dSPACE Prototyping Systems

Accelerated function prototyping for controller development

Highlights

- Control design development and optimization without manual programming
- Intuitive and comprehensive software environment
- Scalable and flexible hardware
- High processing performance
- Broad range of interfaces
- Flexible signal conditioning and power stages

Key Benefits

dSPACE prototyping systems are flexible development systems that let you develop and optimize your control designs for the real plant without manual programming. Design faults are found immediately and corrections can be carried out on the spot. No special expertise is necessary for implementation on the prototyping system. With dSPACE’s implementation software Real-Time Interface (RTI), MATLAB®/Simulink® models can be implemented on the dSPACE hardware automatically. A graphical block library with numerous interface functionalities is provided for configuring inputs and outputs and connecting I/O to the model. With an RTI extension, it is also possible to execute AUTOSAR software components and compositions on dSPACE real-time hardware. As the prototyping hardware is far more powerful than the actual production electronic control unit (ECU) with regard to processing power and memory, you have no hardware limitations to worry about. The scalable, modular and highly configurable RapidPro hardware for signal conditioning and power stages allows you to directly connect the dSPACE prototyping hardware to your sensors and actuators. dSPACE has a wide range of off-the-shelf software and hardware components for building your own tailored prototyping system. In addition, solutions with programmable FPGAs allow numerous fields of application. You can also take advantage of our engineering service to make customer-specific extensions. dSPACE prototyping systems can be used directly in the vehicle, in the laboratory or on test benches. They are well suited for numerous automotive applications (e.g., engine, transmission, hybrid chassis, body), aerospace (e.g., turbine, landing gear, vibration and noise reduction), and other fields of applications (e.g., drives, robotics, industrial automation).

Two Ways of Function Prototyping

There are two basic approaches to rapid control prototyping (RCP): fullpass and bypass.

- With **fullpass**, the prototyping system completely replaces the ECU. All sensors and actuators are connected to this prototyping hardware, and it has full authority to control the plant.

- With function **bypassing**, the prototyping system is used in parallel to the ECU. The synchronized connection between the two systems is carried out via dedicated bypass or ECU interfaces. Typically, only specific software functions that are under development are offloaded from the ECU to the prototyping system. The ECU continues to execute parts of its application code plus the operating system, network management and I/O.
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, quick trials have to be run at an early stage to verify the correctness of the control strategy. Tests in a vehicle or on a test bench therefore have to be carried out even before the new ECU hardware becomes available. Producing an application-specific prototype ECU for this purpose, e.g. by means of a modified production ECU, would be expensive, time-consuming and inflexible. Instead, developers can use a powerful off-the-shelf prototyping system which acts as an experimental ECU, but which has many advantages compared to other solutions. The dSPACE prototyping system allows developers to concentrate completely on the new function design without having to worry about computing power and memory – 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. Function prototyping begins with graphical descriptions of control functions, created and tested in a modeling and simulation environment such as MATLAB® / Simulink® / Stateflow®. Changes to the function model can therefore be carried out quickly and conveniently, and downloaded to the prototyping system via automatic code generation at a click. This procedure provides the shortest possible iteration times. I/O interfaces can be configured and easily integrated into the function model by means of a comprehensive graphical I/O block library which is part of the RTI software (p. 126). Parameters of the control function can be changed and signals can be captured on-the-fly by means of or ControlDesk® Next Generation (p. 160).

Connecting Sensors and Actuators

Signal conditioning and power stages are decisive 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 (p. 462) 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 interface modules for the highest degree of flexibility. Changes in the sensor-actuator setup are no longer a risk for your development project, home-made solutions are no longer necessary.

1) For information on possible product combinations, please contact dSPACE.
Comparison: dSPACE Prototyping System versus ECU

<table>
<thead>
<tr>
<th></th>
<th>dSPACE Prototyping System</th>
<th>ECU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Scalable, floating-point, high processing power</td>
<td>Optimized (i.e. limited) for target application; low processing power</td>
</tr>
<tr>
<td>Memory</td>
<td>Large</td>
<td>Limited</td>
</tr>
<tr>
<td>I/O</td>
<td>Flexible, modular, configurable</td>
<td>Application-specific, fixed</td>
</tr>
<tr>
<td>Signal conditioning and power stages</td>
<td>Flexible, modular, configurable via RapidPro Hardware (basic signal conditioning also integrated in MicroAutoBox(^1))</td>
<td>Application-specific, fixed</td>
</tr>
<tr>
<td>Size</td>
<td>All sizes from small to fairly large, depending on use scenario</td>
<td>Small</td>
</tr>
<tr>
<td>Programming</td>
<td>Graphical, convenient, automatic implementation</td>
<td>Difficult, manual</td>
</tr>
<tr>
<td>Host access</td>
<td>Dedicated high-speed interfaces</td>
<td>ECU-specific (e.g. CAN)</td>
</tr>
</tbody>
</table>

