DS2302 Direct Digital Synthesis 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 contains proprietary information that is protected by copyright. All rights are reserved. The publication may be printed for personal or internal use provided all the proprietary markings are retained on all printed copies. In all other cases, the publication must not be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of dSPACE GmbH.

© 1999 - 2021 by:
dSPACE GmbH
Rathenaustraße 26
33102 Paderborn
Germany

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

AUTERA, ConfigurationDesk, ControlDesk, MicroAutoBox, MicroLabBox, SCALEXIO, SIMPHERA, SYNECT, SystemDesk, TargetLink and VEOS are registered trademarks of dSPACE GmbH in the United States or other countries, or both. Other brand names or product names are trademarks or registered trademarks of their respective companies or organizations.

Contents

About This Reference

Introduction

This reference gives detailed descriptions of the C functions needed to program a DS2302 Direct Digital Synthesis 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.

Implementation

For information on how to implement applications for the DS2302, refer to DS2302 DSP Programming 🕮.

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 Functions

Introduction

Before you can use the DS2302, you have to perform the initialization process.

Note

The initialization function of the processor board must be called before the DS2302 board's initialization function.

Where to go from here

Information in this section

ds2302_init

Syntax

```
void ds2302_init(phs_addr_t base)
```

Include file

`ds2302.h`

Purpose

To initialize the DS2302.

Description

This function must be called before any other DS2302 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

Messages

The following messages are defined:

ID	Type	Message	Description
201	Error	ds2302_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.
-164	Error	ds2302_init(0x??): Board not found!	No DS2302 board could be found at the specified PHS-bus address. Check if the DSxxxx_n_BASE macro corresponds to the I/O board used.
-507	Error	ds2302_init(0x??): Memory allocation error!	The allocation of some dynamic memory for internal data storage has failed.

Example

This example shows how to use this function:

```
void main(void)
{
   init();
   ds2302_init(DS2302_1_BASE);
    ...
}
```

The DS2302 board at address `DS2302_1_BASE` is initialized.

Related topics

References

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

ds2302_dsp_reset_on_ioerr

Syntax

```
void ds2302_dsp_reset_on_ioerr(
   phs_addr_t base,
   long value)
```

Include file

`ds2302.h`

Purpose

To initialize the behavior of the 6 DS2302 DSP's on I/O error flag.

Description

The `value` parameter is written into the DSPIORST bit of the CTRL register.

The DSPIORST flag defines the behavior of the DSP reset lines when the I/O error line on the PHS bus is active. If DSPIORST is 1, all RESET bits of the STP register are or-combined with the PHS-bus I/O error line. In this case an active I/O error resets all DSPs. If DSPIORST is 0, the I/O error line does not affect the RESET bits of the STP register.

For further information, refer to Board Overview (Revision DS2302-01) (PHS Bus System Hardware Reference 🕮).

Note

The `ds2302_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.

value Specifies the value written into DSPIORST bit of CTRL register:

Symbol	Meaning
`DS2302_RESET_DISABLE`	No DSP reset on I/O error
`DS2302_RESET_ENABLE`	DSP reset on I/O error

Return value

None

Example

This example shows how to use this function:

```
void sub_fct()
{
   ...
   ds2302_dsp_reset_on_ioerr(DS2302_1_BASE,
      DS2302_RESET_ENABLE);
   ...
}
```

The 6 DS2302 DSP's will be reset on I/O error.

Related topics

References

ds2302_load_board

Syntax

```
void ds2302_load_board(
   phs_addr_t base,
   UInt32 int_mask,
   UInt32 start_mode,
   unsigned long ch1_app,
   unsigned long ch2_app,
   unsigned long ch3_app,
   unsigned long ch4_app,
   unsigned long ch5_app,
   unsigned long ch6_app)
```

Include file

`ds2302.h`

Purpose

To load and start slave applications to the slave DSPs of the DS2302 board.

Description

To load a slave application to the slave DSP, it must be available in a generated SLC file. Since Release 7.3, the `CL230x` tool automatically generates a SLC file during the slave DSP application build process. You can also use the coffconv utility to convert an object file. Refer to coffconv (DS2302 DSP Programming 🕮). Each SLC file contains an c array with data of the slave application object file and must be included into the DS100x processor board application. Using the `ds2302_load_board` function, you can load slave applications for up to 6 DSP channels, by specifying the names of the corresponding c arrays in the SLC files.

If slave DSP applications should be interrupted by each other (i.e. they contain calls to the `int_xf0` or `int_xf1` functions) you must setup the interrupt request line wiring by using the `int_mask` parameter. This parameter specifies which DSP channel can be interrupted by a certain other DSP channel.

The `DS2302_load_board` function is also used to start the slave applications on the slave DSP. You can start all slave applications synchronously.

Note

If additional DSPs are loaded while other DSPs are still running, a burst of INT1 interrupt requests are requested by the booting DSPs. If you are using INT1 interrupts in your slave applications, you have to acknowledge the INT1 interrupt on the corresponding DSPs by using the `int1_ack()` macro after all DSPs have been booted. Otherwise, no further INT1 interrupts can be executed.

