DS4004 HIL Digital I/O Board

Features

Release 2021-A – May 2021

dSPACE

How to Contact dSPACE

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

How to Contact dSPACE Support

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

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

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

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

Software Updates and Patches

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

Important Notice

Contents

About This Document

Contents

This document provides information on the features of the DS4004 HIL Digital I/O Board. You need this information to implement the DS4004 functions.

The Introduction to the Features of the DS4004 on page 7 provides a diagram of the board's architecture and an overview of the board's hardware and software features.

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.

Introduction to the Features of the DS4004

Introduction

The DS4004 HIL Digital I/O Board comprises 96 digital I/O channels with signal conditioning arranged in three identical ports. Each channel is software-configurable as digital I/O or timing I/O. The board can be connected to up to six independent external supply voltages (two per port).

Where to go from here

Information in this section

DS4004 Architecture

Introduction

The DS4004 provides 96 digital I/O channels with signal conditioning arranged in three ports.

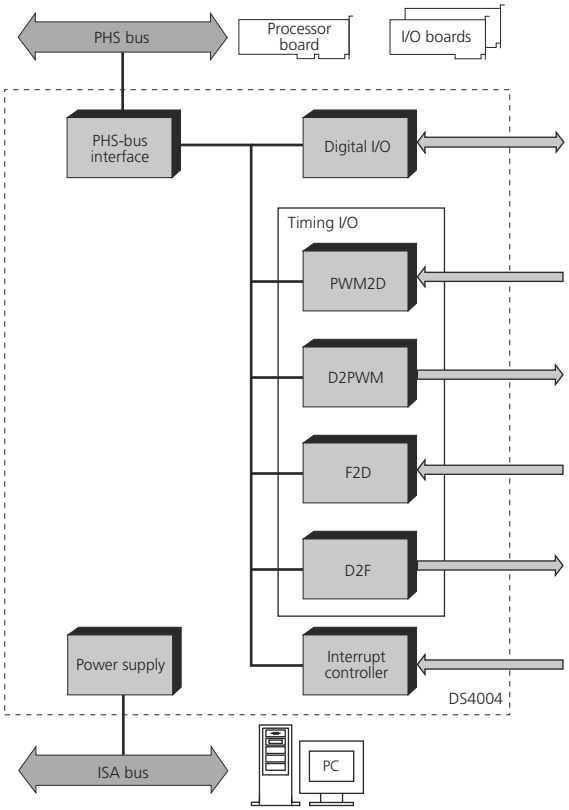
Functional units

The DS4004 includes the following functional units:

Multi I/O interface The DS4004 contains a multi I/O interface that provides digital I/O, timing I/O (including PWM signal generation and measurement), and interrupt control.

Voltage systems The DS4004 supports up to six independent voltage systems in the voltage range 5 V … 60 V. The voltages must be supplied externally.

Block diagram The illustration shows a block diagram of the DS4004.



PWM2D	PWM signal measurement
D2PWM	PWM signal generation
F2D	Frequency measurement
D2F	Square-wave signal generation

DS4004 Feature Overview

Introduction

The DS4004 HIL Digital I/O Board comprises 96 digital I/O channels with signal conditioning arranged in three identical ports. Each channel is software-configurable as digital I/O or timing I/O. The board can be connected to up to six independent external supply voltages (two per port).

Digital I/O

All the 3 x 32 channels of the board can be configured individually as digital input or output, refer to Digital I/O on page 11.

Timing I/O

All the 3 x 32 channels of the board can be configured individually for:

- *PWM signal measurement*, refer to Basics on PWM Signal Measurement (PWM2D) on page 30.
- *PWM signal generation*, refer to Basics on PWM Signal Generation (D2PWM) on page 37.
- *Frequency measurement*, refer to Basics on Frequency Measurement (F2D) on page 48.
- *Square-wave signal generation*, refer to Basics on Square-Wave Signal Generation (D2F) on page 54.

Interrupt control

The DS4004 provides six interrupts, which you can use in your Simulink® model and your handcoded model, refer to Interrupt Control on page 65.

Limitations

There are some limitations when you work with the DS4004. Refer to Limitations on page 73.

DS4004 Interfaces

Introduction

The DS4004 has interfaces for connection to a PHS-bus-based system and external devices.

Integration into a PHS-bus-based system

As an I/O board, the DS4004 is always part of a PHS-bus-based system. While the DS4004 measures and simulates the signals required, the processor board performs the calculation of the real-time model. That is, applications using DS4004 I/O features are implemented on the processor board.

Communication between the processor board and the I/O board is performed via the peripheral high-speed bus, the PHS++ bus, which is called "PHS bus" in the documentation.

Partitioning the PHS bus with the DS802 With the DS802 PHS Link Board you can spatially partition the PHS bus by arranging the I/O boards in several expansion boxes.

The DS802 can be used in combination with many types of available dSPACE I/O boards. However, some I/O boards and some functionalities of specific I/O boards are not supported.

The I/O board support depends on the dSPACE software release which you use. For a list of supported I/O boards, refer to DS802 Data Sheet (PHS Bus System Hardware Reference 🕮).

Connection to external devices

There are three 50-pin I/O connectors (P1, P2, P3) to connect external devices to the DS4004, refer to I/O Connectors (P1, P2, P3) (PHS Bus System Hardware Reference 🕮).

Related topics

Basics

DS4004 Components (PHS Bus System Hardware Reference 🕮)
Signal Connection to External Devices (PHS Bus System Hardware Reference 🕮)

Digital I/O

Introduction

All the 3 x 32 channels of the DS4004 can be used for digital I/O.

Where to go from here

Information in this section

Basics on Standard I/O

Introduction

You can use each of the 3 x 32 channels of the board as digital input or output.

I/O characteristics

Input voltage The input channels are fully operational up to 60 V. After software initialization, the input threshold is set to 2.5 V. You can set the input threshold in the range 1.0 … 23.8 V. The input channels have a hysteresis voltage of approximately 0.2 V. This value is fixed and cannot be changed. For example, if the input threshold is 2.5 V, the input signal must exceed 2.6 V to be detected as high, and must fall below 2.4 V to be detected as low.

Output voltage The output channels of the DS4004 have push-pull drivers running from external power supply sources (VBAT). Each port (P1 … P3) provides two power supply rails (Px_VBAT1 and Px_VBAT2). These power supply rails are independent of one another. Thus, you can use up to six independent supply rails with the DS4004.

The maximum output current per channel is ±50 mA. The functional voltage range is +5 V … +60 V. Self-resetting multifuses protect the outputs against short circuits to GND or to any voltage up to +60 V. Protection against a short circuit to a negative voltage is also provided, if the voltage difference of the output pin to VBAT is below 60 V.

Note

The multifuses are designed for occasional faults only. They are not suitable for failure simulation.

After board initialization, all outputs remain disabled in high impedance (high-Z) state automatically.

Note

Before operating the output channels, you must connect an external power supply (VBAT) to at least one of the two supply rails (VBAT1 or VBAT2) of the port.

Low-side and high-side switches The output state of each I/O channel depends on its individual settings for the low-side switch and the high-side switches. You can use the supply rails L (GND), H1 (VBAT1), and H2 (VBAT2).

- If you set the low-side switch L (GND), the digital output channel is set to low-side switch mode.
- If you set the high-side switch H1 (VBAT1) *or* H2 (VBAT2), the digital output channel is actively driven in high-side switch mode.
- If you set the high-side switches H1 (VBAT1) *and* H2 (VBAT2), the digital output channel is also actively driven in high-side switch mode, but the output voltage is driven to the highest supply voltage (VBAT1 or VBAT2).

- If you set low-side switch L (GND) and the high-side switches H1 (VBAT1) *and/or* H2 (VBAT2), the digital output channel is actively driven in push-pull mode. Push-pull driver mode means that the output source is actively driven to both high and low level.

Switch Settings [1]			Input of the Output Circuit [2]	Output Px_IO1 ... Px_IO32 [2], [3]	Description [3]
L (GND)	H1 (VBAT1)	H2 (VBAT2)			
0	0	0	0 or 1	High-Z	Individual output disabled.[4]
1	0	0	0	GND	Low-side switch
1	0	0	1	High-Z	Low-side switch
0	1	0	0	High-Z	High-side switch set to Px_VBAT1
0	1	0	1	Px_VBAT1	High-side switch set to Px_VBAT1
0	0	1	0	High-Z	High-side switch set to Px_VBAT2
0	0	1	1	Px_VBAT2	High-side switch set to Px_VBAT2
0	1	1	0	High-Z	High-side switch set to maximum value (Px_VBAT1 or Px_VBAT2)
0	1	1	1	max. value (Px_VBAT1 or Px_VBAT2)	High-side switch set to maximum value (Px_VBAT1 or Px_VBAT2)
1	1	0	0	GND	Push-pull output set to Px_VBAT1
1	1	0	1	Px_VBAT1	Push-pull output set to Px_VBAT1
1	0	1	0	GND	Push-pull output set to Px_VBAT2
1	0	1	1	Px_VBAT2	Push-pull output set to Px_VBAT2
1	1	1	0	GND	Push-pull output set to maximum value (Px_VBAT1 or Px_VBAT2)
1	1	1	1	max. value (Px_VBAT1 or Px_VBAT2)	Push-pull output set to maximum value (Px_VBAT1 or Px_VBAT2)

[1] 0 = switch disabled, 1 = switch enabled
[2] Refer to Digital Outputs (PHS Bus System Hardware Reference 🕮)
[3] x is a placeholder for port/connector number 1 ... 3
[4] With RTLib functions, the channel can be used as digital input or PWM input.

For more details, refer to Digital Outputs (PHS Bus System Hardware Reference 🕮).

RTI and RTLib support

You can use the board's digital I/O unit via RTI blocks or RTLib functions.

RTI users refer to

- How to Read a Single Digital Input via RTI on page 15
- How to Read all the 32 Digital Inputs of a Port via RTI on page 16
- How to Write to a Single Digital Output via RTI on page 18
- How to Write to all the 32 Digital Outputs of a Port via RTI on page 20

RTLib users refer to

- Reading a Digital Input via RTLib Functions on page 23
- Writing to a Digital Output via RTLib Functions on page 25

Execution times

The execution times required by the RTLib functions have been measured. For details on the results and the measurement setup, refer to Function Execution Times (DS4004 RTLib Reference 🕮).

I/O mapping

Note

The channels on the board can basically be used as follows:

- Each channel can be used for digital I/O.
- Each channel can be used for timing I/O.
- Channels 1 and 2 (signals Px_IO1 and Px_IO2) of each port can also be used for interrupt control.

However, there are limitations in how you can combine these uses, refer to Conflicting I/O Features on page 74.

The following table shows the mapping of the I/O signals to the corresponding ports and channels, as used in RTI and RTLib.

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO1	Px	1	1	Digital I/O shared with timing I/O and interrupt control
Px_IO2	Px	34	2	Digital I/O shared with timing I/O and interrupt control
Px_IO3	Px	18	3	Digital I/O shared with timing I/O
Px_IO4	Px	2	4	Digital I/O shared with timing I/O
Px_IO5	Px	35	5	Digital I/O shared with timing I/O
Px_IO6	Px	19	6	Digital I/O shared with timing I/O
Px_IO7	Px	3	7	Digital I/O shared with timing I/O
Px_IO8	Px	36	8	Digital I/O shared with timing I/O
Px_IO9	Px	20	9	Digital I/O shared with timing I/O
Px_IO10	Px	4	10	Digital I/O shared with timing I/O
Px_IO11	Px	37	11	Digital I/O shared with timing I/O
Px_IO12	Px	21	12	Digital I/O shared with timing I/O
Px_IO13	Px	5	13	Digital I/O shared with timing I/O
Px_IO14	Px	38	14	Digital I/O shared with timing I/O
Px_IO15	Px	22	15	Digital I/O shared with timing I/O
Px_IO16	Px	6	16	Digital I/O shared with timing I/O
Px_IO17	Px	39	17	Digital I/O shared with timing I/O
Px_IO18	Px	23	18	Digital I/O shared with timing I/O
Px_IO19	Px	7	19	Digital I/O shared with timing I/O
Px_IO20	Px	40	20	Digital I/O shared with timing I/O
Px_IO21	Px	24	21	Digital I/O shared with timing I/O
Px_IO22	Px	8	22	Digital I/O shared with timing I/O
Px_IO23	Px	41	23	Digital I/O shared with timing I/O
Px_IO24	Px	25	24	Digital I/O shared with timing I/O

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO25	Px	9	25	Digital I/O shared with timing I/O
Px_IO26	Px	42	26	Digital I/O shared with timing I/O
Px_IO27	Px	26	27	Digital I/O shared with timing I/O
Px_IO28	Px	10	28	Digital I/O shared with timing I/O
Px_IO29	Px	43	29	Digital I/O shared with timing I/O
Px_IO30	Px	27	30	Digital I/O shared with timing I/O
Px_IO31	Px	11	31	Digital I/O shared with timing I/O
Px_IO32	Px	44	32	Digital I/O shared with timing I/O

[1] x is a placeholder for port/connector number 1 ... 3.

Related topics

References

Digital I/O Functions (DS4004 RTLib Reference 🕮)
Digital Inputs (PHS Bus System Hardware Reference 🕮)
Digital Outputs (PHS Bus System Hardware Reference 🕮)

How to Read a Single Digital Input via RTI

Objective

You can use the DS4004BIT_IN_BLx RTI block to read a single bit of a digital I/O port channel-wise.

Basics

The block provides channel-wise read access to a single bit of a digital I/O port. You can specify the threshold level on which the input signal is interpreted as logical 0 or 1. The block's output is a binary value representing the state of the digital input channel.

