DS4121 ECU Interface Board

RTLib Reference

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 Document

Content

The DS4121 Real-Time Library (RTLib) provides the C functions and macros you need to program the DS4121 ECU Interface Board.

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.
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.

ECU Interface Unit

Where to go from here

Information in this section

Basics

Where to go from here

Information in this section

DS4121 RTLib Functions

Introduction

RTLib functions provide data-typed, format-dependent read and write access to the DPMEM.

Abbreviations

The function names characterize the supported functionality with the following abbreviations:

Abbreviation	Meaning
ob	access to the *o*n*b*oard DPMEM
ext	access to an *ext*ernal DPMEM
p	*p*acked data
s	*s*cattered data
le	*l*ittle *e*ndian
be	*b*ig *e*ndian
int	*int*eger
fl	*fl*oating point
8, 16, 32	number of bits for a data word

Example

The `ds4121_p_int_write8` function writes *8*-bit *int*eger data in a *p*acked format to the DPMEM.

Error messages

For some functions error messages are defined. These messages are reported by ControlDesk. Each message comprises the function name and the PHS-bus offset

(0xXX) of the DS4121 that generates the message. Furthermore, we will give a short description of the reason in this reference.

Example

ds4121_init(0xXX): Board not found! The DS4121 was not found with the given PHS-bus offset.

Related topics

Basics

Data Layout in the DPMEM

Introduction

The DS4121 has a 16-bit dual-port memory (DPMEM) which provides read and write access to ECU variables for the ECU and the DS4121. The data layout in the DPMEM, i.e., the way ECU variables are stored in, written to, and read from the DPMEM, depends on the ECU microprocessor's architecture.

ECU data format

Depending on the microcontroller's bus width, the DS4121 provides byte-wise and/or word-wise data access to the DPMEM. ECU data values are arranged in packed and/or scattered format in the DPMEM. If the ECU values' bits are scattered, only byte-wise (8-bit) data access to the DPMEM is possible. Packed format implies word-wise (16-bit) data access. Thus, an ECU value might need to be divided into 8-bit or 16-bit segments which are stored at two or more different DPMEM locations. To read an ECU data value from the DPMEM, it might be necessary to read bits from various DPMEM locations and put them together to form a single value. To write a single value to the DPMEM, it might be necessary to divide it into several 8- or 16-bit segments and write the bits byte- or word-wise to several DPMEM locations (see example below).

Data arrangement in the DPMEM

The arrangement of ECU values in the DPMEM depends on the data format and the data bus format used by the ECU. The following factors affect the memory addressing:

- Width of the ECU variable (8-bit, 16-bit or 32-bit data)
- Format of the ECU data (packed or scattered data)
- Byte order (little endian or big endian)

Example

This example shows the arrangement of 8-bit, 16-bit and 32-bit ECU data values in a 16-bit DPMEM according to different ECU data formats:

- Arrangement of 8-bit data values in the DPMEM

Value	Data Format	DPMEM
0x12	Packed, upper halfword	[0] 0x1200
	Packed, lower halfword	[0] 0x0012
	Scattered	[0] 0x0012

- Arrangement of 16-bit data values in the DPMEM

Value	Data Format	Byte Order	DPMEM
0x1234	Packed	Little endian	[0] 0x1234
	Packed	Big endian	[0] 0x1234
	Scattered	Little endian	[0] 0x0012 [1] 0x0034
	Scattered	Big endian	[0] 0x0012 [1] 0x0034

- Arrangement of 32-bit data values in the DPMEM

Value	Data Format	Byte Order	DPMEM
0x12345678	Packed	Little endian	[0] 0x5678 [1] 0x1234
	Packed	Big endian	[0] 0x1234 [1] 0x5678
	Scattered	Little endian	[0] 0x0056 [1] 0x0078 [2] 0x0012 [3] 0x0034
	Scattered	Big endian	[0] 0x0012 [1] 0x0034 [2] 0x0056 [3] 0x0078

Related topics

Basics

DS4121 RTLib Functions.......... 8

System Functions

Introduction

System functions allow you to initialize the board, to check the connection to the ECU and to set the mode of the onboard or external DPMEM.

Where to go from here

Information in this section

ds4121_init

Syntax

```
void ds4121_init(phs_addr_t base);
```

Include file

`ds4121.h`

Purpose

To initialize the DS4121.

Parameters

base	PHS-bus base address of the DS4121 board

Description

The board is initialized as follows:

- The registers of the board are initialized.
- The FIFOs are reset.
- The interrupt inputs are set to edge triggered.

Return value

None

Messages

The following message is defined:

Type	Message	Meaning
Error	ds4121_init(): Invalid PHS-bus address 0x????????	The value of the base parameter is not a valid PHS-bus address. This error may be caused if the PHS-bus connection of the I/O board is missing. Check the connection.
Error	ds4121_init(): (0xXX): Board not found!	The DS4121 board was not found at the specified PHS-bus address.Check if the DSxxxx_n_BASE macro corresponds to the I/O board used.

Related topics

References

DS4121_ECUSETUP_Bx (DS4121 RTI Reference 📖)

ds4121_connection_check

Syntax

```
UInt32 ds4121_connection_check(
      phs_addr_t base,
      UInt32 channel);
```

Include file

`ds4121.h`

Purpose

To check if an ECU is connected to the DS4121.

Description

To check if an ECU is connected to the DS4121 at the given channel.

`ds4121_connection_check` generates a warning message when the connection has been lost. An info message is generated when the connection has been reestablished. Additionally, you can evaluate the return value to get the information.

Parameters

base	PHS-bus base address of the DS4121 board
channel	DS4121 channel number; the possible values are 1 or 2.

Return value

Error code. The following macros are predefined:

Return Value	Meaning
DS4121_CONNECTION_OK	The connection between the DS4121 channel and the ECU has been established.
DS4121_CONNECTION_LOST	The connection between the DS4121 channel and the ECU has been lost. This can be caused by missing or faulty wiring or by an ECU without power supply.

Messages

The following information messages are defined:

Type	Message	Meaning
Info	ds4121_connection_check(0xXX): Connection to ECU channel x established!	The DS4121 channel has established a connection to an ECU.
Warning	ds4121_connection_check(0xXX): No connection to ECU channel x!	The DS4121 channel could not establish a connection to an ECU. Check the physical connection between the DS4121 and the ECU and check the power supply of the ECU.

