DS2401 Resistive Sensor Simulation 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

This publication and the contents hereof are subject to change without notice.

Contents

About This Reference

Contents

This reference gives detailed descriptions of the C functions needed to program a DS2401 Resistive Sensor Simulation Board. The C functions can be used to program RTI-specific Simulink S-functions, or to implement your real-time models manually using C programs.

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.

Macros

Introduction

The base address of an I/O board in a PHS-bus-based system has to be defined by using the `DSxxxx_n_BASE` macro.

Base Address of the I/O Board

DSxxxx_n_BASE Macros

When using I/O board functions, you always need the board's base address as a parameter. This address can easily be obtained by using the `DSxxxx_n_BASE` macros, where `DSxxxx` is the board name (for example, DS2001) and `n` is an index which counts boards of the same type. The board with the lowest base address is given index 1. The other boards of the same type are given consecutive numbers in order of their base addresses.

The macros reference an internal data structure which holds the addresses of all I/O boards in the system. The initialization function of the processor board (named `init`) creates this data structure. Hence, when you change an I/O board base address, it is not necessary to recompile the code of your application. For more information on the processor board's initialization function, refer to ds1006_init (DS1006 RTLib Reference) or init (DS1007 RTLib Reference).

> **Note**
>
> The `DSxxxx_n_BASE` macros can be used only after the processor board's initialization function `init` is called.

Example

This example demonstrates the use of the `DSxxxx_n_BASE` macros. There are two DS2001 boards, two DS2101 boards, and one DS2002 board connected to a PHS bus. Their base addresses have been set to different addresses. The following table shows the I/O boards, their base addresses, and the macros which can be used as base addresses:

Board	Base Address	Macro
DS2001	00H	`DS2001_1_BASE`
DS2002	20H	`DS2002_1_BASE`
DS2101	80H	`DS2101_1_BASE`
DS2001	90H	`DS2001_2_BASE`
DS2101	A0H	`DS2101_2_BASE`

Initialization Function

Introduction

Before you can use the DS2401 board, you have to perform the initialization process.

ds2401_board_init

Syntax

```
void ds2401_board_init(phs_addr_t base)
```

Include file

```
ds2401.h
```

Purpose

To initialize the DS2401 board.

Description

> **Note**
>
> - This function must be called before any other DS2401 function can be used.
> - The initialization function of the processor board must be called before the DS2401 board's initialization function.

When the DS2401 board is initialized, simulated resistors are disconnected and default resistors are enabled.

After initialization or reset of the DS2401 board, the default resistores are enabled. You can disable the default resistor using the `ds2401_set_initial_value` or `ds2401_enable_default_resistors` function. After setting a simulated resistor value using the `ds2401_out` or `ds2401_strobe` function, the simulated resistor is connected in parallel to the default resistor if it is still enabled.

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

Return value

None

Messages

The following message is defined:

ID	Type	Message	Description
201	Error	ds2401_init(): Invalid PHS-bus base 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.
-165	Error	ds2401_board_init(0x??): Board not found!	No DS2401 board could be found at the specified PHS-bus address. Check if the DSxxxx_n_BASE macro corresponds to I/O board used.

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
void main(void)
{
   init();
   ds2401_board_init(DS2401_1_BASE);
   ...
}
```

The DS2401 board at address `DS2401_1_BASE` is initialized.

Related topics

References

Base Address of the I/O Board.......... 7
Macros.......... 7

Resistive Sensor Simulation Unit

Introduction

The DS2401 Resistive Sensor Simulation Board features 4 resistor output channels.

> **Note**
>
> You have to initialize the DS2401 board with the `ds2401_board_init` function before you can use one of these functions.

Where to go from here

Information in this section

ds2401_ch_ovld

Syntax

```
int ds2401_ch_ovld(
   phs_addr_t base,
   int channel)
```

Include file

```
ds2401.h
```

Purpose

To check for overload for an individual DS2401 output channel.

Description

This function performs an overload check for an individual channel.

In case the overload protection circuit signals an overload situation at the specified output channel, the value `DS2401_OVERLOAD` is returned. Since the function evaluates the respective OVLD bit in the board's OVLD register, `DS2401_OVERLOAD` is returned only as long as the overload situation is present. Valid channel numbers range from 1 … 4.

> **Note**
>
> The function `ds2401_board_init` must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Details of the Resistive Sensor Simulation Unit (DS2401 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

channel Specifies the logical channel number within the range 1 … 4.

Return values

The following values are returned:

Symbol	Meaning
`DS2401_NO_ERROR`	No overload detected
`DS2401_OVERLOAD`	Overload detected

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
void sub_fct()
{
   int err;
   err = ds2401_ch_ovld(DS2401_1_BASE, 1);
   if(err == DS2401_OVERLOAD)
   error_handler();
   ...
}
```

DS2401 board channel 1 is checked for overload. If an overload has been detected the user function `error_handler()` is called.