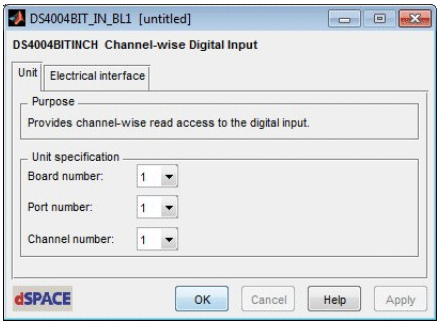
For further information, refer to Basics on Standard I/O on page 12.

Method

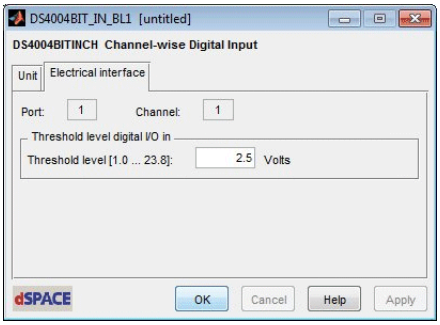
To read a single digital input via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004BIT_IN_BLx block from the library to your Simulink model. Double-click the block to open it.

3 Select the block's **Unit** page to specify the board number, the port number, and the channel number.



4 Select the block's **Electrical interface** page to specify the threshold level for the selected digital input channel in the range 1.0 … 23.8 V.



Result

You added the DS4004BIT_IN_BLx block to your Simulink model and configured it to read a single digital input.

The digital input signal connected to the corresponding pin of the DS4004 can now be examined by the model. The block's outport delivers a binary value to the model representing the state of the signal. If the voltage of the input signal is below the defined threshold level, the block's output is logical 0. If the voltage exceeds the threshold level, the block's output is logical 1.

Related topics

References

DS4004BIT_IN_BLx (DS4004 RTI Reference 🕮)

How to Read all the 32 Digital Inputs of a Port via RTI

Objective

You can use the DS4004BIT_IN32_BLx RTI block to read the 32 bits of a digital I/O port word-wise.

Basics

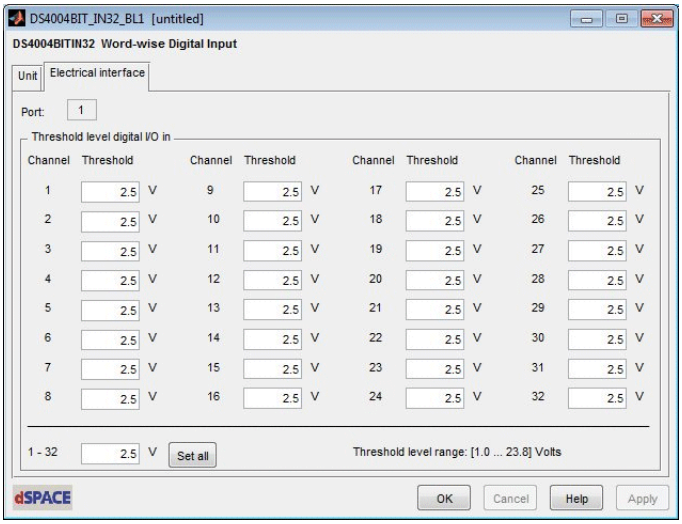
The block provides word-wise read access to the 32 channels (bits) of a digital I/O port. You can specify the threshold level on which an input signal is interpreted as logical 0 or 1 for each channel individually. The block's output is a decimal value in the range 0 ... 4294967295 (0 … 2^{32}-1). The single bits of this value correspond directly to the related channels, i.e., the LSB corresponds to the state of channel 1 and the MSB corresponds to the state of channel 32.

For further information, refer to Basics on Standard I/O on page 12.

Method

To read a digital input via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004BIT_IN32_BLx block from the library to your Simulink model. Double-click the block to open it.
3. Select the block's Unit page to specify the board number and the port number.
4. Select the block's Electrical interface page to specify the threshold level for the input channels in the range 1.0 … 23.8 V.



Tip

You can set the threshold level for all the 32 channels at once by using the Set all button.

Result

You added the DS4004BIT_IN32_BLx block to your Simulink model and configured it to read all the 32 channels (bits) of a digital input port at once.

The digital input signals connected to the corresponding port of the DS4004 can now be examined by the model. The block's output to Simulink is a 32-bit value. Each bit represents one of the 32 input signals. If the voltage of an input signal is

below the defined threshold level, the corresponding bit is set to 0. If the voltage exceeds the threshold level, the bit is set to 1.

Related topics

References

DS4004BIT_IN32_BLx (DS4004 RTI Reference 🕮)

How to Write to a Single Digital Output via RTI

Objective

You can use the **DS4004BIT_OUT_BLx** RTI block to write a single bit of a digital I/O port channel-wise.

Basics

Block description The block provides channel-wise write access to a single bit of a digital I/O port. The block's input is a binary value.

For information on the I/O mapping, refer to Basics on Standard I/O on page 12.

Initialization During the model initialization phase, the **Initial output state** is written to the selected channel (bit) to ensure a defined output during this simulation phase. This is especially useful if the channel is used in a triggered or enabled subsystem that is not executed right from the start of the simulation.

Termination With the block's **Termination** settings, you can specify an output state of the channel on termination to drive your external hardware into a safe final condition.

The possible termination states at the end of the simulation are:

- The output is set to high impedance (high-Z) state.
- The output holds its last output value.
- The output is set to a definite output value.

The specified termination values of I/O channels are set when the simulation executes its termination function by setting the `simState` variable to STOP. If you stop the real-time application by using ControlDesk's **Stop RTP** command, the processor resets immediately without executing termination functions. The current values of the I/O channels are kept and the specified termination values are not set.

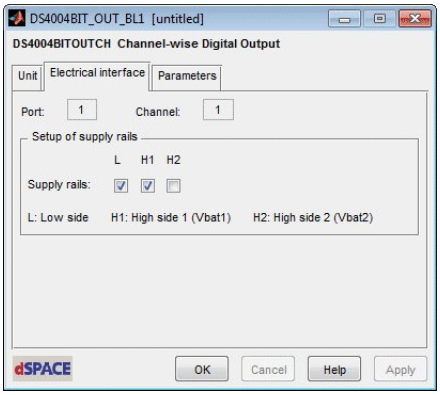
Preconditions

Before operating the output channels, you must connect an external power supply (VBAT) to at least one of the two supply rails (VBAT1 or VBAT2) of the port.

Method

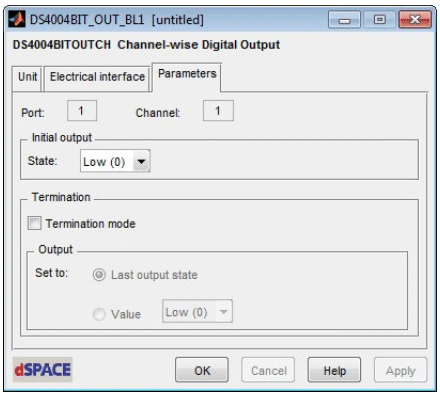
To write to a single digital output via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004BIT_OUT_BLx block from the library to your Simulink model. Double-click the block to open it.
3. Select the block's Unit page to specify the board number, the port number, and the channel number.
4. Select the block's Electrical interface page to specify the low-side and high-side switches.



   To set a switch, enable the corresponding checkbox.

5. Select the block's Parameters page, to specify its initial output state and its termination output state.



6. Specify the State of the Initial output as *Low(0)* or *High(1)*.
7. Specify the Termination for the output channel:
   - To set the output channel to high impedance state, clear the Termination mode checkbox.
   - To hold the last output value, select the Termination mode checkbox and the Last output state option button.
   - To set a definite output value, select the Termination mode checkbox and the Value option button and specify the value as *Low(0)* or *High(1)*.

Result

You added the **DS4004BIT_OUT_BLx** block to your Simulink model, configured it, and specified its output value for the initialization and termination of the simulation.

You can now connect a binary signal to the block's inport in Simulink to control the output channel during real-time simulation. The signal will drive the output voltage of the corresponding pin until the simulation stops. If the output is configured for push-pull mode, a logical 1 in Simulink will cause a high potential on the pin, and a logical 0 GND potential.

Related topics

References

DS4004BIT_OUT_BLx (DS4004 RTI Reference 🕮)
simState (RTI and RTI-MP Implementation Reference 🕮)
Stop RTP (ControlDesk Platform Management 🕮)

How to Write to all the 32 Digital Outputs of a Port via RTI

Objective

You can use the **DS4004BIT_OUT32_BLx** RTI block to write the 32 bits of a digital I/O port word-wise.

Basics

Block description The block provides word-wise write access to the 32 channels (bits) of a digital I/O port. The block's input is a decimal value in the range 0 ... 4294967295 (0 … 2^{32}-1). The single bits of this value correspond directly to the related channels, i.e., the LSB corresponds to the setting for channel 1 and the MSB corresponds to the setting for channel 32.

For information on the I/O mapping, refer to Basics on Standard I/O on page 12.

Initialization During the model initialization phase, the initial digital output states specified by the **Initial output** settings are written to the 32 channels to ensure a defined output during this simulation phase. This is especially useful if the channels are used in a triggered or enabled subsystem that is not executed right from the start of the simulation.

Termination With the block's **Termination** settings, you can specify the output states of the 32 channels on termination to drive your external hardware into a safe final condition.

The possible termination states at the end of the simulation are:

- All digital outputs are set to high impedance (high-Z) state.
- Each output holds its last output value.
- Each output is set to a definite output value.

The specified termination values of I/O channels are set when the simulation executes its termination function by setting the `simState` variable to STOP. If

you stop the real-time application by using ControlDesk's **Stop RTP** command, the processor resets immediately without executing termination functions. The current values of the I/O channels are kept and the specified termination values are not set.

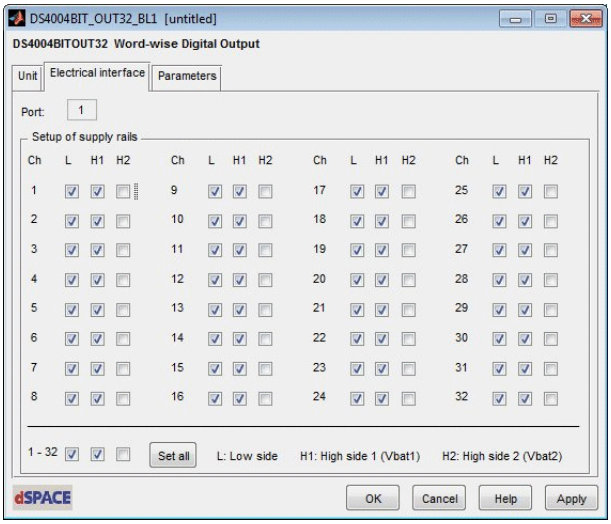
Preconditions

Before operating the output channels, you must connect an external power supply (VBAT) to at least one of the two supply rails (VBAT1 or VBAT2) of the port.

Method

To write to all the 32 digital outputs of a port via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the **DS4004BIT_OUT32_BLx** block from the library to your Simulink model. Double-click the block to open it.
3. Select the block's **Unit** page to specify the board number and the port number.
4. Select the block's **Electrical interface** page to specify the low-side and high-side switches.

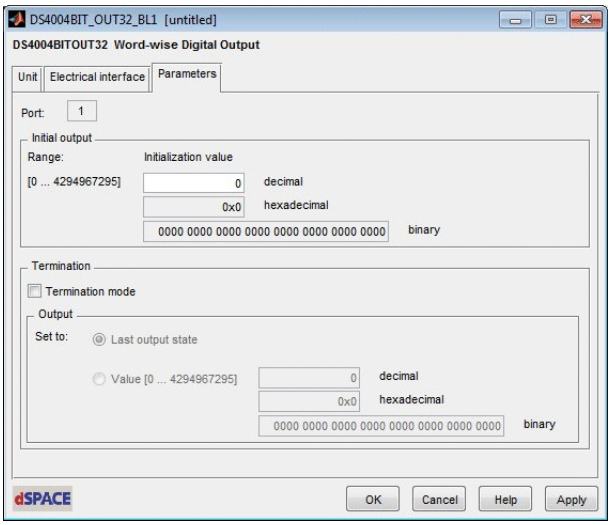


To set a low-side or high-side switch, enable the corresponding checkbox.

Tip

You can set the switches (L, H1, H2) for all the 32 channels at once by using the **1 - 32** (L, H1, H2) checkboxes and the **Set all** button.

5 Select the block's **Parameters** page, to specify its initial output state and its termination output state.



6 Specify the **Initialization value** of the **Initial output** in the range 0 ... 4294967295 (0 … 2^{32}-1).

The value is also displayed in hexadecimal and binary format. The single bits of this value correspond directly to the related channels, i.e., the LSB corresponds to the setting for channel 1 and the MSB corresponds to the setting for channel 32.

7 Specify the **Termination** for the output channels:

- To set all the output channels to high impedance state, clear the **Termination mode** checkbox.
- To hold all the last output values, select the **Termination mode** checkbox and the **Last output state** option button.
- To set definite output values, select the **Termination mode** checkbox and the **Value** option button and enter the desired output value in the range 0 ... 4294967295 (0 … 2^{32}-1).

  The value is also displayed in hexadecimal and binary format. The single bits of this value correspond directly to the related channels, i.e., the LSB corresponds to the setting for channel 1 and the MSB corresponds to the setting for channel 32.

Result

You added the **DS4004BIT_OUT32_BLx** block to your Simulink model, configured it, and specified its output values for the initialization and termination of the simulation.

You can now connect a decimal value to the block's inport in Simulink, to control the corresponding pins of the DS4004 during real-time simulation. For example, the decimal input value of "1762984" in Simulink will cause a bit setting on the port's pins of "0000 0000 0001 1010 1110 0110 1010 1000". If a channel is configured for push-pull mode, each logical 1 will cause a high potential on the corresponding pin, and each logical 0 GND potential.

