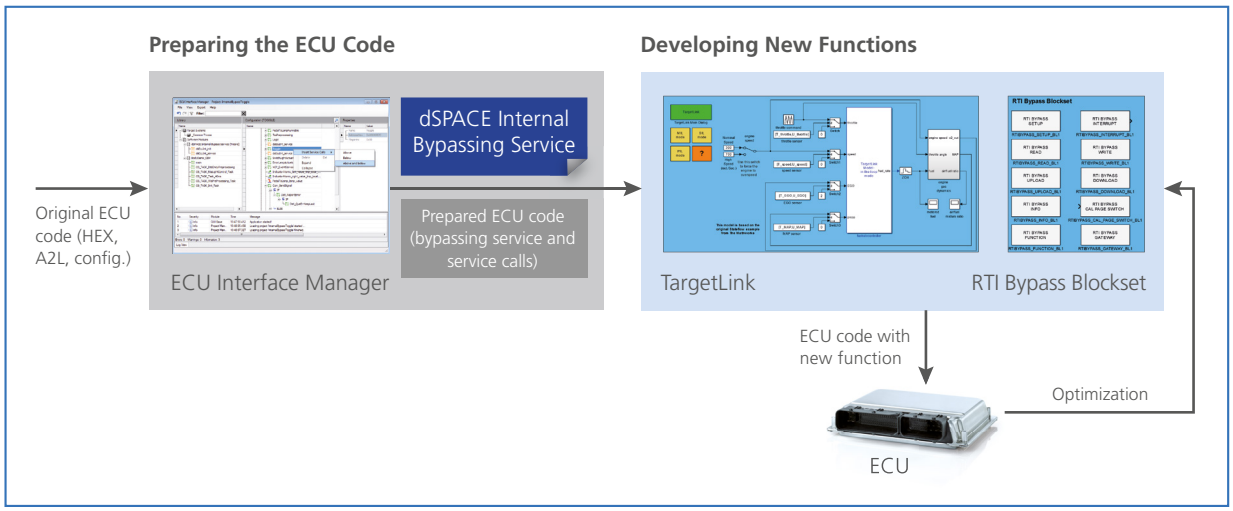
On-Target Bypassing

If a real electronic control unit (ECU) exists and you only want to extend its functionality, it is useful to perform function development right on the ECU. The prerequisites are sufficient I/O and enough free resources. If these prerequisites are met, it makes sense to use TargetLink as the code generator for on-target bypassing to use the free capacities of the ECU as efficiently as possible.



Prototyping directly on the ECU.

TargetLink supports the dSPACE RTI Bypass Blockset (part of the ECU Interface Base Package), making it possible to develop and validate new functions right on the existing production ECU. The handling of the RTI Bypass Blockset is the same as always, which gives you the familiar convenience of using dSPACE RCP systems. At the same time, using TargetLink gives you the flexibility and efficiency of a production code generator. The results on program size and run-time behavior of the generated production code accelerate development and reduce project risks during series production.



In combination with the RTI Bypass Blockset (part of the ECU Interface Base Package), TargetLink implements new ECU functions in the ECU code, which was prepared with the ECU Interface Manager (part of the ECU Interface Base Package).

On-target bypassing with TargetLink opens up all degrees of freedom on the path from functions development to series production.

Most development projects start out with a Simulink model. In this case, TargetLink is used only as the code generator and provides modern optimization options to efficiently use the existing resources on the production ECU that is used for function development. If you already use the TargetLink blockset for modeling, you can also reuse the model from the development phase in series production without any modifications at all. The specifications needed for production code generation are then added to the existing model in the software development phase. This means function developers no longer have to deal with the details of implementation. You can reach maximum production maturity by using a completely specified TargetLink model also for prototyping. This is especially useful for iterative improvements of functions developed with TargetLink. In this case, the model of the improved function can be used immediately in software development, and only minor additions, e.g., for specifying software interfaces, are necessary.

Validating Production Code on a dSPACE Real-Time System Supported by ConfigurationDesk

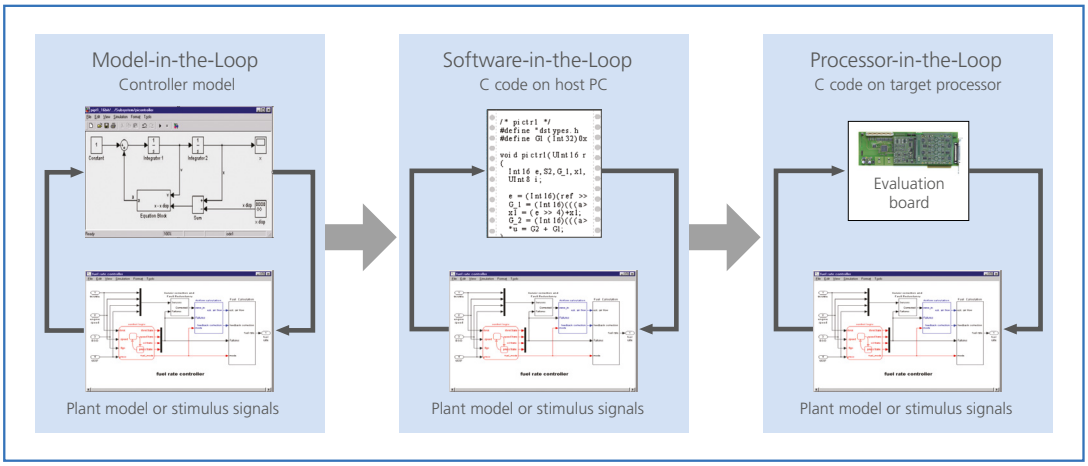
TargetLink-generated production code can be directly validated and tested on a dSPACE real-time system supported by ConfigurationDesk (currently: SCALEXIO and MicroAutoBox III). This helps determine how code potentially works in a realistic environment (real-world connection via the I/O of the real-time system). For more information, please see p. 31.

Built-in Verification and Validation Support

Testing at Various Test Levels

Three Simulation Modes for Testing

Although code generators produce virtually flawless results, when compared to manual programming, the generated code still needs to be tested as well as the underlying specification. TargetLink provides powerful and easy-to-use means to verify the generated code. The code tests are performed in the same simulation environment that was used to specify the underlying simulation model. Functional identity has been achieved when the simulation results match. TargetLink provides a three-step verification process which shows at the click of a button whether the specification (the model) and implementation (the generated code) are functionally identical. On the basis of a controller model, TargetLink performs simulations of the model (model-in-the-loop simulation, MIL), the generated code on the host (software-in-the-loop simulation, SIL), and the generated code on the target (processor-in-the-loop simulation, PIL), without additional model modifications or preparations.



TargetLink performs the three different simulation modes without any changes to the model or the generated code. TargetLink does this automatically in the background.

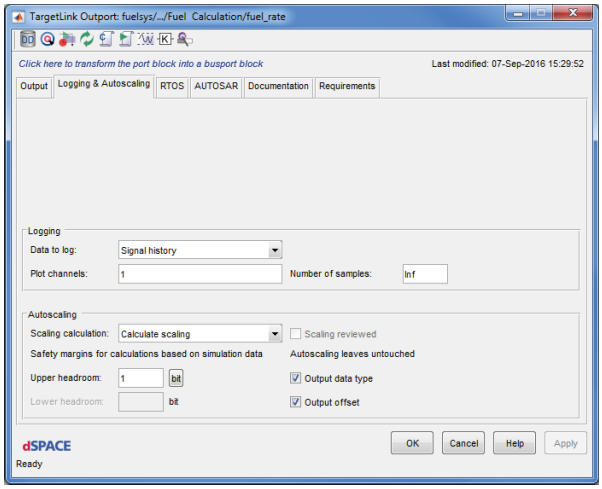
Simulation Mode	How It Works
Model-in-the-loop simulation	■ Data is recorded for reference plots from the simulation model. Signals from selected blocks and state variables are automatically logged by TargetLink. ■ The model-in-the-loop simulation captures the specified behavior of the model that is to be implemented in C code later on. The recorded signal plots act as the reference for the next verification steps. ■ Model-in-the-loop simulation can also be used for detecting overflows of integer variables, and its results are used for simulation-based autoscaling.
Software-in-the-loop simulation	■ The code is generated and replaces the controller blocks in the simulation model (for example, the same plant and stimulus signals). TargetLink does this automatically in the background. ■ You still see the controller blocks, though it is the code that is executed on the host PC instead. The signal plots should be largely identical when compared to the results of model-in-the-loop simulation. If they are not, they can be analyzed to get a better understanding of the cause of the deviation and to fine-tune the fixed-point settings.
Processor-in-the-loop simulation	■ The generated code runs on an embedded processor, but because code that runs correctly on the host PC can still cause trouble, it has to be inspected further. ■ An off-the-shelf evaluation board is connected to the host PC, and the generated code is compiled with the target compiler and downloaded to the evaluation board. ■ TargetLink manages the communication between the host PC and the evaluation board. All these activities are automated and need no user interaction. Simulation on an evaluation board just takes two mouse clicks.

Features and Benefits of the Simulation Concept

Feature	Description	Benefit
MIL/SIL/PIL simulation at the click of a button	Switching from MIL to SIL or PIL simulation requires just one click	■ Powerful simulation environment ■ No need for separate test models, generation of S-functions or manual insertions into test harness models
Integrated data logging	Built-in data logging and result plotting for all simulation modes	■ No model modifications necessary ■ Available for all simulation modes
Direct comparison of MIL/SIL/PIL results	Automatical plotting of all simulation results in the same plot window	■ Display results of simulations in different modes directly and analyze deviations ■ Direct feedback whether code matches model simulation
Detailed signal analysis and deviation plots	Zoom signals to visually inspect deviations, display constraints (e.g., defined ranges), use cursor to scroll through signal histories, display signal values numerically or plot signal deviation	■ Get a clear picture of the signal behavior ■ Especially useful for conversion from floating-point to fixed-point

Integrated Data Logging and Plotting

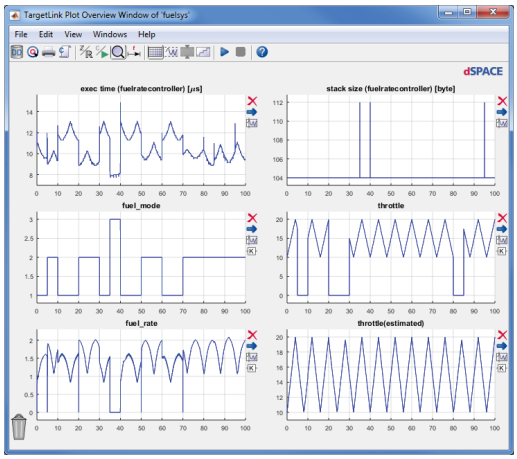
Built-in TargetLink blocks come with an integrated data logging functionality. In the block dialogs, you can specify whether to log the block output signals. No changes to the model are necessary. The data logging is available for all simulation modes without further user interventions.