Related topics

Basics

ECU Interface Unit (DS4121 Features 🕮)

ds4121_fifo_check

Syntax

```
UInt32 ds4121_fifo_check(
     phs_addr_t base,
     UInt32 channel);
```

Include file

`ds4121.h`

Purpose

To check the state of the FIFO of the specified communication channel.

Note

The DS4121 write functions check the FIFO state automatically.

Parameters

base	PHS-bus base address of the DS4121 board
channel	DS4121 channel number; the possible values are 1 or 2.

Return value

Error code. The following macros are predefined:

Return Value	Meaning
DS4121_FIFO_NOT_FULL	The FIFO of the specified channel is not full.
DS4121_FIFO_ALMOST_FULL	There are less than 64 entries free in the FIFO of the specified channel.
DS4121_FIFO_FULL	The FIFO of the specified channel is full.

Related topics

Basics

ECU Interface Unit (DS4121 Features 🕮)

References

ds4121_fifo_reset.......... 14
ds4121_fifo_start.......... 15

ds4121_fifo_reset

Syntax

```
void ds4121_fifo_reset(
      phs_addr_t base,
      UInt32 channel);
```

Include file

`ds4121.h`

Purpose

To reset the FIFO buffer of the specified communication channel.

Parameters

base	PHS-bus base address of the DS4121 board
channel	DS4121 channel number; the possible values are 1 or 2.

Return value

None

Related topics

Basics

References

ds4121_fifo_start

Syntax

```
void ds4121_fifo_start(
      phs_addr_t base,
      UInt32 channel);
```

Include file

`ds4121.h`

Purpose

To release the reset of the FIFO of the specified communication channel after calling `ds4121_fifo_reset`.

Note

For a reset of the FIFO, a period of 200 ns is required between `ds4121_fifo_reset` and `ds4121_fifo_start`.

Parameters

base	PHS-bus base address of the DS4121 board
channel	DS4121 channel number; the possible values are 1 or 2.

Return value

None

Related topics

Basics

ECU Interface Unit (DS4121 Features 📖)

References

ds4121_ecu_version_check

Syntax

```
UInt32 ds4121_ecu_version_check(
      phs_addr_t base,
      UInt32 channel);
```

Include file

`ds4121.h`

Purpose

To check the version of dSPACE's ECU Software Porting Kit used by the ECU connected to the specified channel (refer to ECU Software Porting Kit (DS4121 Features 📖)).

Note

- `ds4121_subint_init` must be called before the `ds4121_ecu_version_check` function can be used.
- Use this function only if your ECU supports the subinterrupts provided by dSPACE's ECU Software Porting Kit.

Parameters

base	PHS-bus base address of the DS4121 board
channel	DS4121 channel number; the possible values are 1 or 2.

Return value

Error code. The following macros are predefined:

Return Value	Meaning
DS4121_ECU_VERSION_OK	The ECU Software Porting Kit version found is compatible with the RTLib version used.
DS4121_ECU_VERSION_NOT_AVAILABLE	The ECU Software Porting Kit version could not be evaluated. This occurs when the ECU is not yet alive.

Return Value	Meaning
DS4121_ECU_VERSION_OBSOLETE	The ECU Software Porting Kit version found is old and not compatible with the RTLib version used. Use the current version of the ECU Software Porting Kit (contact support@dspace.de or call the technical support team) to recompile and download your ECU code.
DS4121_ECU_WRONG_COMPILATION	dSPACE's ECU Software Porting Kit supports different ECU interface boards. The subinterrupt handling on your ECU has been compiled for another dSPACE board and is not compatible with the board used.

Messages

The following messages are defined:

Type	Message	Meaning
Info	ds4121_ecu_version_check(0xXX): ECU Software Porting Kit rev. X.Y.ZZ detected on ch x!	Version X.Y.ZZ of the ECU Software Porting Kit was detected. This version is compatible with the RTLib version used.
Warning	ds4121_ecu_version_check(0xXX): ECU SW Vs. not yet available on ch x!	Normally, the ECU is powered up after the dSPACE system. In this case, the version of the ECU Software Porting Kit is not available yet. This information is displayed if ▪ The ECU is not running, ▪ The ECU is powered up after the dSPACE system or ▪ The ECU is not alive.
Warning	ds4121_ecu_version_check(0xXX): ECU software rev. X.Y or higher required on ch x!	The ECU Software Porting Kit version found is old and not compatible with the RTLib version used. Use the current version of the ECU Software Porting Kit (contact support@dspace.de or call the technical support team) to recompile and download your ECU code.
Warning	ds4121_ecu_version_check(0xXX): ECU software is not compiled for DS4121 on ch x!	dSPACE's ECU Software Porting Kit supports different ECU interface boards. The subinterrupt handling on your ECU has been compiled for another dSPACE board and is not compatible with the board used. For information on how to set up the ECU Software Porting Kit for a DS4121, refer to Generating the ECU Specific File on page 36.

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 🕮)

References

ds4121_subint_init........ 18
Generating the ECU Specific File........ 36

Interrupt Handling

Where to go from here

Information in this section

Information in other sections

Interrupt Handling (DS4121 Features 🕮)

ds4121_subint_init

Syntax

```
void ds4121_subint_init(
      phs_addr_t base,
      UInt32 channel,
      UInt32 nr_sints,
      UInt32 thirty_plus_x_free_dpm_words);
```

Include file

`ds4121.h`

Purpose

To create the subinterrupt receiver for the onboard DPMEM.

Description

A subinterrupt receiver is initialized for the onboard DPMEM. The subinterrupt locations are initialized with "4", that means "ready invalid".

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

nr_sints Number of subinterrupts to be used within the range of 1 … 16

thirty_plus_x_free_dpm_words Start address of `30 + x` (with x = number of subinterrupts) unused words in the onboard DPMEM that is allocated to the subinterrupt module. The usage of the words is defined in `dssint_wb_relation.h`.

Return value

None

Messages

The following messages are defined:

Type	Message	Meaning
Error	ds4121_subint_init (0xXX): Memory allocation error!	No memory could be allocated for the function.
Error	ds4121_subint_init (0xXX): Invalid channel number!	Select the ECU channel 1 or 2.