Related topics

References

DS4004BIT_OUT32_BLx (DS4004 RTI Reference 🕮)
simState (RTI and RTI-MP Implementation Reference 🕮)
Stop RTP (ControlDesk Platform Management 🕮)

Reading a Digital Input via RTLib Functions

Introduction

You can use RTLib functions to read from digital I/O channels.

Related RTLib functions

- `ds4004_init`
- `ds4004_digin_init`
- `ds4004_bit_in`
- `ds4004_bit_in32`

Example

The following code lines show the usage of the relevant RTLib functions:

```
UInt32 value;                    /* store digital value */
ds4004_init(DS4004_1_BASE);      /* initialize the DS4004 */

/* initialize all the channels of port 1 as digital input with threshold 2.5V */
ds4004_digin_init(DS4004_1_BASE,  /* base address of the DS4004 */
        1,                       /* select port 1 */
        0xFFFFFFFF,              /* select all the 32 channels */
        2.5);                    /* set input threshold to 2.5V */

/* read complete port 1 */
value = ds4004_bit_in32(DS4004_1_BASE, /* base address of the DS4004 */
        1);                      /* select port 1 */

/* initialize digital input for channel 2 of port 2 with threshold 5V */
ds4004_digin_init(DS4004_1_BASE,  /* base address of the DS4004 */
        2,                       /* select port 2 */
        DS4004_MASK_CH2,         /* select channel 2 */
        5);                      /* set input threshold to 5V */

/* initialize digital input for channel 3 of port 2  with threshold 12V */
ds4004_digin_init(DS4004_1_BASE,  /* base address of the DS4004 */
        2,                       /* select port 2 */
        DS4004_MASK_CH3,         /* select channel 3 */
        12);                     /* set input threshold to 12V */

/* read channel 2 and channel 3 of port 2 */
value = ds4004_bit_in(DS4004_1_BASE, /* base address of the DS4004 */
        2,                       /* select port 2 */
        DS4004_MASK_CH2 | DS4004_MASK_CH3); /* select ch. 2 and 3 */

/* if signal of channel 10 of port 1 is high */
if(ds4004_bit_in(DS4004_1_BASE, 1, DS4004_MASK_CH10))
{
  ...                            /* do something ... */
}
```

Description

Before reading from I/O channels, you have to initialize the channels and set their threshold levels.

To select channels, you can use the predefined symbols `DS4004_MASK_CH1` ... `DS4004_MASK_CH32`. To select more than one channel, use a logical OR operation for the predefined symbols. The result of the operation is a bit mask for the selected channels.

By using the `ds4004_digin_init` function several times, you can set the threshold level for each channel individually in the range 1 V ... 23.8 V with a resolution of 0.1 V. The input channels have a fixed hysteresis voltage of approximately 0.2 V. For example, if the defined threshold level is 2.5 V, the input signal must exceed 2.6 V to be detected as high (logical 1), and must fall below 2.4 V to be detected as low (logical 0).

The `ds4004_bit_in` function returns a 32-bit value of the logical AND operation between the parameter for the channel mask and the input value of the specified port.

The `ds4004_bit_in32` function returns a 32-bit value which represents all the 32 inputs of the specified port.

Related topics

Basics

Basics on Standard I/O 12

Examples

Example of Digital Input Functionality (DS4004 RTLib Reference 🕮)

References

ds4004_bit_in (DS4004 RTLib Reference 🕮)
ds4004_bit_in32 (DS4004 RTLib Reference 🕮)
ds4004_digin_init (DS4004 RTLib Reference 🕮)
ds4004_init (DS4004 RTLib Reference 🕮)

Writing to a Digital Output via RTLib Functions

Introduction

You can use RTLib functions to write to digital I/O channels.

Related RTLib functions

- `ds4004_init`
- `ds4004_digout_init`
- `ds4004_digout_mode_set`
- `ds4004_bit_out`
- `ds4004_bit_out32`

Example

The following code lines show the usage of the relevant RTLib functions:

```
ds4004_init(DS4004_1_BASE);      /* initialize the DS4004 */

/* initialize all the channels of port 1 as digital output (push-pull mode) */
ds4004_digout_init(DS4004_1_BASE, /* base address of the DS4004 */
        1,                       /* select port 1 */
        0xFFFFFFFF,              /* select all the 32 channels */
        DS4004_LS_ENABLE,        /* enable low-side switch */
        DS4004_HS_VBAT1_ENABLE); /* enable high-side switch VBAT1 */

/* set the initial value for port 1 */
ds4004_bit_out32(DS4004_1_BASE,  /* base address of the DS4004 */
        1,                       /* select port 1 */
        0x00000000);             /* set all the bits to 0 */

/* enable (global) digital output mode for port 1 */
ds4004_digout_mode_set(DS4004_1_BASE, /* base address of the DS4004 */
        DS4004_MASK_PORT1,       /* select port 1 */
        DS4004_DIGOUT_ENABLE);   /* enable digital outputs */

/* write to single bits of port 1 */
ds4004_bit_out(DS4004_1_BASE,    /* base address of the DS4004 */
        1,                       /* select port 1 */
        DS4004_MASK_CH1 | DS4004_MASK_CH32, /* select ch. 1 and 32 */
        0x80000001);             /* set channel 1 and 32 to 1 */

/* write to all the 32 bits of port 1 */
ds4004_bit_out32(DS4004_1_BASE,  /* base address of the DS4004 */
        1,                       /* select port 1 */
        0xAAAA5555);             /* set the port to 0xAAAA5555 */
```

Description

Before operating the output channels, you must connect an external power supply (VBAT) to at least one of the two supply rails (VBAT1 or VBAT2) of the port.

Before writing to I/O channels, you have to call the `ds4004_digout_init` function to initialize the channels and set their low-side and high-side switches.

To select channels, you can use the predefined symbols `DS4004_MASK_CH1` ... `DS4004_MASK_CH32`. To select more than one channel, use a logical OR operation for the predefined symbols. The result of the operation is a bit mask for the selected channels.

By calling the `ds4004_digout_init` function several times, you can initialize low-side and high-side switches of channels individually.

Use the `ds4004_bit_out` function to set I/O channels channel-wise. Use the `ds4004_bit_out32` function to set all the 32 channels (bits) of a specified port.

You have to enable the digital output circuits for the specified port(s) by calling the `ds4004_digout_mode_set` function.

Related topics

Basics

Basics on Standard I/O..... 12

Examples

Example of Digital Output Functionality (DS4004 RTLib Reference 🕮)

References

ds4004_bit_out (DS4004 RTLib Reference 🕮)
ds4004_bit_out32 (DS4004 RTLib Reference 🕮)
ds4004_digout_init (DS4004 RTLib Reference 🕮)
ds4004_digout_mode_set (DS4004 RTLib Reference 🕮)
ds4004_init (DS4004 RTLib Reference 🕮)

Timing I/O

Introduction

All the 3 x 32 channels of the DS4004 can be used for timing I/O.

Where to go from here

Information in this section

Basics on PWM Signal Measurement (PWM2D)

Introduction

You can specify each I/O channel individually as PWM input to analyze the duty cycle and the period of PWM signals.

Basics

To evaluate frequencies and duty cycles, digital pulses have to be measured at high speed and transferred to the processor board that performs the analysis.

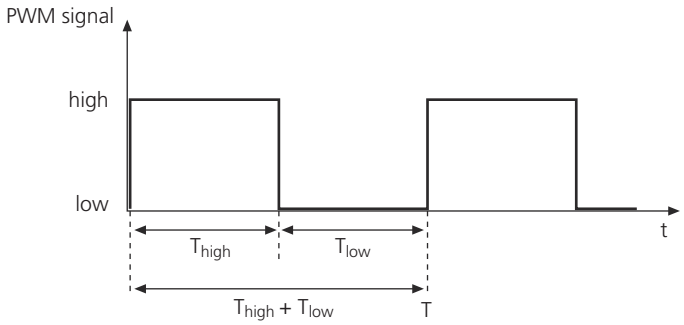
Note

The measurement resolution depends on the selected measurement range for the PWM period.
Due to quantization effects, you will encounter considerable deviations between the input PWM period and the measured PWM period, especially for higher PWM frequencies, refer to Quantization Effects on page 73. To avoid poor measurement resolution, you should therefore select the period range with the best possible resolution (resolution values as small as possible).

PWM characteristics

At 16-bit resolution, you can measure

- The period in the range 6.7 µs ... 107.3 s
- The duty cycle in the range 0 … 100 % (scaled to 0 ... 1)



The measurements of T_{low} and T_{high} are used to calculate

- Period = $T_{low} + T_{high}$
- Duty cycle = $T_{high} / (T_{low} + T_{high})$

For the best possible measurements, you have to select the expected period range of the PWM type signal to be measured.

Measurement range

Depending on the measurement range, the periods can be measured within the following limits and resolutions:

Range	Minimum Period		Maximum Period	Resolution
	Theoretical	Practical		
1	200 ns	6.7 µs	3.27 ms	50 ns
2	400 ns	6.7 µs	6.55 ms	100 ns
3	800 ns	6.7 µs	13.1 ms	200 ns
4	1.6 µs	6.7 µs	26.2 ms	400 ns
5	3.2 µs	6.7 µs	52.4 ms	800 ns

Range	Minimum Period		Maximum Period	Resolution
	Theoretical	Practical		
6	6.4 µs	6.7 µs	104 ms	1.6 µs
7	12.8 µs	12.8 µs	209 ms	3.2 µs
8	25.6 µs	25.6 µs	419 ms	6.4 µs
9	51.2 µs	51.2 µs	838 ms	12.8 µs
10	103 µs	103 µs	1.67 s	25.6 µs
11	205 µs	205 µs	3.35 s	51.2 µs
12	410 µs	410 µs	6.71 s	103 µs
13	820 µs	820 µs	13.4 s	205 µs
14	1.64 ms	1.64 ms	26.8 s	410 µs
15	3.28 ms	3.28 ms	53.6 s	820 µs
16	6.56 ms	6.56 ms	107.3 s	1.64 ms

The maximum period applies to 0% < duty cycle < 100%.

If these ranges are exceeded, the measurement will be faulty. Note the following restrictions with values outside the practical period range:

Range	Restriction
PWM period < theoretical minimum period	No precise measurement (undersampling). Some high or low periods might not be recognized.
PWM period < 6.7 µs, T_{high} or T_{low} < 3 µs	Pulses that you want to measure might be missing. (The frequency range of input stage is exceeded.)
Max. period < PWM period < 2 · max. period	PWM signal measurement with restricted duty cycle.
PWM period > 2 · maximum period	PWM signal measurement with duty cycle alternating between "0" and "1". (The returned period is the maximum period.)

The duty cycle values 0 (input constant low) and 1 (input constant high) are measured properly.

Note

Signal periods and resolution

Each high period and each low period of the measured signal must be longer (not equal) than the resolution to avoid missing pulses.

Update mode

The DS4004 supports two update modes that describe the time interval when the measured values are updated:

Asynchronous mode The measured values are updated *asynchronously* at the end of each T_{high} and T_{low} period of the PWM signal.

Synchronous mode The measured values are updated *synchronously* at the end of each T_{low} period of the PWM signal.

Input voltage

The input channels are fully operational up to 60 V. After software initialization, the input threshold is set to 2.5 V. You can set the input threshold in the range 1.0 … 23.8 V. The input channels have a hysteresis voltage of approximately 0.2 V. This value is fixed and cannot be changed. For example, if the input threshold is 2.5 V, the input signal must exceed 2.6 V to be detected as high, and must fall below 2.4 V to be detected as low.

RTI/RTLib support

You can measure PWM signals via an RTI block or RTLib functions.

- *RTI users* refer to How to Measure a Frequency via RTI on page 51.
- *RTLib users* refer to Measuring a PWM Signal via RTLib Functions on page 36.

Execution times

The execution times required by the RTLib functions have been measured. For details on the results and the measurement setup, refer to Function Execution Times (DS4004 RTLib Reference 🕮).

I/O mapping

Note

The channels on the board can basically be used as follows:

- Each channel can be used for digital I/O.
- Each channel can be used for timing I/O.
- Channels 1 and 2 (signals Px_IO1 and Px_IO2) of each port can also be used for interrupt control.

However, there are limitations in how you can combine these uses, refer to Conflicting I/O Features on page 74.

The following table shows the mapping of the I/O signals to the corresponding ports and channels, as used in RTI and RTLib.

I/O Signal [1)]	Port/ Connector [1)]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO1	Px	1	1	Digital I/O shared with timing I/O and interrupt control
Px_IO2	Px	34	2	Digital I/O shared with timing I/O and interrupt control
Px_IO3	Px	18	3	Digital I/O shared with timing I/O
Px_IO4	Px	2	4	Digital I/O shared with timing I/O
Px_IO5	Px	35	5	Digital I/O shared with timing I/O
Px_IO6	Px	19	6	Digital I/O shared with timing I/O
Px_IO7	Px	3	7	Digital I/O shared with timing I/O
Px_IO8	Px	36	8	Digital I/O shared with timing I/O
Px_IO9	Px	20	9	Digital I/O shared with timing I/O
Px_IO10	Px	4	10	Digital I/O shared with timing I/O

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO11	Px	37	11	Digital I/O shared with timing I/O
Px_IO12	Px	21	12	Digital I/O shared with timing I/O
Px_IO13	Px	5	13	Digital I/O shared with timing I/O
Px_IO14	Px	38	14	Digital I/O shared with timing I/O
Px_IO15	Px	22	15	Digital I/O shared with timing I/O
Px_IO16	Px	6	16	Digital I/O shared with timing I/O
Px_IO17	Px	39	17	Digital I/O shared with timing I/O
Px_IO18	Px	23	18	Digital I/O shared with timing I/O
Px_IO19	Px	7	19	Digital I/O shared with timing I/O
Px_IO20	Px	40	20	Digital I/O shared with timing I/O
Px_IO21	Px	24	21	Digital I/O shared with timing I/O
Px_IO22	Px	8	22	Digital I/O shared with timing I/O
Px_IO23	Px	41	23	Digital I/O shared with timing I/O
Px_IO24	Px	25	24	Digital I/O shared with timing I/O
Px_IO25	Px	9	25	Digital I/O shared with timing I/O
Px_IO26	Px	42	26	Digital I/O shared with timing I/O
Px_IO27	Px	26	27	Digital I/O shared with timing I/O
Px_IO28	Px	10	28	Digital I/O shared with timing I/O
Px_IO29	Px	43	29	Digital I/O shared with timing I/O
Px_IO30	Px	27	30	Digital I/O shared with timing I/O
Px_IO31	Px	11	31	Digital I/O shared with timing I/O
Px_IO32	Px	44	32	Digital I/O shared with timing I/O

[1] x is a placeholder for port/connector number 1 ... 3.

