dSPACE Prototyping Systems and Tools

Developing, validating, and experiencing new functionalities in real environments
Contents

dSPACE Prototyping Systems 3
Fullpassing 4
Bypassing – Externally or Directly on the ECU 5
Production Focus 8
Use Cases 9
Prototyping Hardware and Software 17
Engineering Services 19
dSPACE Prototyping Systems

Accelerated function prototyping for controller development

Highlights
- Test and run new functionalities in real environments
- Implement Simulink® models at the click of a button
- Access all relevant data conveniently during run time
- Choose a system that fits your needs, from high-end prototyping hardware to production ECUs

Key Benefits
dSPACE prototyping systems let you develop, optimize, and test new control strategies in a real environment quickly without manual programming. Rapid prototyping helps you detect design faults and correct them immediately. With the dSPACE implementation software, Real-Time Interface (RTI) and ConfigurationDesk, models designed with MATLAB®/Simulink can automatically be implemented on the dSPACE hardware. Easy-to-use libraries with numerous interface functionalities let you connect inputs and outputs to the model. With the experiment software ControlDesk, you can monitor and tune variables during run time by using graphical instruments. As the prototyping hardware is far more powerful than actual production electronic control units (ECUs) with regard to processing power and memory space, almost no hardware limitations have to be considered. dSPACE prototyping systems can be used as a substitute (fullpassing, p. 4) or an extension (bypassing, p. 5) of the ECU. For building your own tailored prototyping system, dSPACE offers a wide range of off-the-shelf software and hardware components for in-vehicle, laboratory or test bench usage. The components are well suited for automotive, aerospace, medical, robotics and many other application fields.

Development process with rapid prototyping (example: dSPACE MicroLabBox).
Fullpassing

Using a Prototyping System as a Flexible Experimental ECU

If a new ECU or a new set of control functions has to be developed from scratch, often different control strategies have to be tested, modified and benchmarked against each other to find the optimal solution. In order to accelerate the development, quick trials have to be run at an early stage (rapid control prototyping; RCP). Tests in a vehicle or on a test bench have to be carried out even before the new ECU hardware is available. Producing an application-specific prototype ECU for this purpose, e.g., by modifying a production ECU, would be expensive, time-consuming, and inflexible. Instead, developers can use a powerful off-the-shelf prototyping system that acts as an experimental ECU but has many advantages.

The dSPACE prototyping systems allow developers to concentrate completely on the new function design without having to worry about computing power and memory space – the system offers plenty of both, ensuring maximum flexibility. Unlike a production-type ECU, programming a dSPACE prototyping system is easy. This makes it particularly convenient to change ECU function designs without having to do manual programming. A wide range of I/O interfaces can be configured and easily connected to the function model by means of dSPACE implementation software (p. 18). Automatic code generation allows the physical implementation on the prototyping system at the click of a button. For evaluation and calibration, parameters of the control function can be changed and signals can be captured on-the-fly by using ControlDesk. This method provides very short iteration times.

<table>
<thead>
<tr>
<th>dSPACE Prototyping System</th>
<th>ECU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Scalable, floating-point, high processing power</td>
</tr>
<tr>
<td>Memory</td>
<td>Large</td>
</tr>
<tr>
<td>I/O</td>
<td>Flexible, modular, configurable</td>
</tr>
<tr>
<td>Signal conditioning and power stages</td>
<td>Flexible, modular, configurable</td>
</tr>
<tr>
<td>Size</td>
<td>All sizes from small to fairly large, depending on use scenario</td>
</tr>
<tr>
<td>Programming</td>
<td>Easy and intuitive</td>
</tr>
</tbody>
</table>

Fullpassing – the prototyping system completely replaces the ECU.

