RTI CAN Blockset

Reference

For RTI CAN Blockset 3.4.12

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

RTI CAN Blocks 55

About This Reference

Content

This reference provides a full description of the RTI CAN Blockset.

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.

Examples:

- Where you find terms such as `rti<XXXX>` replace them by the RTI platform support you are using, for example, `rti1007`.
- Where you find terms such as `<model>` or `<submodel>` in this document, replace them by the actual name of your model or submodel. For example, if the name of your Simulink model is `smd_1007_sl.slx` and you are asked to edit the `<model>_usr.c` file, you actually have to edit the `smd_1007_sl_usr.c` file.

RTI block name conventions All I/O blocks have default names based on dSPACE's board naming conventions:

- Most RTI block names start with the board name.
- A short description of functionality is added.
- Most RTI block names also have a suffix.

Suffix	Meaning
B	Board number (for PHS-bus-based systems)
M	Module number (for MicroAutoBox II)
C	Channel number
G	Group number
CON	Converter number
BL	Block number
P	Port number
I	Interrupt number

A suffix is followed by the appropriate number. For example, DS2201IN_B2_C14 represents a digital input block located on a DS2201 board. The suffix indicates board number 2 and channel number 14 of the block. For more general block naming, the numbers are replaced by variables (for example, DS2201IN_Bx_Cy).

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 RTI CAN Blockset

Basics on the RTI CAN Blockset

Introduction

The RTI CAN Blockset lets you implement CAN bus communication based on CAN hardware of dSPACE real-time hardware.

Supported CAN hardware

The RTI CAN Blockset supports CAN hardware of the following dSPACE real-time hardware:

- MicroAutoBox II
- MicroLabBox
- Modular systems with a DS2202, DS2210, DS2211, and/or DS4302

Hardware requirements

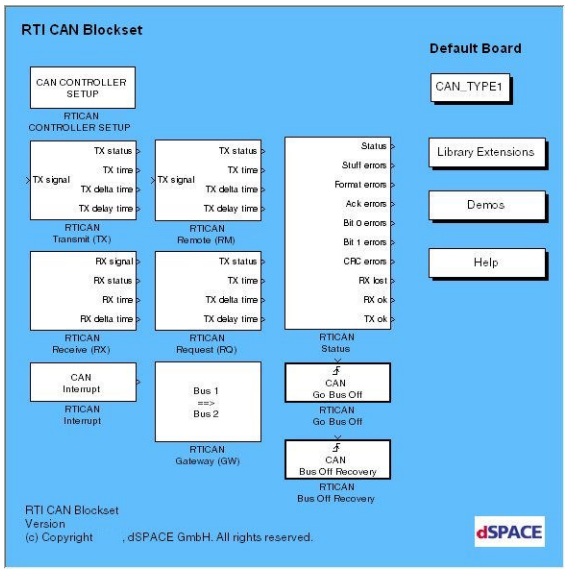
Supported MicroAutoBox II hardware:

DS1401 Base Board	MicroAutoBox II with I/O Boards			
	DS1507	DS1511	DS1513	DS1514
–	✓	✓	✓	–

Library access

You can open the RTI CAN Blockset by entering `rtican` in the MATLAB Command Window.

The following illustration shows the RTI CAN Blockset library.



For reference information, refer to:

- RTI CAN Blocks on page 55
- Library extensions:
  - C252 Support Blocks on page 145
  - TJA1041 Support Blocks on page 157
  - ISO11898-6 Support Blocks on page 169

Note

- To use any of the other RTI CAN blocks, the RTICAN CONTROLLER SETUP block must always be present in your model.
- (Relevant for I/O boards with CAN interface in a dSPACE modular system only) If several I/O boards with CAN interface are connected to different PHS buses of a multiprocessor system, identical board numbers are assigned to the boards. Each I/O board requires its own Setup block. To distinguish the Setup blocks, the Group ID number is set automatically for each block. The ID is used to identify the blocks and cannot be edited.
- The number of CAN messages you can implement in an application with RTI CAN Blockset is limited.
  See Limited Number of CAN Messages on page 186.

Demo models

For Simulink models that show how to use the RTI CAN Blockset, refer to the RTI demo library of the blockset. You will also find the model files in the `%ProgramData%\dSPACE\<InstallationGUID>\Demos\Rtican\` folder.

You can access the `%ProgramData%\dSPACE\<InstallationGUID>` folder via a shortcut in the Windows Start menu below dSPACE RCP and HIL <version>.

Information on installation and licensing

For information on installing dSPACE software and handling dSPACE licenses, refer to What Do You Want To Do? (Installing dSPACE Software 🕮).

Related topics

Basics

Limited Number of CAN Messages 186

References

C252 Support Blocks 145
ISO11898-6 Support Blocks 169
RTI CAN Blocks 55
TJA1041 Support Blocks 157

Working with the RTI CAN Blockset

Where to go from here

Information in this section

Setting Up a CAN Controller

Introduction

To use a dSPACE board with CAN bus interface, you have to set up the CAN controller.

Where to go from here

Information in this section

Initializing the CAN Controller

Introduction

The CAN controller performs serial communication according to the CAN protocol. You can take control of or communicate with other members of a CAN bus via the controller. This means you must configure the CAN controller — called the CAN channel — according to the application.

Standard configuration

You must specify the baud rate for the CAN application and the sample mode:

Sample Mode	Description
1-sample mode	(supported by all dSPACE CAN boards) The controller samples a bit once to determine if it is dominant or recessive.
3-sample mode	(supported by the DS4302 only) The controller samples a bit three times and uses the majority to determine if it is dominant or recessive.

The required bit timing parameters are automatically calculated by the dSPACE CAN software.

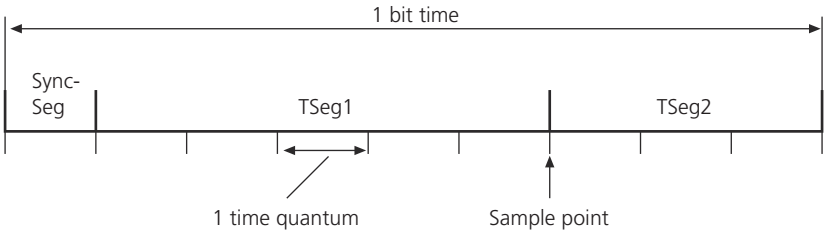
Advanced configuration (bit timing parameters)

The bits of a CAN message are transmitted in consecutive bit times. According to the CAN specification, a bit time consists of two programmable time segments and a synchronization segment:

TSeg1 Timing segment 1. The time before the sample point.