Related topics

Basics

Basics on PWM Signal Generation (D2PWM)..... 37

References

Digital Inputs (PHS Bus System Hardware Reference 🕮)
Timing I/O Functions (DS4004 RTLib Reference 🕮)

How to Measure a PWM Signal via RTI

Objective

You can use the DS4004PWM2D_BLx RTI block to measure the period and duty cycle of a PWM signal.

Basics

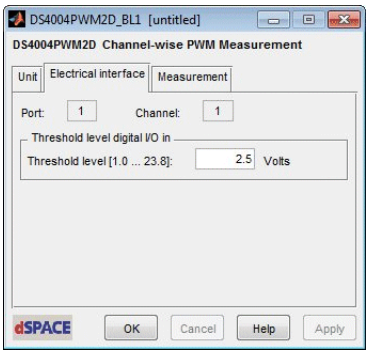
In a Simulink model, the block provides channel-wise read access to the duty cycle and period of a PWM signal. You can specify the threshold level on which the input signal is interpreted as logical 0 or 1.

For further information, refer to Basics on PWM Signal Measurement (PWM2D) on page 30.

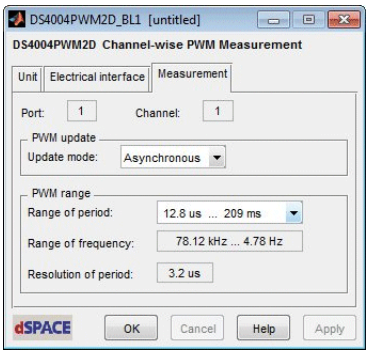
Method

To measure a PWM signal via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004PWM2D_BLx block from the library to your Simulink model. Double-click the block to open it.
3. Select the block's Unit page to specify the board number, the port number, and the channel number.
4. Select the block's Electrical interface page to specify the threshold level for the selected input channel in the range 1.0 … 23.8 V.



5. Select the block's Measurement page.



6. Specify the Update mode as *asynchronous* or *synchronous*.
7. Specify the expected period range for the square-wave signal to be measured.

   The corresponding frequency range and the measurement resolution for the selected period range are displayed (grayed out).

Result

You added the **DS4004PWM2D_BLx** block to your Simulink model and configured it for PWM signal measurement of a single input channel.

The input signal connected to the corresponding pin of the DS4004 can now be examined by the model. The block's outports deliver two values (of double data type) to the model, representing the signal's PWM period and duty cycle. The period of the measured signal is given in seconds. The duty cycle is scaled to the range 0 … 1.

Related topics

References

DS4004PWM2D_BLx (DS4004 RTI Reference 🕮)

Measuring a PWM Signal via RTLib Functions

Introduction

You can use RTLib functions to measure PWM signals.

Related RTLib functions

- `ds4004_init`
- `ds4004_pwm2d_init`
- `ds4004_pwm2d`

Example

The following code lines show the usage of the relevant RTLib functions:

```
dsfloat period, duty;             /* store the period and duty cycle */

ds4004_init(DS4004_1_BASE);       /* initialize the DS4004 */

/* initialize channel 1 of port 1 as PWM input */
ds4004_pwm2d_init(DS4004_1_BASE,  /* base address of the DS4004 */
        1,                        /* select port 1 */
        DS4004_MASK_CH1,          /* select channel 1 */
        2.5,                      /* set input threshold to 2.5V */
        DS4004_TIMING_RANGE1,     /* use timing range 1 (6.7µs ... 3.27ms) */
        DS4004_PWM2D);            /* use normal D2PWM mode (asynchronous) */

/* get the period and duty cycle of the input channel */
ds4004_pwm2d(DS4004_1_BASE,       /* base address of the DS4004 */
        1,                        /* select port 1 */
        1,                        /* select channel 1 */
        &period,                  /* address for the period */
        &duty);                   /* address for the duty cycle */
```

Description

Before measuring PWM signals, you have to initialize the channels as PWM inputs and set their threshold levels.

To minimize the quantization effects on the frequency resolution and duty cycle, you should select the smallest possible period range, refer to Basics on PWM Signal Measurement (PWM2D) on page 30.

For initialization functions, you can use the predefined symbols `DS4004_MASK_CH1` ... `DS4004_MASK_CH32` to select channels. (This does not work for timing I/O run-time functions.) Use a logical OR operation for the predefined symbols to select more than one channel. The result of the operation is a bit mask for the selected channels.

By using the `ds4004_pwm2d_init` function several times, you can initialize different channels individually.

The `ds4004_pwm2d` function measures the period and duty cycle. The results are available by means of the `period` and the `duty` variables.

Related topics

Basics

Basics on PWM Signal Measurement (PWM2D)..... 30

Examples

Example of PWM Signal Measurement (PWM2D) (DS4004 RTLib Reference 🕮)

References

ds4004_init (DS4004 RTLib Reference 🕮)
ds4004_pwm2d (DS4004 RTLib Reference 🕮)
ds4004_pwm2d_init (DS4004 RTLib Reference 🕮)

Basics on PWM Signal Generation (D2PWM)

Introduction

You can specify each I/O channel individually to generate PWM signals.

Basics

PWM signals are square-wave signals with run-time adjustable period and duty cycle.

Asymmetric PWM pulses The pulses of the PWM signals generated by the DS4004 start at the beginning of the PWM period (asymmetric PWM pulses).

With the DS4004, you cannot generate PWM pulses that are centered around the middle of the PWM period (symmetric PWM pulses).

PWM output signals not synchronized If you use several output channels of the DS4004 for PWM signal generation, the generated PWM signals are not synchronized.

Note

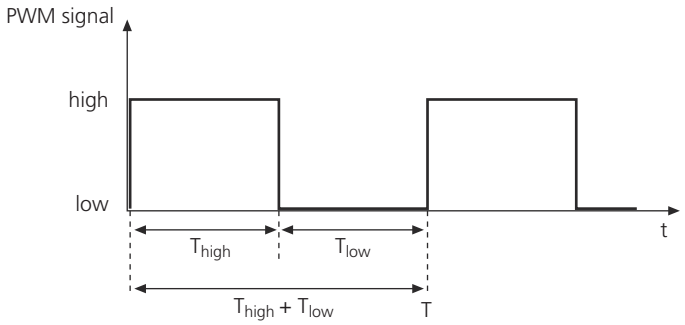
The resolution of the period to be generated depends on the selected period range.
Due to quantization effects, you might encounter considerable deviations between the desired PWM period and the generated PWM period, especially for higher PWM frequencies. To avoid poor frequency resolution, you should therefore select the period range with the best possible resolution (resolution values as small as possible). For details, refer to Quantization Effects on page 73.

PWM characteristics

At 16-bit resolution, you can set

- The period in the range 6.7 µs ... 107.3 s
- The duty cycle in the range 0 … 100 % (scaled to 0 ... 1)

The duty cycle values 0 and 1 yield a constant low and constant high output signal respectively.



The illustration above shows the definitions

- Period = $T_{low} + T_{high}$
- Duty cycle = $T_{high} / (T_{low} + T_{high})$

Limits and resolution

Depending on the selected range, the PWM signals can be generated within the following limits and resolutions:

Range	Minimum Period		Maximum Period	Resolution
	Theoretical	Practical		
1	100 ns	6.7 µs	3.27 ms	50 ns
2	200 ns	6.7 µs	6.55 ms	100 ns

Range	Minimum Period		Maximum Period	Resolution
	Theoretical	Practical		
3	400 ns	6.7 µs	13.1 ms	200 ns
4	800 ns	6.7 µs	26.2 ms	400 ns
5	1.6 µs	6.7 µs	52.4 ms	800 ns
6	3.2 µs	6.7 µs	104 ms	1.6 µs
7	6.4 µs	6.7 µs	209 ms	3.2 µs
8	12.8 µs	12.8 µs	419 ms	6.4 µs
9	25.6 µs	25.6 µs	838 ms	12.8 µs
10	51.2 µs	51.2 µs	1.67 s	25.6 µs
11	103 µs	103 µs	3.35 s	51.2 µs
12	205 µs	205 µs	6.71 s	103 µs
13	410 µs	410 µs	13.4 s	205 µs
14	820 µs	820 µs	26.8 s	410 µs
15	1.64 ms	1.64 ms	53.6 s	820 µs
16	3.28 ms	3.28 ms	107.3 s	1.64 ms

If the period ranges are exceeded, PWM signal generation will be faulty. For values outside these ranges, the period saturates to the minimum period (practical) or the maximum period.

Output characteristics

Output voltage The output channels of the DS4004 have push-pull drivers running from external power supply sources (VBAT). Each port (P1 ... P3) provides two power supply rails (Px_VBAT1 and Px_VBAT2). These power supply rails are independent of one another. Thus, you can use up to six independent supply rails with the DS4004.

The maximum output current per channel is ±50 mA. The functional voltage range is +5 V … +60 V. Self-resetting multifuses protect the outputs against short circuits to GND or to any voltage up to +60 V. Protection against a short circuit to a negative voltage is also provided, if the voltage difference of the output pin to VBAT is below 60 V.

Note

The multifuses are designed for occasional faults only. They are not suitable for failure simulation.

After board initialization, all outputs remain disabled in high impedance (high-Z) state automatically.

Note

Before operating the output channels, you must connect an external power supply (VBAT) to at least one of the two supply rails (VBAT1 or VBAT2) of the port.

Low-side and high-side switches The output state of each I/O channel depends on its individual settings for the low-side switch and the high-side switches. You can use the supply rails L (GND), H1 (VBAT1), and H2 (VBAT2).

- If you set the low-side switch L (GND), the digital output channel is set to low-side switch mode.
- If you set the high-side switch H1 (VBAT1) *or* H2 (VBAT2), the digital output channel is actively driven in high-side switch mode.
- If you set the high-side switches H1 (VBAT1) *and* H2 (VBAT2), the digital output channel is also actively driven in high-side switch mode, but the output voltage is driven to the highest supply voltage (VBAT1 or VBAT2).
- If you set low-side switch L (GND) and the high-side switches H1 (VBAT1) *and/or* H2 (VBAT2), the digital output channel is actively driven in push-pull mode. Push-pull driver mode means that the output source is actively driven to both high and low level.

Switch Settings [1]			**Input of the Output Circuit** [2]	**Output Px_IO1 ... Px_IO32** [2), 3)]	**Description** [3]
L (GND)	**H1 (VBAT1)**	**H2 (VBAT2)**			
0	0	0	0 or 1	High-Z	Individual output disabled.[4]
1	0	0	0	GND	Low-side switch
1	0	0	1	High-Z	Low-side switch
0	1	0	0	High-Z	High-side switch set to Px_VBAT1
0	1	0	1	Px_VBAT1	High-side switch set to Px_VBAT1
0	0	1	0	High-Z	High-side switch set to Px_VBAT2
0	0	1	1	Px_VBAT2	High-side switch set to Px_VBAT2
0	1	1	0	High-Z	High-side switch set to maximum value (Px_VBAT1 or Px_VBAT2)
0	1	1	1	max. value (Px_VBAT1 or Px_VBAT2)	High-side switch set to maximum value (Px_VBAT1 or Px_VBAT2)
1	1	0	0	GND	Push-pull output set to Px_VBAT1
1	1	0	1	Px_VBAT1	Push-pull output set to Px_VBAT1
1	0	1	0	GND	Push-pull output set to Px_VBAT2
1	0	1	1	Px_VBAT2	Push-pull output set to Px_VBAT2
1	1	1	0	GND	Push-pull output set to maximum value (Px_VBAT1 or Px_VBAT2)

Switch Settings [1]			Input of the Output Circuit [2]	Output Px_IO1 ... Px_IO32 [2), 3)]	Description [3)]
L (GND)	H1 (VBAT1)	H2 (VBAT2)			
1	1	1	1	max. value (Px_VBAT1 or Px_VBAT2)	Push-pull output set to maximum value (Px_VBAT1 or Px_VBAT2)

1) 0 = switch disabled, 1 = switch enabled
2) Refer to Digital Outputs (PHS Bus System Hardware Reference 🕮)
3) x is a placeholder for port/connector number 1 ... 3
4) With RTLib functions, the channel can be used as digital input or PWM input.

For more details, refer to Digital Outputs (PHS Bus System Hardware Reference 🕮).

Update mode

The DS4004 supports two update modes that describe the time interval when the new values for duty cycle is set.

Asynchronous mode New values for T_{high} and T_{low} are updated immediately. An update can happen anywhere during the PWM period.

Note

For PWM signal generation with *asynchronous* update, it is possible that a high or low pulse is cut off. This occurs when the new T_{high} or T_{low} value is shorter than the current one and exceeds the time which has elapsed in the current T_{high} or T_{low} period, respectively. The result is a non-constant PWM period during update (i.e. actual $T_{high} + T_{low}$). If this is not desirable, use the synchronous mode instead.

Synchronous mode New values for T_{high} and T_{low} are updated at the next rising edge of the PWM output signal. The output period is constant if $T = T_{high} + T_{low}$ is constant.

