DS2201 Multi-I/O 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

Introduction

This RTLib Reference (Real-Time Library) gives detailed descriptions of the C functions needed to program a DS2201 Multi-I/O 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`

Board Initialization

Introduction

Before you can use the DS2201, you have to perform the initialization process.

ds2201_init

Syntax

```
void ds2201_init(phs_addr_t base)
```

Include file

`ds2201.h`

Purpose

To initialize the DS2201.

Description

All DS2201 registers are initialized to default values:

- Output mode of the D/A converters is set to transparent (see also ds2201_set_outmode on page 23).
- I/O error mode of the D/A converters is set to ’reset output to zero on I/O error’ (see also ds2201_set_errmode on page 22).
- All data registers of the D/A converters are set to zero.

Note

- This function must be called before any other DS2201 function can be used.
- The initialization function of the processor board must be called before the DS2201’s initialization function.

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

Return value

None

Messages

The following messages are defined:

ID	Type	Message	Description
201	Error	ds2201_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.
-161	Error	ds2201_init(0x??): Board not found!	No DS2201 board could be found at the specified PHS-bus address. Check whether the DSxxxx_n_BASE macro corresponds to the I/O board used.
-53	Warning	ds2201_init(0x??): Jumper setting is not matching SW default initialization! STP register: 0x???????? instead of 0x????????	The value of the STP register could not be verified successfully. The DS2201 jumper setting may be not correct. Remove all jumpers.

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main(void)
{
   init();
   ds2201_init(DS2201_1_BASE);
   …
}
```

The DS2201 at address `DS2201_1_BASE` is initialized.

Related topics

References

Base Address of the I/O Board........ 9
ds2201_set_errmode........ 22
ds2201_set_outmode........ 23
Macros........ 9

ADC Unit

Introduction

The following functions are used to access the A/D converters of the DS2201 Multi-I/O Board.

Note

You have to initialize the DS2201 with the `ds2201_init` function before you can use one of these functions.

Where to go from here

Information in this section

Information in other sections

Analog/Digital Conversion (DS2201 Features 📖)
Analog/digital conversion (ADC) is an element of most applications in rapid control prototyping and hardware-in-the-loop simulation.

ds2201_adc_block_in

Syntax

```
void ds2201_adc_block_in(
   phs_addr_t base,
   int count,
   dsfloat *data)
```

Include file

`ds2201.h`

Purpose

To read blocks of channels from the A/D converters.

Description

When the conversion of A/D converter 1 is complete, this function reads the ADC input channels in the order 1 … `count`, converts the input values to floating-point values within the range -1.0 … +1.0, and copies them to the data buffer. Because it is mandatory to read all 4 input channels being served by a single A/D converter, whether or not all 4 input values are actually needed, multiples of 4 channels are read and copied to the data buffer.

> **Note**
>
> Before you can use this function, you must call `ds2201_init` to initialize the board, and `ds2201_adc_start` to start all required A/D converters. If you have not started all of the specified A/D converters the application will hang up.

I/O mapping

For details on the I/O mapping, refer to Analog/Digital Conversion (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

count Specifies the number of input channels to be read within the range 1 … 20.

data Specifies the pointer to the destination data array. The data array must be allocated by the calling program. Its length must be equal to or greater than 4 · ((count + 3) div 4):

Count	Array Length
1 … 4	4
5 … 8	8
9 … 12	12

Count	Array Length
12 … 16	16
17 … 20	20

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
dsfloat adc_value[12];
void sub_fct(void)
{
   ...
   ds2201_adc_start(
      DS2201_1_BASE, DS2201_ADC_GR1 | DS2201_ADC_GR2 | DS2201_ADC_GR3);
   ds2201_adc_block_in(DS2201_1_BASE, 10, adc_value);
   ...
}
```

The ADC values of channels 1 … 10 are read and stored to the `adc_value` array.

Related topics

Basics

ADC Unit (DS2201 Features 📖)

References

Base Address of the I/O Board 9
ds2201_adc_in 15
ds2201_adc_read 17
ds2201_adc_start 18
ds2201_init 11

ds2201_adc_in

Syntax

```
dsfloat ds2201_adc_in(
   phs_addr_t base,
   int channel)
```

Include file

`ds2201.h`

Purpose

To read values from the A/D converters.

Description

After the conversion has finished, this function reads the input value of the specified converter channel and scales it to a floating-point value within the range –1.0 … +1.0.

> **Note**
>
> Before you can use this function, you must call `ds2201_init` to initialize the board, and `ds2201_adc_start` to start all required A/D converters. If you have not started all of the specified A/D converters the application will hang up.

I/O mapping

For details on the I/O mapping, refer to Analog/Digital Conversion (DS2201 Features).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

channel Specifies the channel number within the range 1 … 20.

Return value

Returns the ADC value within the range –1.0 … +1.0.

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to read the ADC value of channel 1:

```
void sub_fct(void)
{
   dsfloat adc_value;

   ds2201_adc_start(DS2201_1_BASE, DS2201_ADC_GR1);
   adc_value = ds2201_in(DS2201_1_BASE, 1);
   ...
}
```

Related topics

Basics

ADC Unit (DS2201 Features 📖)

References

Base Address of the I/O Board	9
ds2201_adc_block_in	14
ds2201_adc_read	17
ds2201_adc_start	18
ds2201_init	11

ds2201_adc_read

Syntax

```
void ds2201_adc_read(
   phs_addr_t base,
   int converter,
   dsfloat *data)
```

Include file

`ds2201.h`

Purpose

To read values from the A/D converters immediately.

Description

This function reads the input value of the 4 channels of the specified converter without waiting for the end-of-conversion flag, scales them to floating-point values within the range –1.0 … +1.0, and copies them to the data buffer. To consider the end of conversion, this function must be used in an interrupt service routine.

Note

Before you can use this function, you must call **`ds2201_init`** to initialize the board, and **`ds2201_adc_start`** to start all required A/D converters.

I/O mapping

For details on the I/O mapping, refer to Analog/Digital Conversion (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

converter Specifies the converter number within the range 1 … 5.

data Specifies the pointer to the destination data array. Its length must be equal to or greater than 4.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to read the A/D converter 1 (channel 1 … 4) in an interrupt service routine:

```
float adc_value[4];
void adc_service(void)
{
   ds2201_adc_read(DS2201_1_BASE, 1, adc_value);
}
```

Related topics

Basics

ADC Unit (DS2201 Features 📖)

References

Base Address of the I/O Board 9
ds2201_adc_block_in 14
ds2201_adc_in 15
ds2201_adc_start 18
ds2201_init 11

ds2201_adc_start

Syntax

```
void ds2201_adc_start(
   phs_addr_t base,
   long mask)
```

Include file

`ds2201.h`

Purpose

To start the A/D conversion of the specified converters.

Note

The `ds2201_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Analog/Digital Conversion (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

mask Specifies the A/D converters that should be started. You can combine the following definitions using the logical OR operation:

Value	Meaning
`DS2201_ADC_GR1`	For channels 1 … 4
`DS2201_ADC_GR2`	For channels 5 … 8
`DS2201_ADC_GR3`	For channels 9 … 12
`DS2201_ADC_GR4`	For channels 13 … 16
`DS2201_ADC_GR5`	For channels 17 … 20
`DS2201_ADC_GR_ALL`	For all channels

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
ds2201_adc_start(DS2201_1_BASE,
         DS2201_ADC_GR1 | DS2201_ADC_GR3);
```

The A/D conversion is started for converter 1 (channels 1 … 4) and converter 3 (channels 9 … 12).

Related topics

Basics

ADC Unit (DS2201 Features 🕮)

References

Base Address of the I/O Board 9
ds2201_adc_block_in 14
ds2201_adc_in 15
ds2201_adc_read 17
ds2201_init 11

DAC Unit

Introduction

The following functions are used to access the D/A converters of the DS2201 Multi-I/O Board.

Note

You have to initialize the DS2201 with the `ds2201_init` function before you can use one of these functions.

Where to go from here

Information in this section

Information in other sections

Digital/Analog Conversion (DS2201 Features 🕮)
Digital/analog conversion (DAC) is required by many control applications to provide the control signal for actuators. In hardware-in-the-loop applications, sensors that provide analog signals have to be simulated.

ds2201_set_errmode

Syntax

```
void ds2201_set_errmode(
   phs_addr_t base,
   int group,
   int errmode)
```

Include file

`ds2201.h`

Purpose

To select the I/O error mode of a D/A converter group.

Description

Another board can activate the I/O error signal of the PHS bus. If the I/O error mode of a D/A converter group is set to 'reset output to zero on I/O error', the DS2201 reacts to the PHS-bus error signal by clearing the DAC outputs until a new output value is written to the channel. This is the default setting of the I/O error mode. If the I/O error mode is set to 'keep output voltage on I/O error', there is no reaction to a PHS-bus error signal.

The error mode can be set for each of the 2 D/A converter groups individually.

> **Note**
>
> The `ds2201_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Digital/Analog Conversion (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

group Specifies the D/A converter group for the error mode setting:

D/A Converter Group	Meaning
DS2201_GROUP1	Sets the specified error mode for channels 1 … 4
DS2201_GROUP2	Sets the specified error mode for channels 5 … 8
DS2201_GROUP_ALL	Sets the specified error mode for all channels

errmode Specifies the I/O error mode:

Value	Meaning
DS2201_ZERO	Reset DAC outputs to zero on I/O error
DS2201_KEEP	Keep DAC output voltage on I/O error

Return value

None

Messages

The following messages are defined:

ID	Type	Message	Description
-50	Error	ds2201_set_errmode(0x??): Board not initialized!	The DS2201 has not been initialized by a preceding call to the `ds2201_init` function.
-54	Error	ds2201_set_errmode(0x??): Specified value is not matching jumper setup! STP register: 0x???????? instead of 0x????????	The value of the STP register could not be verified because the DS2201 jumper setting is not correct.

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
ds2201_set_errmode(DS2201_1_BASE, 2, DS2201_KEEP);
```

The D/A converter group 2, comprising channels 5 … 8, is set to 'keep output voltage on I/O error'.