Parameters

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

int_mask Specifies a bitmask for the interrupt request line wiring. It must be specified if a slave application should be interrupted by a slave application running on another slave DSP (the `int_xf0()` or `int_xf1()` function is used in the slave application). Otherwise, the parameter is irrelevant.

The interrupt sources are specified by an interrupt line wiring (for details, refer to Interrupt Selection (DS2302 DSP Programming 🕮)). Use the following predefined symbols to select the interrupt source. You can combine the symbols for different channels using the OR operator. You must not combine multiple defines of the same channel.

Predefined Symbol	Interrupted Channel	Interrupt Source
DS2302_CH1_IS_INTERRUPTED_BY_CH2_XF0	1	XF0 line of channel 2
DS2302_CH1_IS_INTERRUPTED_BY_CH3_XF0	1	XF0 line of channel 3
DS2302_CH1_IS_INTERRUPTED_BY_CH4_XF0	1	XF0 line of channel 4
DS2302_CH1_IS_INTERRUPTED_BY_CH5_XF0	1	XF0 line of channel 5
DS2302_CH1_IS_INTERRUPTED_BY_CH6_XF0	1	XF0 line of channel 6
DS2302_CH1_IS_INTERRUPTED_BY_CH2_XF1	1	XF1 line of channel 2
DS2302_CH1_IS_INTERRUPTED_BY_CH3_XF1	1	XF1 line of channel 3
DS2302_CH1_IS_INTERRUPTED_BY_CH4_XF1	1	XF1 line of channel 4
DS2302_CH2_IS_INTERRUPTED_BY_CH1_XF0	2	XF0 line of channel 1
DS2302_CH2_IS_INTERRUPTED_BY_CH3_XF0	2	XF0 line of channel 3
DS2302_CH2_IS_INTERRUPTED_BY_CH4_XF0	2	XF0 line of channel 4
DS2302_CH2_IS_INTERRUPTED_BY_CH5_XF0	2	XF0 line of channel 5
DS2302_CH2_IS_INTERRUPTED_BY_CH6_XF0	2	XF0 line of channel 6
DS2302_CH2_IS_INTERRUPTED_BY_CH3_XF1	2	XF1 line of channel 3
DS2302_CH2_IS_INTERRUPTED_BY_CH4_XF1	2	XF1 line of channel 4
DS2302_CH2_IS_INTERRUPTED_BY_CH5_XF1	2	XF1 line of channel 5
DS2302_CH3_IS_INTERRUPTED_BY_CH1_XF0	3	XF0 line of channel 1
DS2302_CH3_IS_INTERRUPTED_BY_CH2_XF0	3	XF0 line of channel 2
DS2302_CH3_IS_INTERRUPTED_BY_CH4_XF0	3	XF0 line of channel 4
DS2302_CH3_IS_INTERRUPTED_BY_CH5_XF0	3	XF0 line of channel 5
DS2302_CH3_IS_INTERRUPTED_BY_CH6_XF0	3	XF0 line of channel 6
DS2302_CH3_IS_INTERRUPTED_BY_CH4_XF1	3	XF1 line of channel 4
DS2302_CH3_IS_INTERRUPTED_BY_CH5_XF1	3	XF1 line of channel 5
DS2302_CH3_IS_INTERRUPTED_BY_CH6_XF1	3	XF1 line of channel 6
DS2302_CH4_IS_INTERRUPTED_BY_CH1_XF0	4	XF0 line of channel 1
DS2302_CH4_IS_INTERRUPTED_BY_CH2_XF0	4	XF0 line of channel 2
DS2302_CH4_IS_INTERRUPTED_BY_CH3_XF0	4	XF0 line of channel 3
DS2302_CH4_IS_INTERRUPTED_BY_CH5_XF0	4	XF0 line of channel 5
DS2302_CH4_IS_INTERRUPTED_BY_CH6_XF0	4	XF0 line of channel 6
DS2302_CH4_IS_INTERRUPTED_BY_CH5_XF1	4	XF1 line of channel 5
DS2302_CH4_IS_INTERRUPTED_BY_CH6_XF1	4	XF1 line of channel 6
DS2302_CH4_IS_INTERRUPTED_BY_CH1_XF1	4	XF1 line of channel 1

Predefined Symbol	Interrupted Channel	Interrupt Source
DS2302_CH5_IS_INTERRUPTED_BY_CH1_XF0	5	XF0 line of channel 1
DS2302_CH5_IS_INTERRUPTED_BY_CH2_XF0	5	XF0 line of channel 2
DS2302_CH5_IS_INTERRUPTED_BY_CH3_XF0	5	XF0 line of channel 3
DS2302_CH5_IS_INTERRUPTED_BY_CH4_XF0	5	XF0 line of channel 4
DS2302_CH5_IS_INTERRUPTED_BY_CH6_XF0	5	XF0 line of channel 6
DS2302_CH5_IS_INTERRUPTED_BY_CH6_XF1	5	XF1 line of channel 6
DS2302_CH5_IS_INTERRUPTED_BY_CH1_XF1	5	XF1 line of channel 1
DS2302_CH5_IS_INTERRUPTED_BY_CH2_XF1	5	XF1 line of channel 2
DS2302_CH6_IS_INTERRUPTED_BY_CH1_XF0	6	XF0 line of channel 1
DS2302_CH6_IS_INTERRUPTED_BY_CH2_XF0	6	XF0 line of channel 2
DS2302_CH6_IS_INTERRUPTED_BY_CH3_XF0	6	XF0 line of channel 3
DS2302_CH6_IS_INTERRUPTED_BY_CH4_XF0	6	XF0 line of channel 4
DS2302_CH6_IS_INTERRUPTED_BY_CH5_XF0	6	XF0 line of channel 5
DS2302_CH6_IS_INTERRUPTED_BY_CH1_XF1	6	XF1 line of channel 1
DS2302_CH6_IS_INTERRUPTED_BY_CH2_XF1	6	XF1 line of channel 2
DS2302_CH6_IS_INTERRUPTED_BY_CH3_XF1	6	XF1 line of channel 3