Bypassing

**Integrating New Functions in Existing Controllers**
In contrast to fullpass applications, where the ECU is completely replaced by the prototyping system, with the external bypass approach only individual parts of the algorithms are shifted to the prototyping system. Sensors and actuators are interfaced via the existing harness of the ECU and only the I/O which is additionally required needs to be connected to the prototyping system. The bypass method is an efficient approach, especially to developing new functions and optimizing existing controller strategies. The original controller executes all functions that remain unchanged, while new algorithms are executed on the prototyping hardware. Using the external bypass approach gives you great flexibility during the design phase, since you have almost no resource constraints in RAM, ROM, processor performance, or I/O channels. Real-time behavior is guaranteed even with complex bypass functions. In addition, the fast autonomous booting of the prototyping systems allow you to validate the behavior of the new function in realistic scenarios, for example, during test drives.

\(^1\) The MicroAutoBox II 1401/1507 does not offer standard I/O or signal conditioning but focuses on the bus and interface features (see p. 446).
External Bypass Methods and Interfaces
To divert the flow of data from the ECU to the prototyping system and back, dSPACE supports several methods and interfaces:

- DPMEM Plug-On Device (POD)
- On-Chip Debug Interface (e.g., JTAG/Nexus, JTAG/OCDS, NBD, AUD)
- XCP on CAN
- XCP on Ethernet (UDP/IP)
- XCP on FlexRay
- CCP

Additional I/O

Prototyping of new functions, additional I/O

Existing functions and I/O

External bypassing: New functions run on the prototyping system, while unchanged algorithms stay on the ECU.

1) For information on possible product combinations, please contact dSPACE.
Address-Based Bypassing
Address-based bypassing is one method for off-loading specific sections of ECU code to a prototyping system for function optimization and development. It can be applied in combination with a dual-port memory (DPMEM) or an on-chip debug interface. The method involves integrating customer-specific code patches into the ECU code. If the input and output variables of functions change, the ECU code typically also needs to be modified. Address-based bypassing allows very fast execution times and minimum latencies in data transmission. For example, using a dSPACE address-based bypass system with a DPMEM plug-on device (POD), the latency for 20 bytes to be sent from the ECU to the prototyping system and back again is only 2 x 8.5 µs.

Service-Based Bypassing
Normally, preparation of the ECU code for bypassing is carried out only once by the ECU supplier. With service-based bypassing as supported by dSPACE, more than 100 functions in the ECU code can be prepared for bypassing by means of service calls (bypass hooks). Afterwards, there is no need to modify the ECU code again. Service calls in the ECU code can be described in an ECU description file (A2L file) and evaluated by the RTI Bypass Blockset, so you can flexibly select the functions to be bypassed and the ECU variables to be read and written in the modeling environment. Moreover, service-based bypassing is a generic solution which can be applied independently of the ECU interface, the processor type, and coding methods. dSPACE provides ECU services for DPMEM PODs and the DCI-GSI1 as well as for XCP, the Universal Measurement and Calibration Protocol. The services can be used for measurement, calibration, ECU flash programming, and bypassing. They are available in C code and have to be compiled and linked to the ECU code only once. In addition, dSPACE provides an example C-code implementation for the synchronization of data using the CCP download mechanism in connection with bypassing via an existing CCP service in the ECU.

Dual-Port Memory (DPMEM)
A dual-port memory allows read and write access by two systems (processors) independently of one another, so that they can exchange data. Each of the two ports has its own address, control, and data channels. DPMEMs are used in bypassing to transmit data between the ECU and the prototyping system via their shared memory with the smallest possible latencies.

Plug-on Device (POD)
Plug-on devices are additional components mounted on an ECU to connect it to the prototyping hardware (for example, the DS4121 ECU Interface Board). PODs contain a dual-port memory (DPMEM) which is connected directly to the address and data bus of the microcontroller, and signal-conditioning tailored to the specific ECU.