Related topics

References

Base Address of the I/O Board 9
Digital/Analog Conversion (DS2201 Features 🕮)
ds2201_dac_load 27
ds2201_dac_out 25
ds2201_dac_strobe 28
ds2201_init 11
ds2201_set_outmode 23

ds2201_set_outmode

Syntax

```
void ds2201_set_outmode(
   phs_addr_t base,
   int group,
   int outmode)
```

Include file

`ds2201.h`

Purpose

To select the output mode of a D/A converter group.

Description

This function can be used to change the output mode settings. Either transparent output mode or latched output mode can be selected. If transparent output mode is selected, the output delay between individual channels depends on the actual output sequence. Latched output mode can be used to update all outputs at the same time using the `ds2201_dac_strobe` function.

The output mode can be set for each of the 2 D/A converter groups individually.

For further information on the output mode, refer to Digital/Analog Conversion (DS2201 Features 🕮).

Note

The `ds2201_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Digital/Analog Conversion (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

group Specifies the D/A converter group for the output mode setting:

D/A Converter Group	Meaning
`DS2201_GROUP1`	Sets the specified output mode for channels 1 … 4
`DS2201_GROUP2`	Sets the specified output mode for channels 5 … 8
`DS2201_GROUP_ALL`	Sets the specified output mode for all channels

outmode Specifies the output mode:

Value	Meaning
`DS2201_TRANS`	Transparent output mode
`DS2201_LATCH`	Latched output mode

Return value

None

Messages

The following messages are defined:

ID	Type	Message	Description
-50	Error	ds2201_set_outmode(0x??): Board not initialized!	The DS2201 has not been initialized by a preceding call to the `ds2201_init` function.
-54	Error	ds2201_set_outmode(0x??): Specified value is not matching jumper setup! STP register: 0x???????? instead of 0x????????	The value of the STP register could not be verified because the DS2201 jumper setting is not correct.

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
ds2201_set_outmode(DS2201_1_BASE, DS2201_GROUP1, DS2201_LATCH);
```

The D/A converter group 1 (channels 1 … 4) is set to latched output mode.

Related topics

References

Base Address of the I/O Board........ 9
Digital/Analog Conversion (DS2201 Features 🕮)
ds2201_dac_load........ 27
ds2201_dac_out........ 25
ds2201_dac_strobe........ 28
ds2201_init........ 11
ds2201_set_errmode........ 22

ds2201_dac_out

Syntax

```
void ds2201_dac_out(
   phs_addr_t base,
   int channel,
   dsfloat value)
```

Include file

`ds2201.h`

Purpose

To write analog output to the D/A converters.

Description

The `ds2201_dac_out` function can be used for immediate data output to a DAC channel of a DS2201.

> **Note**
>
> The `ds2201_init` function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Digital/Analog Conversion (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

channel Specifies the logical channel number in the range 1 … 8.

value Specifies the DAC output value within the range –1.0 … +1.0. The output value is scaled to an output voltage within the range -10 V … +10 V.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
ds2201_dac_out(DS2201_1_BASE, 1, 0.5);
```

The output of channel 1 is set to 5 V voltage.

Related topics

References

Base Address of the I/O Board.......... 9
Digital/Analog Conversion (DS2201 Features 📖)
ds2201_dac_load.......... 27
ds2201_dac_strobe.......... 28
ds2201_init.......... 11
ds2201_set_errmode.......... 22
ds2201_set_outmode.......... 23

ds2201_dac_load

Syntax

```
void ds2201_dac_load(
   phs_addr_t base,
   int channel,
   dsfloat value)
```

Include file

`ds2201.h`

Purpose

To write strobed analog output to the D/A converters.

Description

This function loads the output value to the converter. The value becomes valid after calling the `ds2201_dac_strobe` function. In this way, this function can be used to provide synchronous output of the DS2201 DAC channels.

Note

- The `ds2201_init` function must be called before this function can be used.
- The channel must have been initialized to strobed output mode using the `ds2201_set_outmode` function.

I/O mapping

For details on the I/O mapping, refer to Digital/Analog Conversion (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

channel Specifies the logical channel number in the range 1 … 8.

value Specifies the DAC output value within the range –1.0 … +1.0. The output value is scaled to an output voltage within the range -10 V … +10 V.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
ds2201_dac_load(DS2201_1_BASE, 1, 0.5);
```

The output of channel 1 is loaded for 5 V output voltage.

Related topics

References

Base Address of the I/O Board.. 9
Digital/Analog Conversion (DS2201 Features 🕮)
ds2201_dac_out.. 25
ds2201_dac_strobe.. 28
ds2201_init.. 11
ds2201_set_errmode.. 22
ds2201_set_outmode.. 23

ds2201_dac_strobe

Syntax

```
void ds2201_dac_strobe(
   phs_addr_t base,
   int group)
```

Include file

`ds2201.h`

Purpose

To strobe the D/A converter channels.

Description

The output of the specified channel group is updated simultaneously.

> **Note**
>
> The `ds2201_dac_load` function must be called before this function to load the values.

I/O mapping

For details on the I/O mapping, refer to Digital/Analog Conversion (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

group Specifies the DAC group number for channels to be strobed:

Value	Meaning
`DS2201_GROUP1`	For channels 1 … 4
`DS2201_GROUP2`	For channels 5 … 8
`DS2201_GROUP_ALL`	For all channels

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void sub_fct()
{
   ...
   ds2201_dac_load(DS2201_1_BASE, 1, -0.4);
   ds2201_dac_load(DS2201_1_BASE, 2, 0.9);
   ds2201_dac_strobe(DS2201_1_BASE, DS2201_GROUP1);
   ...
}
```

Channel 1 is loaded for –4 V and channel 2 for +9 V output voltage. The outputs of channel 1 and 2 are updated simultaneously.

Related topics

References

Base Address of the I/O Board	9
Digital/Analog Conversion (DS2201 Features 📖)	
ds2201_dac_load	27
ds2201_dac_out	25
ds2201_init	11
ds2201_set_errmode	22
ds2201_set_outmode	23

Digital I/O Unit

Introduction

The following functions are used to access the digital I/O of the DS2201 Multi-I/O Board.

> **Note**
>
> You have to initialize the DS2201 with the `ds2201_init` function before you can use one of these functions.

Where to go from here

Information in this section

Information in other sections

Digital I/O Unit (DS2201 Features 📖)
The slave DSP's digital I/O unit provides 16 digital I/O lines.

Specifying a Bitmask for the Digital I/O Unit

Introduction

The DS2201 provides a 16-bit right-aligned bitmask to configure digital I/O pins. Each four-digit binary value can be represented by its hexadecimal value:

Binary Value	**Hexadecimal Value**
0000	0x0
0001	0x1
0010	0x2
0011	0x3
0100	0x4
0101	0x5
0110	0x6
0111	0x7
1000	0x8
1001	0x9
1010	0xA
1011	0xB
1100	0xC
1101	0xD
1110	0xE
1111	0xF

You can see the correlation of I/O pin and bitmask value in the example below.

Example

The `ds2201_pin_io_init` function needs a bitmask to configure the I/O pins as input or output. An input pin is specified by a 0, an output pin is specified by a 1. If you want to specify I/O pins 0, 1, 2, 3, 5, 6, 8, 11, 12 and 13 as outputs, you have to specify the following bitmask:

I/O pin	*15*	*14*	*13*	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*
Binary	0	0	1	1	1	0	0	1	0	1	1	0	1	1	1	1
Hexadecimal	0x3				0x9				0x6				0xF			

You have to set the bitmask parameter in hexadecimal format as `0x396F`.

Related topics

References

ds2201_pin_io_clear 34
ds2201_pin_io_in 36
ds2201_pin_io_set 37
ds2201_pin_io_write 39

ds2201_pin_io_init

Syntax

```
void ds2201_pin_io_init(
   phs_addr_t base,
   UInt16 mask)
```

Include file

`ds2201.h`

Purpose

To initialize the digital I/O pins.

Description

This function initializes the slave DSP's 16-bit I/O port. Every pin can be configured as an input or output individually. Output pins are set to low level.

Note

- The `ds2201_init` function must be called before this function can be used.
- You cannot use the digital I/O unit if the 4DF application is running on the slave. For details, refer to Conflicting I/O Features (DS2201 Features 📖).

I/O mapping

For details on the I/O mapping, refer to Digital I/O Unit (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

mask Specifies the 16-bit right-aligned bitmask to configure digital I/O pins. The corresponding bit of the bitmask must be 0 for an input pin, and 1 for an output pin. For information on specifying a bitmask, refer to Specifying a Bitmask for the Digital I/O Unit on page 32.

Return value

None

Messages

The following message is defined:

ID	Type	Message	Description
-50	Error	ds2201_pin_io_init(0x??): Board not initialized!	The DS2201 has not been initialized by a preceding call to the `ds2201_init` function.