start_mode Specifies the start mode. The following symbols are predefined.

Predefined Symbol	Description
DS2302_START_CHANNELS_SYNC	The slave applications are loaded to the DSP and started synchronously. To start the slave applications synchronously, an INT2 interrupt is triggered.
DS2302_START_CHANNELS	The slave applications are loaded to the DSP and started afterwards.
DS2302_NOT_START_CHANNELS	The slave applications are only loaded to the DSP. They are not started.
DS2302_RESET_ALL_CHANNELS	This symbol must be combined by the logical OR operator with one of the symbols described above: e.g., DS2302_START_CHANNELS_SYNC | DS2302_RESET_ALL_CHANNELS. All the 6 DSP channels are reset before the specified applications are loaded. This symbol should be used if the DS2302 board is running old DSP applications from a preceding application load.

ch<n>_app Specifies the address of the object data consisting of an array in the generated SLC file. If no application has to be loaded to a DSP channel, you must specify the NULL symbol.

Return value

None

Example

The following example shows how to use the function. In the example, all 6 channels are reset and then the fgen and sin slave applications are loaded to

channel 1 and channel 2 of the DS2302 board and started synchronously afterwards. The **fgen** application (running on channel 1) can be interrupted by the **sin** application (running on channel 2) via the XF1 line. The **sin** application can be interrupted by the **fgen** application via the XF0 line.

```
#include Slv2302_fgen.slc
#include Slv2302_sin.slc
...
UInt32 int_mask
   ...
   int_mask = DS2302_CH1_IS_INTERRUPTED_BY_CH2_XF1 |
            DS2302_CH2_IS_INTERRUPTED_BY_CH1_XF0;
   ds2302_load_board(DS2302_1_BASE, int_mask,
            DS2302_START_CHANNELS_SYNC | DS2302_RESET_ALL_CHANNELS,
            fgen, sin, NULL, NULL, NULL, NULL);
```

Related topics

Basics

Overview of the Standard Slave Applications (DS2302 DSP Programming 🕮)

References

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

ds2302_load_channel

Syntax

```
void ds2302_load_channel(
   phs_addr_t base,
   UInt32 channel,
   unsigned long *appl_ptr)
```

Include file

`ds2302.h`

Purpose

To load a slave application to a channel.

Description

This function is called by the **ds2302_load_board** function to load slave applications to the channels of a DS2302 board.

Parameters

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

channel Specifies the bit mask with selected channel:

Symbol	Meaning
DS2302_CH1	For channel 1
DS2302_CH2	For channel 2
DS2302_CH3	For channel 3
DS2302_CH4	For channel 4
DS2302_CH5	For channel 5
DS2302_CH6	For channel 6

appl_ptr Specifies the address of the object data consisting of an array in the generated SLC file. For details of the SLC file, refer to ds2302_load_board on page 13.

Return value

None

Messages

The following messages are defined:

ID	Type	Message	Description
-520	Error	ds2302_load_channel(0x??): The application of channel %s requires a DS2302 board of revision 03 or higher!	To run the slave application, a DS2302 board with a higher revision is required.
-508 … -513	Error	ds2302_load_channel(0x??), ch. ??: Load error at offset 0x??	The slave application could not be loaded to the specified channel.

Related topics

References

Base Address of the I/O Board 7
ds2302_load_board 13

ds2302_control_channels

Syntax

```
void ds2302_control_channels(
   phs_addr_t base,
   UInt32 channel,
   UInt32 mode)
```

Include file

ds2302.h

Purpose

To reset or start channels.

Description

This function resets or starts the specified channels. You can start all slave applications synchronously.

Parameters

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

channel Specifies the bit mask with selected channel:

Symbol	Meaning
DS2302_CH1	For channel 1
DS2302_CH2	For channel 2
DS2302_CH3	For channel 3
DS2302_CH4	For channel 4
DS2302_CH5	For channel 5
DS2302_CH6	For channel 6

mode Specifies whether to start or reset the channels. The following symbols are predefined.

Predefined Symbol	Description
DS2302_START_CHANNELS_SYNC	The slave applications are loaded to the DSP and started synchronously. To start the slave applications synchronously, an INT2 interrupt is triggered.
DS2302_START_CHANNELS	The slave applications are loaded to the DSPs and started afterwards.
DS2302_RESET_CHANNELS	The channels are reset. This symbol should be used if the DS2302 board is running old DSP slave applications from a preceding application load.

