dSPACE Internal Bypassing Service

Implementation

For dSPACE Internal Bypassing Service 1.3

Release 2021-A – May 2021

dSPACE

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	info@dspace.de
Web:	http://www.dspace.com

How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: http://www.dspace.com/go/locations
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.
  Tel.: +49 5251 1638-941 or e-mail: support@dspace.de

You can also use the support request form: http://www.dspace.com/go/supportrequest. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the relevant dSPACE License ID or the serial number of the CmContainer in your support request.

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit http://www.dspace.com/go/patches for software updates and patches.

Important Notice

Contents

About This Reference

Contents

This document gives you information on how to implement the dSPACE Internal Bypassing Service in your ECU code. It also provides information on the available functions and macros needed to specify and configure the service.

Note

The *dSPACE Internal Bypassing Service* is part of the installation of ECU Interface Software.

After you install and decrypt ECU Interface Software, you will find the `dSPACEInternalBypassingService_<version>.exe` file in the `%ProgramData%\dSPACE\<InstallationGUID>\dsECU\Services` folder. Run it to install the service and its documentation in a folder of your choice.

You can access the `%ProgramData%\dSPACE\<InstallationGUID>` folder via a shortcut in the Windows Start menu below dSPACE RCP and HIL <version>.

Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
⚠ DANGER	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
⚠ WARNING	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
⚠ CAUTION	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a hazard that, if not avoided, could result in property damage.
Note	Indicates important information that you should take into account to avoid malfunctions.

Symbol	Description
Tip	Indicates tips that can make your work easier.
	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

%name% Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Special folders

Some software products use the following special folders:

Common Program Data folder A standard folder for application-specific configuration data that is used by all users.

```
%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>
```

or

```
%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>
```

Documents folder A standard folder for user-specific documents.

```
%USERPROFILE%\Documents\dSPACE\<ProductName>\
<VersionNumber>
```

Local Program Data folder A standard folder for application-specific configuration data that is used by the current, non-roaming user.

```
%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\
<ProductName>
```

Accessing dSPACE Help and PDF Files

After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the icon in dSPACE Help. The PDF opens on the first page.

License Agreement for the dSPACE Internal Bypassing Service

Introduction

If you want to work with the dSPACE Internal Bypassing Service, you have to accept the License Agreement first.

License Agreement

License agreement

Note

IMPORTANT – USE OF THIS SERVICE IS SUBJECT TO LICENSE RESTRICTIONS READ THIS LICENSE AGREEMENT CAREFULLY BEFORE USING THE SERVICE

This license is a legal Agreement between you, the end user, either individually or as an authorized representative of the company acquiring the license, and dSPACE GmbH acting directly or through its local dSPACE companies or authorized distributors (collectively "dSPACE"), concerning the use of the C code containing the dSPACE Internal Bypassing Service (hereinafter referred to as "the Service") together with any other materials which are provided for use in connection with the Service, including without limitation the executable for installation of the Service, any associated user manual and internal documentation (hereinafter collectively referred to as "the Program"). *BY IMPLEMENTING THE EXECUTABLE AND INSTALLING THE PROGRAM, YOU AGREE TO COMPLY WITH THE FOLLOWING TERMS AND RESTRICTIONS. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT INSTALL OR USE THE PROGRAM AND PROMPTLY RETURN IT TO THE PLACE WHERE YOU OBTAINED IT, OR DELETE THE PROGRAM IF YOU RECEIVED IT ELECTRONICALLY.*

1. Grant of License

Unless explicitly agreed otherwise, dSPACE grants you a nonexclusive license to use the Program and execute the Service as described in the respective product

description or documentation for the sole purpose of product development involving dSPACE tools.

2. Restrictions of Use

You may not market, distribute or transfer copies of the Program, in whole or in part, to third parties (including any subsidiary, affiliate or company under common control with you) or transfer the Program (directly or indirectly) via Internet or network applications (such as Citrix, Microsoft Remote Desktop or other terminal servers) or grant third parties any access to the Program by any means. You may not rent, lease or loan the Program.