The TargetLink block dialog lets you specify whether to log signal histories – regardless of the simulation mode.

Run-Time Analysis for Profiling the Code

Processor-in-the-loop (PIL) simulation can also be used to profile the generated code. During PIL simulation, TargetLink automatically measures execution time and stack consumption directly on the target processor. A code summary lists RAM and ROM usage for each function. These features allow you to evaluate design alternatives, such as selecting different search routines of a look-up table block. You can immediately measure the impact of the change on code efficiency. Sound implementation decisions based on accurate benchmarks become a matter of a few clicks.




```
13
14  ********************************************************************
15  *****  Summary for module: fuelratecontroller
16  ********************************************************************
17      Functions                                              [bytes]
18  --------------------------------------------------------------------
19      Tab2DIntp2I1T1_a                                          288
20      Tab2DS17I2T4169_a                                         372
21      TabIdxS49T1_a                                             132
22      Ca26_Running_ex                                           128
23      FuelingMode_du                                            544
24      SensFailCounter_du                                        348
25      fuelratecontroller                                       2404
26
27
28  ********************************************************************
29  *****  Model summary                                         [bytes]
30  ********************************************************************
31      RAM                                          =           2882
32      ROM without libraries                        =           7065
33      ROM with libraries                           =           7065
34
```

The results of code profiling: execution time measurement, stack size measurement, and code summary of generated code.

Code Coverage Analysis

You can assess how comprehensive tests are by using code coverage analysis. TargetLink offers C0 and C1 coverage analysis, also called statement coverage and decision coverage. During simulation on host or target systems, counters record the frequency of execution for each branch of code. After simulation, a code coverage analysis report is generated with a coverage overview table and a detailed, annotated code listing with the execution count for each block. This report tells you immediately whether the tests cover everything. Code branches that have never been executed are easily identified, and test strategies can be adjusted accordingly. Thus, coverage analysis increases software quality.

	Code Coverage	Total Branches	Reached Branches	Unreached Branches
Fuel system	61.87%	139	86	53
Fuel rate controller	58.91%	129	76	53
Fuel rate calculator	56.56%	122	69	53
Correction redundancy	100%	7	7	0
Airflow controller	100%	10	10	0
Airflow calculation	100%	9	9	0
Airflow subsystem	100%	1	1	0

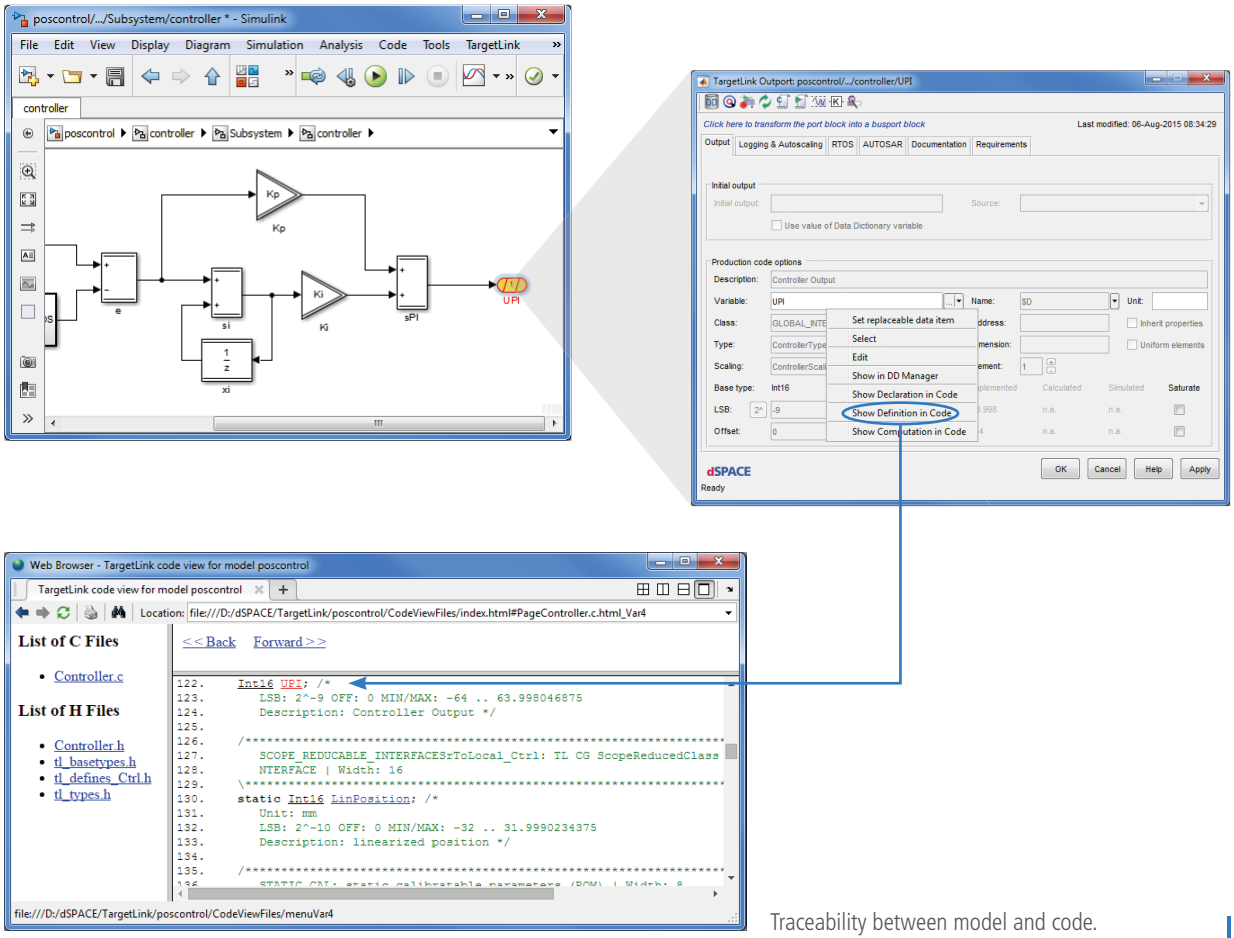
Example of the information provided by code coverage.

```
174          |     if (x <= Aux_S8) {
175      0   |        /* Saturation. */
176          |        return z_table[0];
177          |     }
178          |     else {
179      155 |     }
180          |
181          |     /* Calculate table index. */
182          |     Aux_U8_a = (UInt8) (((UInt8) (((UInt8) x) - ((UInt8) Aux_S8))) >> Aux_U8);
183          |     if (Aux_U8_a >= ((UInt8)(map->Nx - 1))) {
184      0   |        return z_table[(UInt8) (map->Nx - 1)];
185          |     }
186          |     else {
187      155 |     }
188          |     Aux_U8_b = (UInt8) (((UInt8) (((UInt8) x) - ((UInt8) (Aux_U8_a << Aux_U8))
189          |      Aux S8));
```

Annotated code listing with execution count for each code block.

Model-Code Traceability

For improved traceability and simplified code reviews, code files can optionally be generated in HTML format, with hyperlinks for navigation from model to code and vice versa.

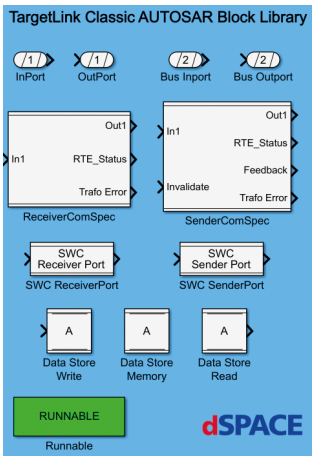


Traceability between model and code.

High-Performance Classic and Adaptive AUTOSAR Support

Classic AUTOSAR

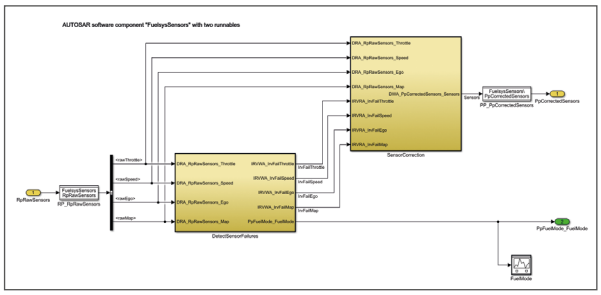
The optional TargetLink AUTOSAR Module makes the modeling, simulation, and code generation features of TargetLink available for designing Classic AUTOSAR software components. TargetLink supports the vast majority of Classic AUTOSAR communication mechanisms and generates genuine RTE AUTOSAR macros.



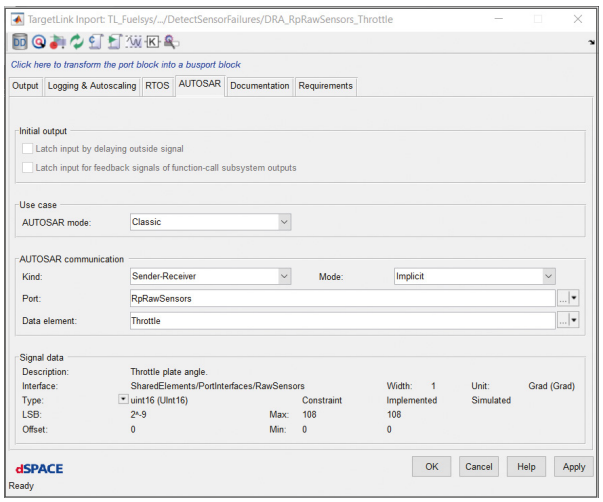
The TargetLink Classic AUTOSAR block library.