TSeg2 Timing segment 2. The time after the sample point.

SyncSeg Used to synchronize the various bus members (nodes).



The following parameters are also part of the advanced configuration:

SP Sample point. Defines the point in time at which the bus voltage level (CAN-H, CAN-L) is read and interpreted as a bit value.

SJW Synchronization jump width. Defines how far the CAN controller can shift the location of the sample point to synchronize itself to the other bus members.

BRP Baud rate prescaler value. The BRP defines the length of one time quantum.

SMPL Sample mode. Either 1-sample or 3-sample mode. Applicable to the DS4302 only.

Except for the SyncSeg parameter, you must define all these parameters via the values of the bit timing registers (BTR0, BTR1), located on the CAN controller.

Note

Setting up bit timing parameters requires advanced knowledge of the CAN controller hardware and the CAN bus hardware.

RTI support

You initialize a CAN controller with the RTICAN CONTROLLER SETUP block.

Refer to RTICAN CONTROLLER SETUP on page 57.

Related topics

References

CAN Transceiver Types

Introduction

To communicate with other bus members in a CAN bus, each bus member is equipped with a CAN transceiver. The transceiver defines the type of wire used for the bus (coaxial, two-wire line, or fiber-optic cables), the voltage level, and the pulse forms used for 0-bit and 1-bit values. The way in which CAN messages are transmitted on a CAN bus therefore significantly depends on the CAN transceiver used.

Note

Make sure that the CAN transceiver type used on the CAN bus matches the type on the dSPACE board you use to connect to the bus.

Terminating the CAN bus

Depending on the CAN transceiver type, you must terminate each CAN bus with resistors at both ends of the bus.

Note

Failure to terminate the bus will cause bit errors due to reflections. These reflections can be detected with an oscilloscope.

Supported transceivers

The following table lists dSPACE hardware and the supported transceivers:

dSPACE Hardware	Transceiver Type
▪ DS2202 ▪ DS2210 ▪ DS2211	ISO11898
DS4302	The following transceiver types are supported: ▪ ISO11898 ▪ RS485 ▪ C252 ▪ Piggyback[1)] **Note** The RTI CAN Blockset does not support transceiver types with different modes, for example single-wire and two-wire operation. Nevertheless, such transceiver types can be applied to the DS4302, but additional user-written S-functions are required to implement the communication between the piggyback module and the CAN controller.
MicroAutoBox II	The following transceiver types are supported: ▪ ISO11898 ▪ ISO11898-6[2), 3)]
MicroLabBox	The following transceiver types are supported: ▪ ISO11898 ▪ ISO11898-6[2)]

1) If none of the above transceivers matches your application or if a TJA1041 transceiver is used, "piggyback" must be selected as the transceiver type.
2) Selecting the ISO11898-6 transceiver type is required to perform partial networking.
3) Supported only by MicroAutoBox II with DS1513 I/O board.

ISO11898 transceiver

ISO11898 defines a high-speed CAN bus that supports baud rates of up to 1 MBd. This is the most commonly used transceiver, especially for the engine management electronics in automobiles.

CAN-H, CAN-L ISO11898 defines two voltage levels:

Level	Description
CAN-H	High if the bit is dominant (3.5 V), floating (2.5 V) if the bit is recessive.
CAN-L	Low if the bit is dominant (1.0 V), floating (2.5 V) if the bit is recessive.

Termination To terminate the CAN bus lines, ISO11898 requires a 120-Ω resistor at both ends of the bus.

ISO11898-6 transceiver

High-speed transceiver that supports partial networking.

Termination To terminate the CAN bus lines, ISO11898-6 requires a 120-Ω resistor at both ends of the bus.

Note

There are some limitations when you use the optional ISO11898-6 transceiver:

- No wake-up interrupt is implemented.
- Partial networking is supported only for the following baud rates:
  - 125 kbit/s
  - 250 kbit/s
  - 500 kbit/s
  - 1000 kbit/s

  Other baud rates can be used for normal CAN operation, but detecting wake-up messages for partial networking is supported only for the baud rates listed above.
- You have to enable **Automatic Wake Up** on the **Parameters Page (RTI<xxxx>_ISO11898_6_SST)** before you build the model. You cannot enable automatic wake-up during run time.
- If the transceiver is in *power on / listen only* mode, the CAN controller does not send an acknowledge message to the transmitter. The transmitter therefore continues to send the message until it receives the acknowledge signal. This might result in a task overrun if an RX interrupt is configured for the CAN controller.
- If the transceiver is in *power on / listen only* mode, it is not able to send CAN messages. Automatic wake-up is not possible if the transceiver is in *power on / listen only* mode. Because no message is sent on the CAN bus by the transceiver in *power on / listen only* mode, CAN arbitration fails. The CAN controller changes to the BUS OFF state. It is not possible to set the BUS state automatically to BUS ON via an interrupt, because the reason for the BUS OFF state still remains. You must set the CAN controller to BUS ON after you have switched the transceiver state to normal, standby, or sleep mode.

RS485 transceiver

The RS485 transceiver supports baud rates of up to 500 kBd. It is often used in the automotive industry. A CAN bus using this transceiver can connect up to 25 CAN nodes.

Termination To terminate the CAN bus lines, a 120-Ω resistor must be used at both ends of the CAN bus.

C252 fault-tolerant transceiver

The C252 fault-tolerant transceiver supports baud rates of up to 125 kBd. Its main feature is on-chip error management, which allows the CAN bus to continue operating even if errors such as short circuits between the bus lines occur.

When this transceiver is used, the CAN bus can interconnect nodes that are widely distributed. You can switch the C252 transceiver between sleep and normal (awake) mode.

Termination There are two ways to terminate the CAN bus lines: Use a 10 kΩ resistor for many connected bus members, or a 1.6 kΩ resistor if the number of bus members is equal to or less than five. The termination resistors are located between CAN-L and RTL and CAN-H and RTH (refer also to the "PCA82C252 Fault-tolerant Transceiver Data Sheet" issued by Philips Semiconductors).