Related topics

Basics

Interrupt Handling (DS4121 Features 🕮)
Specifying DPMEM Addresses Seen from the ECU (DS4121 Features 🕮)

References

DS4121_ECUINT_Bx_CHy_I0, DS4121_ECUINT_Bx_CHy_SIz (DS4121 RTI Reference 🕮)

ds4121_end_of_int_set

Syntax

```
void ds4121_end_of_int_set(
     phs_addr_t base,
     UInt32 channel);
```

Include file

`ds4121.h`

Purpose

To indicate the end of an interrupt to the ECU.

Parameters

base	PHS-bus base address of the DS4121 board
channel	DS4121 channel number; the possible values are 1 or 2.

Return value

None

Related topics

Basics

Interrupt Handling (DS4121 Features 📖)

References

DS4121_ECUINT_Bx_CHy_I0, DS4121_ECUINT_Bx_CHy_SIz (DS4121 RTI Reference 📖)

Reading Data from the ECU

Where to go from here

Information in this section

Information in other sections

ds4121_p_int_read8

Syntax

```
void ds4121_p_int_read8(
      phs_addr_t base,
      UInt32 channel ,
      UInt32 count,
      UInt32 *address,
      UInt32 *halfword,
      UInt8 *value);
```

Include file

`ds4121.h`

Purpose

To read 8-bit integer data from the DPMEM (packed format).

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 📖).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

count Number of values to be read

address Pointer to an array containing the addresses to be read (16-bit DPMEM addresses from the view of the RCP system)

halfword Pointer to an array containing the information for each address which halfword (8 bit) has to be read. The following macros are predefined:

Predefined Macros	Meaning
DS4121_LOWER_HALFWORD	To read the lower halfword from the related 16-bit DPMEM address.
DS4121_UPPER_HALFWORD	To read the upper halfword from the related 16-bit DPMEM address.

value Pointer to an array where the values of the specified addresses will be stored

Return value

None

Related topics

References

DS4121_ECUREAD_Bx_CHy_BLz (DS4121 RTI Reference 📖)

ds4121_s_int_read8

Syntax

```
void ds4121_s_int_read8(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UInt8 *value);
```

Include file

`ds4121.h`

Purpose

To read 8-bit integer data from the DPMEM (scattered format). The scattered format reads only from the lower halfword.

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

count Number of values to be read

address Pointer to an array containing the addresses to be read (16-bit DPMEM addresses from the view of the RCP system)

value Pointer to an array where the values of the specified addresses will be stored

Return value

None

Related topics

References

DS4121_ECUREAD_Bx_CHy_BLz (DS4121 RTI Reference)

ds4121_le_p_int_read16, ds4121_le_p_int_read32, ds4121_le_s_int_read16, ds4121_le_s_int_read32

Syntax

```
void ds4121_le_y_int_readzz(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UIntzz *value);
```

Include file

`ds4121.h`

Purpose

To read *16*-bit or *32*-bit integer data from the DPMEM (little endian, *s*cattered or *p*acked format).

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 🕮).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

count Number of values to be read

address Pointer to an array containing the addresses to be read (16-bit DPMEM addresses from the view of the RCP system)

value Pointer to an array where the values of the specified addresses will be stored

Return value

None

Related topics

References

DS4121_ECUREAD_Bx_CHy_BLz (DS4121 RTI Reference 🕮)

ds4121_be_p_int_read16, ds4121_be_p_int_read32, ds4121_be_s_int_read16, ds4121_be_s_int_read32

Syntax

```
void ds4121_be_y_int_readzz(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UIntzz *value);
```

Include file

`ds4121.h`

Purpose	To read *16*-bit or *32*-bit integer data from the DPMEM (big endian, *s*cattered or *p*acked format).
Description	For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 📖). For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.
Parameters	**base** PHS-bus base address of the DS4121 board **channel** DS4121 channel number; the possible values are 1 or 2. **count** Number of values to be read **address** Pointer to an array containing the addresses to be read (16-bit DPMEM addresses from the view of the RCP system) **value** Pointer to an array where the values of the specified addresses will be stored
Return value	None
Related topics	References DS4121_ECUREAD_Bx_CHy_BLz (DS4121 RTI Reference 📖)

ds4121_le_p_fl_read32, ds4121_le_s_fl_read32, ds4121_be_p_fl_read32, ds4121_be_s_fl_read32

Syntax

```
void ds4121_ob_dpm_xx_y_fl_read32(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UInt32 *value);
```

Include file `ds4121.h`

Purpose	To read 32-bit floating point data from the DPMEM (*b*ig or *l*ittle endian, *s*cattered or *p*acked format).
Description	For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 🕮). For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.
Parameters	**base** PHS-bus base address of the DS4121 board **channel** DS4121 channel number; the possible values are 1 or 2. **count** Number of values to be read **address** Pointer to an array containing the addresses to be read (16-bit DPMEM addresses from the view of the RCP system) **value** Pointer to an array where the values of the specified addresses will be stored
Return value	None
Related topics	References DS4121_ECUREAD_Bx_CHy_BLz (DS4121 RTI Reference 🕮)

Transferring Data to the ECU

Where to go from here

Information in this section

Information in other sections

ds4121_p_int_write8

Syntax

```
void ds4121_p_int_write8(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UInt32 *halfword,
      UInt8 *value);
```

Include file

`ds4121.h`

Purpose

To write 8-bit integer data to the DPMEM (packed format).

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 🕮).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Note

This function uses a FIFO for the write transfer of the specified channel. If the FIFO is full, the function waits until there is enough space available for the data transfer. If the connection to the ECU has been lost during the transfer, the function blocks the application and causes an overload error.

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

count Number of values to be written

address Pointer to an array containing the addresses to be written to (16-bit DPMEM addresses from the view of the RCP system)

halfword Pointer to an array containing the information for each address which halfword (8 bit) has to be written. The following macros are predefined:

Predefined Macros	Meaning
DS4121_LOWER_HALFWORD	To read the lower halfword from the related 16-bit DPMEM address.
DS4121_UPPER_HALFWORD	To read the upper halfword from the related 16-bit DPMEM address.

value Pointer to an array containing the values to be written

Return value

None

Related topics

References

DS4121_ECUWRITE_Bx_CHy_BLz (DS4121 RTI Reference 🕮)

ds4121_s_int_write8

Syntax

```
void ds4121_s_int_write8(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UInt8 *value);
```

Include file

`ds4121.h`

Purpose

To write 8-bit integer data to the DPMEM (scattered format). The scattered format writes only to the lower halfword.