Designing Classic AUTOSAR Software Components

The TargetLink Classic AUTOSAR Block Library provides Classic AUTOSAR structure elements, for example, runnables, ports, and communication interfaces, at the model level. To define a Classic AUTOSAR runnable, a Function block specified as a Classic AUTOSAR runnable is used in a subsystem, similar to regular TargetLink functions. The TargetLink InPort and OutPort blocks are used to define the data exchange between the runnables of one or more software components. Developers can specify properties on the block level and via the TargetLink Data Dictionary.



Modeling Classic AUTOSAR SWCs.



AUTOSAR page in a TargetLink block dialog.

Generating Classic AUTOSAR-Compliant Code

TargetLink generates production code for Classic AUTOSAR software components. Moreover, TargetLink provides special options allowing for Classic AUTOSAR optimizations. Runnables are implemented as C functions, and Classic AUTOSAR communication mechanisms are implemented as run-time environment macros according to the Classic AUTOSAR standard. The same model can be used to generate Classic AUTOSAR-compliant code and standard TargetLink code.

```
void controller_runnable(Rte_Instance instance, Rte_ActivatingEvent_controller_runnable activation)
{
                    ...
   /* TargetLink inport: TL_Controller/Controller_Runnable/ref */
   ref = Rte_IRead_PosController_RequiredSignals_ref(instance);
                    ...
   /* Sum: TL_Controller/Controller_Runnable/e1 */
   S12_e1 = (sint16) ((((sint32) ref) - ((sint32) LinPos)) >> 1);
                    ...
   /* TargetLink outport: TL_Controller/Controller_Runnable/upi */
   upi = Rte_Pim_ACP_1(instance)->S12_sPI1;
   Rte_IWrite_PosController_ProvidedSignal_upi(instance, upi);
                    ...
}
```

The generated C code with RTE macro calls.

Simulating and Testing Classic AUTOSAR Software Components and SWC Compositions

TargetLink simulates Classic AUTOSAR software components in all simulation modes. Multiple software components can be simulated in one simulation run. The communication between software components is simulated to the extent supported by the Simulink design environment. For testing purposes in SIL and PIL, TargetLink generates a stub RTE that maps the RTE macro function calls in the application with SWC code to global variables and generates access functions to global variables.

Helpful Features and Utilities (Examples)

Feature/Utility	Description	Benefits
Importing and exporting software component descriptions.	■ Based on TargetLink AUTOSAR models, ARXML files can be exported directly from TargetLink. ■ Importing and merging existing software component descriptions from the TargetLink Data Dictionary.	■ Seamless Classic AUTOSAR development process with a tool such as SystemDesk and dSPACE AUTOSAR Compare.
Generating a Classic AUTOSAR frame model.	■ Generating a frame model containing the relevant ports and runnables. ■ Inserting the control algorithm into the model frame to obtain a complete Classic AUTOSAR software component. ■ Generating SWC compositions as frames.	■ Easy and convenient transfer of AUTOSAR specifications into the TargetLink model.
Migrating standard TargetLink models to Classic AUTOSAR.	■ TargetLink AUTOSAR Migration Tool for automatically converting individual subsystems to Classic AUTOSAR runnables. ■ Supporting the flexible specification of Classic AUTOSAR properties.	■ Highest flexibility when reusing existing software parts.
Modeling Guide for Classic AUTOSAR	■ Explains how to model and generate code for Classic AUTOSAR software components.	■ Support while developing Classic AUTOSAR software components with the model-based development approach of TargetLink.

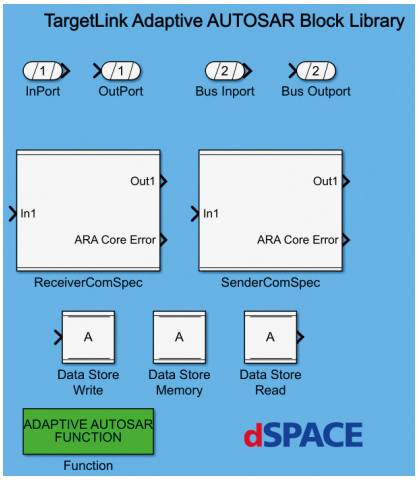
TargetLink in an Classic AUTOSAR Tool Chain

For Classic AUTOSAR software development, TargetLink is typically combined with an AUTOSAR architecture tool such as SystemDesk. A software architecture with multiple components is specified in the architecture tool, while TargetLink is used to design and implement the individual software components. The two tools exchange data via Classic AUTOSAR XML files. Both a top-down approach (starting with the software architecture) or a bottom-up approach (starting with a TargetLink function model) can be used for Classic AUTOSAR round trips. This AUTOSAR round trip can be simplified and automated by using dSPACE AUTOSAR Compare, which lets you compare and manually or automatically merge your AUTOSAR artifacts.

Adaptive AUTOSAR Support

The optional TargetLink Adaptive AUTOSAR Module makes the modeling, simulation, and code generation features of TargetLink available for designing Adaptive AUTOSAR applications. TargetLink supports selected features of Adaptive AUTOSAR, for example:

- Importing and exporting Adaptive AUTOSAR ARXMLs with elements defined by ara::com.
- Importing Adaptive AUTOSAR ARXMLs with elements defined by ara::per.
- Modeling select parts of service-based communication as described by ara::com.
- Modeling select parts of accessing persistent memory as described by ara::per.
- Modeling and simulating error handling as described by ara::com and ara::per.



The TargetLink Adaptive AUTOSAR block library.

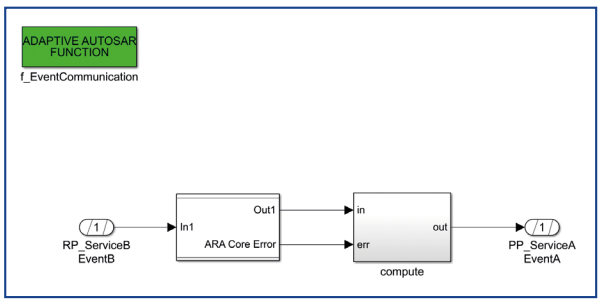
Designing Adaptive AUTOSAR Applications

TargetLink supports the model-based development of functional parts of adaptive applications. It provides different model elements for you to model the functionalities.

The following uses cases are supported to model communication according to ara::com:

- Accessing fields as a service consumer or service provider.
- Sending or receiving events.
- Calling or implementing a method.

In addition, TargetLink supports the modeling of persistent data access, i.e., reading and writing key-value pairs from key-value storage, according to ara::per.



Modeling Adaptive AUTOSAR event communication with TargetLink.

Generating Code for Adaptive Applications

TargetLink generates Adaptive AUTOSAR functions from Method Behavior subsystems and Adaptive AUTOSAR subsystems. The Adaptive AUTOSAR function is a C++ function representing a partial functionality of an adaptive application. This function can be called in the C++ code of an adaptive application. key-value storage, according to ara::per.

Simulating and Testing Adaptive AUTOSAR Models

TargetLink simulates Adaptive AUTOSAR models in MIL and SIL simulation mode. While building the simulation application for SIL simulation, TargetLink uses preprocessor macros and conditional compilation. This allows the simulation of the Adaptive AUTOSAR functions directly in TargetLink without the need of an Adaptive AUTOSAR middleware. For testing purposes, TargetLink supports modeling and simulating error handling as specified by Adaptive AUTOSAR.

Helpful Features and Utilities (Examples)

Feature/Utility	Description	Benefits
MIL and SIL simulation	■ Directly simulate the Adaptive AUTOSAR function in TargetLink without Adaptive AUTOSAR middleware.	■ Easily simulate and validate your Adaptive AUTOSAR functionality in the development environment. TargetLink allows for comfortable and straight-forward testing without setting up, configuring, and maintaining a build chain for Adaptive AUTOSAR.
Modeling of select Adaptive AUTOSAR clusters	■ Model persistent data access as defined by ara::per by reading and writing key-value pairs from a key-value storage. Model communication as defined by ara::com: ■ Accessing fields as a service consumer or service provider. ■ Sending or receiving events. ■ Calling or implementing a method.	■ Convenient and well-known modeling similar to Classic AUTOSAR modeling in TargetLink.
Modeling Guide for Adaptive AUTOSAR	■ Explains how to model and generate code according to Adaptive AUTOSAR for the integration into adaptive applications.	■ Support while developing functional parts of adaptive applications with the model-based development approach of TargetLink.

Automation, Process and Tool Integration

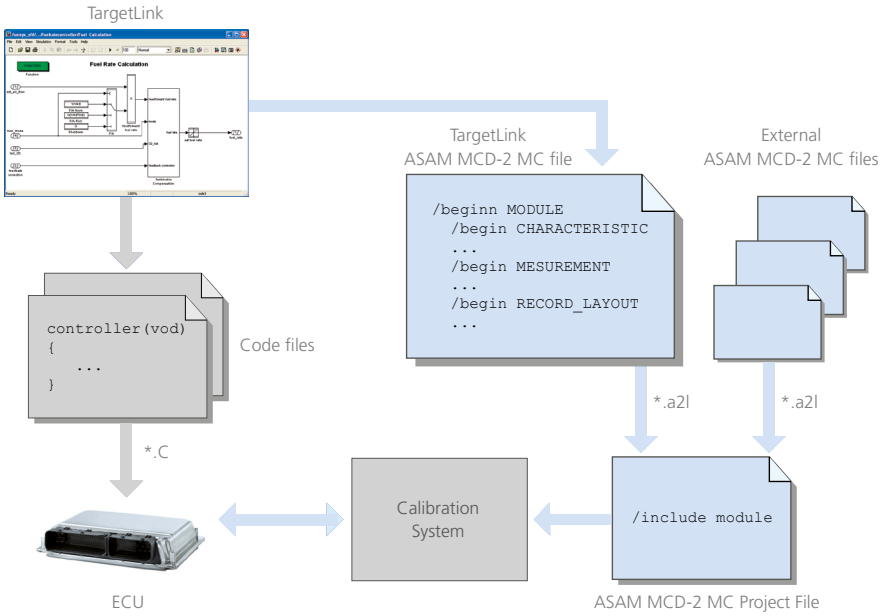
Easy Integration into your Tool Chain

Comprehensive TargetLink API

TargetLink can be easily integrated into existing development environments, because it comes with a comprehensive and fully documented application programming interface (API). This grants full access to all TargetLink properties and settings and allows for processes to be automated while at the same time providing options for interventions in the individual process phases. For example, hook functions allow for user tasks to be performed at all stages of the build process.