Related topics

References

Base Address of the I/O Board	7
ds2401_board_init	9
ds2401_ovld	20
Macros	7

ds2401_data_ready

Syntax

```
int ds2401_data_ready(
   phs_addr_t base,
   long mask)
```

Include file

```
ds2401.h
```

Purpose

To check the data transmission of the specified channel.

Description

This function returns the state of the serial data transmission to the specified DS2401 channel after a resistance value update by using the `ds2401_out` or `ds2401_load` function.

Note

The `ds2401_board_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Details of the Resistive Sensor Simulation Unit (DS2401 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

mask Specifies the bitmap for output channels to be strobed:

Symbol	Meaning
`DS2401_CH_1`	For channel 1
`DS2401_CH_2`	For channel 2
`DS2401_CH_3`	For channel 3
`DS2401_CH_4`	For channel 4
`DS2401_CH_ALL`	For all 4 channels

The definitions can be combined using the logical OR operation.

Return values

The following values are returned:

Value	Meaning
1	Channel is updated
0	Data transmission is in progress

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
void sub_fct()
{
   long state;

   ds2401_load(DS2401_1_BASE, 1, 5000.0);
   ds2401_load(DS2401_1_BASE, 2, 5000.0);
   while( !ds2401_data_ready(DS2401_1_BASE,
      DS2401_CH1 | DS2401_CH_2) )
   ds2401_strobe(DS2401_1_BASE, DS2401_CH_1 | DS2401_CH_2);
   ...
}
```

DS2401 board channels 1 and 2 are loaded for 5 kΩ. After the transmission of the new values to the channels have been finished, the channels are updated.

Related topics

References

Base Address of the I/O Board	7
ds2401_board_init	9
ds2401_load	16
ds2401_out	17
Macros	7

ds2401_enable_default_resistors

Syntax

```
void ds2401_enable_resistors(
   phs_addr_t base,
   int state)
```

Include file

```
ds2401.h
```

Purpose

To enable or disable the default resistors of the DS2401 board.

I/O mapping

For details on the I/O mapping, refer to Details of the Resistive Sensor Simulation Unit (DS2401 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

state Specifies the state of the default resistors.

Symbol	Meaning
`DS2401_DISABLE`	Disables the default resistors.
`DS2401_ENABLE`	Enables the default resistors.

Return value

None

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to enable the default resistors:

```
ds2401_enable_default_resistors(DS2401_1_BASE, DS2401_ENABLE);
```

Related topics

References

ds2401_load

Syntax

```
void ds2401_load(
   phs_addr_t base,
   int channel,
   dsfloat value)
```

Include file

```
ds2401.h
```

Purpose

To load the resistance output value.

Description

The simulated resistor value of an individual channel can be loaded by using the `ds2401_load` function.

The output resistance of the specified channel is loaded to the new output value.

After loading the output values the 4 DS2401 board channels can be updated simultaneously using the `ds2401_strobe` function.

Output values outside the valid range of 10 Ω … 500 kΩ are saturated to the minimum or maximum limit, respectively. Valid channel numbers range from 1 … 4.

Note

- Due to the serial transmission of the new value to the DS2401 channel, the loading after this function call takes about 10 µs. If the loaded channel is strobed before the serial transmission is finished the output is delayed until data transmission is done.
- The `ds2401_data_ready` function can be used to check if the data transmission to the channels has been finished.
- The `ds2401_board_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Details of the Resistive Sensor Simulation Unit (DS2401 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

channel Specifies the logical channel number within the range 1 … 4.

value Specifies the resistance output value within the range 10 Ω … 500 kΩ.

Return value

None

Execution times

For information, refer to Function Execution Times on page 27.

Example

For an example, refer to ds2401_strobe on page 24.

Related topics

References

Base Address of the I/O Board.......7
ds2401_board_init.......9
ds2401_data_ready.......13
ds2401_strobe.......24
Macros.......7

ds2401_out

Syntax

```
void ds2401_out(
   phs_addr_t base,
   int channel,
   dsfloat value)
```

Include file