Note

The TJA1054 transceiver is pin and downward compatible with the C252 transceiver. If the TJA1054 transceiver is on board the DS4302 and you want to use the fault-tolerant transceiver functionality, select "C252" in the RTI CAN CONTROLLER SETUP block. Refer to Unit Page (RTICAN CONTROLLER SETUP) on page 59.

Custom transceivers

The DS4302 allows you to mount up to four customization modules to use transceivers that are not on the DS4302.

Connecting customization modules For instructions on connecting customization modules, refer to Customization Modules (PHS Bus System Hardware Reference 🕮).

Optional TJA1041 transceiver dSPACE provides the optional TJA1041 that you can use as a custom transceiver for the DS4302. For a detailed description of the transceiver and the available transceiver modes, refer to the data sheet of the TJA1041 transceiver.

For details on the RTI support for the TJA1041 transceiver, refer to TJA1041 Support Blocks on page 157.

Note

There are some limitations when you use the optional TJA1041 transceiver:

- No wake-up interrupt is implemented.
- You have to enable **Automatic Wake Up** in the **DS4302_TJA1041_SST** on page 162 block before you build the model. You cannot enable automatic wake-up during run time.
- If the transceiver is in *power on / listen only* mode, the CAN controller does not send an acknowledge message to the transmitter. The transmitter therefore continues to send the message until it receives the acknowledge signal. This might cause a task overrun if an RX interrupt is configured for the CAN controller.
- If the transceiver is in *power on / listen only* mode, it is not able to send CAN messages. Automatic wake-up is not possible if the transceiver is in *power on / listen only* mode. Because no message is sent on the CAN bus by the transceiver in *power on / listen only* mode, CAN arbitration fails. The CAN controller changes to the BUS OFF state. It is not possible to set the BUS state automatically to BUS ON via an interrupt, because the reason for the BUS OFF state still remains. You must set the CAN controller to BUS ON after you have switched the transceiver state to normal, standby, or sleep mode.

DS2211: Selecting the CAN Controller Frequency

Introduction

Depending on the version of your DS2211, the board's CAN controller supports the following maximum clock frequency:

Board Version	Frequency
Up to 4.10	24 MHz
4.10 and higher	36 MHz

To get the board version of your DS2211, open the board's **Properties** dialog in the dSPACE experiment software.

Highest possible frequency automatically selected

The real-time application built with RTI CAN Blockset is compatible with both board versions and frequencies: During board initialization, the highest frequency that is available for the controller is automatically selected, together with the corresponding bit timing values. This applies regardless of the frequency you select in the **Advanced Configuration** dialog. Refer to Advanced Configuration Dialog (RTICAN CONTROLLER SETUP) on page 73.

Related topics

References

Advanced Configuration Dialog (RTICAN CONTROLLER SETUP)..... 73

MicroAutoBox II: Selecting the CAN Controller Frequency

Introduction

Depending on the CAN module version of your MicroAutoBox II, MicroAutoBox II supports the following maximum CAN controller clock frequency:

Module Version	Frequency
Up to 2.0	24 MHz
2.0 and higher	36 MHz
2.1 and higher	64 MHz

To get the CAN module version of your MicroAutoBox II, use the dSPACE experiment software.

Highest possible frequency automatically selected

The real-time application built with RTI CAN Blockset is compatible with both module versions and frequencies: During board initialization, the highest frequency that is available for the controller is automatically selected, together with the corresponding bit timing values. This applies regardless of the frequency you select in the **Advanced Configuration** dialog. Refer to Advanced Configuration Dialog (RTICAN CONTROLLER SETUP) on page 73.

Related topics

References

Advanced Configuration Dialog (RTICAN CONTROLLER SETUP)..... 73

Defining CAN Messages

Introduction

The dSPACE CAN software lets you easily define CAN messages to be transmitted or received.

Message types

You can define a message as a TX, RX, RQ, or RM message:

Message Type	Description
Transmit (TX)	This message is transmitted with a specific identifier. A TX message contains up to 8 bytes of data.
Receive (RX)	This message is *not* transmitted over the bus. An RX message is used only to define how the CAN controller processes a received message. An RX message transfers the incoming data from the CAN controller to the master processor.
Request (RQ)	First part of a *remote transmission request*[1)]. An RQ message is transmitted with a specific identifier to request data. An RQ message does not contain data.
Remote (RM)	Second part of a *remote transmission request*[1)]. An RM message is a TX message that is sent only if the CAN controller has received a corresponding RQ message. The RM message contains the data requested by the RQ message.

[1)] With RTI CAN Blockset, the remote transmission request is divided into an RQ message and an RM message. The meanings of the words "remote" and "request" used in this document do not correspond to those used in CAN specifications.

Message configuration

With RTI CAN Blockset, you have to implement one message block for each message. To define a message to be transmitted, for example, you must implement an RTICAN Transmit (TX) block.

Message configuration by hand You can perform message configuration by hand. In this case, you must specify the message identifier and identifier format (STD, XTD), the length of the data field, and the signals for each message. You also have to specify the start bit and length of each signal.

Message configuration from data file (data file support) You can load a data file containing the configuration of one or more messages. Then you can assign a message defined in the data file to a message block. Refer to Configuring CAN Messages via Data Files on page 32.

Multiple message access

Multiple message access allows you to place several RX or TX blocks with the same identifier and identifier format in one model. You can decode the signals of an RX message in several ways, or place TX blocks in several enabled subsystems to send data in various ways.

Delay time for message transmission

To distribute messages over time and avoid message bursts, you can specify delay times. A message is sent after the delay time. The delay time is a multiple of the time needed to send a CAN message at a given baud rate and identifier format. You can only enter a factor to increase or decrease the delay time.

RTI support

With RTI CAN Blockset, you have to implement one message block for each message. Refer to:

Message Type	RTI Block
Transmit (TX)	RTICAN Transmit (TX) on page 79
Receive (RX)	RTICAN Receive (RX) on page 92
Request (RQ)	RTICAN Request (RQ) on page 103
Remote (RM)	RTICAN Remote (RM) on page 109

Related topics

Basics

Configuring CAN Messages via Data Files..... 32

Implementing a CAN Interrupt

Introduction

The CAN controller transmits and receives messages and handles error management. It is also responsible for generating interrupts to the master processor. You can specify the events on which these interrupts are generated.

A special Bus Failure interrupt and a wake-up interrupt are available for the DS4302.

RTI support

You can implement a CAN interrupt with the **RTICAN Interrupt** block. Refer to RTICAN Interrupt on page 120.