Connecting Sensors and Actuators

Signal conditioning and power stages are essential for the optimal connection of sensors and actuators to prototyping systems. To cover today’s variety of sensors and actuators, dSPACE offers the RapidPro hardware as an optional extension to a dSPACE prototyping system. The very compact, robust and modular RapidPro hardware consists of ready-made and easy-to-insert hardware- and software-configurable I/O-interfaces for the highest degree of flexibility. Changes in the sensor-actuator setup are no longer a risk for your development project, and home-made solutions are no longer necessary.
Bypassing – Externally or Directly on the ECU

New Functions for Existing Controllers
Unlike fullpass approaches, where the ECU is completely replaced by the prototyping system, bypassing is used to develop only individual parts of the ECU software from scratch or modify them, e.g., single control functions. These parts run either on a prototyping system that is connected to an existing ECU (external bypassing) or directly on the ECU (on-target bypassing). dSPACE offers a comprehensive range of hardware and software that help prepare an existing ECU for bypassing and that support the different bypass methods.

Fast and Flexible Bypass Integration
To use external and on-target bypassing, the existing ECU code has to be prepared. With service-based bypassing as supported by dSPACE, virtually any number of functions in the ECU code can be prepared for bypassing by integrating service calls, also known as bypass hooks. These service calls can be flexibly used in the MATLAB®/Simulink® modeling environment for the synchronous measurement and calibration of ECU variables and parameters, for ECU flash programming, and for bypassing.

To help you prepare the ECU code for bypassing, dSPACE offers the ECU Interface Manager with the Binary Code Management Module as an easy-to-use tool with an intuitive, graphical view on the code structure of an ECU.

It gives you a convenient and flexible way to implement bypass services directly and automatically in the binary code of the ECU. There is no need for the ECU supplier to modify the source code and run through the whole production development process and tooling again. This approach saves time and money, and it also increases flexibility. Alternatively, services for bypassing can be manually integrated in the ECU source code. dSPACE offers its generically designed bypass services and service calls as C sources, so they can be compiled and linked with the existing ECU code. Using ConfigurationDesk or the RTI Bypass Blockset, you can conveniently develop new model-based bypass functions in Simulink.

ECU code can be conveniently prepared for bypassing with the help of the dSPACE ECU Interface Manager (part of the ECU Interface Base Package).
External Bypassing

This bypass method is an efficient way to develop new control functions and optimize existing controller strategies. ‘External’ means that a dedicated RCP system is attached to an ECU. The RCP system executes a new control function \( f(x) \) synchronously to the original code that runs on the target ECU. RCP systems have almost no resource constraints with regard to available RAM, ROM (flash) as well as processor performance, and they also provide additional I/O channels. Therefore, even complex Simulink models can be executed as external bypass functions. Real-time behavior is ensured by specific synchronization mechanisms of the ECU interface. In case of a vehicle-in-the-loop simulation, where an ECU is tested in a real vehicle but under (partly) virtual conditions (e.g., virtual traffic or camera object lists), the same method allows you to simulate a virtual environment on a dSPACE real-time system that can be fed into the ECU in real time (external environment bypassing). In more complex hardware setups, you can also connect several ECU interfaces to your RCP system. Using the bypassing approach, it is possible to inject values into one ECU, modify the controller algorithm on another ECU, and capture internal variables of a third ECU, all at the same time. Again, the synchronization mechanisms of the ECU interfaces ensure the real-time behavior of the setup as a whole. The number of ECUs that can be bypassed in parallel is only limited by the available data processing resources on the RCP system.

dSPACE supports numerous interfaces for connecting the prototyping system to the ECU. If the ECU has standard CAN, CAN FD, or Ethernet interfaces, a direct access via the XCP protocol without further hardware is possible. If such bus interfaces are not available for bypassing, and high real-time performance with a high bandwidth is required, you can use the Generic Serial Interface (DCI-GSI2), which is connected to the ECU’s on-chip debug interface, such as NEXUS or DAP. If using such an interface is not possible either, dSPACE also supports ECU-specific plug-on devices (PODs).

Adding or optimizing even comprehensive functions on an existing ECU using external prototyping systems.

Vehicle-in-the-loop simulation: simulating virtual environments which are fed into the ECU in real time, e.g., during test runs.