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 🕮).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Note

This function uses a FIFO for the write transfer of the specified channel. If the FIFO is full, the function waits until there is enough space available for the data transfer. If the connection to the ECU has been lost during the transfer, the function blocks the application and causes an overload error.

Parameters

base	PHS-bus base address of the DS4121 board
channel	DS4121 channel number; the possible values are 1 or 2.
count	Number of values to be written
address	Pointer to an array containing the DPMEM addresses to be written to (16-bit DPMEM addresses from the view of the RCP system)
value	Pointer to an array containing the values to be written

Return value

None

Related topics

References

DS4121_ECUWRITE_Bx_CHy_BLz (DS4121 RTI Reference 🕮)

ds4121_le_p_int_write16, ds4121_le_p_int_write32, ds4121_le_s_int_write16, ds4121_le_s_int_write32

Syntax

```
void ds4121_le_y_int_writezz(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UIntzz *value);
```

Include file

`ds4121.h`

Purpose

To write *16*-bit or *32*-bit integer data (little endian, *s*cattered or *p*acked format) to the DPMEM.

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 🕮).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Note

These functions use a FIFO for the write transfer of the specified channel. If the FIFO is full, the function waits until there is enough space available for the data transfer. If the connection to the ECU has been lost during the transfer, the function blocks the application and causes an overload error.

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

count Number of values to be written

address Pointer to an array containing the DPMEM addresses to be written to (16-bit DPMEM addresses from the view of the RCP system)

value Pointer to an array containing the values to be written

Return value

None

Related topics

References

DS4121_ECUWRITE_Bx_CHy_BLz (DS4121 RTI Reference 📖)

ds4121_be_p_int_write16, ds4121_be_p_int_write32, ds4121_be_s_int_write16, ds4121_be_s_int_write32

Syntax

```
void ds4121_be_y_int_writezz(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UIntzz *value);
```

Include file

`ds4121.h`

Purpose

To write *16*-bit or *32*-bit integer data (big endian, *s*cattered or *p*acked format) to the DPMEM.

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 📖).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Note

These functions use a FIFO for the write transfer of the specified channel. If the FIFO is full, the function waits until there is enough space available for the data transfer. If the connection to the ECU has been lost during the transfer, the function blocks the application and causes an overload error.

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

count Number of values to be written

address Pointer to an array containing the DPMEM addresses to be written to (16-bit DPMEM addresses from the view of the RCP system)

value Pointer to an array containing the values to be written

Return value

None

Related topics

References

DS4121_ECUWRITE_Bx_CHy_BLz (DS4121 RTI Reference 🕮)

ds4121_le_p_fl_write32, ds4121_le_s_fl_write32, ds4121_be_p_fl_write32, ds4121_be_s_fl_write32

Syntax

```
void ds4121_xx_y_fl_write32(
      phs_addr_t base,
      UInt32 channel,
      UInt32 count,
      UInt32 *address,
      UInt32 *value);
```

Include file

`ds4121.h`

Purpose

To write 32-bit floating point data (*b*ig or *l*ittle endian, *s*cattered or *p*acked format) to the DPMEM.

Description

For more information on data types, refer to DS4121 RTLib Functions on page 8 and Supported ECU Data Type Formats (DS4121 Features 🕮).

For information on the data arrangement in the DPMEM, refer to Data Layout in the DPMEM on page 9.

Note

These functions use a FIFO for the write transfer of the specified channel. If the FIFO is full, the function waits until there is enough space available for the data transfer. If the connection to the ECU has been lost during the transfer, the function blocks the application and causes an overload error.

Parameters

base PHS-bus base address of the DS4121 board

channel DS4121 channel number; the possible values are 1 or 2.

count Number of values to be written

address Pointer to an array containing the DPMEM addresses to be written to (16-bit DPMEM addresses from the view of the RCP system)

value Pointer to an array containing the values to be written

Return value

None

Related topics

References

DS4121_ECUWRITE_Bx_CHy_BLz (DS4121 RTI Reference 🕮)

ECU Software Porting Kit

Introduction

The ECU Software Porting Kit allows you to split one ECU-to-RTP interrupt into 16 subinterrupts (0 … 15).

For further information, refer to ECU Software Porting Kit (DS4121 Features 🕮).

Where to go from here

Information in this section

Generating the ECU Specific File

Defines to be Specified in the dsECUframe.c File

Required defines

The following defines have to be specified in the ECU-specific `dsECUframe.c` file:

Define	Description
_DSECU	To identify the ECU target type. ***NOTICE*** **Do not modify this define.**
DSECU_INT16	To specify the integer data type to be used for a width of 16 bit. Possible values are: short, int or long.
DSECU_THIRTY_PLUS_X_FREE_DPM_WORDS	To specify the start address for `30 + x` (with x = number of subinterrupts to be used) successive DPMEM addresses to be used for subinterrupt handling.
DSECU_ECU2RTP_INT	To specify the DPMEM address. When the ECU writes to this location, an interrupt is triggered on the RTP.
DSECU_COMPILATION	To specify the dSPACE ECU interface board. The specification is used for the compilation process: Depending on the specified board, the compilation results in different interrupt triggering DPMEM address. You can combine the predefined macros via the logical operator OR. The following macros are predefined: ▪ DSSINT_WB_COMPILED_FOR_DS4121 Compilation for a DS4121 board. ▪ DSSINT_WB_COMPILED_FOR_DS1401 Compilation for an ECU module of a MicroAutoBox II. ▪ DSSINT_WB_COMPILED_FOR_INDEPENDENT Compilation independent of the hardware.

Related topics

References

dsECUnew.bat.. 50

Initializing the ECU Software Porting Kit

Introduction

To use subinterrupt handling for your ECU, you have to initialize the subinterrupt module.

Where to go from here

Information in this section

Information in other sections

ECU Software Porting Kit (DS4121 Features 🕮)

Word-Based Subinterrupt Handling

DSECU_SINT_INIT

Syntax

```
DSECU_SINT_INIT();
```

Include file

`dsECU.h`

Purpose

To simplify the initialization of the subinterrupt module.