Related topics

References

RTICAN Interrupt..... 120

Using RX Service Support

Concepts for receiving CAN messages

When CAN messages are received, RX blocks access the DPMEM between the master processor and the slave processor.

RTI CAN Blockset provides two concepts for receiving CAN messages:

- Common receive concept
- RX service receive concept

Common receive concept

According to the common receive concept, one data object is created in the DPMEM for each received CAN message. Due to the limited DPMEM size, the number of RX blocks you can use in a model is limited to 100 (200 for the DS4302).

RX service receive concept

When you enable RX service support, one data object is created in the DPMEM for all received CAN messages, and memory on the master processor is used to receive CAN messages. The RX service fills this memory with new CAN data. This concept improves run-time performance.

Tip

In contrast to the common receive concept, the number of RX blocks for which RX service support is enabled is unlimited.

Specifying a message filter When you use RX service, you have to specify a filter to select the messages to receive via RX service. To define the filter, you have to set up a bitmap that represents the message. Each bit position can be assigned 0 (must be matched), 1 (must be matched), or X (don't care). A message is received via RX service only if it matches the bitmap.

You can define the message filter on the RX Service Page (RTICAN CONTROLLER SETUP) on page 69.

Specifying the queue size When you use RX service, you have to specify the maximum number of CAN messages that you expect to receive in a sample step. The memory allocated on the master processor used to queue CAN messages is calculated from the specified maximum number of CAN messages.

Note

If more CAN messages than the specified **Queue size** are received in a sample step, the oldest CAN messages are lost. You should therefore specify the queue size so that no CAN messages are lost.

Example:

A CAN controller is configured to use the baud rate 500 kBd. The slowest RX block assigned to this CAN controller is sampled every 10 ms. At the specified baud rate, a maximum of about 46 CAN messages (STD format) might be received during two consecutive sample steps. To ensure that no CAN message is lost, set the queue size to 46.

Triggering an interrupt when a message is received via RX service You can let an interrupt be triggered when a message is received via RX service.

Note

You cannot let an interrupt be triggered when a message *with a specific ID* is received. An interrupt is triggered each time a message is received via RX service.

You can define the interrupt on the **Unit Page (RTI CAN Interrupt)** on page 121.

Precondition for gatewaying messages Enabling the RX service is a precondition for *gatewaying messages* between CAN controllers.

Refer to Gatewaying Messages Between CAN Buses on page 43.

Precondition for the TX loop back feature RX service allows you to use the *TX loop back feature*. The feature lets you observe whether message transfer over the bus was successful.

You can enable TX loop back on the **Options Page (RTICAN Transmit (TX))** on page 86.

Enabling RX service support

You have to enable RX service support for each CAN controller and for each RX block.

RTI support

- For a CAN controller, you enable the RX service on the **RX Service** page of the **RTICAN CONTROLLER SETUP** block. Refer to RX Service Page (RTICAN CONTROLLER SETUP) on page 69.

- For an RX block, you enable the RX service on the **Options** page of the **RTICAN Receive (RX)** block of the RTICAN CONTROLLER. Refer to Options Page (RTICAN Receive (RX)) on page 99.

Related topics

Basics

Gatewaying Messages Between CAN Buses....43

Removing a CAN Controller (Go Bus Off)

Introduction

If you use several CAN controllers, you can remove the one currently in use from the bus. Data transfer from the master to the slave processor is then stopped. You can select the CAN controller you want to remove from the bus via the **RTICAN Go Bus Off** block.

You can restart data transfer with another CAN controller or the same one with the **RTICAN Bus Off Recovery** block.

RTI support

- To remove a CAN controller from the bus, use the **RTICAN Go Bus Off** block. Refer to RTICAN Go Bus Off on page 129.
- To restart data transfer, use the **RTICAN Bus Off Recovery** block. Refer to RTICAN Bus Off Recovery on page 133.

Related topics

References

RTICAN Bus Off Recovery....133
RTICAN Go Bus Off....129

Getting CAN Status Information

Introduction

You can use the Error Management Logic (EML) of a CAN controller to get error and status information on the CAN bus and the controller. Errors occur, for example, if a CAN controller fails to transmit a message successfully.

CAN controller status information

The controller's EML has two counters: the Receive Error counter and the Transmit Error counter. According to their values, the EML can set the CAN controller to one of the following states:

Counter Value	Error State	Description
Each counter value < 128	Error active	The CAN controller is active. Before turning to the error passive state, the controller sets an error warn (EWRN) bit if one of the counter values is ≥ 96.
At least one counter value ≥ 128	Error passive	The CAN controller is still active. The CAN controller can recover from this state itself.
Transmit Error counter value ≥ 256	Bus off	The CAN controller disconnects itself from the bus. To recover, an external action is required (bus off recovery).

CAN bus status information You can get the following CAN bus status information:

Number of …	Description
Stuff bit errors	Each time more than 5 equal bits in a sequence occurred in a part of a received message where this is not allowed, the appropriate counter is incremented.
Form errors	Each time the format of a received message deviates from the fixed format, the appropriate counter is incremented.
Acknowledge errors	Each time a message sent by the CAN controller is not acknowledged, the appropriate counter is incremented.
Bit 0 errors	Each time the CAN controller tries to send a dominant bit level and a recessive bus level is detected instead, the appropriate counter is incremented. During bus off recovery, the counter is incremented each time a sequence of 11 recessive bits is detected. This enables the controller to monitor the bus off recovery sequence, indicating that the bus is not permanently disturbed.
Bit 1 errors	Each time the CAN controller tries to send a recessive bit level and a dominant bus level is detected instead, the appropriate counter is incremented.
Cyclic redundancy check (CRC) errors	Each time the CRC checksum of the received message is incorrect, the appropriate counter is incremented. The EML also checks the CRC checksum of each message (see Message fields on page 182).
Lost RX messages	Each time a message cannot be stored in the buffer of the CAN controller, the message is lost and an *RX lost error* is detected.
Successfully received RX messages	Each time an RX message is received successfully, the appropriate counter is incremented.
Successfully sent TX messages	Each time a TX message is sent successfully, the appropriate counter is incremented.
(DS4302 only) Status of fault tolerant receiver	The error state of the fault tolerant receiver is reported.
(DS4302 only) Fault tolerant transceiver	The value of the output is increased if a CAN bus events occurs.