These restrictions do not prevent you from providing compiled object code versions of the Service as part of your own ECU code to third parties, subject to the condition that:

- This takes place in the course of a project where (amongst others) dSPACE tools are used, *and*
- The code is used for the sole purpose of product development and not for use in any end product or production.

The recipient of your respective ECU code needs to be instructed accordingly and shall undertake to comply with these restrictions and to agree to the Limitation of Liability according to Clause 4 hereunder. dSPACE reserves the right to ask for written confirmation that appropriate instructions have been issued.

Upon request and at the sole discretion of dSPACE, you may be granted permission to provide the Service itself, in whole or in part, to third parties as part of your own ECU source code, subject to the conditions stated above. To be valid, such permission needs to be granted in writing by dSPACE.

For the avoidance of doubt, in any case any transfer of or granting of access to parts of the Program other than the Service itself is explicitly prohibited.

3. Confidentiality

dSPACE considers the Program to contain valuable intellectual property of dSPACE, the unauthorized disclosure of which could cause irreparable harm to dSPACE. You agree to use reasonable efforts not to disclose the Program to any third parties (including any subsidiary, affiliate or company under common control with you) and not to use the Program other than for the purposes authorized by dSPACE.

4. Limitation of Liability

The Program was designed and tested solely for use in research and product development and is supplied to you by dSPACE exclusively for this purpose. It must be put into operation exclusively by suitably trained and expert operating personnel under strict compliance with the safety measures described in the software documentation. Any use of the Program or compiled object code versions of the Service for purposes and under conditions other than the above, including but not only any use in end products, constitutes inappropriate use.

Any liability by dSPACE under mandatory law, including but not restricted to product liability law, for damages of any kind that may be caused by using the Program or compiled object code versions of the Service in areas other than

product development shall be limited, even to the point of total exclusion, as the case may be. In the event of claims by third parties against dSPACE that are due to such inappropriate use of the Program or of compiled object code versions of the Service by you or with your permission, you agree to indemnify dSPACE against all such claims.

In addition, the regulations on liability according to the General Terms and Conditions of dSPACE as attached to any dSPACE offer apply accordingly. A copy of the General Terms and Conditions can also be obtained at info@dspace.de.

5. Miscellaneous

Any amendments or additions to this Agreement must be made in writing and must be expressly marked as such. This also applies to this written form requirement.

In the event that any of the above terms is or becomes invalid, the remaining terms shall continue in full force and effect.

Any failure to enforce, or any waiver of, any right under this Agreement by dSPACE shall not be construed as a waiver of future rights.

The legal regulations shall apply in addition to the terms of this Agreement, except in cases where they conflict with said terms. This Agreement shall be governed by the laws of the Federal Republic of Germany, excluding the UN Convention on Contracts for the International Sale of Goods (CISG).

Paderborn, Germany, is agreed as the exclusive place of jurisdiction for all disputes arising from or in connection with this Agreement, unless a different place of jurisdiction is mandatory on the basis of legal requirements.

Introduction to the dSPACE Internal Bypassing Service Implementation

Introduction

The dSPACE Internal Bypassing Service is used to control communication between an ECU application and one or more post-integrated functions implementing a specific internal bypass functionality.

Where to go from here

Information in this section

Basics on the dSPACE Internal Bypassing Service

Introduction

The dSPACE Internal Bypassing Service is used to control communication between an ECU application and one or more post-integrated functions implementing a specific internal bypass functionality.