Calibration File Generation

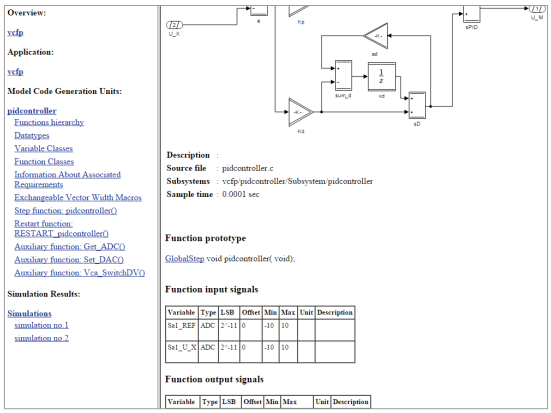
Another important requirement for a code generator is to have close links with calibration systems. ECU code must be prepared for parameter fine-tuning by making calibratable or measurable variables accessible to a calibration system. TargetLink supports the generation of the standardized ASAM MCD-2 MC file format (formerly ASAP2) via the Data Dictionary to make the variables and parameters available for ECU calibration. All major calibration tools support this standard. Because the C code and the ASAM MCD-2 MC file are generated with the same data basis, they are always consistent. This eliminates another error source, and streamlines the development process. TargetLink offers several predefined variable classes for calibratable and measurable variables. You can also specify your own classes, ensuring that each class holds suitable attributes for calibration and/or measurement.



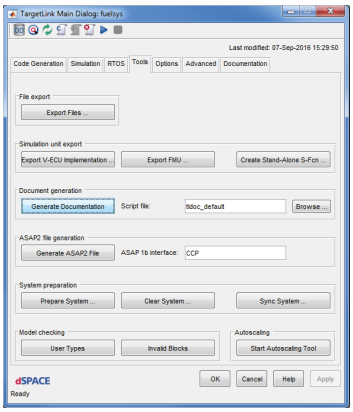
ASAM MCD-2 MC file generation for calibration purposes.

Documentation Generated Automatically

TargetLink not only generates code, it also documents what it does – keeping perfect consistency with the model and the code. An automatically generated document provides information about function interfaces and global variables, and a list of all measurable and adjustable variables, scaling parameters, code generator options and much more. Screenshots of models, subsystems, and simulation plots can also be included. Links to the generated C code are provided. You can specify the documentation you require, for example, the level of detail. Documentation can be generated in the HTML, RTF (for word processing) and PDF formats.



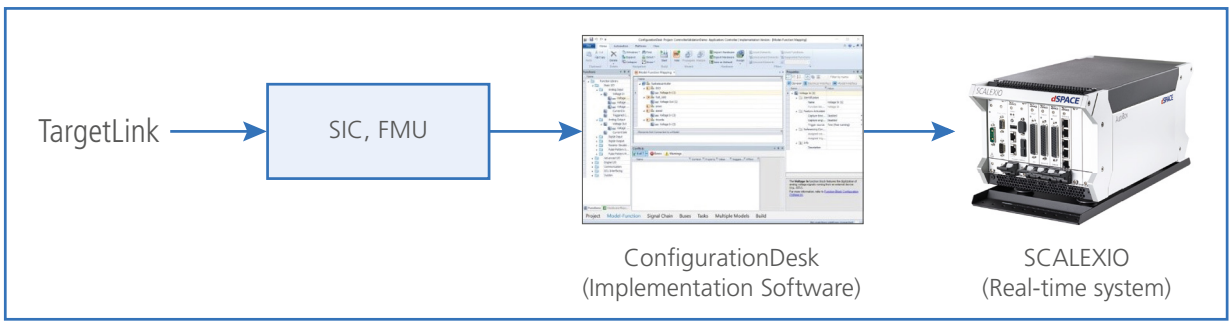
Production code documentation in HTML.



The documentation is easy to generate – at the click of a button.

Validating Production Code on a dSPACE Real-Time System Supported by ConfigurationDesk

TargetLink-generated production code can be directly validated and tested on a dSPACE real-time system supported by ConfigurationDesk (currently: SCALEXIO and MicroAutoBox III). This enables users to quickly determine how code potentially works in a realistic environment (real-world connection via the I/O of the real-time system). Moreover, the processing power, memory space, and I/O options of the real-time hardware allow for extensive test options, which are difficult to provide due to the limited possibilities of a production ECU. TargetLink-generated code can also be used for multiple real-time system-based tests and development tasks in other process steps aside from pure production code generation. The mechanism is possible thanks to the support of container-based workflows for the ConfigurationDesk implementation software. The software is required for connecting to SCALEXIO and MicroAutoBox III.



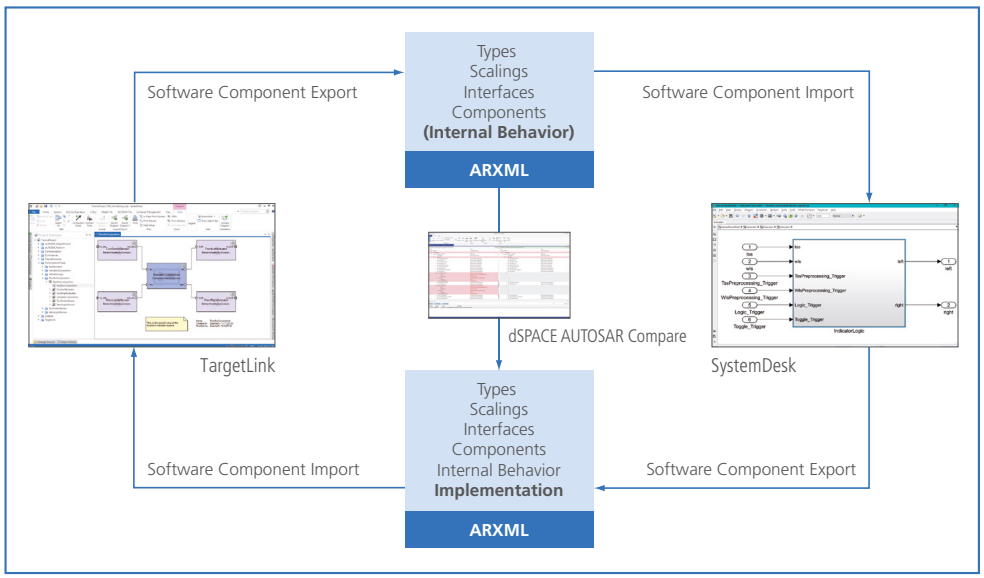
Production code tests on the real-time system.

TargetLink and SystemDesk

TargetLink and SystemDesk are the ideal combination for developing Classic AUTOSAR-compliant software. TargetLink and software architecture tools can exchange AUTOSAR specifications via ARXMLs. Differences in these files can be easily visualized using dSPACE AUTOSAR Compare and hence merged either in the user interface or via dSPACE AUTOSAR Compare automatization capabilities. This is a reliable, transparent way to perform Classic AUTOSAR round trips with minimum user intervention. In addition, TargetLink users have convenient, direct access to virtual ECU (V-ECU) generation in SystemDesk and to VEOS simulation capabilities, which let them evaluate and test the behavior of TargetLink components as parts of complex systems in early development stages, and with the same interfaces throughout all development stages.

Main benefits for TargetLink users:

- Safe and convenient AUTOSAR specification exchange between TargetLink and a software architecture tool on the basis of ARXMLs
- Fast and reliable Classic AUTOSAR round trips between TargetLink and software architecture tools by means of dSPACE AUTOSAR Compare
- Convenient V-ECU generation with SystemDesk for testing SWCs with VEOS in early development stages



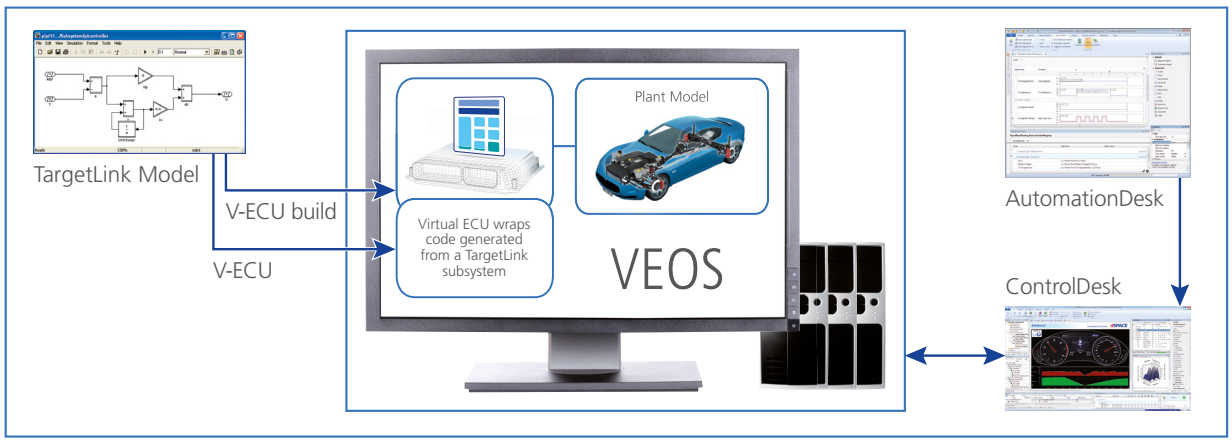
The new, additional product dSPACE AUTOSAR Compare enables users to compare and manually or automatically merge their AUTOSAR artefacts.

TargetLink Connection to VEOS

TargetLink code can run not only in SIL/PIL simulations in the Simulink environment, but also on dSPACE's PC-based simulator VEOS (figure below). This is done by wrapping code in an executable unit called a virtual ECU (V-ECU), which is generated from TargetLink models. TargetLink code can then be executed along with other V-ECUs as well as plant models on VEOS. To capture signals and adjust or test parameters, A2L files exported from TargetLink are used with dSPACE ControlDesk, which provides access to VEOS. Thus, the standard dSPACE tools used for real-time simulation are also used for offline simulation and experimentation with TargetLink-generated code. This provides the following benefits for TargetLink users:

- Early simulation of TargetLink code in large systems consisting of multiple virtual ECUs and plant models, including buses if required
- Convenient experimentation and testing with ControlDesk and other dSPACE tools

The approach shown in the figure below is intended for TargetLink users who do not work on AUTOSAR projects. For AUTOSAR projects, it is recommended to exchange AUTOSAR software components between SystemDesk and TargetLink, and generate the V-ECU from SystemDesk (p. 32).