RTI support To get status information, use the **RTICAN Status** block. Refer to RTICAN Status on page 124.

Related topics

References

RTICAN Status..... 124

CAN Partial Networking

Introduction

Principle of partial networking With CAN partial networking, selected ECUs in a network can be set to sleep mode or shut down if they do not have to run continuously. Wake-up messages then activate specific ECUs as and when required, and for as long as required.

Supported dSPACE real-time hardware Partial networking is possible for the following dSPACE real-time hardware only:

- MicroAutoBox II equipped with the DS1513 I/O board
- MicroLabBox

Specifying wake-up messages The RTI CAN Blockset lets you specify the CAN partial networking wake-up messages by filtering message IDs and message data:

- Filtering message IDs: You can define a message filter to select the messages to use as wake-up messages. The filter uses a bitmask which represents the message. A message passes the filter and is used as wake-up message only if it matches the bitmask.
- Filtering message data: You can mask the data bytes of incoming wake-up messages to determine whether they are valid wake-up messages.

Switching the CAN transceiver to sleep mode The CAN transceiver of the dSPACE real-time hardware is switched to sleep mode via the real-time application.

Tip

(MicroAutoBox II only) You can stop and power down the MicroAutoBox II with the DS1401_POWER_DOWN block from the DS1401 MicroAutoBox II Base Board II library. To set MicroAutoBox II to sleep mode, KL15 must be disconnected from the power supply. For further information, refer to DS1401_POWER_DOWN (MicroAutoBox II RTI Reference 🕮).
This is not possible for MicroLabBox.

Waking up dSPACE real-time hardware You can use partial networking messages to wake up dSPACE real-time hardware after its CAN transceiver is switched to sleep mode.

- (Relevant for MicroAutoBox II only) If a MicroAutoBox II's CAN transceiver is woken up via a partial networking message, the MicroAutoBox II behaves as if

it was powered up manually. Depending on where the real-time application is loaded (flash memory or RAM), the MicroAutoBox II starts the application or waits for further input.

- (Relevant for MicroLabBox only) Unlike MicroAutoBox II, MicroLabBox cannot be powered down and then woken up via partial networking messages. However, the CAN transceiver of MicroLabBox can be set to sleep mode, and then woken up via partial networking messages later on.

RTI support Refer to Partial Networking Page (RTICAN CONTROLLER SETUP) on page 71.

Related topics

References

Partial Networking Page (RTICAN CONTROLLER SETUP)..... 71

Configuring CAN Messages via Data Files

Introduction

Provides information on using a data file such as a DBC file for CAN message configuration.

Where to go from here

Information in this section

How to Configure Data File Handling

Objective

With RTI CAN Blockset, you have to implement one message block for each message. You can configure each message block by hand. You can also load a data file such as a DBC file that defines the configuration of one or more messages, and assign a message defined in the file to a message block.

Method

To configure data file handling

1. Enable data file support globally for your model on the **Data File Support** page of the **RTICAN CONTROLLER SETUP** block.

   Refer to Data File Support Page (RTICAN CONTROLLER SETUP) on page 64.
2. Add the data file you want to use for message configuration to the data file list. This is also done on the **Data File Support** page of the **RTICAN CONTROLLER SETUP** block.

Adding a data file provides the messages for further configuration of message blocks.

Note

The added data file is available only for message blocks configured for the specified CAN controller.

3 Enable data file support for the message block to be configured, and load a message from the data file on the block's **Message** page.

The signals contained in the message loaded to a message block are displayed on the block's **Message Composition** page.

4 Reload the data file after you modify it or to refresh the message configurations. Reloading is done on the **Data File Support** page of the RTICAN CONTROLLER SETUP block.

A log file (`rtican_reload_<modelname>_<date and time>.log`) is automatically generated in the MATLAB working directory.

Result

You configured data file handling.

Related topics

References

Data File Support Page (RTICAN CONTROLLER SETUP)..... 64

How to Use Multiple Data Files

Objective

Using multiple data files lets you work with different message configurations in the same model. This allows you, for example, to switch easily between different test environments.

Changing the data file during run time reduces compiling times, especially for large applications.

Restrictions

Ensure that only one configuration is active at the same time. Otherwise, the sent messages might be overwritten by the next message within the same sample step.

Method

To use multiple data files

1. Enable multiple data file support on the Options page of the RTICAN CONTROLLER SETUP block.

   Refer to Options Page (RTICAN CONTROLLER SETUP) on page 67.

2. Enable the RX service mode if you want to use multiple RX message blocks in your model. Otherwise, only the RX block computed first can receive any incoming message. This is done on the RX Service page of the RTICAN CONTROLLER SETUP block.

   Refer to RX Service Page (RTICAN CONTROLLER SETUP) on page 69.

3. Place several RX blocks with the same message identifier and identifier format in your model.

   You can decode the signal of an RX message in several ways. Assign the corresponding data file to each RX block.

4. Place several TX blocks with the same message identifier and identifier format in your model.

   You can place TX blocks in different enabled subsystems to send messages in various ways. Assign the corresponding data file to each TX block.

Result

You enabled the use of multiple data files.

Related topics

References

Using Data File Formats Other Than DBC

Introduction

RTI CAN Blockset supports the DBC file format. To make the data in a DBC file available to RTI, RTI CAN Blockset provides an interface file specific for the DBC file format that is used as a parser. To use your own data file format, you have to create a specific interface file.

Note

Creating an interface file requires knowledge of programming M files or programming in C++ for creating MEX files.

interface_function

Purpose To parse data file formats other than DBC.

Syntax

```
[MessageList,ErrMsg]=interface_function(FileName)
```

Description To parse data file formats other than DBC, you have to write your own interface file.

Input parameter The following input parameter is available:

Parameter	Description
`FileName`	Path and name of the data file to be parsed.