Access to ECU application and resources

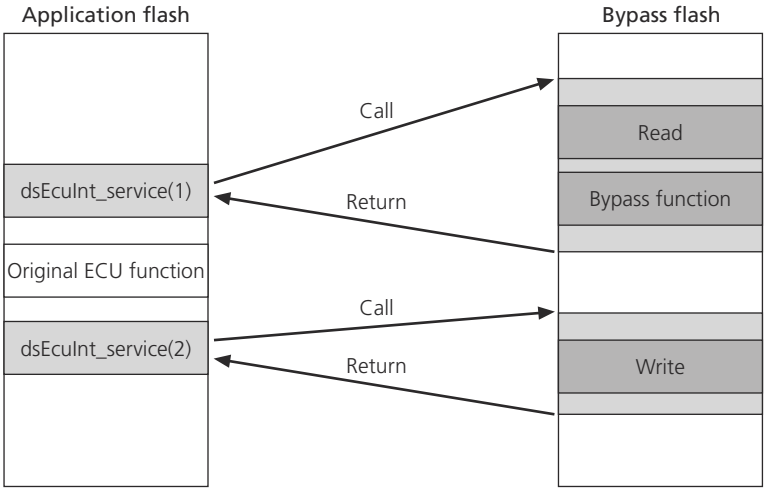
During the development of an ECU, it is often necessary to access the application running on it. For example, values must be modified, and algorithms used to calculate values must be replaced or extended.

The dSPACE Internal Bypassing Service provides interfaces to extend the ECU application by inserting additional functionality by means of internal bypassing.

Internal bypassing

Using the dSPACE Internal Bypassing Service together with the RTI Bypass Blockset allows you to perform internal bypassing (also called on-target prototyping). Functions of the original ECU application are bypassed, and the new ECU functions are executed on the target ECU by additional internal-bypass-specific ECU applications. Specific ECU memory areas are used for internal bypassing.

Typical scenario The following illustration shows a typical scenario for service-based internal bypassing:



Before the original ECU function is calculated, the ECU application is interrupted by a service call. This service call triggers the following actions: The input parameters relevant for internal bypassing are sampled and stored in a buffer located in the bypass RAM, the input parameters are read from the buffer, the bypass function is calculated, and the calculated output values are saved to the buffer. A second service call, placed after the ECU function to be bypassed, triggers the replacement of the ECU's results by the results of the bypass function.

Extended bypassing mechanisms To ensure operational reliability, the following extended bypassing mechanisms can be used in connection with internal bypassing:

- Double buffer mechanism
- Failure checking mechanism

For further information, refer to Extended Bypassing Mechanisms for On-Target Bypassing (RTI Bypass Blockset Reference 🕮).

Related topics

Basics

dSPACE Internal Bypassing Service Files....13
Elementary Data Types....14

dSPACE Internal Bypassing Service Files

Introduction

The functions and configuration options used to implement the dSPACE Internal Bypassing Service are contained in different header and code files which you must compile and link to the ECU code.

File structure

The following tables show the header and code files for the dSPACE Internal Bypassing Service.

File Type	File Name	Description
Common files [1]	dsECUIntcustom.h	Configuration file for custom commands
	dsECUIntSvc.c	Code file of the dSPACE Internal Bypassing Service
	dsECUIntSvc.h	Include file for bypassing
Custom files [2]	dsECUIntCfg.h	Configuration file for the complete service
	dsECUIntcustom.c	Initial definitions of service configuration structures used by the dSPACE Internal Bypassing Service and the external tool.[3]

[1] The common files are fixed files containing the dSPACE Internal Bypassing Service and therefore must not be altered.
[2] The custom files are used for custom configuration and adaptation.
[3] You can change the memory locations of the service configuration structures in the ECU memory at your own risk. Refer to Specifying the Start Addresses of the Service Configurations in the ECU Memory on page 27.

Note

You must compile and link the files to your ECU application. The ECU application only has to include the `dsECUIntSvc.h` header file. You do not have to make any configuration settings in the file. Optionally you can make compiler-specific adaptations.