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   UInt16 mask = 0x0209;
   init();
   ds2201_init();
   ds2201_pin_io_init(DS2201_1_BASE, mask);
   ...
}
```

Pins 0, 3 and 9 are initialized as output, all other pins are used as input, the `mask` parameter contains the bitmask `0x0209` or `0000 0010 0000 1001B`.

Related topics

References

Base Address of the I/O Board.......9
ds2201_init.......11
ds2201_pin_io_clear.......34
ds2201_pin_io_in.......36
ds2201_pin_io_set.......37
ds2201_pin_io_write.......39

ds2201_pin_io_clear

Syntax

```
void ds2201_pin_io_clear(
   phs_addr_t base,
   UInt16 mask)
```

Include file

`ds2201.h`

Purpose

To clear the output of the digital I/O pins.

Description

This function clears individual output bits of the slave DSP's I/O port. A value of 1 in the bitmask clears the corresponding bit and a value of 0 has no effect.

Note

- The `ds2201_init` function must be called before this function can be used.
- The digital I/O pins must have been initialized by using the `ds2201_pin_io_init` function.
- You cannot use the digital I/O unit if the 4DF application is running on the slave. For details, refer to Conflicting I/O Features (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

mask Specifies the 16-bit right-aligned bitmask to configure digital I/O pins. The corresponding bit of the bitmask must be 1 for clearing the pin. For information on specifying a bitmask, refer to Specifying a Bitmask for the Digital I/O Unit on page 32.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   UInt16 mask = 0x0209;
   UInt16 clear = 0x0008;
   init();
   ds2201_init(DS2201_1_BASE);
   ds2201_pin_io_init(DS2201_1_BASE, mask);
   ds2201_pin_io_write(DS2201_1_BASE, mask);
   ds2201_pin_io_clear(DS2201_1_BASE, clear);
   ...
}
```

Pins 0, 3 and 9 are initialized as output and set to high level, all other pins are used as input corresponding to the `mask` parameter. Output pin 3 is cleared, output pins 0 and 9 remain unchanged.

Related topics

References

Base Address of the I/O Board 9
ds2201_init 11
ds2201_pin_io_in 36
ds2201_pin_io_init 33
ds2201_pin_io_set 37
ds2201_pin_io_write 39

ds2201_pin_io_in

Syntax

```
UInt16 ds2201_pin_io_in(
   phs_addr_t base)
```

Include file

`ds2201.h`

Purpose

To read the digital I/O pins.

Description

The `ds2201_pin_io_in` function reads the slave DSP's I/O port pins. It returns the status of the I/O-port pins configured as input, and the data to be transmitted in I/O-port pins configured as output.

Note

- The `ds2201_init` function must be called before this function can be used.
- The digital I/O pins must have been initialized by using the `ds2201_pin_io_init` function.
- You cannot use the digital I/O unit, if the 4DF application is running on the slave. For details, refer to Conflicting I/O Features (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

Return value

The following values are returned:

Value	Meaning
0x0000 … 0xFFFF	Bitmask with state of the digital I/O pins

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   UInt16 mask = 0x0209;
   UInt16 io_port;
   init();
   ds2201_init(DS2201_1_BASE);
   ds2201_pin_io_init(DS2201_1_BASE, mask);
   io_port = ds2201_pin_io_in(DS2201_1_BASE);
   ...
}
```

The state of the 16-bit digital I/O port is read and written to the `io_port` variable.

Related topics

References

Base Address of the I/O Board	9
ds2201_init	11
ds2201_pin_io_clear	34
ds2201_pin_io_init	33
ds2201_pin_io_set	37
ds2201_pin_io_write	39

ds2201_pin_io_set

Syntax

```
void ds2201_pin_io_set(
   phs_addr_t base,
   UInt16 mask)
```

Include file

`ds2201.h`

Purpose

To set output bits of the digital I/O pins.

Description

This function sets individual output bits of the slave DSP's I/O port. A value of 1 in the bitmask sets the corresponding bit and a value of 0 has no effect.

Note

- The `ds2201_init` function must be called before this function can be used.
- The digital I/O pins must have been initialized by using the `ds2201_pin_io_init` function.
- You cannot use the digital I/O unit, if the 4DF application is running on the slave. For details, refer to Conflicting I/O Features (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

mask Specifies the 16-bit right-aligned bitmask to set digital I/O pins. The corresponding bit of the bitmask must be 1 for setting the output pin to high level. For information on specifying a bitmask, refer to Specifying a Bitmask for the Digital I/O Unit on page 32.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   UInt16 mask = 0x0209;
   UInt16 out = 0x0008;
   init();
   ds2201_init(DS2201_1_BASE);
   ds2201_pin_io_init(DS2201_1_BASE, mask);
   ds2201_pin_io_set(DS2201_1_BASE, out);
   ...
}
```

Pins 0, 3 and 9 are initialized as output, all other pins are used as input corresponding to the `mask` parameter. Output pin 3 is set to high level, output pins 0 and 9 remain unchanged.

Related topics

References

Base Address of the I/O Board..9
ds2201_init..11
ds2201_pin_io_clear..34
ds2201_pin_io_in..36
ds2201_pin_io_init..33
ds2201_pin_io_write..39

ds2201_pin_io_write

Syntax

```
void ds2201_pin_io_write(
   phs_addr_t base,
   UInt16 value)
```

Include file

`ds2201.h`

Purpose

To write to the digital I/O pins.

Description

This function writes the specified bitmask to the slave DSP's I/O port. A value of 1 in the bitmask sets the corresponding I/O-port pin to high level and a value of 0 clears it. Pins configured as inputs are not affected.

Note

- The `ds2201_init` function must be called before this function can be used.
- The digital I/O pins must have been initialized by using the `ds2201_pin_io_init` function.
- You cannot use the digital I/O unit if the 4DF application is running on the slave. For details, refer to Conflicting I/O Features (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

value Specifies the bitmask to be written to the I/O port. For information on specifying a bitmask, refer to Specifying a Bitmask for the Digital I/O Unit on page 32.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   UInt16 mask = 0x0209;
   UInt16 out = 0x0008;
   init();
   ds2201_init(DS2201_1_BASE);
   ds2201_pin_io_init(DS2201_1_BASE, mask);
   ds2201_pin_io_write(DS2201_1_BASE, out);
   ...
}
```

Pins 0, 3 and 9 are initialized as output, all other pins are used as input corresponding to the `mask` parameter. Output pin 3 is set to high level, output pins 0 and 9 are set to low level.

Related topics

References

Base Address of the I/O Board 9
ds2201_init 11
ds2201_pin_io_clear 34
ds2201_pin_io_in 36
ds2201_pin_io_init 33
ds2201_pin_io_set 37

Timing I/O Unit

Introduction

The slave-DSP subsystem on the DS2201 provides a timing I/O unit that you can use to generate and measure signals.

Where to go from here

Information in this section

Information in other sections

Timing I/O Unit (DS2201 Features 📖)
The slave DSP's timing I/O unit supports signal generation and signal measurement.

Basics of the Timing I/O Unit

Basics of the Timing I/O Unit

Basics

The slave-DSP subsystem on the DS2201 provides a timing I/O unit that you can use to generate and measure signals.

The RTLib provides functions for generating PWM and square-wave signals. It also provides functions for measuring square-wave signals.

Generating or Measuring square-wave signals

You have to load a slave-DSP application if you want to generate or measure square-wave signals. For details on the slave-DSP applications, refer to Slave-DSP Applications on page 65.

Generating PWM signals

For generating PWM signals you can use the firmware or a slave-DSP application. For details on the slave-DSP applications, refer to Slave-DSP Applications on page 65.

PWM Signal Generation (PWM)

Introduction

The following functions are used to access the PWM (pulse width modulation) signal generation of the DS2201 Multi-I/O Board.

> **Note**
>
> You have to initialize the DS2201 with the `ds2201_init` function before you can use one of these functions.

Where to go from here

Information in this section

Information in other sections

PWM Signal Generation (DS2201 Features 🕮)
The timing I/O unit of the DS2201 can be programmed to generate pulse-width modulated (PWM) signals on up to 6 channels.

ds2201_pwmvar_init

Syntax

```
int ds2201_pwmvar_init(
   phs_addr_t base,
   int outputs,
   dsfloat period)
```

Include file

`ds2201.h`

Purpose

To initialize PWM signal generation.

Description

Either 4 or all 6 PWM channels of the DS2201 are initialized to generate PWM signals at the specified PWM frequency. All duty cycles are initialized with a value of 0. For details on PWM period, duty cycle and resolution, refer to PWM Signal Generation (DS2201 Features 🕮).

Note

- The `ds2201_init` function must be called before this function can be used.
- For PWM periods greater than approximately 42 ms, timer1 is additionally used as a prescaler and thus slave-DSP applications using this timer (2FD6PWM and 4FD4PWM) cannot be used at the same time. Otherwise the master application will hang up.

I/O mapping

For details on the I/O mapping, refer to PWM Signal Generation (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

outputs Specifies the number of channels to be used for PWM generation (4 or 6).

period Specifies the PWM period in seconds. The minimum PWM period is 160 µs, yielding a maximum PWM frequency of 6.25 MHz. The maximum period is 171.8 s. The PWM resolution is 40 ns for PWM periods below approximately 2.6 ms, and 160 ns otherwise.

Return value

This function returns an error code:

Symbol	Meaning
`DS2201_ALLOC_ERROR`	The memory allocation for internal data storage failed.
`DS2201_NO_ERROR`	No error occurred.

Messages

The following messages are defined:

ID	Type	Message	Description
-50	Error	ds2201_pwmvar_init(0x??): Board not initialized!	The DS2201 has not been initialized by a preceding call to the `ds2201_init` function.
-162	Error	ds2201_pwmvar_init(0x??): Memory allocation error!	The memory allocation for internal data storage failed.

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   init();
   ds2201_init();
   ds2201_pwmvar_init(DS2201_1_BASE, 4, 1.0e-3);
   ...
}
```

This example initializes a 4-channel PWM signal generation with a frequency of 1 kHz.

Related topics

References

Base Address of the I/O Board.. 9
ds2201_init.. 11
ds2201_pwmvar.. 45

ds2201_pwmvar

Syntax

```
void ds2201_pwmvar(
   phs_addr_t base,
   int channel,
   dsfloat duty)
```

Include file

`ds2201.h`

Purpose

To update the duty cycle of the PWM generation.

Description

The PWM duty cycle of the selected channel is updated to the new value specified by the `duty` parameter.

Note

- The `ds2201_init` function must be called before this function can be used.
- PWM generation must be initialized by using the `ds2201_pwmvar_init` function.