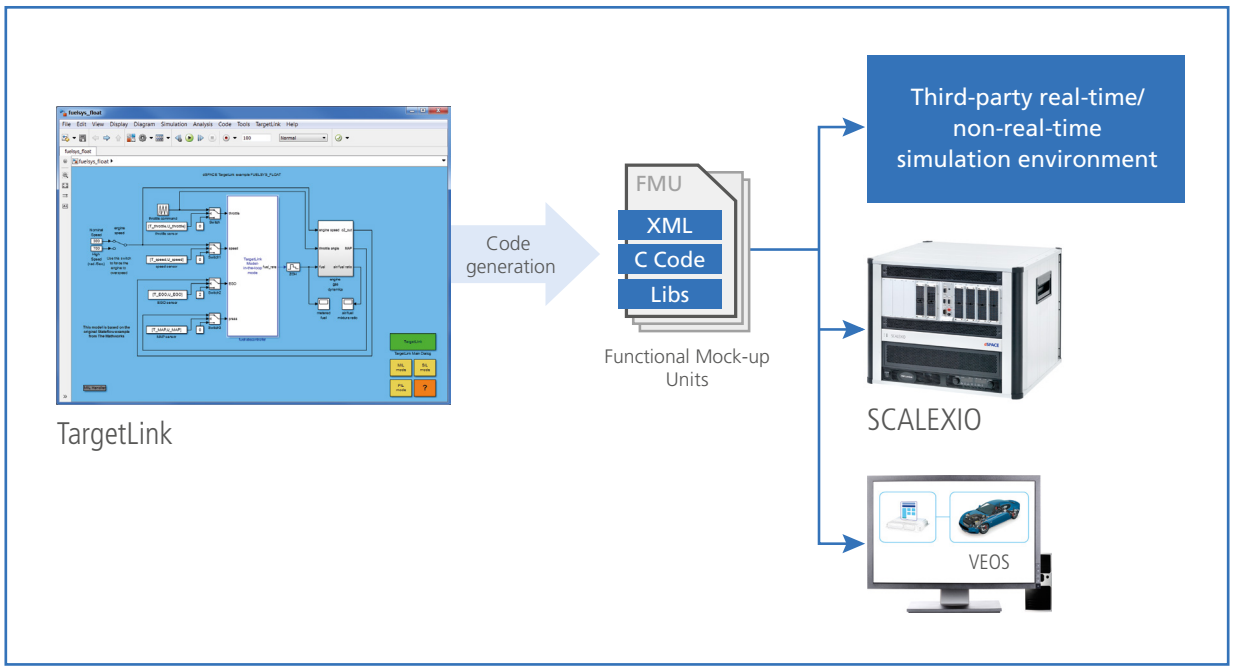
Connection of TargetLink to VEOS for non-AUTOSAR users.

TargetLink FMI Support

Functional Mockup-Units (FMUs) based on the FMI 2.0 for Co-Simulation standard can be exported directly from Simulink/TargetLink models. These FMUs encapsulate the TargetLink-generated production code and can therefore be integrated and simulated in all environments that suitably support the Functional Mockup Interface (FMI) standard. This can be done for real-time and non-real-time simulation platforms. As a result, software developers can create production code software in a familiar development environment and then import and reuse it in different environments without any manual effort.

Main benefits for TargetLink users:

- Model exchange between different environments and domains
- Running TargetLink code easily in third-party simulation environments along with plant models
- Integration of TargetLink into the dSPACE virtual validation tool chain



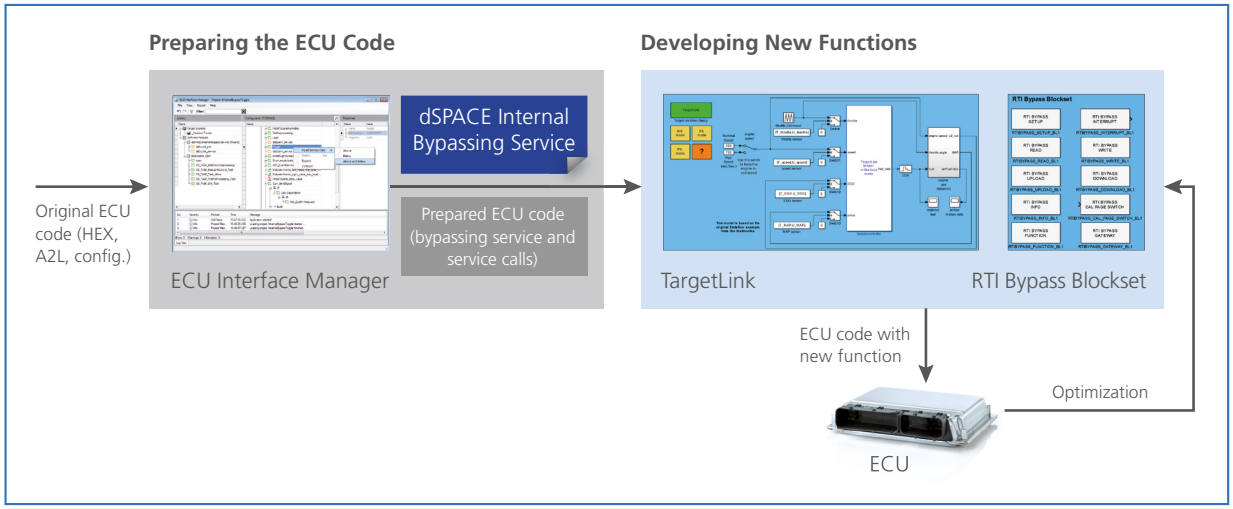
Exporting Functional Mockup-Units (FMUs) from TargetLink to simulation environments that support FMI.

TargetLink and ECU Interface Software

For On-Target Bypassing, i.e., performing function prototyping for functionality extensions directly on an existing ECU, TargetLink can be combined with dSPACE's ECU Interface Software. For more information, please see p. 20.

Main benefits for TargetLink users:

- The existing production ECU can be conveniently used as prototyping hardware.
- ECU resource consumption is kept under control.
- A seamless transition to production development is possible.
- The tool chain can save complex integration loops that involve the ECU supplier.



In combination with the dSPACE RTI Bypass Blockset (part of the ECU Interface Base Package), TargetLink implements new ECU functions in the ECU code, which was prepared with the ECU Interface Manager (part of the ECU Interface Base Package).

Software Quality and Supported Standards

Committed to Highest Quality

TargetLink Certified for ISO 26262, ISO 25119, and IEC 61508

TargetLink is certified by TÜV SÜD (German international certification association) for use in the development of safety-related systems. TÜV confirmed that TargetLink is suitable for software development according to ISO 26262, ISO 25119, IEC 61508, and derivative standards (such as EN 50128, which governs safety-related software on the railways). The certification was based on a number of areas:

- Software development process and software modification process of TargetLink
- Problem handling procedures
- Fitness for purpose in safety-related development according to ISO 26262, ISO 25119, and IEC 61508

TÜV SÜD also approved a reference workflow providing guidance for the model-based development of safety-related software with TargetLink. IEC 61508 is the internationally recognized generic standard for the development of safety-related electronic systems. ISO 26262 is the international automotive standard for the development of safety-related systems in road vehicles. ISO 25119 is the international standard for the development of safety-related parts of control systems in tractors and machinery for agriculture and forestry. Both standards are derived from IEC 61508.

ISO/IEC 15504-Compliant Development Process

ISO/IEC 15504 (also known as SPICE[1)]) is an international standard for software processes. Its underlying concept is that a mature software product requires a mature development process. dSPACE has dedicated itself to an ISO/IEC 15504-compliant development process.

Internal Software Quality Management

An internal quality department, the dSPACE quality management team, proactively manages software quality at dSPACE. The team leads software improvement activities, sets internal standards, conducts internal assessments, and provides consultation services to all software groups. It acts independently and ensures that the highest product quality goals are consistently achieved and sustained.

1) *SPICE: Software Process Improvement and Capability Determination*

AUTOSAR

As a de-facto standard for automotive E/E architectures, AUTOSAR contains specifications for communication interfaces between application functions and basic system functions. The TargetLink AUTOSAR Module makes TargetLink's modeling, simulation and code generation features available for designing Classic AUTOSAR software components (SWCs), see also p. 26. Furthermore, TargetLink supports selected features of Adaptive AUTOSAR (see p. 28) with the TargetLink Adaptive AUTOSAR Module. AUTOSAR Adaptive Platform, also called Adaptive AUTOSAR, is a standard based on a service-oriented architecture that aims at on-demand software updates and high-end functionalities.

FMI

The Functional Mock-up Interface (FMI) is an open standard for the tool-independent exchange and integration of plant models that are provided by various tool vendors. Functional Mockup Units (FMUs) can be exported from TargetLink to simulation environments that support FMI, see also p. 33.

ASAM MCD-2 MC (ASAP2)

Internal ECU variables in measurement and calibration can be defined in the description format ASAM MCD-2 MC. Because a code generator also needs to have close links with calibration systems, TargetLink can export calibration data as ASAM-MCD 2MC file for calibration tools, see also p. 29.

MISRA C

The British MISRA[1)] C standard is a widely accepted C subset for projects in the automotive industry. Its aim is to define rules for avoiding common software errors that can occur when software engineers write software by hand. Most of these rules also make sense for machine-generated code. TargetLink-generated code complies with the vast majority of MISRA C rules. If deviations from the MISRA C standard are a technical necessity, they are identified and well documented. dSPACE provides detailed documentation about TargetLink's MISRA C compliance to all TargetLink customers on request. For more information, please contact TargetLink.Info@dspace.de

1) *MISRA: Motor Industry Software Reliability Association (www.misra.org.uk)*

TargetLink Ecosystem

In combination with TargetLink, several dSPACE tools and numerous third-party tools facilitate high-quality model-based software development. Together, these tools form the TargetLink Ecosystem. For each development phase, a broad range of powerful tools is available, which can be smoothly integrated with TargetLink. Therefore TargetLink users are able to build-up their own highly efficient tool chains for their individual scenarios, also meeting the high requirements of current safety standards such as ISO 26262 or DO-178C.