Example of an external bypassing scenario with dSPACE tools.
**On-Target Bypassing**

If an ECU provides all relevant I/O interfaces and has sufficient free resources, new functions can also be developed directly on the ECU. This reduces development costs, because no additional hardware and wiring harness is necessary. The new functions are executed directly on the target hardware, which means there are no communication latencies to external development hardware. Therefore, new functions can be integrated into very fast control loops. Using dSPACE TargetLink as a code generator allows for a seamless transition to production and a more efficient use of the limited ECU resources. Another benefit is that the additional resource consumption on the ECU can be determined early in the development phase. Furthermore, using the certified TargetLink code generator together with an ECU that was already cleared for production can improve the overall operational safety for the prototyping phase, e.g., during fleet tests.

Adding or optimizing functions and fixing software bugs directly on an existing ECU.

Example of an on-target bypassing scenario with dSPACE tools for an efficient use of the limited ECU resources.
Production Focus

Guidance During Modeling
Not all blocks provided by Simulink® and not all modeling styles are suited for efficient and reliable production code generation. Established modeling guidelines available for the dSPACE production code generator, TargetLink, let you focus on the later production code generation very early on while you are using Simulink Coder™ in the rapid control prototyping (RCP) phase.

Validating Production Code on dSPACE Prototyping Systems and Production ECUs
Production code generated with TargetLink can be executed on dSPACE real-time systems (MicroAutoBox II, SCALEXIO, DS1104, PHS hardware) for validation purposes – early in the development process. This enables you to immediately see the effect that the specific implementation of an algorithm as production code has on a function. You can also validate the production code directly on an existing ECU. In addition to testing the expected behavior, you can check whether the resources of the target ECU are sufficient. Early validation drastically reduces project risks. dSPACE also offers solutions for verifying AUTOSAR application software on a prototyping system (please inquire).
Use Cases

dSPACE prototyping systems can be used in many different application areas of many different industries (automotive, aerospace, transportation, automation, robotics, medical, energy, home appliances, etc.). Whatever your application requires, dSPACE offers a wide range of setups to cover your needs. The following use cases (p. 9-16) are just a few examples of the many possibilities that open up.

Developing Electric Motor Control Strategies

Electric motors are commonly used in many industries. In the automotive industry, electric and hybrid electric vehicles play a major role in strategies for reducing fuel consumption and emissions. The large number of different electric motors and the need to integrate them into existing electric/electronic systems calls for highly flexible development systems. The scope, flexibility, and performance of the I/O interfaces determine whether they can support the desired selection of electric motors and sensors.

Development Environment

dSPACE’s processor- and FPGA-based systems lets you run even the most demanding electric motor applications. Furthermore, dSPACE offers specific I/O interfaces and blocksets for MicroAutoBox II, MicroLabBox as well as modular systems. The powerful I/O with on-board signal processing provides all the typical interfaces for connecting sensors and power stages to support BLDC, and synchronous as well as asynchronous motors. Advanced I/O functions provide PWM control patterns for both basic control methods, block and sine commutation. Comprehensive software support is available so that the full potential of the hardware can be harnessed. To achieve high productivity from the beginning, dSPACE supplies ready-made Simulink demo models with comprehensive control algorithms for BLDC and synchronous motors.

![Diagram of dSPACE prototyping systems and tools](image-url)
Developing and Validating Driver Assistance Systems and Automated Driving Functions

The topic of highly automated driving is the focus of many automobile manufacturers’ development activities. Requirements such as 360° redundant surround view with numerous heterogeneous sensors, high-precision positioning or car connectivity are also challenging topics for tool suppliers. The dSPACE response to this is an end-to-end tool chain for autonomous driving from a single source. Unique rapid prototyping solutions of high-performance platforms and a tailored software environment allow for the development of complete multisensor applications in the vehicle, from perception and fusion algorithms to real-time controls.

Prototyping Functions for ADAS and Automated Driving

Typically, functions for ADAS and automated driving, such as adaptive cruise control (ACC), autonomous emergency braking (AEB), lane keeping assist (LKA), or the intersection assistant, consist of different processing stages.

RTMaps, a prototyping tool for multisensor applications, is typically used on the MicroAutoBox Embedded PC hardware platform for developing perception and sensor fusion algorithms in C++, OpenCV, or Python under Linux or Microsoft® Windows®. The calculation result is then transmitted to the application and control functions on the dSPACE MicroAutoBox II. The MicroAutoBox II performs real-time processing, ensures functional safety, integrates Simulink® and AUTOSAR code, and serves as the interface to the vehicle network, so it can interact with the braking and steering ECUs, for example. The data exchange between RTMaps and the Simulink model on the MicroAutoBox II is established via the RTMaps Interface Blockset.