I/O mapping

For details on the I/O mapping, refer to PWM Signal Generation (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

channel Specifies the logical channel number within the range 1 … 6.

duty Specifies the duty cycle within the range 0.0 … 1.0.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   init();
   ds2201_init(DS2201_1_BASE);
   ds2201_pwmvar_init(DS2201_1_BASE, 4, 1.0e-3);
   ds2201_pwmvar(DS2201_1_BASE, 1, 0.5);
   ...
}
```

The duty cycle of the 1 kHz PWM signal generation of channel 1 is set to 50%.

Related topics

References

Base Address of the I/O Board...9
ds2201_init...11
ds2201_pwmvar_init...43

Square-Wave Signal Generation (D2F)

Introduction

The following functions are used to access the square-wave signal generation of the DS2201 Multi-I/O Board.

Where to go from here

Information in this section

Information in other sections

Square-Wave Signal Generation (D2F) (DS2201 Features 🕮)
The timing I/O unit of the DS2201 can generate square-wave signals with variable frequencies 2 or 4 channels.

ds2201_dtof

Syntax

```
void ds2201_dtof(
   phs_addr_t base,
   int channel,
   dsfloat value)
```

Include file

`ds2201.h`

Purpose

To set the frequency value of the 2-channel square-wave signal generation.

Description

This function controls the frequency of the square-wave signals generated at the DS2201 digital I/O outputs CMP0 and CMP1 if square-wave signal generation is enabled. Use the `ds2201_dtof_enable` function to start the square-wave generation. The frequency resolution of the generated signals is 160 ns.

Note

- The `ds2201_init` function must be called before this function can be used.
- The slave DSP must execute the 2-channel square-wave signal generation program `2fd2df.obj`.

I/O mapping

For details on the I/O mapping, refer to Square-Wave Signal Generation (D2F) (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

channel Specifies the logical channel number 1 or 2.

value Specifies the frequency value within the range 0.01 … 10000 Hz.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to set the generated frequency of channel 1 to 1000 Hz:

```
void main()
{
    init();
    ds2201_init(DS2201_1_BASE);
    ds2201_dtof_enable(DS2201_1_BASE, DS2201_DTOF_ENABLE);
    ds2201_dtof(DS2201_1_BASE, 1, 1000.0);
    ...
}
```

In `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\DS2201\2fd2df`, you can find the `slv_2fd2df_2201_hc` demo application, which demonstrates the usage of the 2-channel square-wave signal generation.

Related topics

References

Base Address of the I/O Board 9
ds2201_dtof_4 50
ds2201_dtof_enable 49
ds2201_init 11

ds2201_dtof_enable

Syntax

```
void ds2201_dtof_enable(
   phs_addr_t base,
   int value)
```

Include file

`ds2201.h`

Purpose

To enable or disable the 2-channel square-wave signal generation.

Description

Square-wave signal generation is disabled by default. Use the `ds2201_dtof` function to set the frequency value.

Note

- The `ds2201_init` function must be called before this function can be used.
- The slave DSP must execute the 2-channel square-wave signal generation program `2fd2df.obj`.

I/O mapping

For details on the I/O mapping, refer to Square-Wave Signal Generation (D2F) (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

value Specifies the status of square-wave signal generation:

Value	Meaning
`DS2201_DTOF_ENABLE`	For enabling square-wave signal generation
`DS2201_DTOF_DISABLE`	For disabling square-wave signal generation

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to enable the 2-channel square-wave signal generation:

```
void main()
{
    init();
    ds2201_init(DS2201_1_BASE);
    ds2201_dtof_enable(DS2201_1_BASE, DS2201_DTOF_ENABLE);
    ds2201_dtof(DS2201_1_BASE, 1, 1000.0);
    ...
}
```

In `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\2fd2df` you can find the `slv_2fd2df_2201_hc` demo application, which demonstrates the usage of the 2-channel square-wave signal generation.

Related topics

References

Base Address of the I/O Board........ 9
ds2201_dtof........ 47
ds2201_dtof_4........ 50
ds2201_init........ 11

ds2201_dtof_4

Syntax

```
void ds2201_dtof_4(
    phs_addr_t base,
    int channel,
    dsfloat value)
```

Include file

`ds2201.h`

Purpose

To set the frequency value of the 4-channel square-wave signal generation.

Description

This function controls the frequency of the square-wave signals generated at the slave DSP's I/O pins IOP0 … IOP3 if the 4-channel square-wave signal generation program is active. The resolution of the generated signals is 15 µs.

Note

- The `ds2201_init` function must be called before this function can be used.
- The slave DSP must execute the 4-channel square-wave signal generation program `4df.obj`.

I/O mapping

For details on the I/O mapping, refer to Square-Wave Signal Generation (D2F) (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

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

value Specifies the frequency value within the range 0.01 … 4000 Hz.

Return value

None

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to set the generated frequency of channel 3 to 1000 Hz:

```
void main()
{
   init();
   ds2201_init(DS2201_1_BASE);
   ds2201_dtof_4(DS2201_1_BASE, 3, 1000.0);
   ...
}
```

In `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\DS2201\4df` you can find the `slv_4df_2201_hc` demo application, which demonstrates the usage of the 4-channel square-wave signal generation.

Related topics

References

Base Address of the I/O Board	9
ds2201_dtof	47
ds2201_dtof_enable	49
ds2201_init	11

Square-Wave Signal Measurement (F2D)

Introduction

The following functions are used to access the square-wave signal measurement of the DS2201 Multi-I/O Board.

Where to go from here

Information in this section

Information in other sections

Square-Wave Signal Measurement (F2D) (DS2201 Features 📖)
The timing I/O unit of the DS2201 can be used to measure square-wave signals on up to 4 channels.

ds2201_ftd

Syntax

```
dsfloat ds2201_ftd(
   phs_addr_t base,
   int channel,
   int *ftoderror,
   int *new)
```

Include file

`ds2201.h`

Purpose

To read the values of the frequency measurement.

Description

The `ds2201_ftd` function reads the measured counter values from the slave DSP's frequency measurement application. The frequency is computed and returned by the function. The valid values for `channel` are 1 … 4 corresponding to the slave DSP's capture inputs CAP0 … CAP3 and depending on the running slave-DSP application.

The `ftoderror` flag indicates if an overflow error has occurred on the selected channel. In this case the returned frequency value must be ignored.

The `new` flag indicates an update of the frequency value. It is set if the value has been updated since the last call of `ds2201_ftd`. You can use this flag to detect zero frequency or the absence of an input signal. For further information, refer to Square-Wave Signal Measurement (F2D) (DS2201 Features 🕮).

The maximum frequency value to be measured by this function depends on the number of channels. For details, refer to Square-Wave Signal Measurement (F2D) (DS2201 Features 🕮).

Note

- The `ds2201_init` function must be called before this function can be used.
- The slave DSP must execute one of the frequency measurement applications:
  - `1fd.obj` for 1-channel frequency measurement
  - `2fd2df.obj` or `2fd6pwm.obj` for 2-channel frequency measurement
  - `4fd4pwm.obj` for 4-channel frequency measurement

I/O mapping

For details on the I/O mapping, refer to Square-Wave Signal Measurement (F2D) (DS2201 Features 🕮).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

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

ftoderror Returns the overflow error flag:

Value	Meaning
0	No error on selected channel
1	Overflow error on selected channel

new Returns the frequency update flag:

Value	Meaning
0	Frequency value not updated since last function call
1	Frequency value updated since last function call

Return value

The following values are returned:

Number of Channels	Value	Meaning
1	0.01 … 60,000.0 Hz	Measured frequency value
2	0.01 … 20,000.0 Hz	
4	0.01 … 10,000.0 Hz	

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
float tmp, frequency;
int   ftoderror, new;
void sub_fct()
{
   tmp = ds2201_ftd(DS2201_1_BASE, 1, &ftoderror, &new)
   if(!ftoderror)
   {
      if(new)
         frequency = tmp;
      else
         frequency = -1.0;
   }
}
```

The measured frequency of channel 1 is written to the variable frequency if no overflow error has occurred. If the frequency value has not been updated since the last call to `ds2201_ftd`, the frequency value is set to –1.0.

In `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\DS2201` you can find some demo applications, which demonstrate the usage of the square-wave signal measurement:

- 1FD for 1-channel frequency measurement
- 2FD2DF for 2-channel frequency measurement and frequency generation
- 2FD6PWM for 2-channel frequency measurement and 6-channel PWM signal generation
- 4FD4PWM for 4-channel frequency measurement and 4-channel PWM signal generation

Related topics

References

Base Address of the I/O Board..9
ds2201_init..11

Serial Interface

Introduction

The following functions are used to access the serial interface of the DS2201 Multi-I/O Board.

Where to go from here

Information in this section

Information in other sections

ds2201_serial_port_init

Syntax

```
void ds2201_serial_port_init(
   phs_addr_t base,
   long baud,
   long databit,
   long parity)
```

Include file

`ds2201.h`

Purpose

To initialize the asynchronous serial port.

Description

This function initializes the asynchronous serial port of the DS2201. After initialization the receive buffer is cleared.

When specifying a baud rate, you will encounter considerable deviations between the desired baud rate and the generated baud rate. For detailed information, refer to Specifying Baud Rates (DS2201 Features 📖).

> **Note**
>
> The **`ds2201_init`** function must be called before this function can be used.

I/O mapping

For details on the I/O mapping, refer to Basics of the Slave DSP (DS2201 Features 📖).