Category	Product	Company
Requirements handling and tracing	■ TargetLink in combination with Simulink Requirements™, and ■ IBM® Rational® DOORS®, ■ PTC™ Integrity, ■ BTC EmbeddedSpecifier® or ■ Microsoft® Excel®	■ www.dspace.com ■ www.mathworks.com ■ www.ibm.com ■ www.ptc.com ■ www.btc-es.de ■ www.microsoft.com
Architecture design	■ SystemDesk ■ DaVinci Developer ■ Enterprise Architect ■ IBM® Rational® Rhapsody ■ EB tresos® Studio	■ www.dspace.com ■ www.vector.com ■ www.sparxsystems.de ■ www.ibm.com ■ www.elektrobit.com
Static model analysis	■ MES Model Examiner® (MXAM)	■ www.model-engineers.com
Modeling support	■ MES Model & Refactor® (MoRe)	■ www.model-engineers.com
AUTOSAR and Model Comparison	■ Model Compare ■ SimDiff ■ dSPACE AUTOSAR Compare	■ www.dspace.com ■ www.ensoft.de
Model coverage analysis	■ BTC EmbeddedTester® ■ Simulink Coverage™	■ www.btc-es.de ■ www.mathworks.com
Test vector generation	■ BTC EmbeddedTester® ■ Reactis® ■ TPT	■ www.btc-es.de ■ www.reactive-systems.com ■ www.piketec.com
Model-based testing	■ BTC EmbeddedTester® ■ MES Test Manager® (MTest) ■ Reactis® ■ TPT	■ www.btc-es.de ■ www.model-engineers.com ■ www.reactive-systems.com ■ www.piketec.com
Formal verifications	■ BTC EmbeddedSpecifier® ■ BTC EmbeddedValidator®	■ www.btc-es.de
Coding guideline checking	■ QA-C ■ PC-lint	■ www.qa-systems.com ■ www.gimpel.com
Code coverage analysis	■ BTC EmbeddedTester® ■ Testwell CTC++	■ www.btc-es.de ■ www.verifysoft.de
Run-time error analysis	■ Astrée ■ Polyspace Code Prover™	■ www.absint.com ■ www.mathworks.com
Resource usage analysis	■ aiT WCET Analyzer ■ StackAnalyzer ■ TimingProfiler	■ www.absint.com

Continuation from p. 37:

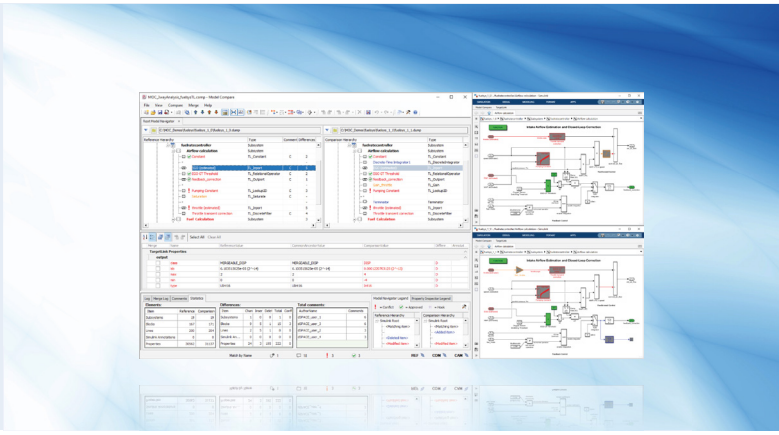
Category	Product	Company
Requirements coverage analysis	■ BTC EmbeddedTester® ■ TPT	■ www.btc-es.de ■ www.piketec.com
Software Testing	■ dSPACE VEOS ■ dSPACE Real-Time Hardware	■ www.dspace.com
Additional MathWorks toolboxes	■ Simulink Control Design™ ■ Simulink Coder™ ■ ...	■ www.mathworks.com
Training and development platform	■ MEDKit on ARM – Motor Control Education Kit	■ www.emerge-engineering.de
ISO 26262 consulting services	■ dSPACE Process Consulting Services ■ MES Consulting and Training	■ www.dspace.com ■ www.model-engineers.com

Model Compare

Comparing MathWorks® Simulink®, Stateflow®, and dSPACE TargetLink models

Highlights

- Intelligent and adjustable comparison algorithms with merge functionality
- Flexible difference filter configuration and highly configurable difference report generation
- Three-way analysis with conflict detection and automatic resolution options
- Built-in TargetLink support



Application Areas

Automatic comparison is a must whenever you work with multiple versions of a model. Model Compare from dSPACE can be used with any TargetLink, MathWorks® Simulink or Stateflow model, and also supports libraries. There is a broad range of use cases for Model Compare, for example:

- Managing different model versions or model variants
- Merging parallel development branches
- Verifying and reviewing model changes

Key Benefits

Model Compare finds all the changes in a model. Even large models can be compared in minutes, which would be practically impossible without tool support. Furthermore, the optional three-way model analysis detects conflicting changes and shows how models have changed rather than only their differences. You can use filters to focus on relevant differences and ignore unimportant ones, such as layout changes or simulation settings. The support for review sessions also enables reliable, safe, and controlled reviews of model changes. The merge support simplifies the synchronization of changes in different versions of a model.

TargetLink Support

Model Compare has built-in support for all TargetLink blocks and properties. They are displayed and handled like ordinary Simulink/Stateflow elements, so you do not have to bother with TargetLink implementation details. Since the semantics of TargetLink properties are known, they are also intelligently handled by the predefined filter options.

Review Sessions and Merge Support

You can associate review comments to block and property differences found by the tool, including date/author tracing. Complex reviews with multiple participants are supported. Detected changes can be transferred from one model to another to merge parallel development branches or manage different model variants. With easy-to-use commands, merging models this way is much less error-prone than it would be by hand. The remaining differences between the models are constantly kept up-to-date, so that you always see the current state of your work.

Main Features and Benefits

Feature	Description	Benefit
Support for TargetLink	■ Model Compare recognizes TargetLink blocks and properties, and handles them just like built-in blocks.	■ There is no need to bother with TargetLink implementation details.
Three-way analysis with conflict detection and automatic resolution option	■ Optional analysis of a third (common ancestor) model ■ Option for automatic conflict resolution	■ Detailed information on changes of the models rather than listing only their differences ■ Different model versions can be merged without user intervention
Powerful comparison algorithm	■ Models from different Simulink® versions are compared; block correspondences are detected even if the names of the blocks have changed; parameter values are compared either in unevaluated form (e.g., "Kp") or in evaluated form (e.g., "5.4").	■ Conversion and upgrade problems can be addressed and conflicting changes can be detected. You get concise comparison results even if blocks were renamed. Different workspace settings can be taken into account.
Flexible filter configuration	■ Model Compare can be configured to filter out unimportant differences, e.g., layout changes or simulation options, according to a variety of criteria. A comparison can be restricted to selected subsystems.	■ You can focus on the differences and model parts that are relevant to you. This greatly improves the efficiency of a comparison.
Convenient result display	■ The comparison results are displayed in synchronized tree views, with differences indicated by customizable color schemes. A statistics window displays the number of changed, added and removed elements.	■ You can easily see which elements correspond to each other. Added, removed and modified elements as well as the number and type of changes can be seen at a glance.
Review support	■ Comments can be associated with individual blocks and properties or with the complete comparison session.	■ Reviews are performed in a reliable, controlled and safe way. You can also use the comment function to structure your own working process.
Traceability from Model Compare to model	■ You can easily highlight Simulink and Stateflow elements by selecting them in Model Compare, and show any differences directly in the compared models.	■ It is easy to view the context of a change and all the differences in a subsystem.
Traceability from model back to Model Compare	■ You can trace model elements in Simulink back to Model Compare.	■ This is helpful to identify blocks and lines in Model Compare and to inspect all their differences at a glance.
Merge support	■ The commands Copy to Right, Copy to Left, and Delete can be used to transfer changes from one model to another. ■ Advanced merge functions for three-way analysis	■ You can merge parallel development branches and transfer changes between different model variants. ■ More convenience by automatically merging and resolving conflicts
Report generation	■ You can save comparison results and associated comments as HTML, PDF and XML reports, model screenshots can be integrated in the difference reports.	■ The information can be archived and published.
Tool automation	■ You can start the comparison via the command line, and reports can be generated automatically. Reports can be saved in XML format for easy processing by external tools.	■ You can process multiple models automatically and incorporate Model Compare into your own tool chain.

Order Information

Product	Order Number
Model Compare	■ MOC

Relevant Software

Software		
Required	Operating system	■ www.dspace.com/goto?os_compatibility
	Integrated development environment	■ MathWorks® MATLAB®/Simulink®/Stateflow®
Optional	Production code generator TargetLink	■ See relevant product information
	dSPACE AUTOSAR Compare	■ DARC