Return value

None

Related topics

References

Base Address of the I/O Board....7
ds2302_init....10
ds2302_load_channel....16

ds2302_module_reset_on_ioerr

Syntax

```
void ds2302_module_reset_on_ioerr(
   phs_addr_t base,
   long value)
```

Include file

`ds2302.h`

Purpose

To specify the behavior of the module reset line depending on the PHS-bus I/O error line.

Description

The `value` parameter specifies the behavior of the module reset line depending on the PHS-bus I/O error line. If an I/O error occurs and `value` is `DS2302_RESET_ENABLE`, the module reset line is activated. If an I/O error occurs and `value` is `DS2302_RESET_DISABLE`, the module reset line will remain inactive.

Note

The `ds2302_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.

value Specifies the behaviour of the module reset line depending on the PHS-bus I/O error line:

Symbol	Meaning
`DS2302_RESET_DISABLE`	No module reset on I/O error
`DS2302_RESET_ENABLE`	Module reset on I/O error

Return value

None

Example

This example shows how to use this function:

```
void sub_fct()
{
   ...
   ds2302_module_reset_on_ioerr(DS2302_1_BASE,
      DS2302_RESET_ENABLE);
   ...
}
```

The 6 DS2302 modules will be reset on I/O error.

Related topics

Basics

Board Overview (as of Revision DS2302-04) (PHS Bus System Hardware Reference 🕮)

References

ds2302_get_board_type

Syntax

```
void ds2302_get_board_type(
   phs_addr_t base,
   long *board_type)
```

Include file

`ds2302.h`

Purpose

To detect the type of DS2302 board.

Description

The `board_type` parameter returns the type of the DS2302 board specified by the `base` parameter.

Note

The `ds2302_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.

board_type Specifies the type number of the detected DS2302 board:

Symbol	Meaning
230201	DS2302 board with 60 MHz DSP's

Return value

None

Example

This example shows how to use this function:

```
void sub_fct()
{
   long type
   ...
   ds2302_get_board_type(DS2302_1_BASE, &type);
   ...
}
```

The type of the DS2302 board at the PHS-bus address **DS2302_1_BASE** is read and stored in the variable type.

Related topics

References

Base Address of the I/O Board....7
ds2302_init....10

ds2302_phspp_init

Syntax

```
void ds2302_phspp_init(
      phs_addr_t base,
      UInt32 mode)
```

Include file

ds2302.h

Purpose

To set the PHS bus from PHS++ mode to PHS mode.

Description

The DS2302 (starting from board revision DS2302-04) provides a PHS++ interface. The board is automatically set to PHS++ mode after initialization. In

PHS++ mode the DS2302 board uses a different PHS-bus register mapping. Software modules, which are written for the DS2302-01 board, like custom access functions or s-Functions, do not work properly. In this case you can set the DS2302-04 board to PHS mode, to avoid problems with these software modules.

Parameters

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

mode Specifies the PHS mode. The following modes are available:

Predefined Symbol	Value	Meaning
DS2302_PHS_MODE	0	PHS mode
DS2302_PHSPP_MODE	1	PHS++ mode

Return value

None

Example

```
#include <ds2302.h>
int main(void)
{
   init();
   ds2302_init(DS2302_1_BASE);  /* initialize DS2302 board */
   ds2302_phspp_init(DS2302_1_BASE, DS2302_PHS_MODE); /* set PHS mode */
   ...
}
```

Related topics

Basics

Architectural Overview (DS2302 DSP Programming 🕮)

References

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

ds2302_dsp_int3_init

Syntax

```
void ds2302_dsp_int3_init(
   phs_addr_t base,
   UInt32 mode)
```

Include file

ds2302.h

Purpose

To enable or disable the INT3 interrupt generation of all the channels.

Parameters

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

mode Specifies whether the INT3 interrupt generation is enabled or disabled, see the following predefined symbols.

Symbol	Description
`DS2302_INT3_DISABLE`	Disables INT3 interrupt generation
`DS2302_INT3_ENABLE`	Enables INT3 interrupt generation

Return value

None

Related topics

References

Memory Access Functions

Introduction

To exchange data between the main application and an application running on a DSP.

Where to go from here

Information in this section

Basics of Memory Access

Introduction

The DS2302 is equipped with DSPs designed for fast and flexible waveform generation. It computes each signal sample just-in-time and outputs it immediately. The following functions allow to exchange data between your main application and an application running on one of the six DSPs. The communication is established via the dual-ported memories of the DDS board.

Note

You have to initialize the DS2302 board with the `ds2302_init` function before you can use one of these functions.

Floating-point conversion

For processor boards (DS1006 and DS1007) a different floating-point format is used. The processor boards use the IEEE floating-point format whereas the slave DSP uses the TI floating-point format. Therefore, floating-point values have to be converted with the `RTLIB_CONV_FLOAT32_TO_IEEE32` or `RTLIB_CONV_FLOAT32_FROM_IEEE32` conversion macros.