Description

This macro performs the following steps:

- Calls `dsecu_define_sender` to initialize the subinterrupt module for your specific ECU. The parameters are derived from the defines in `Defines to be`

`Specified in the dsECUframe.c File`. The relation between the parameters and the macros is as follows:

Parameter	Macro
sender	dssint_sender
nr_subinterrupts	DSECU_SINT_NUMBER
dpm_block_start_addr	DSECU_THIRTY_PLUS_X_FREE_DPM_WORDS
send_addr	DSECU_ECU2RTP_INT

- Writes the version information to the DPMEM.
- Calls `dsecu_startup` to start the alive mechanism (refer to ECU Software Porting Kit (DS4121 Features 🕮)).

Return value None

Example

```
void main(void)
{
   int sint_number;
   DSECU_SINT_INIT();
   while(1)
   {
      sint_number = rand() & 0x000f;
      DSECU_SINT_SEND(sint_number);
      if (DSECU_EOSI_POLL(sint_number) == DSECU_BYPASS_DATA_VALID)
      {
         read_bypass_data();
      }
   }
}
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 🕮)
Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

Defines to be Specified in the dsECUframe.c File........ 36
dsecu_define_sender........ 39
dsecu_startup........ 40

dsecu_define_sender

Syntax

```
void dsecu_define_sender(
      dsecu_sender_type *sender,
      unsigned int nr_subinterrupts,
      unsigned long dpm_block_start_addr,
      unsigned long send_addr);
```

Include file

`dsECU.h`

Purpose

To initialize a subinterrupt handler.

Note

It is recommended to use the macro DSECU_SINT_INIT that provides the necessary parameters automatically.

Parameters

sender Address of the subinterrupt handler

nr_subinterrupts Number of subinterrupts to be used in the application within the range of 1 … 16

dpm_block_start_addr Specifies the start address for `30 + x` (with x = number of subinterrupts to be used) successive DPMEM addresses to be used for subinterrupt handling.

send_addr DPMEM address; if the ECU writes to this location an interrupt will be triggered on the RTP.

Return value

None

Example

This example initializes a subinterrupt handler via the standard macros defined in `Defines to be Specified in the dsECUframe.c File`.

```
dsecu_define_sender(
            &dssint_sender,
            DSECU_SINT_NUMBER,
            DSECU_30_PLUS_X_FREE_DPM_WORDS,
            DSECU_ECU2RTP_INT);
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 🕮)
Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

dsecu_startup

Syntax

```
void dsecu_startup(dsecu_sender_type *sender);
```

Include file

`dsECU.h`

Purpose

To start the alive mechanism (refer to ECU Software Porting Kit (DS4121 Features 🕮)).

Note

This function must be called once when starting an application and after a suspension via `dsecu_sender_suspend`.

Parameters

sender	Address of the subinterrupt handler

Return value

None

Example

```
void main(void)
{
   int startup_flag = 1;
   dsecu_startup(sender);
   while(1) /* background loop*/
   {
      if (*ecu_suspend)
      {
         dsecu_sender_suspend(dssint_sender);
         startup_flag = 0;
      }
      else /* if ECU is not suspended */
      {
         if (!startup_flag)
         {
            dsecu_startup(&dssint_sender);
            startup_flag = 1;

         }
      dsecu_alive(&dssint_sender);
      ...
      }
   }
}
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 📖)
Word-Based Subinterrupt Handling (DS4121 Features 📖)

References

dsecu_alive.. 41
dsecu_sender_suspend.. 43

dsecu_alive

Syntax

```
void dsecu_alive(dsecu_sender_type *sender);
```

Include file

`dsECU.h`

Purpose

To continue the alive mechanism and check if the partner system is still alive (refer to ECU Software Porting Kit (DS4121 Features 📖)).

Note

This function has to be called repetitively in the background loop as long as the connection from the ECU to the RTP is valid.

Parameters

sender	Address of the subinterrupt handler

Return value

None

Example

```
void main(void)
{
   int startup_flag = 1;
   dsecu_startup(sender);
   while(1) /* background loop*/
   {
      if (*ecu_suspend)
      {
         dsecu_sender_suspend(dssint_sender);
         startup_flag = 0;
      }
      else /* if ECU is not suspended */
      {
         if (!startup_flag)
         {
            dsecu_startup(&dssint_sender);
            startup_flag = 1;
         }
         dsecu_alive(&dssint_sender);
         ...
      }
   }
}
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 📖)
Word-Based Subinterrupt Handling (DS4121 Features 📖)

References

dsecu_sender_suspend..43
dsecu_startup..40

dsecu_sender_suspend

Syntax	`void dsecu_sender_suspend(dsecu_sender_type sender);`
Include file	`dsECU.h`
Purpose	To signal to the dSPACE system that the ECU is no longer alive.
Parameters	**sender** Subinterrupt handler
Return value	None

Example

```
void main(void)
{
   int startup_flag = 1;
   dsecu_startup(sender);
   while(1) /* background loop*/
   {
      if (*ecu_suspend)
      {
         dsecu_sender_suspend(dssint_sender);
         startup_flag = 0;
      }
      else /* if ECU is not suspended */
      {
         if (!startup_flag)
         {
            dsecu_startup(&dssint_sender);
            startup_flag = 1;
         }
         dsecu_alive(&dssint_sender);
         ...
      }
   }
}
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 🕮)
Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

Subinterrupt Programming

Introduction

Subinterrupt handling allows you to split the hardware interrupt into 16 subinterrupts.

For more information on the interrupt handling, refer to *Interrupt Handling* and *Word-Based Subinterrupt Handling* in the *DS4121 Features* document.

Where to go from here

Information in this section

DSECU_SINT_SEND

Syntax

```
DSECU_SINT_SEND(subinterrupt_number);
```

Include file

`dsECU.h`

Purpose

To simplify the sending of a subinterrupt.

Description

This macro performs the following steps:

- Calls `dsecu_sint_send` to send a subinterrupt.
- Calls `dsecu_alive` to start the alive mechanism.

Parameters

subinterrupt_number Enter the number of the subinterrupt to be sent within the range of 0 … 15

Return value

None

Example

```
void main(void)
{
   int sint_number;
   DSECU_SINT_INIT();
   while(1)
   {
      sint_number = rand() & 0x000f;
      write_bypass_data(sint_number);
      DSECU_SINT_SEND(sint_number);
      if (DSECU_EOSI_POLL(sint_number) == DSECU_BYPASS_DATA_VALID)
      {
         read_bypass_data(sint_number);
      }
   }
}
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 📖)
Word-Based Subinterrupt Handling (DS4121 Features 📖)

References

dsecu_alive.. 41
dsecu_sint_send.. 46

dsecu_sint_send

Syntax

```
int dsecu_sint_send(
      dsecu_sender_type* sender,
      unsigned int sub_interrupt);
```

Include file

`dsECU.h`

Purpose

To send a subinterrupt.