```
ds2401.h
```

Purpose

To update the resistance output value.

Description

The simulated resistor of an individual channel can be updated by using the `ds2401_out` function.

The output resistance of the specified channel is set to the new output value. Output values outside the valid range of 10 … 500 kΩ are saturated to the minimum or maximum limit, respectively. Valid channel numbers range from 1 … 4.

Note

- The `ds2401_board_init` function must be called before this function can be used.
- After initialization or reset of the DS2401 board, the default resistores are enabled. You can disable the default resistor using the `ds2401_set_initial_value` or `ds2401_enable_default_resistors` function. After setting a simulated resistor value using the `ds2401_out` or `ds2401_strobe` function, the simulated resistor is connected in parallel to the default resistor if it is still enabled.

I/O mapping

For details on the I/O mapping, refer to Details of the Resistive Sensor Simulation Unit (DS2401 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

channel Specifies the logical channel number within the range 1 … 4.

value Specifies the resistance output value within the range 10 Ω … 500 kΩ.

Return value

None

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
#include <brtenv.h>
#define RMAX 500.0e3 /* maximum resistance 500 kΩ */
Float32 r1 = 100.0; /* channel 1 output resistance */
Int32   r1_overload = 0; /* channel 1 overload flag */
void isr_t0() /* timer 0 interrupt service routine */
{
   r1 = ... /* calculate current resistance */
   ds2401_out(DS2401_1_BASE, 1, r1); /* update resistance */
}
main()
{
   init(); /* basic hardware initialization */
   ds2401_board_init(DS2401_1_BASE);
      /* DS2401 initialization */
   ds2401_set_initial_value(
      DS2401_1_BASE, r1, RMAX, RMAX, RMAX);
   isr_t0_start(1.0e-3); /* setup timer interrupts */
   while (1) /* background process */
   {
      RTLIB_BACKGROUND_SERVICE(); /* background loop */
      r1_overload = /* check channel 1 overload status */
      ds2401_ch_ovld(DS2401_1_BASE, 1);
   }
}
```

In the example only channel 1 is used. Channels 2 … 4 are set to the maximum output resistance of 500 kΩ during board initialization.

The output resistance of channel 1 is periodically updated within the timer interrupt service routine `isr_t0` in steps of 1 ms. The application specific calculation of the current output resistance is not shown in the example.

Channel 1 is continually checked for overload within the background process. The DS2401 contains an overload protection circuit that turns of an individual channel in case the maximum current is exceeded. For details, refer to Details of the Resistive Sensor Simulation Unit (DS2401 Features 📖). The overload status may be displayed by using ControlDesk.

Related topics

References

Base Address of the I/O Board.......... 7
ds2401_board_init.......... 9
Macros.......... 7

ds2401_ovld

Syntax

```
int ds2401_ovld(phs_addr_t base)
```

Include file

```
ds2401.h
```

Purpose

To check the output channels for overload.

Description

This function performs a global overload check.

In case the overload protection circuit signals an overload situation at any of the 4 output channels, the value `DS2401_OVERLOAD` is returned. Since the function evaluates the OVLD bits in the board's OVLD register, `DS2401_OVERLOAD` is returned only as long as the overload situation is present.

> **Note**
>
> The function `ds2401_board_init` must be called before this function can be used.

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

Return values

The following values are returned:

Symbol	Meaning
DS2401_NO_ERROR	No overload detected
DS2401_OVERLOAD	Overload detected

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
void sub_fct()
{
   int err;

   err = ds2401_ovld(DS2401_1_BASE);
   if(err == DS2401_OVERLOAD)
   error_handler();
   ...
}
```

All DS2401 board channels are checked for overload. If an overload has been detected the user function `error_handler()` is called.

Related topics

References

Base Address of the I/O Board.. 7
ds2401_board_init.. 9
ds2401_ch_ovld.. 12
Macros.. 7

ds2401_reset

Syntax

```
void ds2401_reset(phs_addr_t base)
```

Include file

```
ds2401.h
```

Purpose

To reset the DS2401 board to the power-up state.

Description

All 4 output channels are switched to their default resistors. The simulated resistors are disabled.

> **Note**
>
> The function **`ds2401_board_init`** must be called before this function can be used.

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

Return value

None

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
void sub_fct(void)
{
   ...
   ds2401_reset(DS2401_1_BASE);
   ...
}
```

The DS2401 board is reset to power-up state.

Related topics

References

Base Address of the I/O Board...7
ds2401_board_init...9
Macros...7

ds2401_set_initial_value

Syntax

```
int ds2401_set_initial_value(
   phs_addr_t base,
   dsfloat val1,
   dsfloat val2,
   dsfloat val3,
   dsfloat val4)
```

Include file

```
ds2401.h
```

Purpose

To initialize the DS2401 board resistors.

Description

The `ds2401_set_initial_value` function sets the simulated resistors of all 4 channels to the initial values val1 … val4. The default resistors are disabled and the outputs are switched to the simulated resistors. The initial output values must be in the range 10 Ω … 500 kΩ.

In case one of the output channels is overloaded, the board is reset to the power-up state and the function sets and returns the error `DS2401_OVERLOAD`.

> **Note**
>
> - In case of an error this function does not perform an exit as many other initialization functions!
> - The `ds2401_board_init` function must be called before this function can be used.