Related topics

References

ds2302_init..... 10

ds2302_read

Syntax

```
void ds2302_read(
   phs_addr_t base,
   int channel,
   long offs,
   void *value)
```

Include file

`ds2302.h`

Purpose

To read data from the DS2302's dual-port memory.

Description

The 32-bit parameter `value` returns the contents of the DS2302s dual-port memory location specified by the `base`, `channel` and `offs` parameters.

Note

- The `channel` parameter must be specified as a 6-bit mask. It is recommended to use the channel definitions from the `ds2302.h` header file. Only a single channel can be read at a time.
- The `ds2302_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channel:

Symbol	Meaning
`DS2302_CH1`	For channel 1
`DS2302_CH2`	For channel 2
`DS2302_CH3`	For channel 3
`DS2302_CH4`	For channel 4
`DS2302_CH5`	For channel 5
`DS2302_CH6`	For channel 6

offs Specifies the dual-port memory source address within the range of 0x0000 … 0x3FFF.

value Specifies the contents of the specified memory location.

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
   long count;
   ...
   ds2302_read(DS2302_1_BASE, DS2302_CH1, 0, &count);
   ...
}
```

The contents of the dual-port memory location at offset 0 is read and written into the variable count.

Related topics

References

ds2302_read_block

Syntax

```
void ds2302_read_block(
   phs_addr_t base,
   int channel,
   long offs,
   long count,
   void *data)
```

Include file

`ds2302.h`

Purpose

To read a data block from the DS2302's dual-port memory.

Description

A block of count 32-bit words is read from the DS2302's dual-port memory specified by `base` and `channel` and starting at address `offs`. The `data` pointer must point to the destination data block where the data shall be stored. The destination data block must be allocated by the calling program.

Note

- The `channel` parameter must be specified as a 6-bit mask. It is recommended to use the channel definitions from the `ds2302.h` header file. Only a single channel can be read at a time.
- The `ds2302_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channel:

Symbol	Meaning
DS2302_CH1	For channel 1
DS2302_CH2	For channel 2
DS2302_CH3	For channel 3
DS2302_CH4	For channel 4
DS2302_CH5	For channel 5
DS2302_CH6	For channel 6

offs Specifies the dual-port memory start address within the range of 0x0000 … 0x3FFF.

count Specifies the number of data words to be read.

data Specifies the source data array.

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
   float x[8];
   ...
   ds2302_read_block(DS2302_1_BASE, DS2302_CH1, 0, 8, x);
}
```

The contents of the dual-port memory of channel 1 from address 0x000 to 0x007 is read and written into the array `x[]`.

Related topics

References

Base Address of the I/O Board 7
ds2302_init 10
ds2302_read_block_float 30

ds2302_read_block_float

Syntax

```
void ds2302_read_block_float(
   phs_addr_t base,
   int channel,
   long offs,
   long count,
   dsfloat *data)
```

Include file

`ds2302.h`

Purpose

To read a data block of float type from the DS2302's dual-port memory.

Description

A block of count 32-bit words is read from the DS2302's dual-port memory specified by `base` and `channel` and starting at `offs` address. The `data` pointer must point to the destination data block where the data shall be stored. The destination data block must be allocated by the calling program.

Implicitly the function converts the value from TI to IEEE754 floating point format.

Note

- The `channel` parameter must be specified as a 6-bit mask. It is recommended to use the channel definitions from the `ds2302.h` header file. Only a single channel can be read at a time.
- The `ds2302_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channel:

Symbol	Meaning
`DS2302_CH1`	For channel 1
`DS2302_CH2`	For channel 2
`DS2302_CH3`	For channel 3
`DS2302_CH4`	For channel 4

Symbol	Meaning
DS2302_CH5	For channel 5
DS2302_CH6	For channel 6

offs Specifies the dual-port memory start address within the range of 0x0000 … 0x3FFF.

count Specifies the number of data words to be read.

data Specifies the source data array.

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
   dsfloat x[8];
   ds2302_read_block_float(DS2302_1_BASE, DS2302_CH1, 0, 8, x);
   ...
}
```

The contents of the dual-port memory of channel 1 from address 0x000 to 0x007 is read and written into the array `x[]`.

Related topics

References

Base Address of the I/O Board....7
ds2302_init....10
ds2302_read_block....28

ds2302_read_float

Syntax

```
void ds2302_read_float(
   phs_addr_t base,
   int channel,
   long offs,
   dsfloat *value)
```

Include file

`ds2302.h`

Purpose

To read data of float type from the DS2302's dual-port memory.

Description

The `value` parameter returns the contents of the DS2302s dual-port memory location specified by the `base`, `channel` and `offs` parameters.

Implicitly the function converts the value from TI to IEEE754 floating point format.

Note

- The `channel` parameter must be specified as a 6-bit mask. It is recommended to use the channel definitions from the `ds2302.h` header file. Only a single channel can be read at a time.
- The `ds2302_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channel:

Symbol	Meaning
`DS2302_CH1`	For channel 1
`DS2302_CH2`	For channel 2
`DS2302_CH3`	For channel 3
`DS2302_CH4`	For channel 4
`DS2302_CH5`	For channel 5
`DS2302_CH6`	For channel 6

offs Specifies the dual-port memory source address within the range of 0x0000 … 0x3FFF.

value Specifies the contents of the specified memory location.

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
   dsfloat count;
   ...
   ds2302_read(DS2302_1_BASE, DS2302_CH1, 0, &count);
   ...
}
```

The contents of the dual-port memory location at offset 0 is read and written into the `count` variable.