Note

It is recommended to use the macro DSECU_SINT_SEND that provides the necessary parameters automatically.

Parameters

sender Address of the subinterrupt handler

sub_interrupt Enter the number of the subinterrupt to be sent within the range of 0 … 15

Return value

Error code; the following macros are predefined:

Return Value	Meaning
DSECU_INTERRUPT_SENT	The subinterrupt was sent successfully.
DSECU_INTERRUPT_NOT_SENT	The subinterrupt could not be sent.

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 🕮)
Word-Based Subinterrupt Handling (DS4121 Features 🕮)

DSECU_EOSI_POLL

Syntax

```
DSECU_EOSI_POLL(subinterrupt_number);
```

Include file

`dsECU.h`

Purpose

To simplify the polling for the end-of-subinterrupt message. This function sets the parameter `sender` automatically.

Parameters

subinterrupt_number Number of the subinterrupt within the range of 0 … 15

Return value

None

Example

```
void main(void)
```

```
{
   int sint_number;
   DSECU_SINT_INIT();
   while(1)
   {
      sint_number = rand() & 0x000f;
      DSECU_SINT_SEND(sint_number);
      if (DSECU_EOSI_POLL(sint_number) == DSECU_BYPASS_DATA_VALID)
      {
         read_bypass_data();
      }
   }
}
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 🕮)
Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

dsecu_eosi_poll.. 48

dsecu_eosi_poll

Syntax

```
int dsecu_eosi_poll(
      dsecu_sender_type sender,
      int sint_number);
```

Include file

`dsECU.h`

Purpose

To poll the end-of-subinterrupt message until the RTP acknowledges the subinterrupt, that means that the bypass data are valid.

Note

Use the DSECU_EOSI_POLL macro, because it provides the necessary parameters automatically.

Parameters

sender	Address of the subinterrupt handler
sint_number	Number of the subinterrupt within the range of 0 … 15

Return value

Error code; the following macros are predefined:

Return Value	Meaning
DSECU_BYPASS_DATA_VALID	The bypass data are valid.
DSECU_BYPASS_DATA_INVALID	The bypass data are invalid.

Example

```
if (DSECU_BYPASS_DATA_VALID == dsecu_eosi_poll(dssint_sender,sint))
{
   read_data();
}
```

Related topics

Basics

ECU Software Porting Kit (DS4121 Features 🕮)
Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

DSECU_EOSI_POLL.. 47

Host Utilities

dsECUnew.bat

Purpose

To create the new working directory and copy `dsECUframe.c` to the new directory.

Description

To work with the ECU Software Porting Kit, you have to create the working directory for the ECU application and copy the `Defines to be Specified in the dsECUframe.c File` file to this directory using the `dsECUnew.bat` function.

For already existing ECU projects, copy the `dsECUframe.c` file to the working directory.

Some ECU-specific settings must be made in the `Defines to be Specified in the dsECUframe.c File` file for the specific ECU used for your application. This file has to be included into your ECU build project. A template of the file is located in `<RCP_HIL_InstallationPath>\dsECU\PortingKit`.

Syntax

```
dsECUnew Directory
```

Location `<RCP_HIL_InstallationPath>\Exe`

Info messages

The following information message is defined:

Type	Message	Description
Info	The ECU project "<projectname> contains a dsecuframe.c file yet. The existing file will be opened.	The directory already exists and there is a file `dsECUframe.c` in the directory. The file will be opened.

Related topics

References

Defines to be Specified in the dsECUframe.c File.. 36

Word-Based Subinterrupt Handling

Introduction

The Word-Based Subinterrupt module provides functions to extend one hardware interrupt to multiple software subinterrupts.

Where to go from here

Information in this section

dssint_wb_define_receiver

Syntax

```
dssint_wb_receiver_type *dssint_wb_define_receiver(
      long target,
      unsigned int nr_subinterrupts,
      unsigned long dpm_block_start_addr,
      unsigned long receiver_addr,
      dssint_wb_write_fcn_t write_fcn,
      dssint_wb_read_fcn_t read_fcn,
      dssint_wb_read_block_fcn_t read_block_fcn);
```

Include file

`Dssint_wb.h`

Purpose

To create a receiver handler for the word-based subinterrupt handling.

Description

The function reads from the `receiver_addr` to enable interrupt triggering by the sender. It defines an interrupt receiver and returns a handle to it. The handle identifies the appropriate subinterrupt vector and receiving information table for a specific sender.

Parameters

target Address of the target memory, for example, a PHS bus address or COM port number

nr_subinterrupts Number of subinterrupts to be used in the application within the range of 1 … 16

dpm_block_start_addr Specifies the start address for `30 + x` (with x = number of subinterrupts to be used) successive DPMEM addresses to be used for subinterrupt handling.

receiver_addr Memory location where the acknowledgment information from the receiver is passed

write_fcn Address of a function that performs a write access to the dual-port memory

read_fcn Address of a function that performs a read access to the dual-port memory

read_block_fcn Address of a function that performs a block read access to the dual-port memory

Return value

Address of an interrupt receiver. The functions returns "0" if an error has occurred.

Example

```
ds4121_subint_receiver[int_line][base_index] =
      (dssint_wb_receiver_type*)dssint_wb_define_receiver(
            base,
            nr_sints,
            thirty_plus_x_free_dpm_words,
            acknowledge_addr,
            (dssint_wb_write_fcn_t)write_fcn,
            (dssint_wb_read_fcn_t)read_fcn,
            (dssint_wb_read_block_fcn_t)read_block_fcn);
```

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 📖)

dssint_wb_subint_disable

Syntax

```
void dssint_wb_subint_disable(
      dssint_wb_receiver_type *receiver,
      unsigned int sub_interrupt);
```

Include file

`Dssint_wb.h`

Purpose

To disable a subinterrupt.

Description

After initialization all subinterrupts are enabled. You must disable the subinterrupt explicitly via this function.

Parameters

receiver	Receiver handler where the subinterrupt is located in.
sub_interrupt	Subinterrupt to be disabled within the range of 0 … 15

Return value

None

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 📖)

References

dssint_wb_subint_enable

Syntax