Parameters

base Specifies the PHS-bus base address. Refer to Base Address of the I/O Board on page 9.

baud Specifies the baud rate for serial transmission within the range 5 … 390,625 baud. For detailed information, refer to Basics of the Slave DSP (DS2201 Features 📖).

databit Specifies the number of data bits to be transmitted:

Value	Meaning
`DS2201_6_DATABIT`	For 6 data bits
`DS2201_7_DATABIT`	For 7 data bits
`DS2201_8_DATABIT`	For 8 data bits
`DS2201_9_DATABIT`	For 9 data bits

parity Specifies the parity check:

Value	Meaning
`DS2201_NO_PARITY`	Disables parity check
`DS2201_ODD_PARITY`	Enables odd parity check
`DS2201_EVEN_PARITY`	Enables even parity check

Return value

None

Messages

The following messages are defined:

ID	Type	Message	Description
-50	Error	ds2201_serial_port_init(): Board at offset 0x?? not initialized!	The DS2201 has not been initialized by a preceding call to the **`ds2201_init`** function.
-49	Error	ds2201_serial_port_init(): Unable to clear receive buffer!	The error occurs if it is not possible to empty the buffer. A possible cause for this error is the reception of data during initialization. The error can also be caused by a hardware failure of the slave DSP.

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void main()
{
   init();
   ds2201_init();
   ds2201_serial_port_init(DS2201_1_BASE, 9765,
   DS2201_8_DATABIT, DS2201_NO_PARITY)
   ...
}
```

The serial port is initialized to a baud rate of 9765 baud, 8 data bits are transmitted, the parity check is disabled.

Related topics

Basics

Serial Interface (DS2201 Features 📖)

References

Base Address of the I/O Board.......... 9
ds2201_init.......... 11
ds2201_read_scon.......... 59
ds2201_read_serial_port_init.......... 58
ds2201_rx_byte.......... 60
ds2201_tx_byte.......... 62

ds2201_read_serial_port_init

Syntax

```
void ds2201_read_serial_port_init(
   phs_addr_t base,
   long *baud,
   long *databit,
   long *parity)
```

Include file

`ds2201.h`

Purpose

To read the initialization of the asynchronous serial port.

Note

The `ds2201_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 9.

baud Returns the baud rate for serial transmission within the range 5 … 390,625 baud.

databit Returns the number of data bits to be transmitted:

Value	Meaning
DS2201_6_DATABIT	For 6 data bits
DS2201_7_DATABIT	For 7 data bits
DS2201_8_DATABIT	For 8 data bits
DS2201_9_DATABIT	For 9 data bits

parity Returns the state of the parity check:

Value	Meaning
`DS2201_NO_PARITY`	Disables parity check
`DS2201_ODD_PARITY`	Enables odd parity check
`DS2201_EVEN_PARITY`	Enables even parity check

Return value

None

Example

This example shows how to read the initialization of the serial port:

```
void sub_fct()
{
   long baud;
   long databits;
   long parity;
   ds2201_read_serial_port_init(
   DS2201_1_BASE, &baud, &databits &parity);
   ...
}
```

Related topics

References

Base Address of the I/O Board.......9
ds2201_init.......11
ds2201_read_scon.......59
ds2201_rx_byte.......60
ds2201_serial_port_init.......56
ds2201_tx_byte.......62

ds2201_read_scon

Syntax

```
UInt16 ds2201_read_scon(phs_addr_t base)
```

Include file

`ds2201.h`

Purpose

To read the serial control register (SCON).

Description

This function reads the serial control register (SCON) of the DS2201 serial port. For more information about the SCON register, refer to the *TMS320C1x User's Guide* (search for *TMS320C1x* at http://www.ti.com).

> **Note**
>
> The `ds2201_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 9.

Return value

The following values are returned:

Value	Meaning
`0x0000` … `0xFFFF`	Contents of SCON register.

Example

This example shows how to read the SCON register of the serial port:

```
void sub_fct()
{
   UInt16 scon;
   scon = ds2201_read_scon(DS2201_1_BASE);
   ...
}
```

Related topics

References

Base Address of the I/O Board	9
ds2201_init	11
ds2201_read_serial_port_init	58
ds2201_rx_byte	60
ds2201_serial_port_init	56
ds2201_tx_byte	62

ds2201_rx_byte

Syntax

```
int ds2201_rx_byte(
   phs_addr_t base,
   unsigned long *data)
```

Include file `ds2201.h`

Purpose To read a received data byte from the asynchronous serial port.

> **Note**
>
> The `ds2201_init` and `ds2201_serial_port_init` functions 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 9.

data Specifies the address where the received data byte is written.

Return value The return value can be 0, 1 or greater than 1. If there was a receive error (>1), the status bits of the SCON register are copied into the return value (see also `ds2201_read_scon`). They can be tested using the corresponding error flag definitions. The predefined symbols can be combined using the logical OR operator.

Value	Predefined Symbol	Meaning
0	`DS2201_NO_ERROR`	Received data successfully read
1	`DS2201_RB_EMPTY`	No new data received
>1	`DS2201_BBF_MSK`	The receive buffer is full (it contains 2 bytes). You must call the `ds2201_rx_byte` function a second time to get all received data bytes.
	`DS2201_ROV_MSK`	The receive buffer is full (it contains 2 bytes) and a third data byte has been sent to the serial port. The third data byte has been lost. You must call the `ds2201_rx_byte` function a second time to get the second received data byte.
	`DS2201_FERR_MSK`	The serial port has detected a framing error. This means that the serial port has received an unexpected number of bits. This error can occur if transmitter and receiver are using different settings for data bytes, stop bits or parity check.
	`DS2201_PERR_MSK`	The serial port has detected a parity error. Some of the received bits are corrupted.

Execution times For information, refer to Function Execution Times on page 75.

Example

This example shows how to use the function:

```
void sub_fct()
{
   int error;
   UInt8 data;
   error = ds2201_rx_byte(DS2201_1_BASE, &data);
   if(error & (DS2201_FERR_MSK | DS2201_PERR_MSK)
      return ERROR;
   else if(!error)
      return DATA_RECEIVED;
   else if(error == DS2201_RB_EMPTY)
      return NO_NEW_DATA;
   ...
}
```

The serial port is read. If a framing or parity error has been detected, `ERROR` is returned.

If the data byte has been received successfully, `DATA_RECEIVED` is returned. If no new data is available, `NO_NEW_DATA` is returned.

Related topics

References

Base Address of the I/O Board........ 9
ds2201_init........ 11
ds2201_read_scon........ 59
ds2201_read_serial_port_init........ 58
ds2201_serial_port_init........ 56
ds2201_tx_byte........ 62

ds2201_tx_byte

Syntax

```
int ds2201_tx_byte(
   phs_addr_t base,
   UInt8 data)
```

Include file

`ds2201.h`

Purpose

To transmit bytes via the serial port.

Note

The `ds2201_init` and `ds2201_serial_port_init` functions 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 9.

data Specifies the output data value to be transmitted within the range 0x00 … 0xFF.

Return value

The following values are returned:

Value	Meaning
`DS2201_NO_ERROR`	Data was transmitted successfully
`DS2201_TB_FULL`	Write buffer is full, data not transmitted

Execution times

For information, refer to Function Execution Times on page 75.

Example

This example shows how to transmit a data byte via the serial port:

```
void sub_fct()
{
   int error;
   Uint8 data = 0x12;
   error = ds2201_tx_byte(DS2201_1_BASE, data);
   if(!error)
      return OK;
   else
      return ERROR;
   ...
}
```

If the transmission fails, `ERROR` is returned. If the byte is transmitted successfully, `OK` is returned.

Related topics

References

Slave-DSP Applications

Introduction

This section contains information on the ready-to-use slave-DSP applications.

Where to go from here

Information in this section

Basics of Slave-DSP Applications

Introduction

The applications described in this section have been written for the TMS320E14 DSP used as a slave DSP on the DS2201 Multi-I/O Board.

The slave DSP has two modes of operation. After power-up or after a slave-DSP reset, the TMS320E14's microcomputer mode is selected and the code contained in the firmware is executed. The firmware provides:

- Communication functions
- Digital I/O functions
- PWM signal generation

For special applications the second operation mode can be applied. Slave-DSP applications can be downloaded to the slave's program RAM. When an application has been downloaded, the DSP is switched to microprocessor mode to execute the code in program RAM. In each application, firmware-equivalent functions are included besides the specific code to provide the communication mechanisms to the processor board. The available slave-DSP applications are:

Slave-DSP Application	Description
1FD	Frequency measurement (F2D) within the range 0.01 Hz … 60 kHz on 1 channel.
2FD2DF	▪ Square-wave signal generation (D2F) within the range 0.01 Hz … 10 kHz on 2 channels. ▪ Frequency measurement (F2D) within the range 0.01 Hz … 20 kHz on 2 channels.
2FD6PWM	▪ Frequency measurement (F2D) within the range 0.01 Hz … 10 kHz on 2 channels. ▪ PWM signal generation on 6 channels.
4DF	Square-wave signal generation (D2F) within the range 0.01 Hz … 4 kHz on 4 channels.
4FD4PWM	▪ Frequency measurement (F2D) within the range 0.01 Hz … 10 kHz on 4 channels. ▪ PWM signal generation on 4 channels.

The slave-DSP application example programs are copied to `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201` during software installation:

For information on the slave DSP, refer to Basics of the Slave DSP (DS2201 Features 📖).

Note

Not all of the I/O features of the DS2201 can be used if a slave-DSP application has been loaded. For detailed information on the conflicts, refer to Conflicting I/O Features (DS2201 Features 📖).

How to Load DS2201 Slave-DSP Applications

Introduction

To use the DS2201 slave DSP for square-wave signal generation or measurement, it is necessary to load an appropriate slave-DSP application.

Note

This instruction must be done only for a handcoded application. Using RTI, the required slave-DSP application will be loaded automatically.

Loading slave-DSP applications

The DS2201 slave-DSP application cannot be loaded directly to the slave DSP. It must be loaded via the processor board. To do this, the object file for the slave DSP has been converted into a C file. The C file must be included in the processor board application. It contains a loader function which loads the application to the slave DSP.

Method

To load a slave-DSP application

1. Copy the C file of the required slave-DSP application from `<RCP_HIL_InstallationPath>\Slave\DS2201` to your application folder.
2. Include the generated C file, for example `1fd.c`, in your processor board application.
3. Invoke the loader function within the processor board application.

   Example:

```
#include "1fd.c"
...
void main(void)
{
   init();
   load_slave_1fd(DS2201_1_BASE);
   ...
}
```

Tip

The C files containing the load functions for the five DS2201 standard slave-DSP applications are already available in `<RCP_HIL_InstallationPath>\Slave\DS2201`.

Demo Program for 1FD Application

Introduction

The `slv_1fd_2201_hc.c` demo program measures the frequency on one channel.

Physical connections

Connect pin CAP0 of the DS2201's I/O connector to a function generator with TTL output level. For information on the connector pinouts, refer to Digital I/O Connector (P4) (PHS Bus System Hardware Reference 📖).

Opening the demo

A backup file for working with the demo is installed in `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\1fd`. You can open the `slv_1fd_2201_hc.zip` in ControlDesk.

Experimenting with the demo

The measured frequency, the flag status and the execution time of the `ds2201_ftd` function can be observed in ControlDesk.

If the frequency is too high, either an overload error or an ftod error occurs. These errors are displayed in the layout. An overload error pauses program execution and can be cleared by using the Clear Error button. An ftod error signals that the frequency value measured by the slave DSP is not correct.

The new signal lamp in the layout shows the new flag status of the `ds2201_ftd` function. The new flag is set if the measured frequency value has been updated since the last call to `ds2201_ftd`.

Related topics

HowTos

How to Load DS2201 Slave-DSP Applications.. 67

References

ds2201_ftd... 52

Demo Program for 2FD2DF Application

Introduction

The `slv_2fd2df_2201_hc.c` demo program measures the frequency on two channels and generates square-wave signals on two channels.

Physical connections

Connect pins CAP0, CAP1 of the DS2201's I/O connector to a function generator with TTL output level and connect pins CMP0, CMP1 to an oscilloscope. For information on the connector pinouts, refer to Digital I/O Connector (P4) (PHS Bus System Hardware Reference 🕮).

Opening the demo

A backup file for working with the demo is installed in `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\2fd2df`. You can open the `slv_2fd2df_2201_hc.zip` in ControlDesk.

Experimenting with the demo

The measured frequency, the flag status and the execution time of the `ds2201_ftd` function can be observed.

If the frequency is too high, either an overload error or an ftod error occurs. These errors are displayed in the ControlDesk window. An overload error pauses program execution and can be cleared by using the **Clear Error** button. An ftod error signals that the frequency value measured by the slave DSP is not correct.

The **new** signal lamp in the layout shows the new flag status of the `ds2201_ftd` function. The new flag is set if the measured frequency value has been updated since the last call to `ds2201_ftd`.

To enable 2-channel square-wave signal generation, the `ds2201_dtof_enable` function is called in the initialization part of the `main` function. The frequency of the generated square-wave signals is set by using the `ds2201_dtof` function. The values can be adjusted between 100 Hz and 10 kHz in the layout. The execution time of `ds2201_dtof` is also displayed in the layout.

Related topics

HowTos

How to Load DS2201 Slave-DSP Applications.. 67

References

ds2201_dtof.. 47
ds2201_dtof_enable.. 49
ds2201_ftd... 52

Demo Program for 2FD6PWM Application

Introduction

The `slv_2fd6pwm_2201_hc.c` demo program measures the frequency on two channels and generates PWM signals on 6 channels.

Physical connections

Connect pins CAP0, CAP1 of the DS2201's I/O connector to a function generator with TTL output level and connect pins CMP0 … CMP3, CAP2, CAP3 to an oscilloscope. For information on the connector pinouts, refer to Digital I/O Connector (P4) (PHS Bus System Hardware Reference 📖).

Opening the demo

A backup file for working with the demo is installed in `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\2fd6pwm`. You can open the `slv_2fd6pwm_2201_hc.zip` in ControlDesk.

Experimenting with the demo

The measured frequency, the flag status and the execution time of the `ds2201_ftd` function can be observed.

If the frequency is too high, either an overload error or an ftod error occurs. These errors are displayed in the layout. An overload error pauses program execution and can be cleared by using the **Clear error** button. An ftod error signals that the frequency value measured by the slave DSP is not correct.

The **new** signal lamp in the layout shows the new flag status of the `ds2201_ftd` function. The new flag is set if the measured frequency value has been updated since the last call to `ds2201_ftd`.

The `ds2201_pwmvar` function is used to control the duty cycle of the generated PWM signals. The duty cycle can be adjusted in the layout between 0 (0%) and 1 (100%). The PWM channel and the PWM frequency can also be selected. If the PWM frequency has been changed in the layout, the `ds2201_pwmvar_init` function is called. The execution time of `ds2201_pwmvar` is also displayed.

Related topics

HowTos

How to Load DS2201 Slave-DSP Applications.. 67

References

ds2201_ftd... 52
ds2201_pwmvar.. 45
ds2201_pwmvar_init.. 43

Demo Program for 4DF Application

Introduction

The `slv_4df_2201_hc.c` demo program generates square-wave signals on 4 channels.

Physical connections

Connect pins IOP0 … IOP3 of the DS2201's I/O connector to an oscilloscope. For information on the Connector Pinouts, refer to Digital I/O Connector (P4) (PHS Bus System Hardware Reference 🕮).

Opening the demo

A backup file for working with the demo is installed in `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\4df`. You can open the `slv_4df_2201_hc.zip` in ControlDesk.

Experimenting with the demo

The square-wave signal generation can be observed.

The frequency of each channel can be adjusted between 0.01 Hz and 4 kHz in the layout. The execution time of the `ds2201_dtof_4` function, which is used to control the frequency of the generated square-wave signal, is also displayed.

Related topics

HowTos

How to Load DS2201 Slave-DSP Applications.. 67

References

ds2201_dtof_4.. 50

Demo Program for 4FD4PWM Application

Introduction

The `slv_4fd4pwm_2201_hc.c` demo program measures the frequency on 4 channels and generates PWM signals on 4 channels.

Physical connections

Connect pins CAP0 … CAP3 of the DS2201's I/O connector to a function generator with TTL output level and connect pins CMP0 … CMP3 to an oscilloscope. For information on the connector pinouts, refer to Digital I/O Connector (P4) (PHS Bus System Hardware Reference 🕮).

Opening the demo

A backup file for working with the demo is installed in `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\4fd4pwm`. You can open the `slv_4fd4pwm_2201_hc.zip` in ControlDesk.

Experimenting with the demo

The measured frequency, the flag status and the execution time of the `ds2201_ftd` function can be observed.

If the frequency is too high, either an overload error or an ftod error occurs. These errors are displayed in the layout. An overload error pauses program execution and can be cleared by using the Clear Error button. An ftod error signals that the frequency value measured by the slave DSP is not correct and must be ignored.

The new signal lamp in the layout shows the new flag status of the `ds2201_ftd` function. The new flag is set if the measured frequency value has been updated since the last call to `ds2201_ftd`.

The `ds2201_pwmvar` function is used to control the duty cycle of the generated PWM signals. The duty cycle can be adjusted between 0 (0%) and 1 (100%) in the layout. The PWM channel and the PWM frequency can also be selected. If the PWM frequency has been changed, the `ds2201_pwmvar_init` function is called. The execution time of `ds2201_pwmvar` is also displayed.

Related topics

HowTos

How to Load DS2201 Slave-DSP Applications..67

References

ds2201_ftd..52
ds2201_pwmvar..45
ds2201_pwmvar_init...43

Demo Program for Using the Slave-DSP's Digital I/O Unit

Introduction

The `slv_ioport_2201_hc.c` demo program shows you how to access the slave-DSP's digital I/O unit (pins IOP0 … IOP15).

Physical connections

Connect the pins to serve as inputs to a function generator with TTL output level and the pins to serve as outputs to an oscilloscope. For information on the Connector Pinouts, refer to Digital I/O Connector (P4) (PHS Bus System Hardware Reference 🕮).

Opening the demo

A backup file for working with the demo is installed in `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\io_port`. You can open the `slv_ioport_2201_hc.zip` in ControlDesk.

Experimenting with the demo

The settings of the I/O pins can be observed.