RTI/RTLib support

You can generate PWM signals via an RTI block or RTLib functions.

- *RTI users* refer to How to Generate a PWM Signal via RTI on page 43.
- *RTLib users* refer to Generating a PWM Signal via RTLib Functions on page 47.

Execution times

The execution times required by the RTLib functions have been measured. For details on the results and the measurement setup, refer to Function Execution Times (DS4004 RTLib Reference 🕮).

I/O mapping

Note

The channels on the board can basically be used as follows:

- Each channel can be used for digital I/O.
- Each channel can be used for timing I/O.
- Channels 1 and 2 (signals Px_IO1 and Px_IO2) of each port can also be used for interrupt control.

However, there are limitations in how you can combine these uses, refer to Conflicting I/O Features on page 74.

The following table shows the mapping of the I/O signals to the corresponding ports and channels, as used in RTI and RTLib.

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO1	Px	1	1	Digital I/O shared with timing I/O and interrupt control
Px_IO2	Px	34	2	Digital I/O shared with timing I/O and interrupt control
Px_IO3	Px	18	3	Digital I/O shared with timing I/O
Px_IO4	Px	2	4	Digital I/O shared with timing I/O
Px_IO5	Px	35	5	Digital I/O shared with timing I/O
Px_IO6	Px	19	6	Digital I/O shared with timing I/O
Px_IO7	Px	3	7	Digital I/O shared with timing I/O
Px_IO8	Px	36	8	Digital I/O shared with timing I/O
Px_IO9	Px	20	9	Digital I/O shared with timing I/O
Px_IO10	Px	4	10	Digital I/O shared with timing I/O
Px_IO11	Px	37	11	Digital I/O shared with timing I/O
Px_IO12	Px	21	12	Digital I/O shared with timing I/O
Px_IO13	Px	5	13	Digital I/O shared with timing I/O
Px_IO14	Px	38	14	Digital I/O shared with timing I/O
Px_IO15	Px	22	15	Digital I/O shared with timing I/O
Px_IO16	Px	6	16	Digital I/O shared with timing I/O
Px_IO17	Px	39	17	Digital I/O shared with timing I/O
Px_IO18	Px	23	18	Digital I/O shared with timing I/O
Px_IO19	Px	7	19	Digital I/O shared with timing I/O
Px_IO20	Px	40	20	Digital I/O shared with timing I/O
Px_IO21	Px	24	21	Digital I/O shared with timing I/O
Px_IO22	Px	8	22	Digital I/O shared with timing I/O
Px_IO23	Px	41	23	Digital I/O shared with timing I/O
Px_IO24	Px	25	24	Digital I/O shared with timing I/O
Px_IO25	Px	9	25	Digital I/O shared with timing I/O
Px_IO26	Px	42	26	Digital I/O shared with timing I/O
Px_IO27	Px	26	27	Digital I/O shared with timing I/O
Px_IO28	Px	10	28	Digital I/O shared with timing I/O
Px_IO29	Px	43	29	Digital I/O shared with timing I/O

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO30	Px	27	30	Digital I/O shared with timing I/O
Px_IO31	Px	11	31	Digital I/O shared with timing I/O
Px_IO32	Px	44	32	Digital I/O shared with timing I/O

[1] x is a placeholder for port/connector number 1 ... 3.

Related topics

Basics

Basics on PWM Signal Measurement (PWM2D)..... 30

References

Digital Outputs (PHS Bus System Hardware Reference 🕮)
Timing I/O Functions (DS4004 RTLib Reference 🕮)

How to Generate a PWM Signal via RTI

Objective

You can use the DS4004D2PWM_BLx RTI block to specify an I/O channel for PWM signal generation.

Basics

Block description The block's Simulink inputs – the period and the duty cycle – can be changed during run time. To avoid saturation effects when generating the PWM signal, the input value for the period must remain in the selected period range.

Initialization During the model initialization phase, the PWM output signal is either generated with an initial period or set to constant low or high potential. This is especially useful if a channel is used in a triggered or enabled subsystem that is not executed at the start of the simulation. With **Initial period** and **Initial duty cycle**, the channel has a defined output during this simulation phase.

Termination With the block's **Termination** settings, you can specify an output behavior of the channel on termination to drive your external hardware into a safe final condition.

The possible output behaviors at the end of the simulation are:

- The output is set to high impedance (high-Z) state.
- The output holds the last duty cycle and period.
- The output is set to a definite duty cycle and period.

The specified termination values of I/O channels are set when the simulation executes its termination function by setting the `simState` variable to STOP. If you stop the real-time application by using ControlDesk's **Stop RTP** command,

the processor resets immediately without executing termination functions. The current values of the I/O channels are kept and the specified termination values are not set.

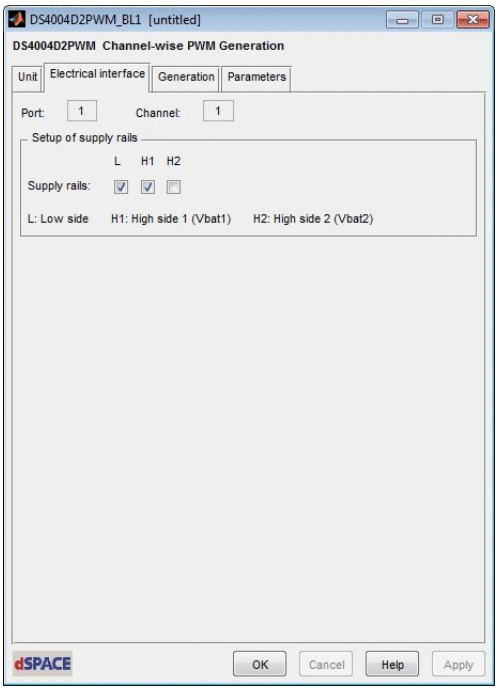
For further information, refer to Basics on PWM Signal Generation (D2PWM) on page 37.

Method

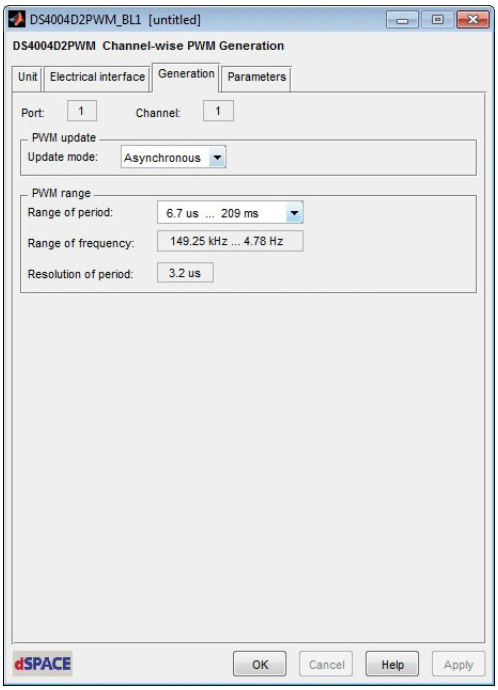
To generate a PWM signal via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004D2PWM_BLx block from the library to your Simulink model. Double-click the block to open it.
3. Select the block's Unit page to specify the board number, the port number, and the channel number.
4. Select the block's Electrical interface page to specify the low-side and high-side switches.

   To set a switch, enable the corresponding checkbox.



5 Select the block's **Generation** page.

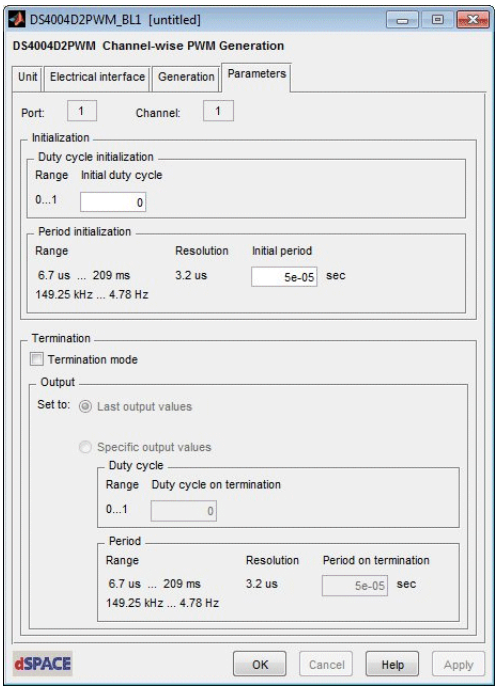


6 Specify the **Update mode** for the PWM signal to be generated as *asynchronous* or *synchronous*.

7 Specify the **Range of period** for the PWM signal to be generated.

The corresponding frequency range and the resolution for the selected period range are displayed (grayed out).

8 Select the block's **Parameters** page, to specify the block's initial output behavior and its termination output behavior.



9 Specify the **Initial duty cycle** in the range 0 ... 1 and the **Initial period** in the stated range. The duty cycle values 0 and 1 yield a constant low and constant high output signal respectively.

10 Specify the **Termination** at the end of the simulation:

- To set the output channel to high impedance state, clear the **Termination mode** checkbox.
- To hold the last duty cycle and period when the simulation terminates, select the **Termination mode** checkbox and the **Last output values** option button.
- To set a definite duty cycle and period at the end of the simulation, select the **Termination mode** checkbox and the **Specific output values** option button, and specify the values.

  The duty cycle values 0 and 1 yield a constant low and constant high output signal respectively. The **Period on termination** should remain in the stated period range.

Result

You added the **DS4004D2PWM_BLx** block to your Simulink model and configured with it an output channel for PWM signal generation.

Related topics

Basics

Basics on PWM Signal Generation (D2PWM)..... 37

References

DS4004D2PWM_BLx (DS4004 RTI Reference 🕮)
simState (RTI and RTI-MP Implementation Reference 🕮)
Stop RTP (ControlDesk Platform Management 🕮)

Generating a PWM Signal via RTLib Functions

Introduction

You can use RTLib functions to generate PWM signals.

Related RTLib functions

- `ds4004_init`
- `ds4004_d2pwm_init`
- `ds4004_digout_mode_set`
- `ds4004_d2pwm`

Example

The following code lines show the usage of the relevant RTLib functions:

```
ds4004_init(DS4004_1_BASE);       /* initialize the DS4004 */

/* initialize channel 1 of port 1 as PWM output */
ds4004_d2pwm_init(DS4004_1_BASE,  /* base address of the DS4004 */
        1,                        /* select port 1 */
        DS4004_MASK_CH1,          /* select channel 1 */
        DS4004_LS_ENABLE,         /* enable low-side switch */
        DS4004_HS_VBAT1_ENABLE,   /* enable high-side switch to VBAT1 */
        DS4004_TIMING_RANGE1,     /* use timing range 1 (6.7µs ... 3.27ms) */
        DS4004_D2PWM);            /* use normal PWM generation (asynchronous) */

/* enable (global) digital output mode for port 1 */
ds4004_digout_mode_set(DS4004_1_BASE, /* base address of the DS4004 */
        DS4004_MASK_PORT1,        /* select port 1 */
        DS4004_DIGOUT_ENABLE);    /* enable digital outputs for the port */

/* set period and duty cycle for the PWM output */
ds4004_d2pwm(DS4004_1_BASE,       /* base address of the DS4004 */
        1,                        /* select port 1 */
        1,                        /* select channel 1 */
        1e-3,                     /* set period = 1ms */
        0.2);                     /* set duty cycle = 20% */
```

Description

Before generating PWM signals, you have to initialize the desired channels as PWM outputs and set their low-side and high-side switches.

Due to quantization effects, you might encounter considerable deviations between the desired PWM period TP and the generated PWM period, especially for higher PWM frequencies. To avoid poor frequency resolution, you should therefore select the frequency range with the best possible resolution (resolution values as small as possible), refer to Basics on PWM Signal Generation (D2PWM) on page 37.

For initialization functions, you can use the predefined symbols `DS4004_MASK_CH1` ... `DS4004_MASK_CH32` to select channels. (This does not work for timing I/O run-time functions.) Use a logical OR operation for the predefined symbols to select more than one channel. The result of the operation is a bit mask for the selected channels.

By using the `ds4004_pwm2d_init` function several times, you can initialize different channels individually.

When you have made all your settings, enable the output mode for the port by calling the `ds4004_digout_mode_set` function.

Calling the `ds4004_d2pwm` function sets the period and duty cycle for the selected channel.

Related topics

Basics

Basics on PWM Signal Generation (D2PWM)..... 37

Examples

Example of PWM Signal Generation (D2PWM) (DS4004 RTLib Reference 🕮)

References

ds4004_d2pwm (DS4004 RTLib Reference 🕮)
ds4004_d2pwm_init (DS4004 RTLib Reference 🕮)
ds4004_digout_mode_set (DS4004 RTLib Reference 🕮)
ds4004_init (DS4004 RTLib Reference 🕮)

Basics on Frequency Measurement (F2D)

Introduction

You can specify each I/O channel individually to measure the frequency of square-wave signals.

Frequency ranges

At 21-bit resolution, you can measure frequencies in the range 0.3 mHz … 150 kHz.

Note

The measurement resolution depends on the selected frequency range.
To get the best possible resolution of the measured square-wave signal, you should select the frequency range with the best possible resolution (the frequency range with the lowest possible range number).