Related topics

Basics

Elementary Data Types....14

Elementary Data Types

Data types

The `dsECUIntCfg.h` file defines the data types for the dSPACE Internal Bypassing Service. The default values listed below match most ECU processors:

```
typedef unsigned char       DSECUINT_UInt8;
typedef signed char         DSECUINT_Int8;
typedef unsigned short      DSECUINT_UInt16;
typedef signed short        DSECUINT_Int16;
typedef unsigned long       DSECUINT_UInt32;
typedef signed long         DSECUINT_Int32;
typedef float               DSECUINT_Flt32;
typedef double              DSECUINT_Flt64;
```

Include files

`dsECUIntCfg.h`

Related topics

Basics

dSPACE Internal Bypassing Service Files 13

Integration of the dSPACE Internal Bypassing Service

Introduction

The ECU application must be instrumented by the dSPACE Internal Bypassing Service.

Where to go from here

Information in this section

dSPACE Internal Bypassing Service API

Introduction

The dSPACE Internal Bypassing Service provides API functions for initializing the service and performing command processing.

Service initialization

The service initialization is performed by the `dsEcuInt_init` initialization function, which must always be integrated into the ECU code. The function must be called before any other dSPACE Internal Bypassing Service function. A typical location in the ECU code is where other initializations of the ECU are performed.

Foreground service

The foreground service of the dSPACE Internal Bypassing Service is called to execute functions provided by the internal bypass functionality of the RTI Bypass Blockset. These bypass functions perform one or more of the following actions, depending on the bypass model:

- Sampling data (copying data from the ECU application RAM to ECU RAM areas available for internal bypassing)
- Calculating new data by using bypass functions
- Stimulating data (copying data from ECU RAM areas reserved for internal bypassing back to the ECU RAM used by the ECU application)

The foreground service is performed by the `dsEcuInt_service` function, which must be integrated into the ECU code. The function call can be placed in any function, for example, so that time-synchronous or crankshaft-synchronous bypassing can be performed. Two scenarios are possible for bypassing:

- Bypassing with two function calls

  One function call is placed before or at the beginning of the function to be bypassed, and one function call is placed at the end of or after the function to be bypassed. The first function call triggers a bypass function for sampling data and calculating the bypass results. The second function call triggers a bypass function for overwriting the result of the ECU function with the bypass results.
- Bypassing with one function call

  The function call is placed at the end of or after the function to be bypassed, or, if the function results are not used again in the current ECU task, at the end of the task. The function call triggers a bypass function for sampling data, calculating the bypass results and overwriting the result of the ECU function with the bypass results.

Related topics

Basics

Integrating the dSPACE Internal Bypassing Service in the ECU Code..... 16

Integrating the dSPACE Internal Bypassing Service in the ECU Code

Introduction

The ECU application must be instrumented by dSPACE Internal Bypassing Service API functions. Different functions must be called according to the purpose the service is to be used for.

General integration

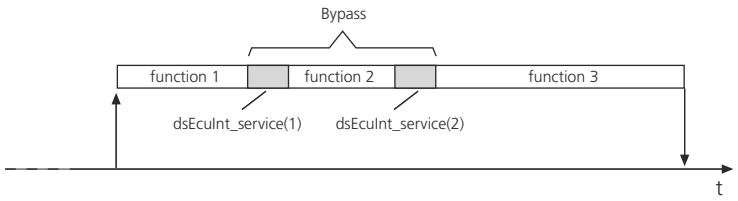
The initialization function must always be integrated in the ECU code. The `dsEcuInt_init` initialization function is called during start-up prior to any ECU task.

Integration for internal bypassing

The ECU application can consist of several ECU tasks which are called at fixed time intervals (time rasters) or synchronously to specific ECU events (for example, crankshaft-synchronously).

Bypassing with two function calls The `dsEcuInt_service` function is called to trigger the bypass functionality at the beginning and end of the ECU function or task to be bypassed.

Example: To perform bypassing on an ECU with the dSPACE Internal Bypassing Service using two foreground service calls, you have to instrument an ECU task as shown by the example in the illustration below.



The ECU task consists of three functions, and function 2 is to be bypassed. The `dsEcuInt_service` function is called at the beginning of function 2 (to read the function arguments and calculate the new data) and at the end of function 2 (to write the calculated data back to the ECU application).