Related topics

References

ds2302_read_msg

Syntax

```
void ds2302_read_msg(
   phs_addr_t base,
   UInt32 channel)
```

Include file

`ds2302.h`

Purpose

To read messages, sent from a DS2302 slave-DSP for debugging purposes.

Description

This function reads messages, sent from a DS2302 slave-DSP for debugging purposes. The messages are transferred to the master processor boards message module and can be observed using the **Message Viewer** of the experiment software. This function looks for new messages in the Dual-Port Memory of the specified DSP channel and prints all new messages read since the last call to `ds2302_read_msg()`. This function should be called in the background of your application.

Note

When using messages the dual-port memory offset range 0x3A00 … 0x3F7F must not be used by the application.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channels:

Symbol	Meaning
DS2302_CH1	For channel 1
DS2302_CH2	For channel 2
DS2302_CH3	For channel 3
DS2302_CH4	For channel 4
DS2302_CH5	For channel 5
DS2302_CH6	For channel 6
DS2302_ALL_CH	For all 6 channels

The definitions can be combined using the logical OR operation.

Return value

None

Example

This example shows how to use the function:

```
int main(void)
{
...
  /* background loop */
  while(1)
  {
    RTLIB_BACKGROUND_SERVICE();
    /* read messages from channel 1 */
    ds2302_read_msg(DS2302_1_BASE, DS2302_CH1);
  }
}
```

In the Message Viewer you receive for example the following:

```
ds1006: 18:19:46 [#5] ds1006 - RTLIB: DSP #1: Application
started. (1)(0xCA)
```

Related topics

Basics

Basics of Memory Access..... 26

References

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

ds2302_write

Syntax

```
void ds2302_write(
   phs_addr_t base,
   int channel,
   long offs,
   void *value)
```

Include file

`ds2302.h`

Purpose

To write data to the DS2302's dual-port memory.

Description

The 32-bit parameter **value** is written to the DS2302's dual-port memory location specified by the **base**, **channel**, and **offs** parameters.

The channels to be written are selected by a 6-bit mask. This allows writing to multiple channels with a single write operation. You can use the channel definitions from the **ds2302.h** header file.

Note

The **ds2302_init** function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channels:

Symbol	Meaning
DS2302_CH1	For channel 1
DS2302_CH2	For channel 2
DS2302_CH3	For channel 3
DS2302_CH4	For channel 4
DS2302_CH5	For channel 5
DS2302_CH6	For channel 6
DS2302_ALL_CH	For all 6 channels

The definitions can be combined using the logical OR operation.

offs Specifies the dual-port memory source address within the range of 0x0000 … 0x3FFF.

value Specifies the data value to be written to the dual-port memory (can be either float or long).

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
   float freq = 100.5;
   ...
   ds2302_write(DS2302_1_BASE, DS2302_CH1, 0x120, &freq);
   ...
}
```

The contents of the variable freq is written into the dual-port memory location at offset 0x120.

Related topics

References

Base Address of the I/O Board 7
ds2302_init 10
ds2302_write_block 36
ds2302_write_block_float 38
ds2302_write_float 40

ds2302_write_block

Syntax

```
void ds2302_write_block(
   phs_addr_t base,
   int channel,
   long offs,
   long count,
   void *data)
```

Include file

`ds2302.h`

Purpose

To write a data block to the DS2302's dual-port memory.

Description

A block of count 32-bit words is written to the DS2302's dual-port memory specified by `base` and `channel` and starting at address `offs`. The `data` pointer must point to the source data block.

Note

The `ds2302_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channels:

Symbol	Meaning
`DS2302_CH1`	For channel 1
`DS2302_CH2`	For channel 2
`DS2302_CH3`	For channel 3
`DS2302_CH4`	For channel 4
`DS2302_CH5`	For channel 5
`DS2302_CH6`	For channel 6
`DS2302_ALL_CH`	For all 6 channels

The definitions can be combined using the logical OR operation.

offs Specifies the dual-port memory start address within the range of 0x0000 … 0x3FFF.

count Specifies the number of data words to be written.

data Specifies the data value to be written to the dual-port memory (can be either float or long).

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
   float x[8] = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0};
   ...
   ds2302_write_block(DS2302_1_BASE,
   DS2302_CH1 | DS2302_CH2 | DS2302_CH3, 0, 8, x);
   ...
}
```

The contents of array x is written to the dual-port memory of channels 1, 2, and 3 starting at address 0x0000.

Related topics

References

ds2302_write_block_float

Syntax

```
void ds2302_write_block_float(
   phs_addr_t base,
   int channel,
   long offs,
   long count,
   dsfloat *data)
```

Include file

`ds2302.h`

Purpose

To write a data block of float type to the DS2302's dual-port memory.