You can measure frequencies in the following ranges:

Range	Minimum Frequency	Maximum Frequency	Resolution
1	9.54 Hz	150 kHz	50 ns
2	4.77 Hz	150 kHz	100 ns
3	2.39 Hz	150 kHz	200 ns
4	1.20 Hz	150 kHz	400 ns
5	0.60 Hz	150 kHz	800 ns
6	0.30 Hz	150 kHz	1.6 µs
7	0.15 Hz	150 kHz	3.2 µs
8	75 mHz	78.12 kHz	6.4 µs
9	38 mHz	39.06 kHz	12.8 µs
10	19 mHz	19.53 kHz	25.6 µs
11	10 mHz	9.76 kHz	51.2 µs
12	5.0 mHz	4.88 kHz	103 µs
13	2.5 mHz	2.44 kHz	205 µs
14	1.2 mHz	1.22 kHz	410 µs
15	0.6 mHz	610.35 Hz	820 µs
16	0.3 mHz	305.17 Hz	1.64 ms

If these ranges are exceeded, the measurement will be faulty. Note the following restrictions with values outside the frequency ranges:

Range	Restriction
Frequency < f_{min}	The measurement returns a frequency of 0 Hz.
Frequency > f_{max}	Faulty measurement because of undersampling effects.

Note

Signal periods and resolution

Each high period and each low period of the measured signal must be longer (not equal) than the resolution to avoid missing pulses.

Input voltage

The input channels are fully operational up to 60 V. After software initialization, the input threshold is set to 2.5 V. You can set the input threshold in the range 1.0 … 23.8 V. The input channels have a hysteresis voltage of approximately 0.2 V. This value is fixed and cannot be changed. For example, if the input threshold is 2.5 V, the input signal must exceed 2.6 V to be detected as high, and must fall below 2.4 V to be detected as low.

RTI/RTLib support

You can measure square-wave signal frequencies via an RTI block or RTLib functions.

- *RTI users* refer to How to Measure a Frequency via RTI on page 51.
- *RTLib users* refer to Measuring the Signal Frequency via RTLib Functions on page 53.

Execution times

The execution times required by the RTLib functions have been measured. For details on the results and the measurement setup, refer to Function Execution Times (DS4004 RTLib Reference 🕮).

I/O mapping

Note

The channels on the board can basically be used as follows:

- Each channel can be used for digital I/O.
- Each channel can be used for timing I/O.
- Channels 1 and 2 (signals Px_IO1 and Px_IO2) of each port can also be used for interrupt control.

However, there are limitations in how you can combine these uses, refer to Conflicting I/O Features on page 74.

The following table shows the mapping of the I/O signals to the corresponding ports and channels, as used in RTI and RTLib.

I/O Signal [1)]	Port/ Connector [1)]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO1	Px	1	1	Digital I/O shared with timing I/O and interrupt control
Px_IO2	Px	34	2	Digital I/O shared with timing I/O and interrupt control
Px_IO3	Px	18	3	Digital I/O shared with timing I/O
Px_IO4	Px	2	4	Digital I/O shared with timing I/O
Px_IO5	Px	35	5	Digital I/O shared with timing I/O
Px_IO6	Px	19	6	Digital I/O shared with timing I/O
Px_IO7	Px	3	7	Digital I/O shared with timing I/O
Px_IO8	Px	36	8	Digital I/O shared with timing I/O
Px_IO9	Px	20	9	Digital I/O shared with timing I/O
Px_IO10	Px	4	10	Digital I/O shared with timing I/O

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO11	Px	37	11	Digital I/O shared with timing I/O
Px_IO12	Px	21	12	Digital I/O shared with timing I/O
Px_IO13	Px	5	13	Digital I/O shared with timing I/O
Px_IO14	Px	38	14	Digital I/O shared with timing I/O
Px_IO15	Px	22	15	Digital I/O shared with timing I/O
Px_IO16	Px	6	16	Digital I/O shared with timing I/O
Px_IO17	Px	39	17	Digital I/O shared with timing I/O
Px_IO18	Px	23	18	Digital I/O shared with timing I/O
Px_IO19	Px	7	19	Digital I/O shared with timing I/O
Px_IO20	Px	40	20	Digital I/O shared with timing I/O
Px_IO21	Px	24	21	Digital I/O shared with timing I/O
Px_IO22	Px	8	22	Digital I/O shared with timing I/O
Px_IO23	Px	41	23	Digital I/O shared with timing I/O
Px_IO24	Px	25	24	Digital I/O shared with timing I/O
Px_IO25	Px	9	25	Digital I/O shared with timing I/O
Px_IO26	Px	42	26	Digital I/O shared with timing I/O
Px_IO27	Px	26	27	Digital I/O shared with timing I/O
Px_IO28	Px	10	28	Digital I/O shared with timing I/O
Px_IO29	Px	43	29	Digital I/O shared with timing I/O
Px_IO30	Px	27	30	Digital I/O shared with timing I/O
Px_IO31	Px	11	31	Digital I/O shared with timing I/O
Px_IO32	Px	44	32	Digital I/O shared with timing I/O

[1] x is a placeholder for port/connector number 1 ... 3.

Related topics

Basics

Basics on Square-Wave Signal Generation (D2F) 54

References

Digital Inputs (PHS Bus System Hardware Reference 🕮)
Timing I/O Functions (DS4004 RTLib Reference 🕮)

How to Measure a Frequency via RTI

Objective

You can use the DS4004F2D_BLx RTI block to measure the frequency of a square-wave signal.

Basics

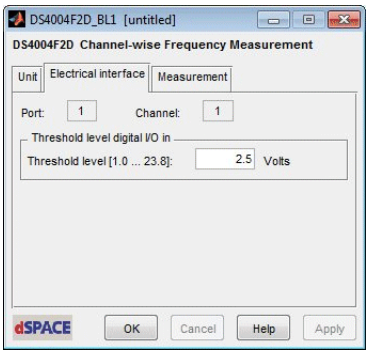
In a Simulink model, the block provides channel-wise read access to the frequency of a square-wave signal. You can specify the threshold level on which the input signal is interpreted as logical 0 or 1.

For further information, refer to Basics on Frequency Measurement (F2D) on page 48.

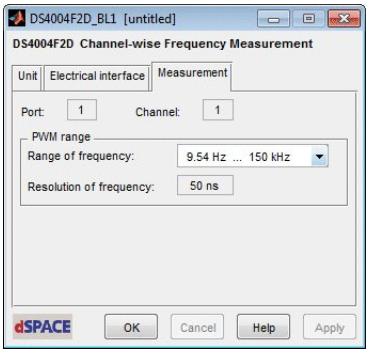
Method

How to measure a frequency via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004F2D_BLx block from the library to your Simulink model. Double-click the block to open it.
3. Select the block's Unit page to specify the board number, the port number, and the channel number.
4. Select the block's Electrical interface page to specify the threshold level for the selected input channel in the range 1.0 … 23.8 V.



5. Select the block's Measurement page to specify the expected frequency range for the square-wave signal to be measured.



The measurement resolution for the selected frequency range is displayed (grayed out).

Result

You added the DS4004F2D_BLx block to your Simulink model and configured it for frequency measurement of a single input channel.

The input signal connected to the corresponding pin of the DS4004 can now be examined by the model. The block's outport delivers a value (of double data type) to the model, representing the measured frequency.

Related topics

Basics

Basics on Frequency Measurement (F2D)..... 48

References

DS4004F2D_BLx (DS4004 RTI Reference 🕮)

Measuring the Signal Frequency via RTLib Functions

Introduction

You can use RTLib functions to measure the frequency of square-wave signals.

Related RTLib functions

- `ds4004_init`
- `ds4004_f2d_init`
- `ds4004_f2d`

Example

The following code lines show the usage of the relevant RTLib functions:

```
dsfloat frequency;

ds4004_init(DS4004_1_BASE);      /* initialize the DS4004 */

/* initialize channel 1 of port 1 for frequency measurement */
ds4004_f2d_init(DS4004_1_BASE,   /* base address of the DS4004 */
        1,                       /* select port 1 */
        DS4004_MASK_CH1,         /* select channel 1 */
        2.5,                     /* set input threshold level to 2.5V */
        DS4004_TIMING_RANGE1);   /* use timing range 1 (9.54Hz ... 150kHz) */

/* get the signal frequency of the input channel */
frequency = ds4004_f2d(DS4004_1_BASE, /* base address of the DS4004 */
        1,                       /* select port 1 */
        1);                      /* select channel 1 */
```

Description

Before measuring frequencies, you have to initialize the desired channels for frequency measurement and set their threshold levels.

To get the best resolution of the measured square-wave signal, you should always use the timing range (frequency range) with the lowest possible range number, refer to Basics on Frequency Measurement (F2D) on page 48.

For initialization functions, you can use the predefined symbols `DS4004_MASK_CH1` ... `DS4004_MASK_CH32` to select channels. (This does not work for timing I/O run-time functions.) Use a logical OR operation for the predefined symbols to select more than one channel. The result of the operation is a bit mask for the selected channels.

By using the `ds4004_f2d_init` function several times, you can initialize different channels individually.

Calling the `ds4004_f2d` function returns the measured frequency (in Hz) for the selected channel. To measure the frequency of several channels, call the `ds4004_f2d` function more often.

Related topics

Basics

Basics on Frequency Measurement (F2D) 48

Examples

Example of Frequency Measurement (F2D) (DS4004 RTLib Reference 🕮)

References

ds4004_f2d (DS4004 RTLib Reference 🕮)
ds4004_f2d_init (DS4004 RTLib Reference 🕮)
ds4004_init (DS4004 RTLib Reference 🕮)

Basics on Square-Wave Signal Generation (D2F)

Introduction

You can specify each I/O channel individually to generate square-wave signals with variable frequencies.

Frequency ranges and resolution

At 20-bit resolution, you can generate frequencies in the range 0.3 mHz … 150 kHz.

Note

The resolution of the square-wave signal to be generated depends on the selected frequency range.
To get the best signal resolution, you should select the frequency range with the best possible resolution (the frequency range with the lowest possible range number). For example, if your desired frequency is 100 Hz, you should use frequency range 1 (9.54 Hz … 150 kHz) rather than frequency range 2 (4.77 Hz … 150 kHz).

You can generate frequencies in the following ranges:

Range	Minimum Frequency	Maximum Frequency	Resolution
1	9.54 Hz	150 kHz	100 ns
2	4.77 Hz	150 kHz	200 ns
3	2.39 Hz	150 kHz	400 ns
4	1.20 Hz	150 kHz	800 ns
5	0.60 Hz	150 kHz	1.6 µs
6	0.30 Hz	150 kHz	3.2 µs
7	0.15 Hz	150 kHz	6.4 µs
8	75 mHz	78.12 kHz	12.8 µs
9	38 mHz	39.06 kHz	25.6 µs
10	19 mHz	19.53 kHz	51.2 µs
11	10 mHz	9.76 kHz	103 µs
12	5.0 mHz	4.88 kHz	205 µs
13	2.5 mHz	2.44 kHz	410 µs
14	1.2 mHz	1.22 kHz	820 µs
15	0.6 mHz	610.35 Hz	1.64 ms
16	0.3 mHz	305.17 Hz	3.28 ms

If these ranges are exceeded, square-wave signal generation will be faulty. For values outside these ranges, note the following restrictions:

Range	Restriction
Frequency < f_{min}	The frequency is set to 0 Hz. Depending on the setting for the *zero frequency mode*[1], the digital output ▪ Is set to low level ▪ Is set to high level ▪ Or holds its last (high or low) level.
Frequency > f_{max}	The frequency saturates to f_{max}.

[1] Refer to **Generation Page (DS4004D2F_BLx)** (DS4004 RTI Reference 🕮) or **`ds4004_d2f_init`** (DS4004 RTLib Reference 🕮)

Output characteristics

Output voltage The output channels of the DS4004 have push-pull drivers running from external power supply sources (VBAT). Each port (P1 … P3) provides two power supply rails (Px_VBAT1 and Px_VBAT2). These power supply rails are independent of one another. Thus, you can use up to six independent supply rails with the DS4004.

The maximum output current per channel is ±50 mA. The functional voltage range is +5 V … +60 V. Self-resetting multifuses protect the outputs against short circuits to GND or to any voltage up to +60 V. Protection against a short

circuit to a negative voltage is also provided, if the voltage difference of the output pin to VBAT is below 60 V.

Note

The multifuses are designed for occasional faults only. They are not suitable for failure simulation.

After board initialization, all outputs remain disabled in high impedance (high-Z) state automatically.

Note

Before operating the output channels, you must connect an external power supply (VBAT) to at least one of the two supply rails (VBAT1 or VBAT2) of the port.

Low-side and high-side switches The output state of each I/O channel depends on its individual settings for the low-side switch and the high-side switches. You can use the supply rails L (GND), H1 (VBAT1), and H2 (VBAT2).

- If you set the low-side switch L (GND), the digital output channel is set to low-side switch mode.
- If you set the high-side switch H1 (VBAT1) *or* H2 (VBAT2), the digital output channel is actively driven in high-side switch mode.
- If you set the high-side switches H1 (VBAT1) *and* H2 (VBAT2), the digital output channel is also actively driven in high-side switch mode, but the output voltage is driven to the highest supply voltage (VBAT1 or VBAT2).
- If you set low-side switch L (GND) and the high-side switches H1 (VBAT1) *and/or* H2 (VBAT2), the digital output channel is actively driven in push-pull mode. Push-pull driver mode means that the output source is actively driven to both high and low level.