Bypassing with one function call Alternatively, bypassing can be performed by only one `dsEcuInt_service` function call. It is placed at the end of the function to be bypassed, or, if the function results are not used in the current task, at the end of the ECU task (see dSPACE Internal Bypassing Service API on page 15).

Related topics

Basics

dSPACE Internal Bypassing Service API 15

dSPACE Internal Bypassing Service API Functions

Introduction

Some functions of the dSPACE Internal Bypassing Service API must be called by the ECU application to initialize the service or to trigger actions provided by the internal bypass tool.

Besides the API functions described in the following topics, there are further functions and options for advanced service configuration. For information, contact dSPACE Support.

Where to go from here

Information in this section

dsEcuInt_init

Syntax

```
void dsEcuInt_init (void)
```

Include file

`dsEcuIntSvc.h`

Purpose

To initialize the dSPACE Internal Bypassing Service.

Description	The `dsEcuInt_init` function initializes the data structure containing information on the service configuration structure used by the dSPACE Internal Bypassing Service. The data structure is located in the ECU RAM memory. The `dsEcuInt_init` function must be called prior to any other function of the dSPACE Internal Bypassing Service. A typical location in the ECU code is where other initializations of the ECU are performed. For further information, refer to Integrating the dSPACE Internal Bypassing Service in the ECU Code on page 16. The internal bypass code is also initialized via the `dsEcuInt_init` function, too. You can define additional functionality to be performed during initialization from within the internal bypassing tool (RTI Bypass Blockset) by declaring and using a service instance assigned to the service ID 0. For example, you can use an RTI Bypass Interrupt block, for which a service instance associated with service ID 0 is specified, to call a subsystem that performs some post-initialization actions.
Parameters	None
Return value	None
Example	`dsEcuInt_init();`
Related topics	References dsEcuInt_service....20 dSPACE Internal Bypassing Service API Functions....19

dsEcuInt_service

Syntax	`DSECUINT_Int16 dsEcuInt_service(DSECUINT_UInt16 SvcId)`
Include file	`dsEcuIntSvc.h`
Purpose	To start the dSPACE Internal Bypassing Service foreground service.
Description	The function calls a bypass hook which is used to start a dedicated action according to the related bypass model of the internal bypassing tool (RTI Bypass Blockset with internal bypass option).

For bypassing purposes, one or two function calls are necessary, depending on the bypass scenario:

- Bypassing with two function calls

  One function call is needed at the beginning of the function or task to be bypassed (to read the arguments of the functions to be bypassed and typically execute the bypass functions). The other function call is needed at the end of the function or task to be bypassed (to write the results calculated by the internal-bypass-specific applications back to the ECU application).
- Bypassing with one function call

  The function call is placed at the end of the function or task to be bypassed (to read the arguments of the functions to be bypassed, to execute the bypass functions and to write the function results back to the ECU application).

For further information, refer to Integrating the dSPACE Internal Bypassing Service in the ECU Code on page 16.

Parameters

SvcId Specifies the service ID in the range 1 … 65535.

Note

The service ID 0 is reserved for initializing the external tool.

Return value

The function returns an error code.

The following error definitions can be used as flags, that is, they can be combined with a binary OR operation:

Predefined Symbol	Meaning
DSECUINT_SVC_ACTIVE	This service is enabled. Data has been copied.
DSECUINT_SVC_INACTIVE	All service calls are disabled.

Example

```
dsEcuInt_service(5);
```

Related topics

References

dsEcuInt_init 19
dSPACE Internal Bypassing Service API Functions 19

dSPACE Internal Bypassing Service Configuration Options

Introduction

The features of the dSPACE Internal Bypassing Service can be configured.

Where to go from here

Information in this section

Basics on Configuring Features of the dSPACE Internal Bypassing Service

Feature configuration

The features of the dSPACE Internal Bypassing Service can be configured. Configuration is done by means of conditional compilation with preprocessor instructions (#define) in the dSPACE Internal Bypassing Service configuration file `dsECUIntCfg.h`.