Universal Measurement and Calibration Protocol (XCP)
The XCP protocol is the successor to the CAN Calibration Protocol CCP and standardized by ASAM (www.asam.net). XCP technology is independent of the transport layer, so XCP can be used with different physical layers such as CAN, FlexRay or Ethernet. dSPACE provides an XCP service implementation that is fully compatible with the ASAM standard for various processor platforms. Apart from basic features such as measurement, calibration, page switching, and ECU flash programming, the dSPACE XCP implementation also supports function bypassing.
**Function Bypassing Without Sampling Step Delay**

Implementation example (1) of function bypassing using XCP on CAN.

**Function Bypassing With Sampling Step Delay**

Implementation example (2) of function bypassing using XCP on CAN.
Early Focus on the Production ECU

Streamlining the Process
Developing and optimizing new control functions with dSPACE prototyping systems speeds up the whole ECU development process by giving you early feedback if a control function is ready for or worth taking into production. To streamline the process from prototyping to production, dSPACE prototyping systems allow you to think ahead and take production aspects into account while still at an early stage of development.

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\(^1\) are ideal for an early focus on the production code. Another option is the dedicated license-free TargetLink® Blockset Stand-Alone (p. 227), which offers RCP users a subset of Simulink blocks that are highly suitable for production code generation without implementation data having to be entered at the prototyping phase. If you follow guidelines and use the TargetLink Blockset, you can use the same models for rapid control prototyping and the generation of highly efficient production code.

Validation of Production Code on dSPACE Prototyping Systems
In some cases it is useful to verify the production code on the prototyping system. For that purpose, TargetLink’s Stand-Alone Model Manager can be used to generate a stand-alone S-function for Simulink that runs the TargetLink generated production code in a non-TargetLink environment. From there it is possible to generate a real-time application using Real-Time Interface (RTI) (p. 126) from dSPACE even without having TargetLink installed.

\(^1\) Please contact dSPACE for more information.
Rapid Control Prototyping and ECU Calibration

More and more often, control engineers are considering production issues when they work on rapid control prototyping projects. A well-established approach for this is to include modelling style guides during control design and to define distinct labels for measurement and calibration variables in terms of Simulink® objects in the MATLAB® workspace. This way you can use the same controller models and the same labels throughout the development phases.

With ControlDesk® Next Generation (p.160) for prototyping and ECU calibration you can easily exchange and compare calibration and measurement data from the different development stages. Data sets in ControlDesk can also be fed back into the MATLAB/Simulink environment. In addition, the same instrument panels (layouts) can be used without modification for prototyping and ECU calibration.
AUTOSAR in Rapid Control Prototyping

Rapid control prototyping Benefits AUTOSAR Development
The AUTOSAR standard allows a high level of exchangeability and reusability and provides a way to connect software components from different suppliers. Rapid control prototyping systems enable you to quickly verify how AUTOSAR software components (SWCs) behave and interact with the real plant in different development phases. A rapid control prototyping system does not require a complex, time-consuming configuration of AUTOSAR basic software and gives you much more flexibility than a later implementation on a production ECU.

Specific Use Cases
Newly created AUTOSAR SWCs benefit from being verified before they are implemented on a production ECU. With rapid control prototyping, behavior differences between initial specification and the production implementation can be detected easily. The differences usually result from the specification being misunderstood or are simply side effects of the production implementation. When new control algorithms are developed in Simulink, some parts of the control strategy might already exist as AUTOSAR SWCs. These parts can be reused and combined with the new control algorithm in a single working environment without having to be recreated. This means that a proof of concept of the overall control strategy can be carried out in an early stage.

Toolchain and Workflow
dSPACE offers a broad portfolio of tools covering different aspects of a stepwise AUTOSAR development process. SystemDesk® (p. 112) covers the architectural design of AUTOSAR ECUs and networks, while the production code generator TargetLink (p. 218) can generate AUTOSAR SWCs from TargetLink models. AUTOSAR SWCs originating from these tools as well as from other sources (e.g., native AUTOSAR-compliant C code) can be embedded into the familiar Simulink environment using the RTI AUTOSAR Package (p. 150). This lets you connect AUTOSAR SWCs to other Simulink blocks and make use of the comprehensive dSPACE RTI implementation software (p. 126). The resulting models can be verified via PC-based simulation, and also implemented and executed on the dSPACE real-time hardware (p. 286). The ability to use rapid control prototyping at different stages of the development process increases reliability and enables fast design iterations.
System Components

Implementation Software

Real-Time Interface

Once you have created your function models, you can automatically implement them on the prototyping hardware with the help of Real-Time Interface (RTI) (p. 126) – without any programming effort at all. You can even configure complex I/O interfaces directly in the block diagram. And you can immediately try out new ideas by simply inserting them in your model design – all without writing a single line of code. Several RTI extensions are available, e.g., for connecting dSPACE systems to CAN, LIN, FlexRay, and Ethernet networks, for bypassing and for AUTOSAR applications, and for integrating FPGA models in dSPACE systems. It is also no problem to implement C-coded models.