![Diagram of Prototyping Functions for ADAS and Automated Driving](image)

**Sensors**
- Video
- Radar, lidar
- Map, GNSS
- V2X

**Perception & Fusion**
- Data processing
- Data fusion
- Localization
- Environment model

**Application**
- Trajectory planning
- Decision making
- Control functions

**Actuators**
- Steering
- Brakes
- Engine, transmission
- V2X, HMI

RTMaps (Open Source Computer Vision, opencv.org)
MicroAutoBox II with Embedded PC
Simulink® model
Real-time application
Developing Engine Controls

Saving fuel and reducing emissions (NOx, CO2) requires continuous research on new operating processes for internal combustion engines. New control strategies have to be developed on the test bench and verified by test drives. The efficient development of these strategies requires systems that can communicate with the vehicle’s bus systems and provide the flexibility to connect various sensors and actuators to control the combustion process. New control strategies like homogeneous charge compression ignition (HCCI) or controlled auto ignition (CAI) require a very precise and extremely fast closed-loop control.

Development Environment

A combination of MicroAutoBox II and RapidPro ensures convenient advanced engine control development on the test bench and in the vehicle. The MicroAutoBox II prototyping system can execute computation-intensive embedded software algorithms, directly generated from Simulink models, and provides the necessary automotive bus interfaces to connect to other electronic control units. MicroAutoBox II also supports cylinder-pressure-based control development with its fast analog input channels. By using the FPGA resource of MicroAutoBox II, you can even implement extremely fast control loops with an in-cycle response to enable control design for CAI or HCCI strategies. RapidPro is a scalable and modular system which adds off-the-shelf signal conditioning and power stage capabilities to the prototyping system. It also provides engine-specific I/O interfaces, such as modules for lambda and knock sensors, and power stage modules for valves, relays and electric motors used in auxiliary units. The figure below shows a common setup for engine control development.
Chassis Control Development

When developing and testing new chassis control strategies, e.g., for higher driving safety, comfort and agility, high-processing power is needed to calculate the controller models. The wide variety of sensors and actuators used in chassis control applications requires high flexibility for the signal conditioning and power stage hardware.

Development Environment

For chassis control development, dSPACE offers signal conditioning and power stage capabilities through RapidPro hardware in combination with a dSPACE prototyping system (MicroAutoBox II or AutoBox). This is an easy way for you to start developing chassis control strategies, because you can quickly adapt the sensors and actuators to your prototyping system. In this scenario, the RapidPro hardware works as an extension to the dSPACE AutoBox, which provides the high processing power and I/O interfaces. The RapidPro SC Unit and various signal conditioning modules provide the links to typical chassis control sensors (e.g., pressure, acceleration, yaw rate). The RapidPro power unit provides dedicated power stages for driving actuators such as DC electric motors or valves.
Optimizing Vehicle Dynamics for Motorcycles

Modern motorcycles require sophisticated control strategies for vehicle dynamics, such as anti-lock braking systems (ABS), automatic stability control (ASC), dynamic traction control (DTC), and dynamic damping control (DDC). When developing and validating new controllers on the real vehicle, changing the center of gravity by installing an additional prototyping system affects the motorcycle’s driving behavior. This poses a crucial problem for vehicle dynamics development during test drives. In addition, the strong vibrations of a motorcycle’s frame pose a challenge to conventional prototyping systems. On-target prototyping on the factory-mounted ECU allows rapid control prototyping (RCP) without any impact on the vehicle driving behavior.

Development Environment

With the ECU Interface Software (including the ECU Interface Manager, the Binary Code Management Module, RTI Bypass Blockset and the On-Target Module), a complete, model-based tool chain for on-target bypassing is available. It is suitable for incremental improvements, bug-fixing, optimizations, and complete replacements of control functions within existing ECU software. It lets you quickly integrate new control functions directly into the HEX code of the existing ECU software. Modifications to the ECU’s software architecture are not required, neither are access to the C source code and build environment of the OEM and ECU supplier. Thus, the tool chain increases the level of independence from production software builds, and allows for fast development and prototyping iterations.