Switch Settings [1]			**Input of the Output Circuit** [2]	**Output Px_IO1 ... Px_IO32** [2), 3)]	**Description** [3]
L (GND)	**H1 (VBAT1)**	**H2 (VBAT2)**			
0	0	0	0 or 1	High-Z	Individual output disabled.[4]
1	0	0	0	GND	Low-side switch
1	0	0	1	High-Z	Low-side switch
0	1	0	0	High-Z	High-side switch set to Px_VBAT1
0	1	0	1	Px_VBAT1	High-side switch set to Px_VBAT1
0	0	1	0	High-Z	High-side switch set to Px_VBAT2
0	0	1	1	Px_VBAT2	High-side switch set to Px_VBAT2
0	1	1	0	High-Z	High-side switch set to maximum value (Px_VBAT1 or Px_VBAT2)
0	1	1	1	max. value (Px_VBAT1 or Px_VBAT2)	High-side switch set to maximum value (Px_VBAT1 or Px_VBAT2)
1	1	0	0	GND	Push-pull output set to Px_VBAT1
1	1	0	1	Px_VBAT1	Push-pull output set to Px_VBAT1

Switch Settings [1]			Input of the Output Circuit [2]	Output Px_IO1 ... Px_IO32 [2), 3)]	Description [3)]
L (GND)	H1 (VBAT1)	H2 (VBAT2)			
1	0	1	0	GND	Push-pull output set to Px_VBAT2
1	0	1	1	Px_VBAT2	Push-pull output set to Px_VBAT2
1	1	1	0	GND	Push-pull output set to maximum value (Px_VBAT1 or Px_VBAT2)
1	1	1	1	max. value (Px_VBAT1 or Px_VBAT2)	Push-pull output set to maximum value (Px_VBAT1 or Px_VBAT2)

[1] 0 = switch disabled, 1 = switch enabled
[2] Refer to Digital Outputs (PHS Bus System Hardware Reference 🕮)
[3] x is a placeholder for port/connector number 1 ... 3
[4] With RTLib functions, the channel can be used as digital input or PWM input.

For more details, refer to Digital Outputs (PHS Bus System Hardware Reference 🕮).

RTI/RTLib support

You can generate square-wave signals via an RTI block or RTLib functions.

- *RTI users* refer to How to Generate a Square-Wave Signal via RTI on page 59.
- *RTLib users* refer to Generating a Square-Wave Signal via RTLib Functions on page 62.

Execution times

The execution times required by the RTLib functions have been measured. For details on the results and the measurement setup, refer to Function Execution Times (DS4004 RTLib Reference 🕮).

I/O mapping

Note

The channels on the board can basically be used as follows:

- Each channel can be used for digital I/O.
- Each channel can be used for timing I/O.
- Channels 1 and 2 (signals Px_IO1 and Px_IO2) of each port can also be used for interrupt control.

However, there are limitations in how you can combine these uses, refer to Conflicting I/O Features on page 74.

The following table shows the mapping of the I/O signals to the corresponding ports and channels, as used in RTI and RTLib.

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO1	Px	1	1	Digital I/O shared with timing I/O and interrupt control
Px_IO2	Px	34	2	Digital I/O shared with timing I/O and interrupt control
Px_IO3	Px	18	3	Digital I/O shared with timing I/O
Px_IO4	Px	2	4	Digital I/O shared with timing I/O

I/O Signal [1]	Port/ Connector [1]	Pin	Channel/Bit Number	Description/Conflicts
Px_IO5	Px	35	5	Digital I/O shared with timing I/O
Px_IO6	Px	19	6	Digital I/O shared with timing I/O
Px_IO7	Px	3	7	Digital I/O shared with timing I/O
Px_IO8	Px	36	8	Digital I/O shared with timing I/O
Px_IO9	Px	20	9	Digital I/O shared with timing I/O
Px_IO10	Px	4	10	Digital I/O shared with timing I/O
Px_IO11	Px	37	11	Digital I/O shared with timing I/O
Px_IO12	Px	21	12	Digital I/O shared with timing I/O
Px_IO13	Px	5	13	Digital I/O shared with timing I/O
Px_IO14	Px	38	14	Digital I/O shared with timing I/O
Px_IO15	Px	22	15	Digital I/O shared with timing I/O
Px_IO16	Px	6	16	Digital I/O shared with timing I/O
Px_IO17	Px	39	17	Digital I/O shared with timing I/O
Px_IO18	Px	23	18	Digital I/O shared with timing I/O
Px_IO19	Px	7	19	Digital I/O shared with timing I/O
Px_IO20	Px	40	20	Digital I/O shared with timing I/O
Px_IO21	Px	24	21	Digital I/O shared with timing I/O
Px_IO22	Px	8	22	Digital I/O shared with timing I/O
Px_IO23	Px	41	23	Digital I/O shared with timing I/O
Px_IO24	Px	25	24	Digital I/O shared with timing I/O
Px_IO25	Px	9	25	Digital I/O shared with timing I/O
Px_IO26	Px	42	26	Digital I/O shared with timing I/O
Px_IO27	Px	26	27	Digital I/O shared with timing I/O
Px_IO28	Px	10	28	Digital I/O shared with timing I/O
Px_IO29	Px	43	29	Digital I/O shared with timing I/O
Px_IO30	Px	27	30	Digital I/O shared with timing I/O
Px_IO31	Px	11	31	Digital I/O shared with timing I/O
Px_IO32	Px	44	32	Digital I/O shared with timing I/O

[1] x is a placeholder for port/connector number 1 ... 3.

Related topics

Basics

Basics on Frequency Measurement (F2D) 48

References

Digital Outputs (PHS Bus System Hardware Reference 🕮)
Timing I/O Functions (DS4004 RTLib Reference 🕮)

How to Generate a Square-Wave Signal via RTI

Objective

You can use the DS4004D2F_BLx RTI block to generate a square-wave signal for a specified output channel.

Basics

Block description The block's Simulink input – the frequency – can be changed during run time. To avoid saturation effects when generating the square-wave signal, the input value for the frequency must remain in the selected frequency range.

Initialization During the model initialization phase, the output signal is either generated with an initial frequency or set to zero. This is especially useful if a channel is used in a triggered or enabled subsystem that is not executed at the start of the simulation. With **Initial frequency**, the channel has a defined output during this simulation phase.

Termination With the block's **Termination** settings, you can specify an output behavior of the channel on termination to drive your external hardware into a safe final condition.

The possible output behaviors at the end of the simulation are:

- The output is set to high impedance (high-Z) state.
- The output holds the last frequency.
- The output is set to a definite frequency.

The specified termination values of I/O channels are set when the simulation executes its termination function by setting the `simState` variable to STOP. If you stop the real-time application by using ControlDesk's **Stop RTP** command, the processor resets immediately without executing termination functions. The current values of the I/O channels are kept and the specified termination values are not set.

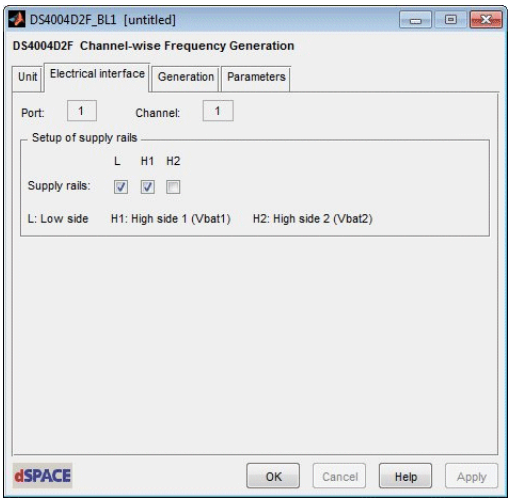
For further information, refer to Basics on Square-Wave Signal Generation (D2F) on page 54.

Method

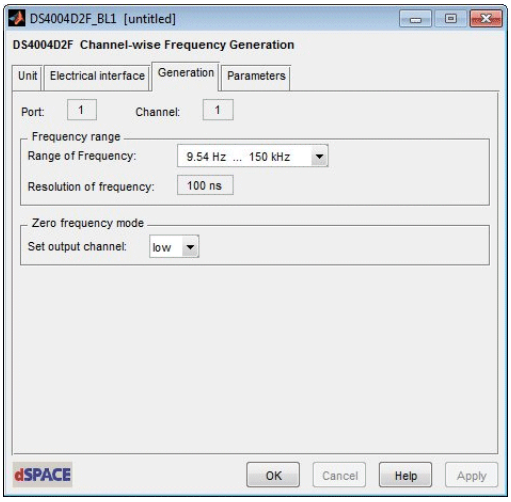
To generate a PWM signal via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004D2F_BLx block from the library to your Simulink model. Double-click the block to open it.
3. Select the block's **Unit** page to specify the board number, the port number, and the channel number.
4. Select the block's **Electrical interface** page to specify the low-side and high-side switches.

To set a switch, enable the corresponding checkbox.



5 Select the block's Generation page.

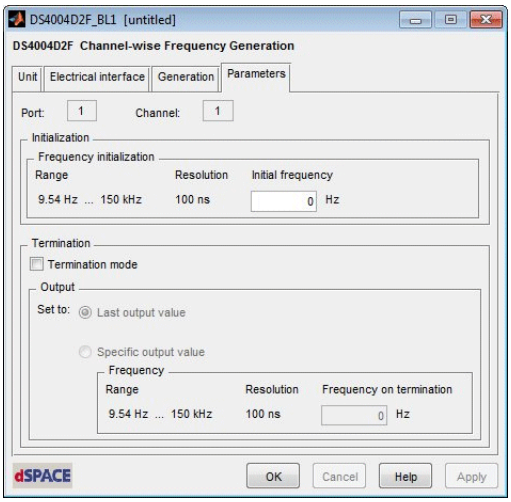


6 Specify the Range of frequency.

The frequency resolution for the selected frequency range is displayed (grayed out).

7 Select Set out channel as *low*, *high*, or *hold*, to specify the Zero frequency mode (the behavior of the output channel if the frequency falls below the lower limit of the frequency range).

8 Select the block's **Parameters** page to specify its initial output behavior and its termination output behavior.



9 Enter the **Initial frequency** at the start of the simulation in Hz.

The frequency should remain in the stated frequency range. Otherwise, it is saturated to f_{max} or set to 0 Hz.

10 Specify the **Termination** at the end of the simulation:

- To set the output channel to high impedance state, clear the **Termination mode** checkbox.
- To hold the last frequency when the simulation terminates, select the **Termination mode** checkbox and the **Last output value** option button.
- To set a definite frequency at the end of the simulation, select the **Termination mode** checkbox and the **Specific output value** option button, and specify the **Frequency on termination** in Hz.

  The frequency should remain in the stated frequency range. Otherwise, it is saturated to f_{max} or set to 0 Hz.

Result

You added the **DS4004D2F_BLx** block to your Simulink model and configured with it an output channel for square-wave signal generation.

Related topics

Basics

Basics on Square-Wave Signal Generation (D2F) 54

References

DS4004D2F_BLx (DS4004 RTI Reference 🕮)
simState (RTI and RTI-MP Implementation Reference 🕮)
Stop RTP (ControlDesk Platform Management 🕮)

Generating a Square-Wave Signal via RTLib Functions

Introduction

You can use RTLib functions to generate square-wave signals.

Related RTLib functions

- `ds4004_init`
- `ds4004_d2f_init`
- `ds4004_digout_mode_set`
- `ds4004_d2f`

Example

The following code lines show the usage of the relevant RTLib functions:

```
ds4004_init(DS4004_1_BASE);       /* initialize the DS4004 */

/* initialize channel 1 of port 1 for frequency generation */
ds4004_d2f_init(DS4004_1_BASE,    /* base address of the DS4004 */
        1,                        /* select port 1 */
        DS4004_MASK_CH1,          /* select channel 1 */
        DS4004_LS_ENABLE,         /* enable low-side switch */
        DS4004_HS_VBAT1_ENABLE,   /* enable high-side switch to VBAT1 */
        DS4004_TIMING_RANGE1,     /* use timing range 1 (9.54Hz ... 150kHz)*/
        DS4004_D2F_LOW);          /* set output low for too small frequencies */

/* enable (global) digital output mode for port 1 */
ds4004_digout_mode_set(DS4004_1_BASE, /* base address of the DS4004 */
        DS4004_MASK_PORT1,        /* select port 1 */
        DS4004_DIGOUT_ENABLE);    /* enable digital outputs for the port */

/* set frequency for the square-wave signal on channel 1 of port 1 */
ds4004_d2f(DS4004_1_BASE,         /* base address of the DS4004 */
        1,                        /* select port 1 */
        1,                        /* select channel 1 */
        1000);                    /* set frequency to 1kHz */
```

Description

Before generating square-wave signals, you have to initialize the desired channels for signal generation and set their low-side and high-side switches.

To get the best resolution for the signal to be generated, always use the timing range (frequency range) with the lowest possible range number, refer to Basics on Square-Wave Signal Generation (D2F) on page 54.

For initialization functions, you can use the predefined symbols `DS4004_MASK_CH1` ... `DS4004_MASK_CH32` to select channels. (This does not work for timing I/O run-time functions.) Use a logical OR operation for the predefined symbols to select more than one channel. The result of the operation is a bit mask for the selected channels.

By using the `ds4004_d2f_init` function several times, you can initialize different channels individually.

When you have made all your settings, enable the output mode for the used port(s) by calling the function `ds4004_digout_mode_set`. You can use the

predefined symbols `DS4004_MASK_PORT1` ... `DS4004_MASK_PORT3` for this function to select ports. Use a logical OR operation for the predefined symbols to select more than one port. The result of the operation is a bit mask for the selected ports.

Calling the `ds4004_d2f` function sets the frequency for the selected channel. To set the frequency of several channels, call the `ds4004_d2f` function several times.

Related topics

Basics

Basics on Square-Wave Signal Generation (D2F)..... 54

Examples

Example of Square-Wave Signal Generation (D2F) (DS4004 RTLib Reference 🕮)

References

ds4004_d2f (DS4004 RTLib Reference 🕮)
ds4004_d2f_init (DS4004 RTLib Reference 🕮)
ds4004_digout_mode_set (DS4004 RTLib Reference 🕮)
ds4004_init (DS4004 RTLib Reference 🕮)

Interrupt Control

Introduction

The DS4004 provides six interrupts, which can be used as trigger sources in Simulink® models and handcoded models.

Where to go from here

Information in this section

Basics on Interrupt Handling

Introduction

Interrupts from the DS4004 are sent to the master interrupt controller of the connected processor board via the interrupt lines of the PHS bus.