Implementation Software (p.126)
Real-Time Hardware

Typically, prototyping hardware is several times more powerful with regard to processor performance and RAM/ROM resources than the later production ECU itself, to keep you free of hardware restrictions in the design phase. The hardware of dSPACE prototyping systems consists of high-performance processors which can calculate your controller models in microseconds. Connection to the outside world is established by a broad range of I/O interfaces.

Single-Board Hardware
With single-board hardware (p. 288), you have everything on one card. The boards contain a comprehensive selection of I/O interfaces as well as a powerful real-time processor. Our single-board hardware can run in PCs (DS1104) or expansion boxes (DS1103).

Modular Hardware
Modular hardware (p. 304) gives you optimum scalability and flexibility. You can choose from our wide range of boards to put together exactly the prototyping hardware you need for your project. The boards can be inserted in expansion boxes with different numbers of slots for laboratory and in-vehicle use.

MicroAutoBox Hardware
The MicroAutoBox hardware (p. 444) is available in several variants that provide a fixed set of commonly used I/O for various applications. Its special strength is the unique combination of high processing power, comprehensive I/O, and an extremely compact and robust design. It is specifically designed for in-vehicle usage. Besides standard I/O, MicroAutoBox offers variants with interfaces for all major automotive bus systems like CAN, LIN, FlexRay, as well as Ethernet interfaces.

RapidPro Hardware
The RapidPro hardware (p. 462) works as an extension to dSPACE prototyping systems. Software- and hardware-configurable off-the-shelf signal conditioning and power stage modules can be mounted on the RapidPro units to set up individual systems. This flexible concept allows components to be reconfigured and extended in later projects or if requirements change, with a minimum of effort. For several standard applications dSPACE offers predefined configurations of RapidPro hardware.
Test and Experiment Software

ControlDesk® Next Generation
ControlDesk Next Generation (p. 160) is a comprehensive experiment environment based on instruments that lets you intuitively manage, control, and automate your experiments. It is ideally suited to instrumenting your controller model for rapid control prototyping. It gives you access to your model's parameters and signals during run time. It offers a wide range of instruments such as sliders, gauges, switches, and knobs as well as a look-up tables and plotters for convenient and intuitive handling. The multi-row Variable Array Instrument lets you compactly view several variables (simulation, ECU and bus) all at once.

MLIB/MTRACE
MLIB/MTRACE (p. 216), the MATLAB-dSPACE Interface Library, gives you access to the prototyping hardware from within MATLAB.

Engineering Services for Function Prototyping

dSPACE offers numerous services to assist you with your function prototyping activities. Whatever you would like to do with dSPACE prototyping systems, you can be sure that we are ready to support you with:

- Consulting and engineering for special bypassing, signal conditioning, and power stage issues
- Engineering for RCP tool introduction and process adaptation (application scenarios, consulting, pilot projects, specific training and coaching)
- Customer-specific hardware development (e.g., interface extensions, hardware modification) and integration of third-party products
- dSPACE Engineering Services (p. 106)
Use Case:  
Development of Electric Motor Control Strategies

Today, 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 call for highly flexible development systems. The scope, flexibility, and performance of the I/O interfaces frequently determine whether they can support the desired selection of three-phase electric motors and encoders.

Development Environment

The ACMC solution for MicroAutoBox II\(^1\) (p. 444) or AutoBox (p. 436) is a complete, in-vehicle solution for the universal control of three-phase electric drives. It consists of a modular AutoBox with a processor board (p. 304), and an FPGA-based I/O board (DS5202, p. 404) or a MicroAutoBox II (Variant 1401/1511/1512 with the DS1553 ACMC piggy back module\(^1\)). The FPGA-based I/O provides all the typical I/O interfaces for connecting sensors and power stages supporting BLDC, synchronous, and asynchronous motors. In addition, the performance of the FPGA is also used to generate the 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 of control algorithms for BLDC motors and synchronous motors.