Return parameters The following return parameters are available:

Parameter	Description
`MessageList`	Array with a list of all messages and all configured signals, message comments, and signal comments. The description is empty if a fatal error occurs. The array must have the following structure:

```
MessageList                          1-by-1 structure
.NumMsg                              integer
.Message (1:NumMsg)                  1-by-n structure array
            .MsgName.                string
            .MsgId                   integer
            .MsgIdFormat             integer [0="STD", 1="XTD"]
            .MsgLength               integer
            .MsgOnBlocknameSelected  integer [0=no, default1=yes](X)
            .NumSignals              integer
            .MsgIdNumFormat          integer [0="dec", 1="hex"]  (X)
            .MsgComment              1-by-n cell array of strings, max.
                                     512 characters in total, each line is a cell string
            .Signal (1:NumSignals)   1-by-n structure array
                 .SgnName                 string
                 .SgnStartBit.            integer
                 .SgnLength               integer
                 .SgnType                 integer    [0="standard", 1="mode", 2="mode dependent"]
                 .ModeValue               integer   [-1|0,1,2,3] -1 = no ModeValue
                 .DataType                integer
                 .SLDataType              string    [default = "SS_DOUBLE"]
                 .ByteLayout              integer   [0="little endian",
                                                     1="big endian", 2="motorola back"]
                 .Factor                  float
                 .Offset                  float
                 .Min                     float
                 .Max                     float
                 .SgnUnit                 string
                 .SgnComment              1-by-n cell array of strings, max.
                                          512 characters in total, each line is a cell string
```

Parameter	Description
	Note ▪ `MsgName`: All the messages must be sorted alphabetically in ascending order. ▪ `MsgOnBlocknameSelected`, `MsgIdNumFormat`: Must be added to be fully compatible with the internal message structures. ▪ `SgnStartBit`: All the signals must be sorted by ascending start bits ▪ The sequence of the struct members is free, but all members are required. ▪ The structure above shows the default values for each property. If your parser does not support all properties, enter the default values to complete the structure.
`ErrMsg`	Empty if no errors occurred. Otherwise, an error or warning string is returned: ▪ An error is returned if the interface function returns an empty `MessageList` structure. In that case, the string defined in the `ErrMsg` of the interface function is issued. ▪ A warning is returned if the interface function returns data for the `MessageList` structure, but `ErrMsg` is not empty. The data is assumed to be valid.

Implementing Custom Message Encoding and Decoding

Introduction

You can manipulate a CAN message after the standard encoding sequence (TX message, RM message), or before the standard decoding sequence (RX message). This lets you, for example, write a checksum value to the last data byte of a message.

Where to go from here

Information in this section

Basics on Custom Message Encoding and Decoding

Introduction

Manipulating a CAN message requires that you implement custom message encoding (TX message, RM message) and/or decoding (RX message) in a separate C file. In the file, you have to implement the desired encoding and/or decoding C functions.

Use case

You can implement custom message encoding, for example, to calculate a checksum for the first 7 data bytes of a message, and write the checksum value to the 8th data byte before it is transmitted.

Accessible CAN message data

First function parameter In your encoding/decoding C function, you can access the CAN message data via the first function parameter `UInt32 *canbytes`. This parameter points to an array (data type: UInt32) with 8 elements, each containing a message data byte. The first element contains the first data byte and so on.

Last function parameter The last function parameter of the encoding/decoding function is always `msgstruct`. This parameter points to the

dSPACE CAN-board-specific message structure object (<dSPACE CAN board>_canMsg). You can evaluate the actual message length by checking the field msgstruct->datalen.

Optional data exchange between model and C function

You can exchange data between the Simulink model and the C functions. You can specify optional inports and outports for the message blocks for this.

Optional inport To import data from the Simulink model to the C function, you can specify an optional inport. You have to define its width. Its data type is Double.

Enabling the optional inport is useful if you want to manipulate the encoding/decoding algorithm from within the model.

Optional outport To export data from the C function to the Simulink model, you can specify an optional outport. You have to define its width. Its data type is Double.

Enabling the optional outport is useful if you want to provide status information to the model.

C function prototypes

To implement custom message encoding and/or decoding, you can use one of the following C function prototypes. They depend on whether you enable the optional inport and/or outport.

Optional inport and outport enabled:

```
void function_name(UInt32 *canbytes, dsfloat *userinport, dsfloat
*useroutport, rtican_msg_info msgstruct)
```

Optional inport enabled, outport disabled:

```
void function_name(UInt32 *canbytes, dsfloat *userinport,
rtican_msg_info msgstruct)
```

Optional inport disabled, outport enabled:

```
void function_name(UInt32 *canbytes, dsfloat *useroutport,
rtican_msg_info msgstruct)
```

Where

`canbytes` Points to an array with 8 elements, each containing a message data byte. The first element contains the first data byte and so on.

`msgstruct` Points to the dSPACE CAN board-specific message structure object (<dSPACE CAN board>_canMsg).

RTI support

- You can specify custom encoding for TX messages on the *Message Encoding* page of the RTICAN Transmit (TX) block. Refer to Custom Encoding Page (RTICAN Transmit (TX)) on page 89.
- You can specify custom encoding for RM messages on the *Message Encoding* page of the RTICAN Remote (RM) block. Refer to Custom Encoding Page (RTICAN Remote (RM)) on page 117.

- You can specify custom decoding for RX messages on the *Custom Decoding* page of the RTICAN Receive (RX) block. Refer to Custom Decoding Page (RTICAN Receive (RX)) on page 100.

Related topics

HowTos

How to Implement Custom Message Encoding and Decoding 39

Examples

Example of Custom Message Encoding and Decoding 41

How to Implement Custom Message Encoding and Decoding

Objective

Implementing custom message encoding lets you you, for example, write a checksum value to the last data byte of a message.