Prototyping on the Target ECU

On-target bypassing means using an existing ECU as the prototyping system. This method relies on the free processing power, memory resources, and the availability of bus interfaces, sensor interfaces and actuator interfaces on the target ECU. You therefore do not have to install an additional, dedicated prototyping system, and you can simply distribute the new functionality to multiple test vehicles. Using the original ECU makes rapid control prototyping under extreme environmental conditions and in safety-critical applications easier.
Prototyping for Active Noise and Vibration Cancellation

The noise produced by helicopters, particularly as they come into land, is far from pleasant, and their vibrations also affect the pilot. Noise and vibration are caused by the air currents in the plane of the rotors. Noise is generated when one rotor blade collides with the vortex trail of the rotor blade in front. One way to reduce the noise emissions as well as the overall vibrations of a helicopter is to use additional, piezoelectrically controlled flaps in the rotor blades.

Development Environment

A SCALEXIO AutoBox with a processor board and an FPGA board is the optimal prototyping system for noise and vibration reduction applications. It has enough processing power to provide the fast control of the multiple piezo actuators per rotor that control the flaps’ angle of attack. The input for the control strategy comes from pressure sensors in the rotor blades and additional microphones on the landing skid, which measure the air pressure curves. Synchronous preprocessing of multiple channels can be conducted on the FPGA board without any impact on the processor board. Because the prototyping system has a modular setup, it can be flexibly scaled in case more microphone channels or more piezo actuators are needed. Model-based development of the control strategy is not only possible on the processor board but also on the FPGA board if very short response times are required.

---

**SCALEXIO AutoBox**
- DS6001 Processor Board
- DS2655 FPGA Base Board
- DS2655M1 Multi-I/O Module

**Software**
- MATLAB®
- Simulink®
- Stateflow®
- Simulink Coder™
- ConfigurationDesk
- RTI FPGA Programming Blockset
- ControlDesk

**Piezoelectrically controlled flaps**

**Host**

**Microphones on the landing skids and pressure sensors in the rotor blades**

**Computed angle of attack**

**Noise measurement**

**Modification of the flaps’ angle of attack**
Prototyping Medical Microsystems

In the area of cardiac rhythm management, devices such as artificial pacemakers can profit from optimized control strategies, reduced power consumption, and improved reliability. An intelligent blood flow sensor helps improve the measurement and control of the blood flow through the heart, preserve the remaining battery life, and detect defective lead wires. With rapid control prototyping, the functionality and performance of existing medical Microsystems, such as ultra-small monitoring devices and therapy aids as well as intelligent implants can be enhanced and improved. Rapid control prototyping has been successfully used even in laboratory tests with live probands to adjust and control their heart rates, both under anesthesia and awake.

Developing and Testing Medical Devices

Safety and reliability play a crucial role in the development of medical devices. New functions must have an optimal design and undergo extensive testing. In many cases, capturing and preprocessing signals is an integral part of function development. With MicroLabBox, you can outsource extensive and computation-intensive signal preprocessing tasks, such as filtering or signal analysis, to an integrated FPGA. Connecting BNC cables directly to MicroLabBox for processing analog signals minimizes the influence of external interferences on the signal and makes it possible to achieve a high signal quality. During or after the development of the medical device, MicroLabBox can also be used as a testing system. With it, you can reproducibly simulate many different environment conditions, e.g., based on test algorithms or existing measurement data. This increases the medical device’s maturity, saves time, reduces costs, and minimizes the risks compared to tests on a living organism.
Developing Functions for Multilevel Converters

The increasing use of renewable energy sources, such as wind and solar power, inevitably leads to fluctuations in energy supply. To not overload the energy grid, control mechanisms for balancing these fluctuations are required. This is where multilevel converters come into play, as they control the amount of energy fed to the grid. They are also used when electric energy is transferred to the consumers over long distances, because long-distance power distribution requires energy grids with different line voltages.