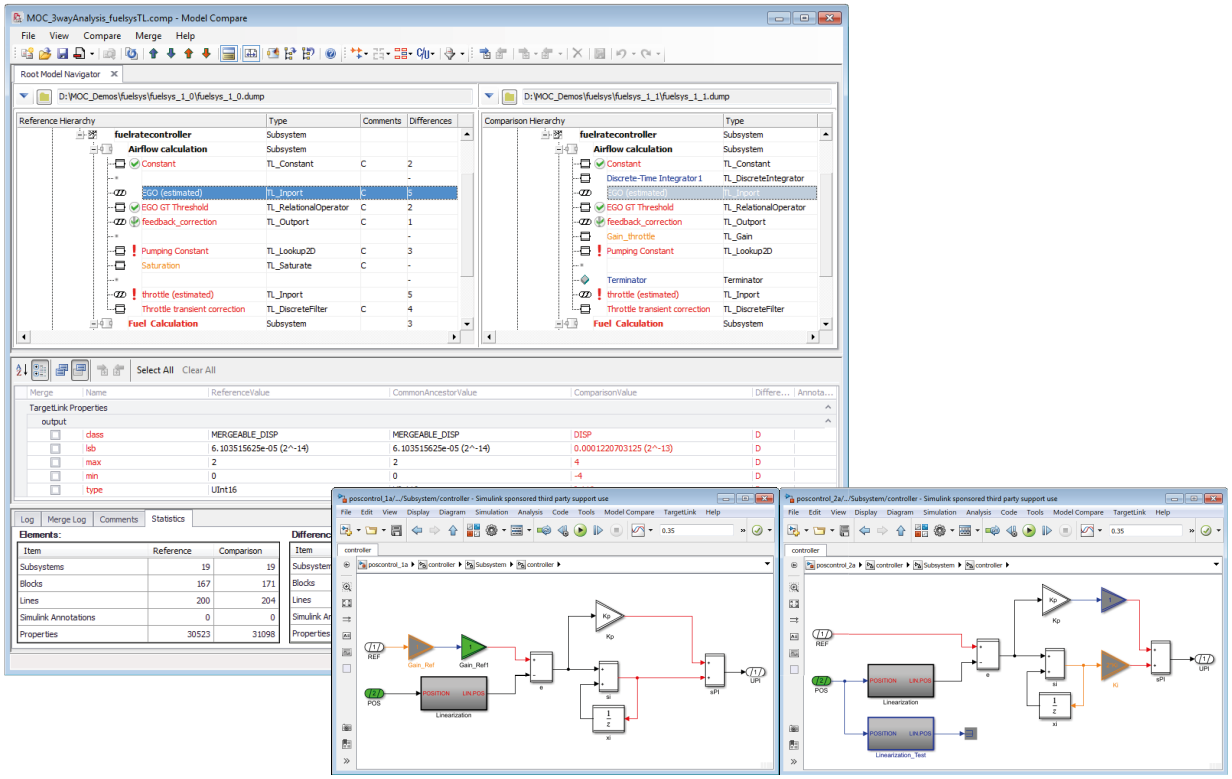
NEW: Model Compare 3.2

Feature Area	Improvement
Model comparison	■ New difference overview for a quick overview of modified model parts and for faster comparison results by selecting the changed model parts only instead of the entire model.
Workflow	■ Improved workflow integration through additional command line interface (CLI) options, which improves efficiency when using Model Compare via CLI or from a version control system, such as Git.
Documentation	■ New documentation for using Model Compare with Git, including all relevant information to set up the product as a diff tool and/or merge tool in your Git environment. ■ New tutorial video: Model Compare Quick Guide

Graphical Display of Differences

Model Compare shows all model differences clearly arranged in two synchronized tree views, where changed, added and removed elements are indicated by customizable colors. While navigating through the model hierarchy, all property differences can be inspected in the Property Inspector. Predefined and flexible filter configurations improve the efficiency of the comparison and let you adjust the view to your individual needs.

Differences can be traced from Model Compare directly to the Simulink/TargetLink models, where the corresponding elements are indicated by customizable colors. Thus, the differences can be easily inspected in the context of the models. In addition, elements of Simulink/TargetLink models can also be traced back to Model Compare to inspect all their differences at a glance.

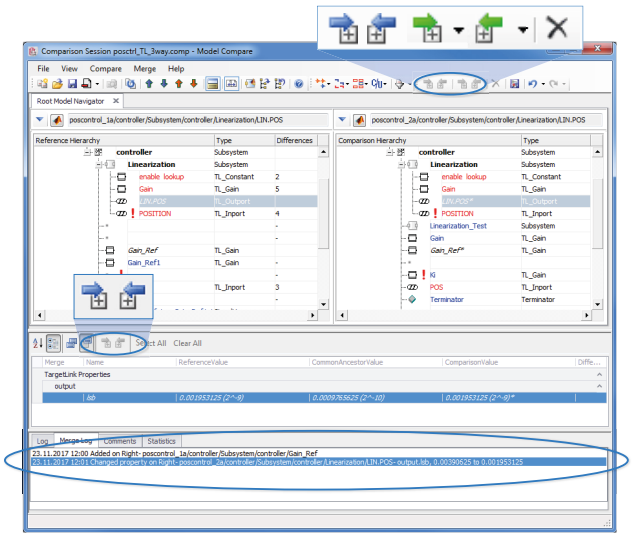


Graphical display of differences in Model Compare and directly in Simulink/TargetLink models.

Merge Support and Automatic Conflict Resolution

With Model Compare you can eliminate differences between two Simulink/TargetLink models by merging the two models. Changes of model elements as well as individual properties can be transferred from one model to the other by commands Copy to Right, Copy to Left or Delete, if these changes exist in only one model. After performing a three-way analysis, additional comfort copy commands are available for automatically merging or resolving conflicts more conveniently.

A smart line handling mechanism automatically adjusts signal lines of copied or deleted blocks. If a block is copied, the related signal lines are copied as well. If a block is deleted, the related signal lines are automatically deleted or reconnected. All merge operations are logged in the merge log window.

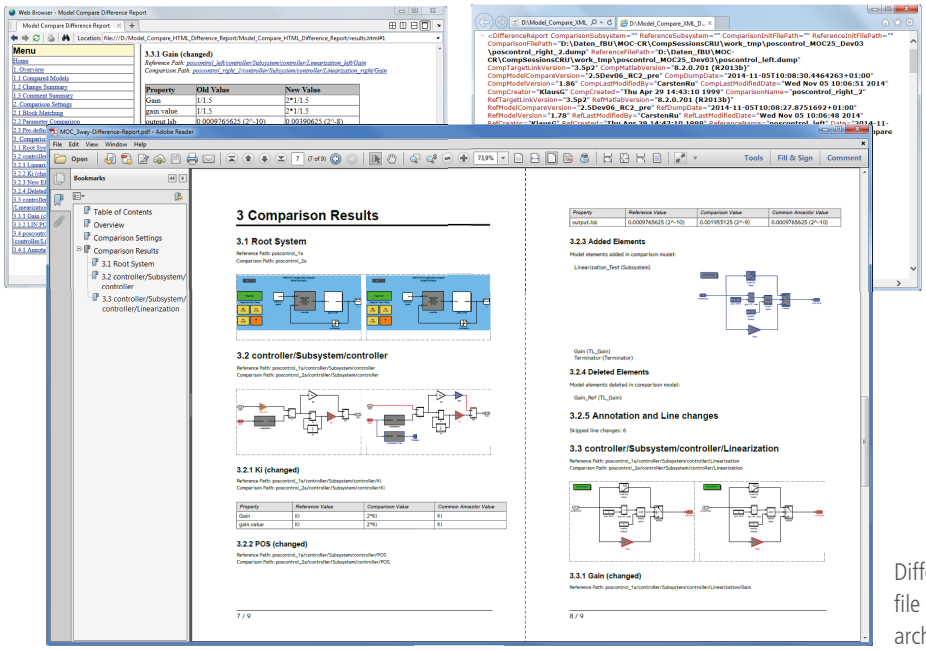


Model Compare Merge Support.

Report Generation

Comparison results and associated comments can be saved as difference reports in HTML, PDF and XML format. The generated reports also include all the comments that were created during a review. Thus, Model Compare´s difference reports are also a means of filing review results.

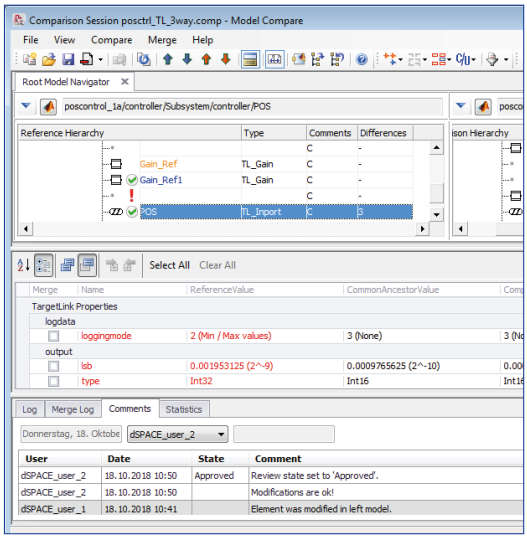
There are advanced configuration options for customizing the report according to individual preferences. You can specify the subsystem level up to which screenshots are provided and the level of detail of the report.



Difference reports in different file formats for publishing and/or archiving the comparison results.

Review Support

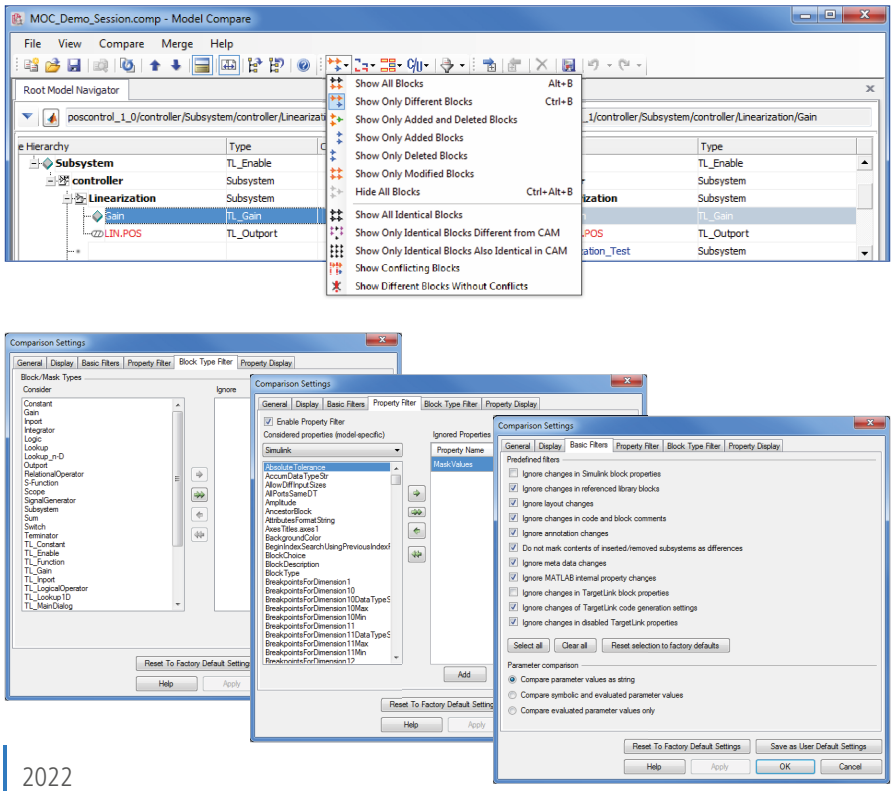
With Model Compare you can associate review comments with the found block and property differences or even with the complete comparison session. Remaining differences can be marked as approved. Time stamps as well as author information are added automatically by the tool. Thus, Model Compare supports even complex reviews with multiple participants.