The status of the I/O-port pins is shown in the upper part of the layout. The port status is read by using the `ds2201_pin_io_in` function. Red signal lamps indicate I/O-port pins with a high level, green lamps indicate a low level. Unconnected inputs are high.

All I/O-port pins are configured as inputs when the program is started. To configure an I/O-port pin as output, activate the check button in the Init I/O Port Direction section in the lower part of the layout. The I/O port is initialized by the `ds2201_pin_io_init` function.

To change the status of I/O-port pins configured as output, use the check buttons in the output section in the middle part of the layout. Activating a check button sets the corresponding pin immediately. Deactivating a check button clears the corresponding pin. The `ds2201_pin_io_write` function is used for this.

Related topics

HowTos

How to Load DS2201 Slave-DSP Applications.. 67

References

ds2201_pin_io_in.. 36
ds2201_pin_io_init.. 33
ds2201_pin_io_write... 39

Demo Program for Using the Serial Interface

Introduction

The `slv_rs232_2201_hc.c` demo program shows you how to use the slave-DSP's serial interface. The serial interface of the connected PC has to be configured with a baud rate of 9600 baud, a transmission length of 8 data bits and no parity check.

Physical connections

Connect the DS2201 Board via Serial Interface Connector to a PC. For information on the connector pinouts, refer to Digital I/O Connector (P4) (PHS Bus System Hardware Reference 📖).

Opening the demo

A backup file for working with the demo is installed in `<RCP_HIL_InstallationPath>\Demos\DS100x\IOBoards\ds2201\rs232`. You can open the `slv_rs232_2201_hc.zip` in ControlDesk.

Experimenting with the demo

The DS2201 transmits one character from the ASCII charset each second. The status of the transmission, the number of transmitted data, the data itself and the execution times for sending and receiving can be observed in the layout.

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.

Where to go from here

Information in this section

Influence of the Processor Board on the Execution Times

Introduction

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

- CPU clock and bus clock frequency of the controller board used
- Optimization level of the compiler and the usage of inlining
- The parameters used
- Behavior of the master-slave communication
- Application loaded on the slave DSP

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

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.

Note

Because of the unpredictable behavior of the execution times, caused by slave-DSP interrupts, you have to consider peak values when calculating the execution time for your application.

The properties of the DS1006 processor boards used are:

CPU clock:	2.6 GHz / 3.0 GHz
Bus clock:	133.16 MHz

Related topics

References

Influence of the Slave DSP on the Execution Times

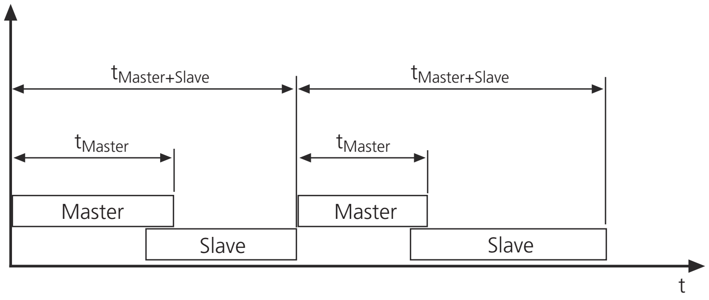
Slave DSP's workload and interrupts

The execution times of slave-DSP access functions depend on the workload of the slave DSP. The master needs a longer time for a slave-DSP access if the slave DSP:

- is just executing a previous command
- has to respond to the master
- is executing an interrupt-driven application (slave-DSP application loaded)

The interrupt service routines used for signal generation and frequency measurement suspend the background task. The number of interrupts again depends on the number of channels used and the frequency specified for signal generation or frequency measurement. It is therefore not possible to give a formula for calculating the slave-DSP execution times.

The execution times are divided into the time that the master takes to execute a function (*Master*) and the time that contains the response time of the slave DSP (*Master* + *Slave*).



The next call of a function with slave-DSP access will not be executed until the slave DSP has finished its last function execution. For two consecutive function calls with slave-DSP access you have to consider the *Master* + *Slave* execution times.

Note

Because of the unpredictable behavior of the execution times, caused by slave-DSP interrupts, you have to consider peak values when calculating the execution time for your application.

Related topics

References

Measured Execution Times for Functions With Slave-DSP Access.. 81

Measured Execution Times for Functions Without Slave-DSP Access

Where to go from here

Information in this section

Execution Times of the Initialization

Introduction

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**
`ds2201_init`	51.25 µs	65.16 µs

Related topics

Basics

Information on the Test Environment..76

Execution Times of the ADC Unit

Introduction

The following execution times have been measured for the ADC unit:

Function	Used Groups/Channels	Mean Execution Time	
		DS1006 with 2.6 GHz	**DS1006 with 3.0 GHz**
`ds2201_adc_start`	–	0.02 µs	0.02 µs

`ds2201_adc_start`
`+ ds2201_adc_in`
	–	34.59 µs	34.58 µs

Function	Used Groups/Channels	Mean Execution Time	
		DS1006 with 2.6 GHz	DS1006 with 3.0 GHz
ds2201_adc_start + ds2201_adc_block_in	All groups (20 channels)	44.40 µs	43.91 µs
ds2201_adc_in	–	2.89 µs	2.88 µs
ds2201_adc_read	–	2.33 µs	2.35 µs
ds2201_adc_block_in	Up to 4 channels Up to 8 channels Up to 12 channels Up to 16 channels Up to 20 channels	2.92 µs 5.26 µs 7.59 µs 9.91 µs 12.23 µs	2.93 µs 5.25 µs 7.57 µs 9.89 µs 12.20 µs

Related topics

Basics

Information on the Test Environment.. 76

Execution Times of the DAC Unit

Introduction

The following execution times have been measured for the DAC unit:

Function	Used Groups/Channels	Mean Execution Time	
		DS1006 with 2.6 GHz	DS1006 with 3.0 GHz
ds2201_set_errmode		1.38 µs	1.39 µs
ds2201_set_outmode		1.39 µs	1.39 µs
ds2201_dac_out		0.03 µs	0.03 µs
ds2201_dac_load		0.02 µs	0.03 µs
ds2201_dac_strobe	All groups	0.02 µs	0.02 µs

Related topics

Basics

Information on the Test Environment.. 76

Measured Execution Times for Functions With Slave-DSP Access

Introduction

The function execution times depend on the application, which is loaded on the slave DSP.

Where to go from here

Information in this section

Execution Times When Using the Standard Firmware

Timing I/O unit

The following mean execution times have been measured for the timing I/O unit:

Function	DS1006 with 2.6 GHz		DS1006 with 3.0 GHz	
	Master	**Master + Slave**	**Master**	**Master + Slave**
`ds2201_pwmvar_init`	91.60 µs	103.03 µs	91.62 µs	103.02 µs
`ds2201_pwmvar`	0.61 µs	11.53 µs	0.61 µs	11.47 µs

Digital I/O unit

The following mean execution times have been measured for the digital I/O unit:

Function	DS1006 with 2.6 GHz		DS1006 with 3.0 GHz	
	Master	**Master + Slave**	**Master**	**Master + Slave**
`ds2201_pin_io_init`	11.55 µs	22.96 µs	11.49 µs	22.89 µs
`ds2201_pin_io_clear`	0.60 µs	11.53 µs	0.60 µs	11.53 µs
`ds2201_pin_io_in`	11.77 µs	12.38 µs	11.78 µs	12.38 µs
`ds2201_pin_io_set`	0.60 µs	11.51 µs	0.60 µs	11.47 µs
`ds2201_pin_io_write`	0.60 µs	11.53 µs	0.60 µs	11.53 µs

Serial Interface

The following mean execution times have been measured for the serial interface:

Function	DS1006 with 2.6 GHz		DS1006 with 3.0 GHz	
	Master	**Master + Slave**	**Master**	**Master + Slave**
`ds2201_serial_port_init`	57.80 µs	58.41 µs	57.75 µs	58.35 µs
`ds2201_rx_byte`	11.80 µs	12.40 µs	11.78 µs	12.38 µs
`ds2201_tx_byte`	11.80 µs	22.34 µs	11.81 µs	22.34 µs

Execution Times When Using the 1FD Application on DS1006-Based Systems

Introduction

Because the execution times depend on the frequency, the measurement considers two different test conditions:

- The 1FD application measures a 1 kHz signal.
- The 1FD application measures a signal with the maximal frequency of 60 kHz.

Timing I/O unit

The following mean execution time has been measured for the timing I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency: 1 kHz (in)		Frequency: 60 kHz (in$_{max}$)		Frequency: 1 kHz (in)		Frequency: 60 kHz (in$_{max}$)	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_ftd`	63.43 µs	64.03 µs	1.35 µs	1.35 µs	63.80 µs	64.40 µs	1.31 µs	1.31 µs

Digital I/O unit

The following mean execution times have been measured for the digital I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency: 1 kHz (in)		Frequency: 60 kHz (in$_{max}$)		Frequency: 1 kHz (in)		Frequency: 60 kHz (in$_{max}$)	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_pin_io_init`	16.00 µs	33.92 µs	391.37 µs	940.54 µs	15.40 µs	33.30 µs	396.00 µs	939.83 µs
`ds2201_pin_io_clear`	0.60 µs	15.36 µs	0.63 µs	405.55 µs	0.61 µs	16.42 µs	0.61 µs	384.64 µs
`ds2201_pin_io_in`	15.71 µs	16.31 µs	390.57 µs	391.18 µs	15.68 µs	16.28 µs	407.61 µs	408.21 µs
`ds2201_pin_io_set`	0.60 µs	15.40 µs	0.63 µs	403.18 µs	0.61 µs	15.19 µs	0.61 µs	405.71 µs
`ds2201_pin_io_write`	0.60 µs	15.29 µs	0.63 µs	393.76 µs	0.61 µs	15.53 µs	0.61 µs	389.59 µs

Serial Interface