Development Environment
To develop and validate control algorithms, e.g., for grid injections, solar and wind power inverters, charging stations, and power transmission between different power grids, a SCALEXIO real-time system is the perfect choice. Based on the compact DS6001 Processor Board with a powerful real-time processor and extensible with several I/O boards and a scalable FPGA subsystem, the SCALEXIO LabBox can be adopted to cover all computational and I/O requirements for various use cases. dSPACE’s scalable FPGA subsystem allows switching times of less than 1μs and can be scaled to support up to 420 switches in a converter application. At all times, you benefit from SCALEXIO’s very high real-time performance with powerful processor and latest FPGA technology.
# Prototyping Hardware and Software

## dSPACE Hardware for Rapid Prototyping

<table>
<thead>
<tr>
<th>Product Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Compact Systems</strong></td>
<td></td>
</tr>
</tbody>
</table>
| DS1104 (Single-Board Hardware) | ■ PCI/PCIe-based controller board for use in standard PCs  
■ Real-time processor and I/O on a single board  
■ Cost-effective  
■ Ideal for developing smaller control applications and for education purposes |
| MicroLabBox Hardware       | ■ Compact all-in-one development system for laboratory purposes  
■ Powerful real-time processor and comprehensive I/O on a single system  
■ User-programmable FPGA  
■ Dedicated electric motor control features  
■ Ethernet and CAN bus interfaces  
■ Easy I/O access via integrated connector panel |
| MicroAutoBox Hardware      | ■ Extremely compact and robust stand-alone system for rapid control prototyping, tests, and data acquisition in tough environments, e.g., in vehicles or airplanes  
■ Powerful real-time processor, programmable FPGA, comprehensive I/O and automotive bus interfaces on a single system  
■ MicroAutoBox Embedded PC – shock- and vibration-proof MicroAutoBox II extension with Intel® Core i7 processor for Windows®- or Linux-based applications  
■ MicroAutoBox Embedded SPU (Sensor Processing Unit) – MicroAutoBox II extension or stand-alone system with multicore ARM® CPU and embedded NVIDIA® GPU for developing perception and sensor fusion algorithms  
■ MicroAutoBox Embedded DSU (Data Storage Unit) – Compact and robust storage hardware to extend Embedded PC or Embedded SPU with up to 64 TB capacity for recording large volumes of data |
| RapidPro Hardware          | ■ Compact, robust and modular hardware that extends dSPACE prototyping systems with extra signal conditioning and power stages  
■ Software- and hardware-configurable off-the-shelf modules  
■ For in-vehicle, laboratory and test bench use |
| ECU Interface Hardware     | ■ DCI-GSI2: compact and robust ECU interface hardware for external bypassing, measurement, ECU calibration, and flash programming |
| **Modular Systems**        |                                                                                                                                                                                                             |
| SCALEXIO Hardware          | ■ Next generation modular real-time system with optimum scalability and high performance for challenging applications  
■ Available as a laboratory system (SCALEXIO LabBox) and as an in-vehicle system (SCALEXIO AutoBox)  
■ Wide range of comprehensive I/O, automotive bus and network interfaces, programmable FGPA, and powerful real-time processors for various applications  
■ Synchronous coupling of multiple systems (to increase computation power or provide even more I/O channels) |
| PHS Hardware               | ■ Modular real-time system  
■ Available as a laboratory system (with Expansion Box) and robust in-vehicle system (with AutoBox)  
■ Wide range of comprehensive I/O, automotive bus interfaces, programmable FGPA, and powerful real-time processors for various applications  
■ Synchronous coupling of multiple systems (to increase computation power or provide even more I/O channels) |
## dSPACE Software for Rapid Prototyping