Method

To implement custom message encoding and decoding

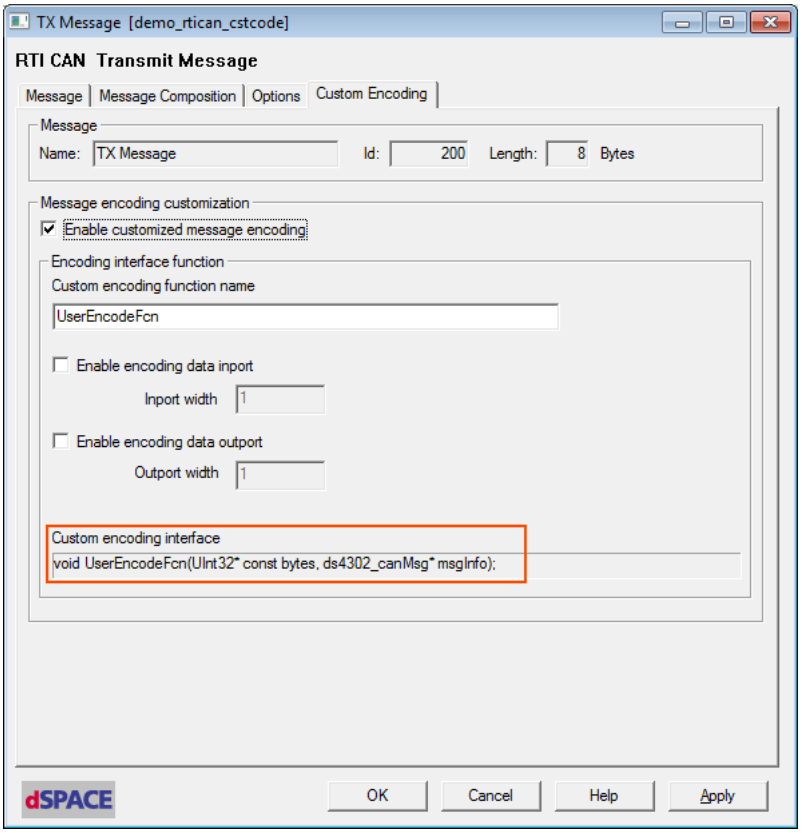
1. To implement custom encoding, open the TX or RM block of the message on which you want to perform custom encoding.

   To implement custom decoding, open the RX block of the message on which you want to perform custom decoding.
2. On the **Custom Encoding** page of the TX or RM block, click **Enable customized message encoding**.

   On the **Custom Encoding** page of the RX block, click **Enable customized message decoding**.
3. For the TX or RM block, enter the name of the C function to be used for encoding in the **Custom encoding function name** edit field.

   For the RX block, enter the name of the C function to be used for decoding in the **Custom decoding function name** edit field.

   The illustration below shows the **Custom Encoding** page of a TX block for a CAN controller on the DS4302. The **Custom encoding interface** field displays the required C function declaration.



4 If your encoding or decoding C function requires additional input from the Simulink model, click **Enable encoding data inport** or **Enable decoding data inport**, and specify the width of the inport.

   If your encoding or decoding C function provides additional output to the Simulink model, click **Enable encoding data outport** or **Enable decoding data outport**, and specify the width of the outport.

5 Create a C file including the `rtican_usercoding.h` header file.

   The header file is automatically generated by RTI CAN Blockset.

Note

Do not change the header file.

6 In the C file, implement the desired encoding/decoding functions. They must comply with the C function declaration displayed in the **Custom encoding/decoding interface** field of the block.

7 Save the C file to the model's working folder.

8 Open the user makefile of the model.

   The user makefile is automatically generated by RTI. For details, refer to User Makefile (USR.MK File) (RTI and RTI-MP Implementation Reference 🕮).

9 In the user makefile, add the C file name as an additional C source file (USER_SRCS).

This includes the C file in the next model compilation process.

Result

You have implemented custom message encoding and decoding.

Related topics

Basics

Basics on Custom Message Encoding and Decoding 37

Examples

Example of Custom Message Encoding and Decoding 41

References

Custom Decoding Page (RTICAN Receive (RX)) 100
Custom Encoding Page (RTICAN Remote (RM)) 117
Custom Encoding Page (RTICAN Transmit (TX)) 89

Example of Custom Message Encoding and Decoding

Introduction

The demo library of RTI CAN Blockset provides the 'User Encoding and Decoding Interface' demo for the DS4302. You have to adjust the demo to your dSPACE CAN board.

Demo files

The demo consists of the following files:

- `demo_rtican_usercoding.slx`
  This is the demo model file.
- `demo_rtican_coding_utility.c`
  This is the C file containing one encoding function and one decoding function. The encoding function is referenced by a TX block in the demo model. The decoding function is referenced by an RX block in the demo model.
- `demo_rtican_usercoding_usr.mk`
  This is the user makefile for the demo model. The makefile lists the `demo_rtican_coding_utility.c` as an additional C source file (`USER_SRCS`). This includes the C file in the model compilation process. For details, refer to User Makefile (USR.MK File) (RTI and RTI-MP Implementation Reference 🕮).

You can find the demo files in the `%ProgramData%\dSPACE\<InstallationGUID>\Demos\RTICAN\Usrcoding\` folder.

You can access the `%ProgramData%\dSPACE\<InstallationGUID>` folder via a shortcut in the Windows Start menu below dSPACE RCP and HIL <version>.

Adjusting the demo to your dSPACE CAN board

The demo model and the encoding/decoding functions implemented in the `demo_rtican_coding_utility.c` file are specific to the DS4302.

Adjusting the model To reuse the demo, you have to adjust the demo model to your specific dSPACE CAN board.

Adjusting the C functions The encoding/decoding C functions point to a CAN message data structure that is specific to each dSPACE CAN board. Refer to Accessible CAN message data on page 37. For this reason, you also have to adjust the encoding/decoding C functions to your specific dSPACE CAN board if you want to reuse the demo on a board other than the DS4302.

Related topics

Basics

Basics on Custom Message Encoding and Decoding 37

HowTos

How to Implement Custom Message Encoding and Decoding 39

Gatewaying Messages Between CAN Buses

Introduction

Gatewaying means exchanging CAN messages between controllers connected to different CAN buses.

There are different gateway methods.

Where to go from here

Information in this section

Guidelines for Selecting the Gateway Method

Introduction

To provide guidelines for selecting the best gateway method.

Gatewaying single messages

You should gateway single messages in the following cases:

- There are many CAN messages in your model, but you want to gateway only a few of them.
- You want to perform custom message encoding/decoding on the messages to be gatewayed.

For instructions, refer to How to Gateway Single Messages on page 44.

Gatewaying all messages using the RTICAN Gateway block

You should gateway all messages using the RTICAN Gateway (GW) block in the following cases:

- There are many CAN messages in your model, and you want to gateway all or most of them.
- You also want to perform custom message encoding/decoding only on a few messages to be gatewayed.

For instructions, refer to How to Gateway All Messages Using the RTICAN Gateway (GW) Block on page 45.

Related topics

HowTos

How to Gateway Single Messages

Objective

With RTI CAN Blockset, you can exchange single messages between CAN controllers connected to different CAN buses.

Guidelines

For guidelines on selecting the best gateway method, refer to Guidelines for Selecting the Gateway Method on page 43.

Restrictions

Gatewaying is possible only between *two* CAN controllers. You cannot implement a gateway between three or more CAN controllers. This means, you cannot exchange messages between controllers connected to three or more CAN buses.