External tools	The dSPACE Internal Bypassing Service supports one external tool at a time.
Related topics	Basics dSPACE Internal Bypassing Service Files 13

DSECUINT_SERVICE_REMOVED

Syntax	`#define DSECUINT_SERVICE_REMOVED`
Purpose	To enable or disable completely the dSPACE Internal Bypassing Service.
Description	`DSECUINT_SERVICE_REMOVED` disables the compilation of the dSPACE Internal Bypassing Service. A common way to disable the dSPACE Internal Bypassing Service is to pass this option to the compiler as a parameter.
Parameters	None
Example	`#define DSECUINT_SERVICE_REMOVED`
Related topics	Basics Basics on Configuring Features of the dSPACE Internal Bypassing Service 23

DSECUINT_SVC_RESOLVE_SERVICE_ID

Syntax	`#define DSECUINT_SVC_RESOLVE_SERVICE_ID`
Purpose	To specify the algorithm used to search for the service ID in a list.

Description

`DSECUINT_SVC_RESOLVE_SERVICE_ID` chooses between three different search algorithms to find the service ID in a list of services. The search function is used in the `dsEcuInt_service` function. The first two methods are linear, starting at the end or the beginning of the list. The last method is a binary search. It is faster, but the function code is more complex.

Parameters

The following parameters are available:

Predefined Symbol	Meaning
0	Linear search starting at the end of the list.
1	Linear search starting at the beginning of the list.
2	Binary search.

Example

```
#define DSECUINT_SVC_RESOLVE_SERVICE_ID (2)
```

Related topics

Basics

Basics on Configuring Features of the dSPACE Internal Bypassing Service..... 23

DSECUINT_CONST

Syntax

```
#define DSECUINT_CONST
```

Purpose

To place data in the ROM or RAM.

Description

`DSECUINT_CONST` is used to place service structures in the ROM or RAM. For most applications, the default define value 'const' can be used and the structures are placed in the ROM. If no value is specified for the define, the service structures are placed in the RAM.

Parameters

... Define that is set if the structures are placed in the RAM.

Example

```
#define DSECUINT_CONST const
```

Related topics

Basics

Basics on Configuring Features of the dSPACE Internal Bypassing Service..... 23

DSECUINT_SVC_PID

Syntax

```
#define DSECUINT_SVC_PID
```

Purpose

To enable or disable data abstraction for position-independent data.

Description

`DSECUINT_SVC_PID` is useful for specific implementations of the service where the code is compiled to be relocated (position-independent code) and the global variables must be located in a global offset table (GOT).

For standard implementations, the default value DSECUINT_DISABLED must not be changed.

Parameters

The following parameters are available:

Predefined Symbol	Meaning
DSECUINT_ENABLED	Enables data abstraction for position-independent code.
DSECUINT_DISABLED	Disables data abstraction for position-independent code.

Example

```
#define DSECUINT_SVC_PID DSECUINT_DISABLED
```

Related topics

Basics

Basics on Configuring Features of the dSPACE Internal Bypassing Service..... 23

Configuring the ECU Memory Locations of the Service Configurations

Introduction

The ECU memory locations of the service configurations used by the dSPACE Internal Bypassing Service and the external tool are defined in the `dsECUIntCfg.h` and `dsECUIntcustom.c` files.

Specifying the Start Addresses of the Service Configurations in the ECU Memory

Introduction

The ECU memory locations of the service configurations used by the dSPACE Internal Bypassing Service and the external tool are defined in the `dsECUIntCfg.h` and `dsECUIntcustom.c` files. You can change the locations of the service configurations in the ECU memory by adding compiler-specific section or location directives modifying the start addresses in the `dsECUIntCfg.h` and `dsECUIntcustom.c` files.