Review comments even from different users are supported.

Advanced Filter Options

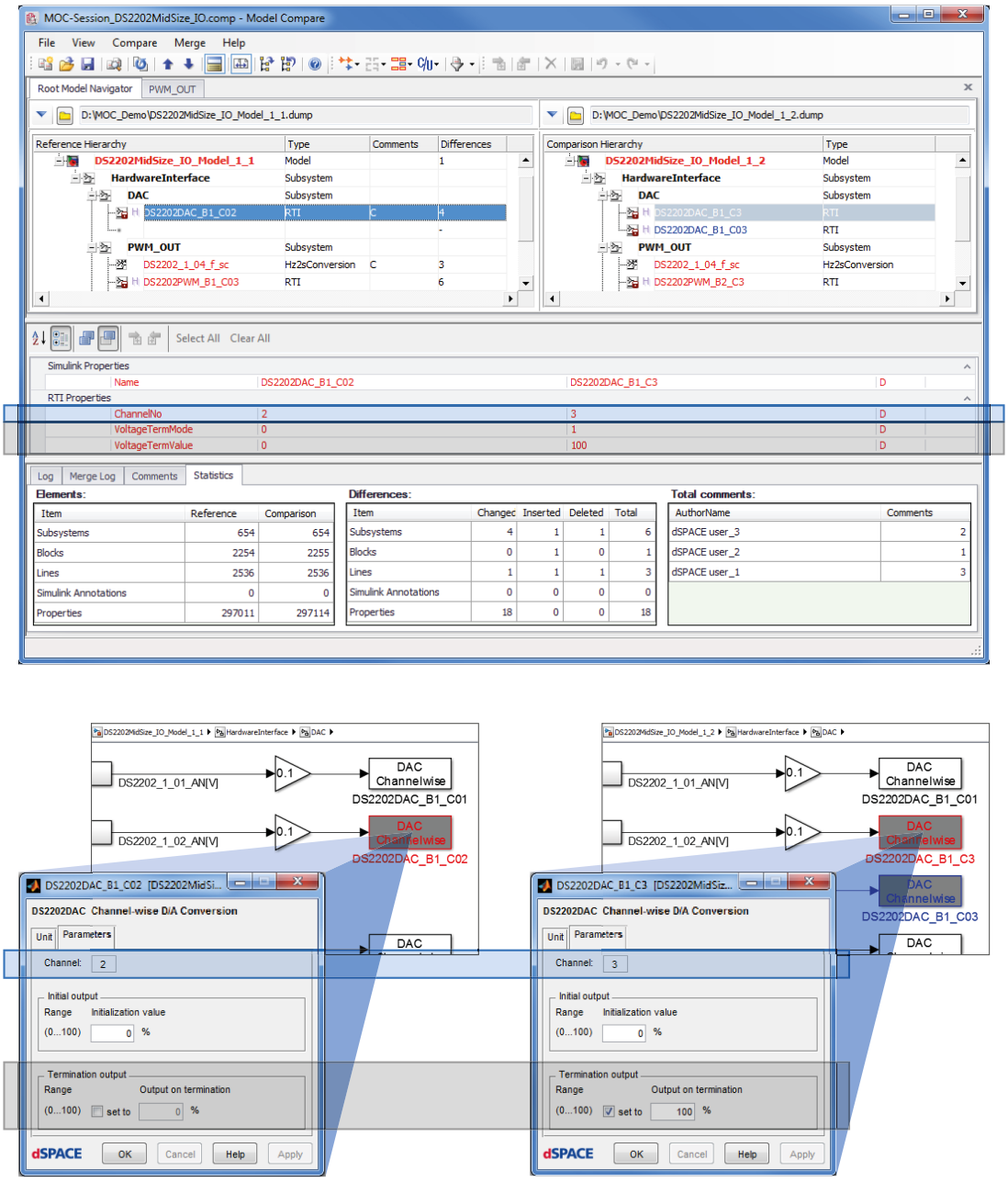
To make your work as efficient as possible, Model Compare provides several filter options. Thus, you can focus on the differences and model parts that are relevant to you and your current work: Via display filters you can specify which model elements to show in the hierarchy display. You can use predefined filters to focus on a specific kind of difference or filter out unimportant ones such as layout changes or simulation settings. You can also define your own filters to exclude element properties or even entire model elements from the comparison. To reuse the defined filter settings in other projects, you can save them as favorites or export them as XML files.



Display filter, block type and property filter as well as predefined basic filters for ensuring efficient model comparisons.

Customizing Block Comparison

In order to make the comparison as efficient as possible also for models from other domains, Model Compare provides an add-on mechanism. This mechanism lets you use hook scripts to integrate block-specific knowledge in the comparison of any number of Simulink-based models. Differences in mask variables or block dialog parameters can therefore be displayed immediately, a method that has already been established for TargetLink models. Thus, Model Compare provides a concise and efficient model comparison also for models from other domains, such as rapid control prototyping (RCP) and hardware-in-the-loop (HIL) simulation.



Example of 'customizing block comparison via a hook' for a block from RTI block library: Changes in the block properties Channel No. and Termination output block properties can be easily viewed in the Model Compare GUI.

dSPACE AUTOSAR Compare

Comparison of two AUTOSAR files made easy

Highlights

- Comparison of Classic and Adaptive AUTOSAR elements
- Visualization of differences in a tree view
- Automatic detection of dependencies
- Manual or automatic rule-based merging of differences
- Integration into the git version control system

Application Areas

With AUTOSAR (ARXML) files, as with any distributed development, it is helpful to know which parts of a file have been changed by colleagues, customers, or suppliers. There are products that visualize text-based differences. However, the XML-based AUTOSAR standard and the many element dependencies in ARXML files often make text-based comparisons too complex. By providing element-level comparisons, dSPACE AUTOSAR Compare gives you a much better overview of relevant differences.

Key Benefits

dSPACE AUTOSAR Compare is a dedicated comparison tool for Classic and Adaptive AUTOSAR for detecting and merging the differences between two AUTOSAR files. dSPACE AUTOSAR Compare always considers dependencies between the elements. It compares the elements, including all their properties and child elements, in a tree, giving you a clear overview. You can use filters to detect differences, missing elements, and even irrelevant differences, such as a different UUID. dSPACE AUTOSAR Compare can be integrated into the git version control system to view differences between different revisions of ARXML files and to assist with merge conflicts.

Comfortable Merge

When you know what differences exist between two files, you can easily copy individual elements from one side to the other side. You always have full control over which elements you copy and overwrite. dSPACE AUTOSAR Compare supports you in every single merge operation by highlighting all elements referenced by the one you selected and asking you whether to copy them as well. It considers not only the top level, but all levels.

Rule-Based Merge

Do you have a defined workflow in which you always want to adopt certain elements from one product, but discard others? Or do you repeatedly face problems when you use products from different manufacturers in your AUTOSAR workflow? dSPACE AUTOSAR Compare lets you write rules to automate these repeating tasks. For the exchange between SystemDesk and TargetLink, which is common practice for many customers, dSPACE provides such a rule file.

Automation

All functions can also be automated via an API. This is realized by a special console application, which ensures that you receive meaningful status messages. In combination with the rule-based merge, you can automate the complete AUTOSAR workflow. To ensure that you always have an overview and can work according to your work processes, dSPACE AUTOSAR Compare also lets you generate reports on differences and the performed merge actions.

Functionality Overview

Functionality	Description
General	■ Detection of differences and visualization for an easy and convenient overview of changes in AUTOSAR artifacts ■ Integration into the git version control system ■ Tree-based manual merge for full control over the merge process ■ Intelligent detection of dependencies supports manual merge operations ■ Automatic rule-based merging enhances safety and ease-of-use, reducing error-prone and repetitive work when merging differences ■ Predefined rule set for work with TargetLink and SystemDesk ■ Individual rule set definition for a faster development process without any project-specific workarounds ■ Dedicated report generation for comparison and merge tasks ■ Command line interface for automatic use cases, such as continuous integration ■ Support for Classic AUTOSAR versions from 4.0 up to R20-11 ■ Support for Adaptive AUTOSAR version R20-11

Order Information

Product	Order Number
dSPACE AUTOSAR Compare	■ DARC

Relevant Software

Software		Order Number
Optional	■ SystemDesk	■ See relevant product information
	■ TargetLink	■ See relevant product information
	■ Model Compare	■ See relevant product information

www.dspace.com

Germany
dSPACE GmbH
Rathenaustraße 26
33102 Paderborn
Tel.: +49 5251 1638-0
Fax: +49 5251 16198-0
info@dspace.de

United Kingdom
dSPACE Ltd.
Unit B7 · Beech House
Melbourn Science Park
Melbourn
Hertfordshire · SG8 6HB
Tel.: +44 1763 269 020
Fax: +44 1763 269 021
info@dspace.co.uk

France
dSPACE SARL
7 Parc Burospace
Route de Gisy
91573 Bièvres Cedex
Tel.: +33 169 355 060
Fax: +33 169 355 061
info@dspace.fr

Croatia
dSPACE Engineering d.o.o.
Ulica grada Vukovara 284
10000 Zagreb
Tel.: +385 1 4400 700
Fax: +385 1 4400 701
info@dspace.hr

China
dSPACE Mechatronic Control
Technology (Shanghai) Co., Ltd.
Unit 01-02,06-09, 19F/L
Middle Xizang Rd. 168
The Headquarters Building
200001 Shanghai
Tel.: +86 21 6391 7666
Fax: +86 21 6391 7445
infochina@dspace.com

Japan
dSPACE Japan K.K.
10F Gotenyama Trust Tower
4-7-35 Kitashinagawa
Shinagawa-ku
Tokyo 140-0001
Tel.: +81 3 5798 5460
Fax: +81 3 5798 5464
info@dspace.jp

USA and Canada
dSPACE Inc.
50131 Pontiac Trail
Wixom · MI 48393-2020
Tel.: +1 248 295 4700
Fax: +1 248 295 2950
info@dspaceinc.com

Korea
dSPACE Korea Co. Ltd.
16th floor, Dongwon Building
60 Mabang-ro
Seocho-gu
06775 Seoul, Republic
of Korea
Tel.: +82 2 570 9100
info@dspace.kr

04/2022