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

val1 Specifies the resistor value of channel 1 within the range 10 Ω … 500 kΩ

val2 Specifies the resistor value of channel 2 within the range 10 Ω … 500 kΩ

val3 Specifies the resistor value of channel 3 within the range 10 Ω … 500 kΩ

val4 Specifies the resistor value of channel 4 within the range 10 Ω … 500 kΩ

Return values

The following values are returned:

Symbol	Meaning
`DS2401_NO_ERROR`	No error occurred
`DS2401_OVERLOAD`	Output channel overloaded

Messages

The following messages are defined:

ID	Type	Message	Description
-50	Error	ds2401_set_initial_value(0x??): Board not initialized!	The DS2401 board has not been initialized by a preceding call to the `ds2401_board_init` function.
-166	Error	ds2401_ set_initial_value(0x??): Channel(s) overloaded! OVLD = 0x????????	One or more of the 4 output channels are overloaded. The board is reset to the power-up state. The value OVLD in the error message shows the contents of the OVLD register of the DS2401 board.

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
void main(void)
{
   int err;
   init();
   ds2401_board_init(DS2401_1_BASE);
   err = ds2401_set_initial_value(
      DS2401_1_BASE, 1.0e3, 1.0e3, 1.0e3, 1.0e3);
   if(err == DS2401_OVERLOAD)
   error_handler();
   ...
}
```

The 4 DS2401 board simulated resistors are set to 1 kΩ. In case of an overload error the user function `error_handler` is called.

Related topics

References

ds2401_strobe

Syntax

```
void ds2401_strobe(
   phs_addr_t base,
   long mask)
```

Include file

```
ds2401.h
```

Purpose

To strobe the DS2401 output channels.

Description

The simulated resistor values of the specified channels are updated simultaneously to the values loaded by the `ds2401_load` function.

Note

- Take care that the loading to the channels has already been finished. Therefore you can use the `ds2401_data_ready` function.
- The `ds2401_board_init` function must be called before this function can be used.
- After initialization or reset of the DS2401 board, the default resistores are enabled. You can disable the default resistor using the `ds2401_set_initial_value` or `ds2401_enable_default_resistors` function. After setting a simulated resistor value using the `ds2401_out` or `ds2401_strobe` function, the simulated resistor is connected in parallel to the default resistor if it is still enabled.

I/O mapping

For details on the I/O mapping, refer to Details of the Resistive Sensor Simulation Unit (DS2401 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 7.

mask Specifies the bitmap for output channels to be strobed:

Symbol	Meaning
`DS2401_CH_1`	For channel 1
`DS2401_CH_2`	For channel 2
`DS2401_CH_3`	For channel 3
`DS2401_CH_4`	For channel 4
`DS2401_CH_ALL`	For all 4 channels

The definitions can be combined using the logical OR operation.

Return value

None

Execution times

For information, refer to Function Execution Times on page 27.

Example

This example shows how to use this function:

```
void sub_fct()
{
   ds2401_load(DS2401_1_BASE, 1, 5000.0);
   ds2401_load(DS2401_1_BASE, 2, 5000.0);
   while( !ds2401_data_ready(DS2401_1_BASE,
      DS2401_CH1 | DS2401_CH_2) )
   ds2401_strobe(DS2401_1_BASE, DS2401_CH_1 | DS2401_CH_2);
   ...
}
```

DS2401 board channels 1 and 2 are loaded for 5 kΩ. After the transmission of the new values to the channels have been finished, the channels are simultaneously updated.

Related topics

References

Function Execution Times

Introduction

To give you the mean function execution times and basic information on the test environment used.

Where to go from here

Information in this section

Information on the Test Environment

Introduction

The execution times of the C functions can vary, since they depend on different factors. The measured execution times are influenced by the test environment used.

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
CPU clock	2.6 GHz / 3.0 GHz
Bus clock	133 MHz

Measured Execution Times

Introduction

Execution times are available for the following RTLib units:

- Initialization on page 28
- Resistive sensor simulation unit on page 28

Note

The following execution times contain mean values for a sequence of I/O accesses. The execution time of a single call might be lower because of buffered I/O access.

Initialization

The following execution time has been measured for the initialization function.

Function	Mean Execution Time	
	DS1006 with 2.6 GHz	DS1006 with 3.0 GHz
ds2401_board_init	2.04 ms	2.04 ms

Resistive sensor simulation unit

The following execution times have been measured for the resistive sensor simulation unit:

Function	Mean Execution Time	
	DS1006 with 2.6 GHz	DS1006 with 3.0 GHz
ds2401_set_initial_value	1.00 ms	1.00 ms
ds2401_reset	2.00 ms	2.00 ms
ds2401_out	0.05 µs	0.08 µs
ds2401_load	0.04 µs	0.06 µs
ds2401_strobe	0.02 µs	0.02 µs
ds2401_ovld	0.59 µs	0.71 µs
ds2401_ch_ovld	0.59 µs	0.59 µs
ds2401_data_ready	0.77 µs	0.59 µs