Modifying the memory locations of the service configurations

The `dsECUIntCfg.h` and `dsECUIntcustom.c` files provide the initial definitions and the external declarations of the service configuration structures used by the dSPACE Internal Bypassing Service and the external tool. You can change the locations of the service configurations in the ECU memory by specifying new start addresses for them in the `dsECUIntCfg.h` and `dsECUIntcustom.c` files. Use compiler-specific statements (e.g., pragma sections) for this.

Note

It is your responsibility to ensure that the service configuration structures are moved to permissible and appropriate memory locations.

Below is an example of changing the start addresses in the `dsECUIntCfg.h` and `dsECUIntcustom.c` files.

Note

The start addresses of the service configuration structures specified in the dSPACE Internal Bypassing Service and in the associated IF_DATA element of the ECU's A2L file must match. Ensure that the address values in the IF_DATA element are updated to the location addresses of the service configuration structures when you rebuild your application or, if you work with an explicit address configuration, after you modified the configured address.

Example

The following excerpt from the `dsECUIntcustom.c` file shows the modification of the start addresses of the service configurations using the GNU GCC compiler.

```
 ...
/* custom configuration structure definitions *********************************/
/* Start of custom external tool configuration definition */
#pragma section .svccfgextsect
volatile DSECUINT_CONST DSECUINT_Service_Config_External_Tool DsEcuIntSvcExtToolCfg =
  DSECUINT_SVC_CONFIG_DEFAULT_EXTERNAL_TOOL;
#pragma section
/* End of custom external tool configuration definition */

/* Start of custom internal module configuration definition */
#pragma section .svccfgintsect
volatile DSECUINT_Service_Config_Internal_Service DsEcuIntSvcIntModuleCfg = DSECUINT_SVC_CONFIG_DEFAULT_INTERNAL_MODULE;
#pragma section
/* End of internal module configuration definition */
 ...
```

The following excerpt from the `dsECUIntCfg.h` file shows the corresponding modifications for the external declarations of the service configurations using the GNU GCC compiler.

```
 ...
/* custom configuration structure definitions *********************************/
/* Start of custom external tool configuration definition */
#pragma section .svccfgextsect
extern volatile DSECUINT_CONST DSECUINT_Service_Config_External_Tool DsEcuIntSvcExtToolCfg;
#pragma section
/* End of custom external tool configuration definition */

/* Start of custom internal module configuration definition */
#pragma section .svccfgintsect
extern volatile DSECUINT_Service_Config_Internal_Service DsEcuIntSvcIntModuleCfg;
#pragma section
/* End of internal module configuration definition */
 ...
```

The sections `svccfgextsect` and `svccfgintsect` must be defined in the compiler-specific linker command file.

Related topics

Basics

dSPACE Internal Bypassing Service Files..... 13

dSPACE Internal Bypassing Service Interface Description

Introduction

To communicate with an ECU with integrated dSPACE Internal Bypassing Service, the interface used for bypassing must be described.

Interface Description

Introduction

ASAM MCD-2 MC (A2L) files contain a variable description of an ECU. To bypass an ECU using the RTI Bypass Blockset, the ECU's A2L file must contain an IF_DATA element specific to the interface provided by the dSPACE Internal Bypassing Service.

Required interface-specific information

The A2L file must include an `IF_DATA` element specific to the bypass interface used. The `IF_DATA` element contains interface-specific information. This information must comply with a special format, which is described in ASAP2 Meta Language (AML) files. The AML specification of the interface must also be contained in the A2L file.

The `IF_DATA` element describing the dSPACE Internal Bypassing Service implementation must be set to the configuration which results from the ECU microcontroller used, the service configuration, and the addresses the internal bypassing service configuration structures are linked to.

Note

Differences between the service configuration and the interface description might prevent the internal bypass tool from accessing the ECU.

For information on the interface-specific `IF_DATA` element and details on its data format, refer to the Interface Description Data Reference 🕮.

Related topics

References

dSPACE Internal Bypassing Service Configuration Options..... 23

C

D

I

L

S