Description

A block of `count` 32-bit words is written to the DS2302's dual-port memory specified by `base` and `channel` and starting at `offs` address. The `data` pointer must point to the source data block.

Implicitly the function converts the value from IEEE754 to TI floating point format.

Note

The `ds2302_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channels:

Symbol	Meaning
`DS2302_CH1`	For channel 1
`DS2302_CH2`	For channel 2
`DS2302_CH3`	For channel 3
`DS2302_CH4`	For channel 4
`DS2302_CH5`	For channel 5
`DS2302_CH6`	For channel 6
`DS2302_ALL_CH`	For all 6 channels

The definitions can be combined using the logical OR operation.

offs Specifies the dual-port memory start address within the range of 0x0000 … 0x3FFF.

count Specifies the number of data words to be written.

data Specifies the data value to be written to the dual-port memory (can be either float or long).

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
   dsfloat x[8] = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0};
   ds2302_write_block_float(DS2302_1_BASE,
      DS2302_CH2 | DS2302_CH3, 0, 8, x);
...
}
```

The contents of array x is written to the dual-port memory of channels 2 and 3 starting at address 0x0000.

Related topics

References

ds2302_write_float

Syntax

```
void ds2302_write_float(
   phs_addr_t base,
   int channel,
   long offs,
   dsfloat *value)
```

Include file

`ds2302.h`

Purpose

To write data of float type to the DS2302's dual-port memory.

Description

The `value` parameter is written to the DS2302's dual-port memory location specified by the `base`, `channel`, and `offs` parameters.

Implicitly the function converts the value from IEEE754 to TI floating point format.

The channels to be written are selected by a 6-bit mask. This allows writing to multiple channels with a single write operation. You may use the channel definitions from the `ds2302.h` header file.

Note

The `ds2302_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channels:

Symbol	Meaning
DS2302_CH1	For channel 1
DS2302_CH2	For channel 2
DS2302_CH3	For channel 3
DS2302_CH4	For channel 4
DS2302_CH5	For channel 5
DS2302_CH6	For channel 6
DS2302_ALL_CH	For all 6 channels

The definitions can be combined using the logical OR operation.

offs Specifies the dual-port memory source address within the range of 0x0000 … 0x3FFF.

value Specifies the data value to be written into dual-port memory.

Return value

None

Example

This example shows how to use the function:

```
void sub_fct()
{
  dsfloat freq = 100.5;
  ...
  ds2302_write_float(DS2302_1_BASE, DS2302_CH1, 0x120, &freq);
  ...
}
```

The contents of the `freq` variable is written into the dual-port memory location at offset 0x120.

Related topics

References

Base Address of the I/O Board 7
ds2302_init 10
ds2302_write 35
ds2302_write_block 36
ds2302_write_block_float 38

Interrupt Functions

Introduction

The following function can be use to program the interrupt for the DS2302.

Note

You have to initialize the DS2302 board with the `ds2302_init` function before you can use one of these functions.

Where to go from here

Information in this section

ds2302_INT3

Syntax

```
void ds2302_INT3(
   phs_addr_t base,
   int channel,
   UInt32 value)
```

Include file `ds2302.h`

Purpose To request the DS2302 INT3 interrupt.

Description The `value` parameter is written into the dual-port memory location 0x3FFF (address 0x403FFF as seen by the DSP) of the DS2302 channels specified by the `base` and `channel` parameters. This requests an interrupt INT3 on the specified DSPs. The dual-port memory location 0x3FFF is reserved for INT3 interrupts and cannot be used for other purposes.

The channels to be written are selected by a 6-bit mask. This allows writing to multiple channels with a single write operation. Use the channel definitions from the `ds2302.h` header file or a combination of them.

The data value being written to the dual-port memory location 0x3FFF can be arbitrary chosen. It can be read by the respective DSP(s) and may be used to implement an interrupt driven data transfer.

Note

The `ds2302_init` function must be called before this function can be used.

I/O mapping For details on the I/O mapping, refer to Mapping of I/O Signals (PHS Bus System Hardware Reference 🕮).

Parameters

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

channel Specifies the bit mask with selected channels:

Symbol	Meaning
`DS2302_CH1`	For channel 1
`DS2302_CH2`	For channel 2
`DS2302_CH3`	For channel 3
`DS2302_CH4`	For channel 4
`DS2302_CH5`	For channel 5
`DS2302_CH6`	For channel 6
`DS2302_ALL_CH`	For all 6 channels

The definitions can be combined using the logical OR operation.

value Specifies the data value to be written into DS2302 dual-port memory location 0x3FFF.

Return value

None

Example

This example shows how to use this function:

```
void sub_fct()
{
   ...
   ds2302_INT3(DS2302_1_BASE, DS2302_CH1 | DS2302_CH2, 33);
   ...
}
```

An INT3 interrupt is requested on the first two channels of the DS2302 by writing 33 into the dual-port memory location 0x3FFF of channels 1 and 2.

Related topics

References

Base Address of the I/O Board 7
ds2302_init 10
ds2302_read_isr 45
ds2302_write_isr 47

ds2302_read_isr

Syntax