Interrupt lines of the PHS bus

Each dSPACE I/O board is connected to a dSPACE processor board via the PHS bus. The PHS bus provides 8 interrupt lines, enabling the I/O boards to generate interrupts on the processor board. Interrupts from the I/O board are sent to the master interrupt controller of the connected processor board via the interrupt lines of the PHS bus.

The 8 interrupt lines are connected to the inputs of the master interrupt controller on the processor board. I/O boards supporting interrupts each have a slave interrupt controller unit (ICU) with 8 interrupt inputs. Assignment of slave interrupt controller outputs to PHS-bus interrupt lines is programmable. Only one I/O board can use the same PHS-bus interrupt line at a time. The configuration consisting of a cascaded master and slave interrupt controllers permits a total of 64 interrupts in a system.

Interrupt handling with RTI/RTLib

Interrupt handling varies depending on whether you use RTI blocks or RTLib functions for your application:

Interrupt-driven subsystems in RTI You can use interrupts or subinterrupts to trigger interrupt-driven subsystems in your Simulink model. When the task in one system has finished, the interrupt handler automatically creates an "End of interrupt" (EOI) message to indicate the state to other units. Refer to Tasks Driven by Interrupt Blocks (RTI and RTI-MP Implementation Guide 🕮).

Handcoded models If you use handcoded models, you have to program the interrupt handling yourself. RTLib provides the interrupt handlers and functions required.

Related topics

Basics

Basics on DS4004 Interrupts.....66
Interrupt Controller (DS1006 Features 🕮)
Interrupt Controller (DS1007 Features 🕮)

References

PHS-Bus Interrupt Handling (DS1006 RTLib Reference 🕮)
PHS-Bus Interrupt Handling (DS1007 RTLib Reference 🕮)

Basics on DS4004 Interrupts

Introduction

You can use channels 1 and 2 of each port for interrupt control.

Interrupts of the DS4004

The slave interrupt controller unit (ICU) of the DS4004 uses 6 of the 8 interrupt lines of the PHS bus.

You can specify the external input signal for interrupt generation via RTI blocks or RTLib functions.

The possible settings are:

- Edge type (falling edge, rising edges, or both edges)
- Threshold level (1 V … 23.8 V)

The minimum pulse length to detect an interrupt is 2 µs, when the threshold level is set to the half signal level.

Input voltage

The input channels are fully operational up to 60 V. After software initialization, the input threshold is set to 2.5 V. You can set the input threshold in the range 1.0 … 23.8 V. The input channels have a hysteresis voltage of approximately 0.2 V. This value is fixed and cannot be changed. For example, if the input threshold is 2.5 V, the input signal must exceed 2.6 V to be detected as high, and must fall below 2.4 V to be detected as low.

RTI and RTLib support

You can specify interrupts via RTI and RTLib.

- *RTI users* refer to How to Specify Interrupt Generation via RTI on page 68.
- *RTLib users* refer to Specifying Interrupt Control via RTLib Functions on page 69.

Execution times

The execution times required by the RTLib functions have been measured. For details on the results and the measurement setup, refer to Function Execution Times (DS4004 RTLib Reference 🕮).

I/O mapping

Note

The channels on the board can basically be used as follows:

- Each channel can be used for digital I/O.
- Each channel can be used for timing I/O.
- Channels 1 and 2 (signals Px_IO1 and Px_IO2) of each port can also be used for interrupt control.

However, there are limitations in how you can combine these uses, refer to Conflicting I/O Features on page 74.

The following table shows the mapping of the interrupt signals to the corresponding ports and channels, as used in RTI and RTLib.

Slave ICU Channel	Interrupt Source I/O Mapping				
	Interrupt Signal	Conflicting I/O Signal	Port	Channel	Pin
0	INT0	P1_IO1	P1	1	1
1	INT1	P1_IO2	P1	2	34
2	INT2	P2_IO1	P2	1	1
3	INT3	P2_IO2	P2	2	34
4	INT4	P3_IO1	P3	1	1

Slave ICU Channel	Interrupt Source I/O Mapping				
	Interrupt Signal	Conflicting I/O Signal	Port	Channel	Pin
5	INT5	P3_IO2	P3	2	34
6	not used				
7	not used				

Related topics

Basics

Basics on Interrupt Handling..... 65

HowTos

How to Specify Interrupt Generation via RTI..... 68

References

DS4004_HWINT_BLx (DS4004 RTI Reference 🕮)
ds4004_int_mode_set (DS4004 RTLib Reference 🕮)

How to Specify Interrupt Generation via RTI

Objective

You can use the DS4004_HWINT_BLx RTI block to specify and perform interrupt generation.

Basics

The block manages the interrupt handling for the DS4004 board. It makes the interrupt of the DS4004 board available as a trigger source in a Simulink® model.

Note

There must not be several interrupt blocks in the same model which are identically configured regarding the board number, the port number, and the channel number.

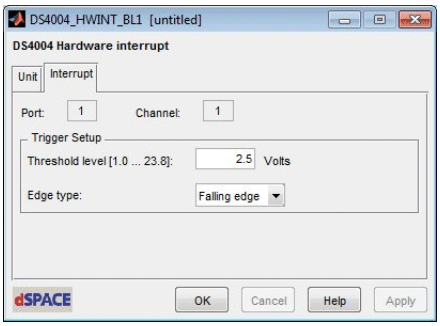
Method

To specify interrupt generation via RTI

1. Open the DS4004 RTI blockset to access the board's Simulink library.
2. Drag the DS4004_HWINT_BLx block from the library to your Simulink model. Double-click the block to open it.

3 Select the block's **Unit** page to specify the board number, the port number, and the channel number.

4 Select the block's **Interrupt** page to specify the threshold level and the edge type for the external input signal for interrupt generation.



Result

You added the **DS4004_HWINT_BLx** block to your Simulink model and configured it for interrupt generation. You can use the block's outport to trigger a subsystem when an interrupt occurs.

Related topics

Basics

Basics on DS4004 Interrupts 66

References

DS4004_HWINT_BLx (DS4004 RTI Reference 🕮)

Specifying Interrupt Control via RTLib Functions

Introduction

You can use RTLib functions to specify and perform interrupt control.

Related RTLib functions and macros

- `ds4004_init`
- `ds4004_digin_init`
- `ds4004_int_mode_set`
- `install_phs_int_vector` (is part of the processor board's RTLib functions)
- `RTLIB_INT_ENABLE` (is part of the processor board's RTLib functions)

Example

The following code lines show the usage of the relevant RTLib functions:

```
void int0()                        /* ISR for interrupt 0 of the DS4004 */
{
msg_info_printf(0, 0, "Interrupt 0");
}

int main()
{

ds4004_init(DS4004_1_BASE);        /* initialize the DS4004 */

/* initialize channel 1 of port 1 as digital input with threshold 2.5V */
ds4004_digin_init(DS4004_1_BASE,   /* base address of the DS4004 */
        1,                         /* select port 1 */
        DS4004_MASK_CH1,           /* select channel 1 */
        2.5);                      /* set input threshold to 2.5V */

/* enable interrupt for channel 1 of port 1 on rising edge */
ds4004_int_mode_set(DS4004_1_BASE, /* base address of the DS4004 */
        1,                         /* select port 1 */
        DS4004_MASK_CH1,           /* select channel 1 */
        DS4004_INT_RISING);        /* select rising edge */

/* install ISR for interrupt 0 */
install_phs_int_vector(DS4004_1_BASE, /* base address of the DS4004 */
        0,                         /* select the slave interrupt number */
        int0);                     /* specify the entry point address of the
                                   interrupt handler */

/* global enable of interrupts */
RTLIB_INT_ENABLE();

} /* main() */
```

Description

Before enabling an interrupt control, you have to initialize the desired channel as digital input by calling the `ds4004_digin_init` function. With this function you can also set the threshold level for interrupt generation in the range 1 V … 23.8 V with a resolution of 0.1 V.

To enable the interrupt control for the initialized input, call the `ds4004_int_mode_set` function and set the triggering edge(s). To select the channel(s), you can use the predefined symbols `DS4004_MASK_CH1` and `DS4004_MASK_CH2`. To select both channel(s), use a logical OR operation for the predefined symbols. The result of the operation is a bit mask for the selected channels.

To use the interrupt, the interrupt service routine (ISR) must be registered by calling the `install_phs_int_vector` function.

To enable interrupts globally, call the `RTLIB_INT_ENABLE` macro.

Related topics

Basics

Basics on DS4004 Interrupts....66
Basics on Interrupt Handling....65

Examples

Example of Interrupt Generation (DS4004 RTLib Reference 🕮)

References

ds4004_digin_init (DS4004 RTLib Reference 🕮)
ds4004_init (DS4004 RTLib Reference 🕮)
ds4004_int_mode_set (DS4004 RTLib Reference 🕮)
install_phs_int_vector (DS1006 RTLib Reference 🕮)
install_phs_int_vector (DS1007 RTLib Reference 🕮)
RTLIB_INT_ENABLE (DS1006 RTLib Reference 🕮)
RTLIB_INT_ENABLE (DS1007 RTLib Reference 🕮)

Limitations

Introduction

There are some limitations you have to take into account when working with the DS4004.

Where to go from here

Information in this section

Information in other sections

Quantization Effects

Introduction

Signal generation and measurement are only feasible within the limits of the resolution of the timing I/O unit. The limited resolution causes quantization errors that increase with increasing frequencies.

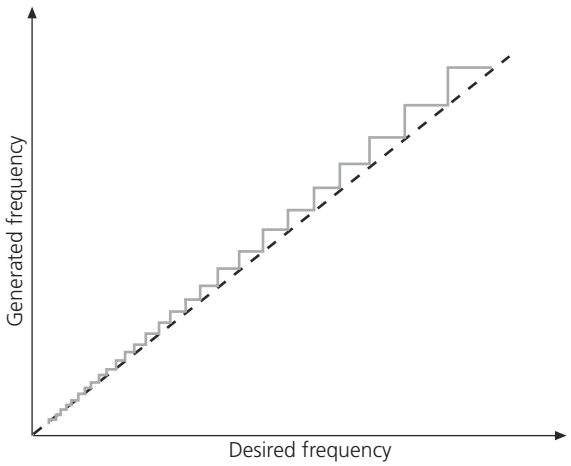
When performing square-wave signal generation (D2F), for example, you will encounter considerable deviations between the desired frequency and the generated frequency, especially for higher frequencies. The (quantized) generated signal frequencies can be calculated according to the following equation:

$$f = \frac{1}{n \cdot R}$$

where R is the resolution (in s), and n is a positive integer.

Example

The following illustration shows the increasing quantization effects for increasing desired frequencies:



You should therefore select the range with the best possible resolution.

Conflicting I/O Features

Conflicts for the DS4004

Note

The channels on the board can basically be used as follows:

- Each channel can be used for digital I/O.
- Each channel can be used for timing I/O.
- Channels 1 and 2 (signals Px_IO1 and Px_IO2) of each port can also be used for interrupt control.

However, there are limitations in how you can combine these uses, see below.

Limitations using RTI blocks

- You cannot use a channel as input (DS4004BIT_IN_BLx, DS4004BIT_IN32_BLx, DS4004PWM2D_BLx, or DS4004F2D_BLx) and as output (DS4004BIT_OUT_BLx, DS4004BIT_OUT32_BLx, DS4004D2PWM_BLx, or DS4004D2F_BLx) at the same time.
- You cannot use an input channel for different input functionalities (DS4004BIT_IN_BLx, DS4004BIT_IN32_BLx, DS4004PWM2D_BLx, or DS4004F2D_BLx) at the same time.

- You cannot use an output channel for different output functionalities (DS4004BIT_OUT_BLx, DS4004BIT_OUT32_BLx, DS4004D2PWM_BLx, or DS4004D2F_BLx) at the same time.
- You cannot use a channel as output (DS4004BIT_OUT_BLx, DS4004BIT_OUT32_BLx, DS4004D2PWM_BLx, or DS4004D2F_BLx) and for interrupt control (DS4004_HWINT_BLx) at the same time.
- You cannot use a channel as digital input (DS4004BIT_IN_BLx, DS4004BIT_IN32_BLx, DS4004PWM2D_BLx, or DS4004F2D_BLx) and for interrupt control (DS4004_HWINT_BLx) at the same time if there are different settings for the channel's threshold level.

Limitations using RTLib functions

- You cannot use a channel as input (`ds4004_bit_in`, `ds4004_bit_in32`, `ds4004_pwm2d`, or `ds4004_f2d`) and as output (`ds4004_bit_out`, `ds4004_bit_out32`, `ds4004_d2pwm`, or `ds4004_d2f`) at the same time.
- You cannot use a channel as digital output (`ds4004_bit_out` or `ds4004_bit_out32`) and as timing I/O output (`ds4004_d2pwm` or `ds4004_d2f`) at the same time.
- You cannot use a channel as digital input (`ds4004_bit_in` or `ds4004_bit_in32`) and as timing I/O input (`ds4004_pwm2d` or `ds4004_f2d`) at the same time if there are different settings for the channel's threshold level.
- You cannot use a channel as input (`ds4004_bit_in`, `ds4004_bit_in32`, `ds4004_pwm2d`, or `ds4004_f2d`) and for interrupt control at the same time if there are different settings for the channel's threshold level.
- You cannot use a channel for PWM signal measurement (`ds4004_pwm2d`) and for frequency measurement (`ds4004_f2d`) at the same time.
- You cannot use a channel for PWM signal generation (`ds4004_d2pwm`) and for square-wave signal generation (`ds4004_d2f`) at the same time.
- You cannot use a channel as digital output (`ds4004_bit_out` or `ds4004_bit_out32`) and for interrupt control at the same time.

Tip

- You can use a channel for channel-wise in functionality (`ds4004_bit_in`) and word-wise in functionality (`ds4004_bit_in32`) at the same time.
- You can use a channel for channel-wise out functionality (`ds4004_bit_out`) and word-wise out functionality (`ds4004_bit_out32`) at the same time.