```
void dssint_wb_subint_enable(
      dssint_wb_receiver_type *receiver,
      unsigned int sub_interrupt);
```

Include file

`Dssint_wb.h`

Purpose

To enable a subinterrupt.

Description

After initialization all subinterrupts are enabled. You can use this function to enable a subinterrupt after it has been disabled via `dssint_wb_subint_disable`.

Parameters

receiver	Receiver handler where the subinterrupt is located in.
sub_interrupt	Subinterrupt to be enabled within the range of 0 … 15.

Return value

None

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 📖)

References

dssint_wb_decode

Syntax

```
int dssint_wb_decode(
      dssint_wb_receiver_type *receiver);
```

Include file

`Dssint_wb.h`

Purpose

To find out which interrupts are pending.

Description

This function is called repetitively within an interrupt handler. It processes the interrupt information of the receiver data structure that was given by `dssint_wb_acknowledge`, determines the pending subinterrupt with the highest priority and returns it to the handler. The pending subinterrupt with highest priority is the one with the smallest subinterrupt number.

Parameters

receiver	Receiver handler where the subinterrupt is located in.

Return value

Number of the pending subinterrupt with highest priority. If there is no pending subinterrupt left, the function returns SINT_WB_NO_SUBINT ("–1").

Example

```
void sint_isr(void)
{
   int sint_number;
   dssint_wb_acknowledge(RECEIVER);
   do
   {
      sint_number = dssint_wb_decode(RECEIVER);
      if (sint_number > -1)
      {
         sint_counter[sint_number]++;
         dssint_wb_end_of_sint_set(RECEIVER, sint_number);
      }
   } while(sint_number > -1);
}
```

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

dssint_wb_acknowledge

Syntax

```
void dssint_wb_acknowledge(
      dssint_wb_receiver_type *receiver);
```

Include file

`Dssint_wb.h`

Purpose

To acknowledge all pending subinterrupts.

Parameters

receiver	Receiver handler where the subinterrupt is located in.

Return value

None

Example

```
void sint_isr(void)
{
   int sint_number;
   dssint_wb_acknowledge(RECEIVER);
   do
   {
      sint_number = dssint_wb_decode(RECEIVER);
      if (sint_number > -1)
      {
         sint_counter[sint_number]++;
         dssint_wb_end_of_sint_set(RECEIVER, sint_number);
      }
   } while(sint_number > -1);
}
```

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

dssint_wb_subint_reset

Syntax

```
void dssint_wb_subint_reset(
      dssint_wb_receiver_type *receiver,
      unsigned int sub_interrupt);
```

Include file

`Dssint_wb.h`

Purpose

To clear a pending subinterrupt.

Parameters

receiver	Receiver handler where the subinterrupt is located in.
sub_interrupt	Subinterrupt to be enabled within the range of 0 … 15.

Return value

None

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

dssint_wb_end_of_sint_set

Syntax	`void dssint_wb_end_of_sint_set(`
	`     dssint_wb_receiver_type *receiver,`
	`     unsigned int sub_interrupt);`

Include file	`Dssint_wb.h`
Purpose	To finish a pending subinterrupt.
Parameters	**receiver** Receiver handler where the subinterrupt is located in. **sub_interrupt** Subinterrupt to be enabled within the range of 0 … 15.
Return value	None
Related topics	Basics Word-Based Subinterrupt Handling (DS4121 Features 🕮) References dssint_wb_acknowledge........ 56 dssint_wb_decode........ 55

dssint_wb_startup

Syntax	`void dssint_wb_startup(`
	`     dssint_wb_type *handler);`

Include file	`Dssint_wb.h`

Purpose

To start the alive mechanism (refer to ECU Software Porting Kit (DS4121 Features 📖)).

Note

This function must be called once when starting an application and after a suspension via `dssint_wb_suspend`.

Parameters

handler	Handler where the subinterrupt receiver is located in.

Return value

None

Example

```
void main(void)
{
   int startup_flag = 1;
   dssint_wb_startup(RECEIVER);
   ...
   while(1) /* background loop */
   {
      if (!suspend_flag)
      {
         if (!startup_flag)
         {
            dssint_wb_startup(RECEIVER);
            startup_flag = 1;
         }
         dssint_wb_alive(RECEIVER);
      }
      else
      {
         dssint_wb_suspend(RECEIVER);
         startup_flag = 0;
      }
      ...
   }
}
```

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 📖)

References

dssint_wb_alive 60
dssint_wb_suspend 61

dssint_wb_alive

Syntax

```
void dssint_wb_alive(dssint_wb_type *handler);
```

Include file

`Dssint_wb.h`

Purpose

To continue the alive mechanism and check if the partner system is still alive (refer to ECU Software Porting Kit (DS4121 Features 🕮)).

Note

This function has to be called repetitively in the background loop as long as the connection from the ECU to the RTP is valid.

Parameters

handler	Handler where the subinterrupt receiver is located in.

Return value

None

Example

```
void main(void)
{
   int startup_flag = 1;
   dssint_wb_startup(RECEIVER);
   ...
   while(1) /* background loop */
   {
      if (!suspend_flag)
      {
         if (!startup_flag)
         {
            dssint_wb_startup(RECEIVER);
            startup_flag = 1;
         }
         dssint_wb_alive(RECEIVER);
      }
      else
      {
         dssint_wb_suspend(RECEIVER);
         startup_flag = 0;
      }
      ...
   }
}
```

Related topics	Basics Word-Based Subinterrupt Handling (DS4121 Features 📖) References dssint_wb_startup .... 58 dssint_wb_suspend 61

dssint_wb_suspend

Syntax

```
void dssint_wb_suspend(dssint_wb_type *handler);
```

Include file

`Dssint_wb.h`

Purpose

To signal to the dSPACE system that the ECU is no longer alive.

Parameters

handler	Handler where the subinterrupt receiver is located in.

Return value

None

Example