<table>
<thead>
<tr>
<th>Product Category</th>
<th>Description</th>
</tr>
</thead>
</table>
| Implementation Software (Real-Time Interface, ConfigurationDesk) | - Automatic implementation of MATLAB®/Simulink®/Stateflow® models on dSPACE hardware  
- Graphical I/O configuration via comprehensive libraries  
- Implementation of behavior model code and I/O function code on dSPACE hardware  
- Comprehensive support and extensions for buses and networks  
- Real-Time Interface (RTI) for PHS-based systems, MicroAutoBox II, MicroLabBox, DS1104  
- ConfigurationDesk for SCALEXIO systems |
| Bus Interface Software | - Comprehensive tools for configuring and implementing the bus communication in real-time applications  
- Support of LIN, CAN, CAN FD, FlexRay, and Ethernet (including SOME/IP)  
- Support of various description formats (DBC, LDF, FIBEX, AUTOSAR)  
- Convenient configuration by means of a graphical user interface without the need for programming  
- Various simulation and manipulation options included (e.g., CRC, Counter, SecOC) |
| ControlDesk | - Comprehensive experiment environment based on graphical instruments for intuitively managing, controlling, and automating experiments  
- Access to the model's parameters and signals of the real-time application during run time  
- Wide range of configurable graphical instruments such as sliders, gauges, switches, and knobs as well as look-up tables and plotters for convenient and intuitive handling  
- For applications from in-vehicle optimized use to photorealistic instruments  
- Bus Navigator: Add-on for conveniently monitoring, logging, and replaying bus and network traffic; support of LIN, CAN, CAN FD, FlexRay, Ethernet (including SOME/IP), and related description files |
| Platform API Package | - Automation libraries for accessing dSPACE systems  
- Provides a convenient way to download and control models, and also gives high-level access to the model variables for reading, writing, stimulating, capturing, etc.  
- Support of Python and any .NET-compliant language  
- Compliant with ASAM XIL API standard |
| ECU Interface Base Package | - Provides real-time access to production ECUs during RCP, HIL and field tests  
- Supports efficient function bypassing directly on existing ECUs or together with separate prototyping hardware  
- Lets you automatically prepare the ECU access on the basis of HEX code  
- New package combines the functionalities of the RTI Bypass Blockset and the ECU Interface Manager  
- Optional Binary Code Management Module for HEX-code-based service integration  
- Optional On-Target Module for implementing new bypass functions directly on the target ECU |
| RTMaps – Real-time Multisensor Applications | - Diagram-based development of multisensor applications under Windows® and Linux on x86 or ARM® platforms  
- Comprehensive component libraries for sensors, vehicle networks, and data processing  
- Easy integration of C++, Python and Simulink® code  
- Support of MicroAutoBox Embedded PC and MicroAutoBox Embedded SPU |
## Engineering Services

dSPACE offers numerous services to assist you with your prototyping activities. Whatever your use case for the dSPACE prototyping systems, we are ready to support you with:

### Rapid Prototyping

<table>
<thead>
<tr>
<th>Service (Examples)</th>
<th>Details (Examples)</th>
</tr>
</thead>
</table>
| Custom I/O solutions               | - Custom I/O interfaces (e.g. customized FPGA-based I/O for SCALEXIO)  
- Custom Real-Time-Interface (RTI) I/O blocksets or custom I/O functions for ConfigurationDesk  
- Modifying and designing signal conditioning and power stage modules for RapidPro hardware  
- Integrating third-party hardware and software  
- Developing and integrating customer-specific I/O and camera interfaces for MicroAutoBox Embedded PC and MicroAutoBox Embedded SPU  
- Modifying and designing signal conditioning and power stage modules for RapidPro hardware  
- Integrating third-party hardware and software  
- Developing and integrating customer-specific I/O and camera interfaces for MicroAutoBox Embedded PC and MicroAutoBox Embedded SPU  
- Developing custom-specific components for RTMaps |
| Turn-key prototyping systems       | - Concept design  
- System configuration  
- Signal list / wiring harness specification and assembly  
- I/O model and system tests  
- On-site commissioning  
- Hands-on training |
| ECU interface solutions            | - Development of ECU-specific plug-on devices and connector adapters for the dSPACE bypass interface DCI-GSI2  
- Support for setting up HEX-code-based external and on-target bypassing projects with the ECU Interface Manager  
- ECU-specific configuration of dSPACE bypass services and integration into ECU source code  
- Development and customer-specific adaptation of ECU flash programming tools |
| Tool introduction and training     | - Support during evaluations and pilot projects  
- Hands-on, customer-specific training |