---

\(^1\) As of December 2010, the AC Motor Control Solution for MicroAutoBox II is not yet available. Please contact dSPACE for information on availability.
Use Case: In-Vehicle Function Development and Test of Map-Based Driver Assistance Applications

Modern, map-based driver assistance systems are one way to solve the challenges posed by the road traffic of tomorrow. Efficient development of these systems requires a tool chain featuring flexible and configurable access to map data. During control design the prototype of the Advanced Driver Assistance System (ADAS) has to be integrated into the vehicle like a real ECU, and communicate with the vehicle’s bus systems (such as vehicle CAN). The digital map data and GPS coordinates must be available to the prototyping system as well.

Development Environment

MicroAutoBox and AutoBox are compact prototyping solutions for executing computing-intensive embedded software and integrating it in a vehicle’s electrical system. They can be configured with all the interfaces necessary for map-based driver assistance systems. In a typical case, the sensors of the vehicle provide the GPS coordinates. If required, a dedicated sensor box with high-quality sensors for position finding (GPS antennas, gyroscopes) can also be used. The ADAS Research Platform (ADAS RP) from NAVTEQ is a development environment for map-based driver assistance systems that runs on Windows® PCs. ADAS RP provides basic functions such as the visualization of maps, route planning and the indication of the vehicle position with respect to the digital map. It also serves as an electronic horizon provider and sends information of the route ahead including associated attributes like slopes and curvatures, for example via a network service. ADAS RP evaluates the vehicle’s position data and provides the electronic horizon accordingly.

Roles and Signals

Position data from in-vehicle sensors or a dedicated sensor box (e.g. from NAVTEQ) is acquired and transferred to ADAS RP. This matches the vehicle’s position to the digital map and broadcasts the electronic horizon. The dSPACE system, which is connected via Ethernet, receives the data and decodes it with the ADAS RP Blockset. The electronic horizon is then available to the ADAS controller to be developed on the MicroAutoBox or AutoBox.

1) For 2011, dSPACE plans to offer a MicroAutoBox II with an integrated Embedded PC. The ADAS controller model and ADAS RP will then both run on the same platform. For more information on the release of the Embedded PC extension, please contact dSPACE.
Use Case:
Engine Control Development

Saving fuel and reducing CO₂ require continuous research on new operating processes for internal combustion engines. New control strategies need to be developed on the testbench and verified by test drives. Efficient development of these control strategies requires development systems that can communicate with the vehicle’s bus systems and provide the flexibility to connect various sensors and actuators needed to control the combustion process.

Development Environment
The combination of MicroAutoBox or AutoBox and the RapidPro system allows comfortable advanced engine control development on the testbench and in the vehicle. The prototyping system MicroAutoBox can execute processing intensive embedded software algorithms, directly generated from Simulink® models, and provides the necessary automotive bus interfaces to connect to other electronic control units. MicroAutoBox also supports cylinder pressure based control development with its fast analog input channels. RapidPro is a scalable and modular system which adds of-the-shelf signal conditioning and power stage capabilities to the prototyping system. RapidPro provides engine specific I/O interfaces such as modules for lambda and knock sensors or power stage modules for valves, relays and electric motors used in ancillary units.

Predefined RapidPro configuration
With the predefined engine control configuration, a complete setup of I/O interfaces is provided for operating typical 4 or 6 cylinder engines. With this basic set of interfaces, current development trends such as downsizing and direct injection are well covered. The flexibility of RapidPro still allows it to change the predefined configuration when new requirements emerge.
Use Case: 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 stages hardware.

Development Environment

For chassis control development, dSPACE offers predefined RapidPro configurations (p. 466) in addition to a dSPACE prototyping system (MicroAutoBox, p. 444). This bundle of hardware easily gets you going on developing chassis control strategies because common sensors and actuators in chassis control applications can easily be adapted to your prototyping system. In this scenario, the RapidPro hardware works as an extension to the dSPACE MicroAutoBox. The RapidPro SC Unit and various signal conditioning modules are used to provide the link to typical sensors (e.g. pressure, acceleration, yaw rate) used in chassis control applications. The RapidPro power unit provides dedicated power stages for driving actuators such as DC electric motors or valves.

You can perform interactive diagnostics with feedback to the model using the Real-Time Interface (RTI) RapidPro Diagnostics Blockset for Simulink. Using dSPACE’s ConfigurationDesk® software you can easily configure the RapidPro system and monitor the hardware states during operation.