```
void main(void)
{
   int startup_flag = 1;
   ...
   dssint_wb_startup(RECEIVER);
   ...
   while(1) /* background loop */
   {
      if (!suspend_flag)
      {
         if (!startup_flag)
         {
            dssint_wb_startup(RECEIVER);
            startup_flag = 1;
         }
         dssint_wb_alive(RECEIVER);
      }
      else
      {
         dssint_wb_suspend(RECEIVER);
         startup_flag = 0;
      }
      ...
   }
}
```

Related topics

Basics

Word-Based Subinterrupt Handling (DS4121 Features 🕮)

References

dssint_wb_alive.. 60

Function Execution Times

Function Execution Times

Introduction

Function execution times for DS4121 are measured with a standardized test environment.

Test environment

The execution time of a function can vary, since it depends on different factors, for example:

- CPU clock and bus clock frequency of the processor board used
- Optimization level of the compiler
- Use of inlining parameters

The test programs that are used to measure the execution time of the functions listed below have been generated and compiled with the default settings of the `down<xxxxx>` tool (optimization and inlining). The execution times in the tables below are always the mean measurement values.

The properties of the processor boards used are:

	DS1006	**DS1006 Multicore**
CPU clock	2.6 GHz / 3.0 GHz	2.8 GHz
Bus clock	133 MHz	133 MHz

Overview

Execution times of the following RTLib functions are listed below.

- System functions
- Functions for reading data from the ECU
- Functions for transferring data to the ECU

System functions

The following execution times have been measured for system functions (functions for initializing the board, accessing the FIFO and checking the ECU connection):

Function	Execution Time	
	DS1006	DS1006 Multicore
ds4121_init	87.79 µs	109.03 µs
ds4121_fifo_check	0.66 µs	0.61 µs
ds4121_fifo_reset	0.66 µs	0.63 µs
ds4121_fifo_start	0.66 µs	0.63 µs
ds4121_connection_check	0.82 µs	0.66 µs

Functions for reading data from the ECU

The following execution times have been measured for the functions used to read data from the ECU:

Function	Execution Time		
	DS1005	DS1006	DS1006 Multicore
ds4121_p_int_read8	(0.153 + n · 0.200) µs 1)	(-0.522 + n · 0.529) µs1)	(0.146 + n · 0.517) µs1)
ds4121_s_int_read8	(0.214 + n · 0.186) µs 1)	(-0.522 + n · 0.529) µs1)	(0.146 + n · 0.517) µs1)
ds4121_le_p_int_read16, ds4121_le_p_int_read32, ds4121_le_s_int_read16, ds4121_le_s_int_read32	(0.163 + n · 0.190) µs, (0.164 + n · 0.387) µs, (0.164 + n · 0.387) µs, (0.088 + n · 1.062) µs 1)	(-0.515 + n · 0.529) µs (-1.041 + n · 1.057) µs (-1.042 + n · 1.057) µs (-2.088 + n · 2.111) µs1)	(0.146 + n · 0.517) µs (0.138 + n · 1.046) µs (0.139 + n · 1.046) µs (0.147 + n · 2.075) µs1)
ds4121_be_p_int_read16, ds4121_be_p_int_read32, ds4121_be_s_int_read16, ds4121_be_s_int_read32	(0.162 + n · 0.190) µs, (0.164 + n · 0.387) µs, (0.163 + n · 0.387) µs, (0.152 + n · 0.749) µs 1)	(-0.515 + n · 0.529) µs (-1.042 + n · 1.057) µs (-1.041 + n · 1.057) µs (-2.094 + n · 2.111) µs1)	(0.146 + n · 0.517) µs (0.136 + n · 1.046) µs (0.138 + n · 1.046) µs (0.136 + n · 2.092) µs1)
ds4121_le_p_fl_read32, ds4121_le_s_fl_read32, ds4121_be_p_fl_read32, ds4121_be_s_fl_read32	(0.164 + n · 0.387) µs (0.126 + n · 1.074) µs (0.164 + n · 0.387) µs (0.188 + n · 0.762) µs1)	(-1.044 + n · 1.059) µs (-2.066 + n · 2.089) µs (-1.044 + n · 1.059) µs (-2.073 + n · 2.089) µs1)	(0.139 + n · 1.046) µs (0.145 + n · 2.076) µs (0.139 + n · 1.046) µs (0.138 + n · 2.092) µs1)

1) n is the number of values to be read.

Functions for transferring data to the ECU

The following execution times have been measured for the functions used to transfer data to the ECU:

Function	Execution Time		
	DS1005	DS1006	DS1006 Multicore
ds4121_p_int_write8	(0.264 + n · 0.141) µs1)	(-0.085 + n · 0.138) µs1)	(0.509 + n · 0.139) µs1)
ds4121_s_int_write8	(0.262 + n · 0.141 µs1)	(-0.083 + n · 0.136) µs1)	(0.510 + n · 0.139) µs1)
ds4121_le_p_int_write16, ds4121_le_p_int_write32, ds4121_le_s_int_write16, ds4121_le_s_int_write32	(0.261 + n · 0.141) µs (0.114 + n · 0.291) µs (0.112 + n · 0.291) µs (-0.080 + n · 0.583) µs1)	(-0.083 + n · 0.136) µs (-0.248 + n · 0.309) µs (-0.260 + n · 0.309) µs (0.003 + n · 0.606) µs1)	(0.510 + n · 0.139) µs (0.352 + n · 0.303) µs (0.353 + n · 0.303) µs (0.057 + n · 0.613) µs1)
ds4121_be_p_int_write16, ds4121_be_p_int_write32, ds4121_be_s_int_write16, ds4121_be_s_int_write32	(0.261 + n · 0.141) µs (0.114 + n · 0.291) µs 0.113 + n · 0.291) µs (-0.081 + n · 0.583) µs	(-0.086 + n · 0.135) µs (-0.249 + n · 0.309) µs (-0.249 + n · 0.309) µs (0.002 + n · 0.606) µs1)	(0.513 + n · 0.139) µs (0.353 + n · 0.303) µs (0.352 + n · 0.303) µs (0.059 + n · 0.613) µs1)

Function	Execution Time		
	DS1005	**DS1006**	**DS1006 Multicore**
ds4121_le_p_fl_write32, ds4121_le_s_fl_write32, ds4121_be_p_fl_write32, ds4121_be_s_fl_write32	(0.163 + n · 0.290) µs (-0.081 + n · 0.583) µs (0.161 + n · 0.290) µs (-0.080 + n · 0.583) µs[1]	(-0.231 + n · 0.308) µs (-0.522 + n · 0.611) µs (-0.226 + n · 0.308) µs (-0.521 + n · 0.611) µs[1]	(0.355 + n · 0.303) µs (0.058 + n · 0.613) µs (0.352 + n · 0.303) µs (0.074 + n · 0.613) µs[1]

[1] n is the number of values to be written.