The following mean execution times have been measured for the serial interface:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency: 1 kHz (in)		Frequency: 60 kHz (in$_{max}$)		Frequency: 1 kHz (in)		Frequency: 60 kHz (in$_{max}$)	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_serial_port_init`	87.74 µs	88.35 µs	2.58 µs	2.58 µs	87.70 µs	88.30 µs	2.54 ms	2.54 ms
`ds2201_rx_byte`	15.81 µs	16.41 µs	393.71 µs	394.32 µs	15.76 µs	16.36 µs	383.08 µs	383.68 µs
`ds2201_tx_byte`	16.31 µs	33.32 µs	398.65 µs	948.61 µs	16.07 µs	33.09 µs	399.68 µs	946.23 µs

Execution Times When Using the 2FD2DF Application on DS1006-Based Systems

Introduction

Because the execution times depend on the frequency, the measurement considers two different test conditions:

- The 2FD2DF application generates a 1 kHz square-wave signal and measures a 1 kHz square-wave signal.
- The 2FD2DF application generates a 10 kHz square-wave signal and measures a 20 kHz square-wave signal.

Timing I/O unit

The following mean execution times have been measured for the timing I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency: 1 kHz (in) / 1 kHz (out)		Frequency: 20 kHz (in$_{max}$) / 10 kHz (out$_{max}$)		Frequency: 1 kHz (in) / 1 kHz (out)		Frequency: 20 kHz (in$_{max}$) / 10 kHz (out$_{max}$)	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_ftd`	117.79 µs	118.39 µs	4.60 ms	4.60 ms	118.04 µs	118.64 µs	4.33 ms	4.33 ms
`ds2201_dtof`	46.53 µs	74.40 µs	2.44 ms	3.48 ms	46.05 µs	73.71 µs	2.44 ms	3.57 ms
`ds2201_dtof_enable`	0.61 µs	18.54 µs	0.62 µs	1.02 µs	0.60 µs	17.66 µs	0.60 µs	1.09 ms

Digital I/O unit

The following mean execution times have been measured for the digital I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency: 1 kHz (in) / 1 kHz (out)		Frequency: 20 kHz (in$_{max}$) / 10 kHz (out$_{max}$)		Frequency: 1 kHz (in) / 1 kHz (out)		Frequency: 20 kHz (in$_{max}$) / 10 kHz (out$_{max}$)	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_pin_io_init`	21.97 µs	53.64 µs	1.18 ms	2.67 ms	22.86 µs	54.54 µs	1.21 ms	2.79 ms
`ds2201_pin_io_clear`	0.60 µs	21.95 µs	0.63 µs	1.19 µs	0.61 µs	22.53 µs	0.61 µs	1.22 ms
`ds2201_pin_io_in`	23.21 µs	23.81 µs	1.21 mµs	1.21 ms	23.77 µs	24.36 µs	1.25 ms	1.25 ms
`ds2201_pin_io_set`	0.60 µs	22.81 µs	0.63 µs	1.25 ms	0.61 µs	21.95 µs	0.61 µs	1.35 ms
`ds2201_pin_io_write`	0.60 µs	23.31 µs	0.63 µs	1.27 µs	0.61 µs	22.21 µs	0.61 µs	1.24 ms

Serial Interface

The following mean execution times have been measured for the serial interface:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency: 1 kHz (in) / 1 kHz (out)		Frequency: 20 kHz (in$_{max}$) / 10 kHz (out$_{max}$)		Frequency: 1 kHz (in) / 1 kHz (out)		Frequency: 20 kHz (in$_{max}$) / 10 kHz (out$_{max}$)	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_serial_port_init`	150.51 µs	151.12 µs	6.43 µs	6.43 µs	149.75 µs	150.35 µs	6.11 µs	6.11 µs
`ds2201_rx_byte`	22.19 µs	22.79 µs	1.24 µs	1.24 µs	22.96 µs	23.56 µs	1.26 µs	1.26 µs
`ds2201_tx_byte`	23.39 µs	54.48 µs	1.28 µs	2.89 µs	22.91 µs	53.96 µs	1.18 µs	2.64 µs

Execution Times When Using the 2FD6PWM Application on DS1006-Based Systems

Introduction

Because the execution times depend on the frequency, the measurement considers two different test conditions:

- The 2FD6PWM application measures a 1 kHz square-wave signal and generates a PWM signal on 6 channels.
- The 2FD6PWM application measures a 20 kHz square-wave signal and generates a PWM signal on 6 channels.

Timing I/O unit

The following mean execution times have been measured for the timing I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (in): 1 kHz		Frequency (in$_{max}$): 20 kHz		Frequency (in): 1 kHz		Frequency (in$_{max}$): 20 kHz	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_ftd`	78.01 µs	78.61 µs	179.01 µs	179.63 µs	78.32 µs	78.92 µs	179.10 µs	179.70 µs
`ds2201_pwmvar_init`	165.24 µs	186.52 µs	370.94 µs	420.91 µs	165.34 µs	186.44 µs	382.87 µs	432.60 µs
`ds2201_pwmvar`	0.61 µs	17.08 µs	0.63 µs	42.45 µs	0.61 µs	16.82 µs	0.61 µs	34.30 µs

Digital I/O unit

The following mean execution times have been measured for the digital I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (in): 1 kHz		Frequency (in$_{max}$): 20 kHz		Frequency (in): 1 kHz		Frequency (in$_{max}$): 20 kHz	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_pin_io_init`	17.33 µs	38.41 µs	28.68 µs	78.42 µs	17.17 µs	38.22 µs	34.39 µs	83.26 µs
`ds2201_pin_io_clear`	0.60 µs	17.01 µs	0.61 µs	34.14 µs	0.61 µs	17.34 µs	0.61 µs	33.56 µs
`ds2201_pin_io_in`	17.30 µs	17.90 µs	23.01 µs	23.62 µs	17.55 µs	18.15 µs	34.50 µs	35.10 µs
`ds2201_pin_io_set`	0.60 µs	17.49 µs	0.61 µs	46.20 µs	0.61 µs	17.05 µs	0.61 µs	34.22 µs
`ds2201_pin_io_write`	0.60 µs	16.99 µs	0.61 µs	29.04 µs	0.61 µs	16.70 µs	0.61 µs	33.96 µs

Serial Interface

The following mean execution times have been measured for the serial interface:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (in): 1 kHz		Frequency (in$_{max}$): 20 kHz		Frequency (in): 1 kHz		Frequency (in$_{max}$): 20 kHz	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_serial_port_init`	102.37 µs	102.98 µs	242.08 µs	242.69 µs	102.09 µs	102.69 µs	232.64 µs	233.24 µs
`ds2201_rx_byte`	17.45 µs	18.06 µs	26.33 µs	26.94 µs	18.12 µs	18.72 µs	34.14 µs	34.74 µs
`ds2201_tx_byte`	17.55 µs	38.09 µs	33.90 µs	82.79 µs	17.81 µs	38.35 µs	36.17 µs	85.36 µs

Execution Times When Using the 4FD4PWM Application on DS1006-Based Systems

Introduction

Because the execution times depend on the frequency, the measurement considers two different test conditions:

- The 4FD4PWM application measures a 1 kHz square-wave signal and generates a PWM signal on 4 channels.
- The 4FD4PWM application measures a 10 kHz square-wave signal and generates a PWM signal on 4 channels.

Timing I/O unit

The following mean execution times have been measured for the timing I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (in): 1 kHz		Frequency (in$_{max}$): 10 kHz		Frequency (in): 1 kHz		Frequency (in$_{max}$): 10 kHz	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_ftd`	85.97 µs	86.58 µs	150.91 µs	151.52 µs	103.78 µs	104.37 µs	136.79 µs	137.39 µs
`ds2201_pwmvar_init`	122.47 µs	143.76 µs	259.28 µs	280.87 µs	127.69 µs	148.80 µs	235.75 µs	257.18 µs
`ds2201_pwmvar`	0.61 µs	16.99 µs	0.63 µs	42.11 µs	0.61 µs	16.40 µs	0.61 µs	17.16 µs

Digital I/O unit

The following mean execution times have been measured for the digital I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (in): 1 kHz		Frequency (in$_{max}$): 10 kHz		Frequency (in): 1 kHz		Frequency (in$_{max}$): 10 kHz	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_pin_io_init`	17.39 µs	38.54 µs	16.54 µs	53.01 µs	16.97 µs	38.25 µs	38.92 µs	70.61 µs
`ds2201_pin_io_clear`	0.60 µs	17.22 µs	0.61 µs	17.21 µs	0.61 µs	16.81 µs	0.61 µs	24.52 µs
`ds2201_pin_io_in`	17.79 µs	18.39 µs	18.26 µs	18.88 µs	18.12 µs	18.72 µs	17.17 µs	17.77 µs
`ds2201_pin_io_set`	0.60 µs	17.13 µs	0.61 µs	16.75 µs	0.61 µs	17.55 µs	0.61 µs	37.06 µs
`ds2201_pin_io_write`	0.60 µs	17.34 µs	0.61 µs	17.45 µs	0.61 µs	17.11 µs	0.61 µs	48.32 µs

Serial Interface

The following mean execution times have been measured for the serial interface:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (in): 1 kHz		Frequency (in$_{max}$): 10 kHz		Frequency (in): 1 kHz		Frequency (in$_{max}$): 10 kHz	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_serial_port_init`	102.05 µs	102.65 µs	167.91 µs	168.52 µs	135.12 µs	135.72 µs	135.37 µs	135.97 µs
`ds2201_rx_byte`	17.75 µs	18.35 µs	47.65 µs	48.26 µs	20.07 µs	20.67 µs	20.66 µs	21.26 µs
`ds2201_tx_byte`	17.62 µs	38.16 µs	50.25 µs	70.65 µs	44.97 µs	70.72 µs	44.47 µs	70.86 µs

Execution Times When Using the 4DF Application on DS1006-Based Systems

Introduction

Because the execution times depend on the frequency, the measurement considers two different test conditions:

- The 4DF application generates a 1 kHz square-wave signal on 4 channels.
- The 4DF application generates a 4 kHz square-wave signal on 4 channels.

Timing I/O unit

The following mean execution time has been measured for the timing I/O unit:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (out): 1 kHz		Frequency (out$_{max}$): 4 kHz		Frequency (out): 1 kHz		Frequency (out$_{max}$): 4 kHz	
	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave	Master	Master + Slave
`ds2201_dtof_4`	15.00 µs	30.19 µs	15.23 µs	30.42 µs	15.16 µs	30.32 µs	15.16 µs	30.32 µs

Serial Interface

The following mean execution times have been measured for the serial interface:

Function	DS1006 with 2.6 GHz				DS1006 with 3.0 GHz			
	Frequency (out): 1 kHz		Frequency (out$_{max}$): 4 kHz		Frequency (out): 1 kHz		Frequency (out$_{max}$): 4 kHz	
	Master	Slave	Master	Slave	Master	Slave	Master	Slave
`ds2201_serial_port_init`	96.50 µs	97.11 µs	96.56 µs	97.17 µs	96.55 µs	97.15 µs	96.60 µs	97.20 µs
`ds2201_rx_byte`	19.15 µs	19.75 µs	20.15 µs	20.75 µs	19.63 µs	20.23 µs	19.65 µs	20.24 µs
`ds2201_tx_byte`	19.42 µs	38.07 µs	19.72 µs	38.34 µs	19.72 µs	38.34 µs	19.72 µs	38.33 µs