```
void ds2302_read_isr(
   phs_addr_t base,
   long *value)
```

Include file

`ds2302.h`

Purpose

To read the interrupt select register (ISR) of the DS2302 board.

Description

The `value` parameter returns the contents of the 18-bit interrupt select field in the DS2302's interrupt select register (ISR), i.e. bits D14 … D31. Bits D0 … D13 are read as zeros.

Note

The `ds2302_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.

value Specifies the contents of the ISR register within the range of 0x00000000 … 0xFFFC0000.

Return value

None

Example

For an example, refer to ds2302_write_isr on page 47.

Related topics

References

ds2302_speedchk

Syntax

```
void ds2302_speedchk(
   phs_addr_t base,
   dsfloat *exec_min,
   dsfloat *exec_max)
```

Include file

`ds2302.h`

Purpose

To get the execution time of the DS2302 timer interrupt service routine.

Description

The minimum and maximum execution time values of the 6 DS2302 board channels are stored in arrays pointed to by `exec_min` and `exec_max`.

The applications of the DS2302 board channels to be measured must execute the `speed_check()` macro.

The execution time values of unused channels or channels which are not executing the `speed_check()` macro are set to 0.0.

Note

The `ds2302_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.

exec_min Specifies the array for the minimum execution times of all 6 DS2302 board channels (in µs).

exec_max Specifies the array for the maximum execution times of all 6 DS2302 board channels (in µs).

Return value

None

Example

This example shows how to use this function:

```
void sub_fct()
{
   dsfloat exec_min[6];
   dsfloat exec_max[6];
   ...
   ds2302_speedchk(DS2302_1_BASE, exec_min, exec_max);
   ...
}
```

The execution times of the 6 DS2302 channels timer interrupt service routines are read. The minimum execution times are stored in the `exec_min` array and the maximum execution times are stored in the `exec_max` array. The values are given in µs.

Array element 0 contains the value of channel 1, element 1 contains the value of channel 2 and so on.

Related topics

References

Base Address of the I/O Board 7
ds2302_init 10

ds2302_write_isr

Syntax

```
void ds2302_write_isr(
   phs_addr_t base,
   UInt32 value)
```

Include file

`ds2302.h`

Purpose

To write data to the DS2302's interrupt select register (ISR).

Description

The parameter `value` is written to the 18-bit interrupt select field in the DS2302's interrupt select register (ISR), i.e. bits D14 … D31. Bits D0 … D13 remain unchanged.

Note

- The `ds2302_write_isr` function does not shift the bits to be written into position. You must place the bits into the proper positions.
- The `ds2302_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.

value Specifies the data value to be written to the ISR register within the range 0x00000000 … 0xFFFC0000.

Return value

None

Example

This example shows how to use this function:

```
#define ISR1_MASK 0x0001C000
void sub_fct()
{
   UInt32 isr;
   ...
   ds2302_read_isr(DS2302_1_BASE, &isr);
   isr &= ~ISR1_MASK;
   isr |= (0x04 << 14);
   ds2302_write_isr(DS2302_1_BASE, isr);
   ...
}
```

The current status of the interrupt select register is read. The 3 bits of the channel 1 interrupt select field are set to the new value 0x04 without changing the interrupt select fields of the remaining channels. This selects the XF0 line of channel 5 as the interrupt source for the INT1 interrupt of channel 1.

Related topics

References

Base Address of the I/O Board 7
ds2302_init 10
ds2302_read_isr 45

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<xxxx> 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

Measured Execution Times

Initialization functions

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

Function	Execution Time	
	DS1006 with 2.6 GHz	DS1006 with 3.0 GHz
ds2302_init	90.20 µs	113.93 µs
ds2302_get_board_type	0.02 µs	0.70 µs
ds2302_dsp_reset_on_ioerr	0.74 µs	0.80 µs
ds2302_module_reset_on_ioerr	0.75 µs	0.80 µs
ds2302_phspp_init	0.11 µs	0.14 µs

Memory access functions

The following execution time has been measured for the memory access functions:

Function	Execution Time	
	DS1006 with 2.6 GHz	DS1006 with 3.0 GHz
ds2302_read	2.11 µs	2.16 µs
ds2302_read_float	2.11 µs	2.16 µs
ds2302_read_block	1.382 + n * 0.706 µs	1.431 + n * 0.718 µs
ds2302_read_block_float	1.396 + n * 0.718 µs	1.435 + n * 0.726 µs
ds2302_write	0.75 µs	0.81 µs
ds2302_write_float	0.76 µs	0.82 µs
ds2302_write_block	0.415 + n * 0.349 µs	0.463 + n * 0.349 µs
ds2302_write_block_float	0.425 + n * 0.351 µs	0.479 + n * 0.349 µs

Interrupt functions

The following execution time has been measured for the interrupt functions:

Function	Execution Time DS1006 with 2.6 GHz	 DS1006 with 3.0 GHz
`ds2302_INT3`	0.75 µs	0.81 µs
`ds2302_read_isr`	0.71 µs	0.70 µs
`ds2302_write_isr`	0.74 µs	0.79 µs
`ds2302_speedchk`	17.40 µs	17.81 µs

Related topics

References

Information on the Test Environment....49

B

C

D

E

F

I

L

M

T