Method

To gateway single messages

1. Add two RTICAN CONTROLLER SETUP blocks to your model.
2. Configure the RTICAN CONTROLLER SETUP blocks according to the two CAN controllers between which you want to gateway messages. You do not have to enable RX service.
3. For each message to be transmitted from bus 1 to bus 2, add a TX block and an RX block to your model.
4. Assign the RX blocks to the CAN controller connected to bus 1.
5. Assign the TX blocks to the CAN controller connected to bus 2.
6. If desired, specify custom message encoding/decoding.

Result

You implemented a (unidirectional) gateway for single messages.

Tip

You can implement custom message encoding/decoding also by using Simulink blocks that you place between the relevant RX/TX blocks.

Related topics

Basics

Guidelines for Selecting the Gateway Method....43

References

RTICAN CONTROLLER SETUP....57

How to Gateway All Messages Using the RTICAN Gateway (GW) Block

Objective

Using the **RTICAN Gateway (GW)** block, you can easily exchange all messages between two CAN buses.

Guidelines

For guidelines on selecting the best gateway method, refer to Guidelines for Selecting the Gateway Method on page 43.

RTICAN Gateway block

Instead of implementing a TX block and an RX block for each message to be gatewayed, you can use the RTICAN Gateway (GW) block. The block transmits all the messages received from CAN bus 1 to CAN bus 2, and vice versa.

Gatewaying via the RTICAN Gateway (GW) block is possible in the following cases:

- Between CAN controllers on the same dSPACE CAN board
- Between CAN controllers on different I/O boards with CAN interface if the boards are connected to the *same* processor board. For example, gatewaying is possible for a DS4302 and a DS2202 if they are connected to the same processor board.

Preconditions

RX service support must be enabled for both CAN controllers. Otherwise, RX messages will not be gatewayed.

Restrictions

- Gatewaying via the RTICAN Gateway (GW) block is *not* possible in the following cases:
  - Between a CAN controller on MicroAutoBox II and a CAN controller on any other dSPACE CAN board
  - Between a CAN controller on MicroLabBox and a CAN controller on any other dSPACE CAN board

- Gatewaying is possible only between *two* CAN controllers. You cannot implement a gateway between three or more CAN controllers. This means, you cannot exchange messages between controllers connected to three or more CAN buses.
- You cannot implement custom message encoding or decoding for message exchange via the **RTICAN Gateway (GW)** block. The block transfers all messages immediately after they are received.

  However, you can implement custom message encoding or decoding for single messages, and also exchange the messages via the **RTICAN Gateway (GW)** block. Implement a gateway for the messages as described in How to Gateway Single Messages on page 44, and enable RX service support for all RX messages as described in Enabling RX service support on page 27.
- You can exclude single messages from being exchanged using the **RTICAN Gateway (GW)** block. However, only messages that are defined within database files can be excluded from being exchanged.
- If you use both identifier formats (XTD and STD) in your model, there must be two **RTICAN Gateway (GW)** blocks in the model – one for each format. The **RTICAN Gateway (GW)** block can transfer messages in one format only.

Method

To gateway all messages using the RTICAN Gateway (GW) block

1. Add two **RTICAN CONTROLLER SETUP** blocks to your model.
2. Configure the **RTICAN CONTROLLER SETUP** blocks according to the two CAN controllers between which you want to gateway.
3. Enable RX service support for both **RTICAN CONTROLLER SETUP** blocks.
4. Add an **RTICAN Gateway (GW)** block to the model.

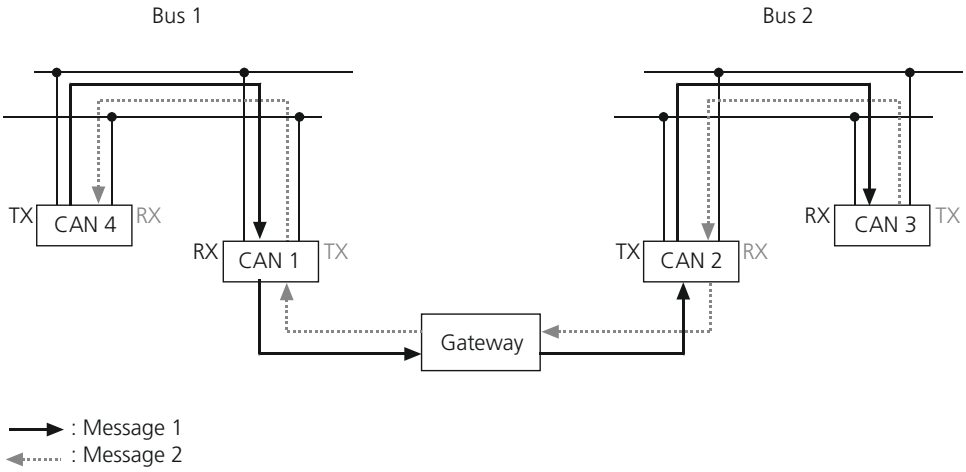
   If you use both identifier formats (XTD and STD) in your model, add a second **RTICAN Gateway (GW)** block.
5. In the **RTICAN Gateway (GW)** block(s), specify the CAN controllers between which you want to gateway messages.
6. If desired, exclude messages from being gatewayed. This is done in the **RTICAN Gateway (GW)** block(s).
7. Add a TX block to your model for each TX message on which you want to perform custom message encoding.
8. Add an RX block to your model for each RX message on which you want to perform custom message decoding.
9. Enable RX service support for the added RX blocks. This allows the **RTICAN Gateway (GW)** block to also exchange the RX messages on which you want to perform custom message decoding. Refer to Enabling RX service support on page 27.
10. Specify custom message decoding in the RX blocks.

Result

You implemented a gateway for all messages.

Example

The following figure shows a bidirectional gateway. The controllers 'CAN 1' and 'CAN 2' perform the message exchange between bus 1 and bus 2. They are located on dSPACE CAN boards.



Message 1 Controller 'CAN 4' sends message 1 to bus 1. The gateway block receives the message via controller 'CAN 1'. Message 1 is transported by using the memory of the processor board to controller 'CAN 2' via the gateway block. Controller 'CAN 2' transmits message 1 on bus 2. 'CAN 3' receives message 1.

Message 2 Controller 'CAN 3' sends message 2 to bus 2. The gateway block receives the message via controller 'CAN 2'. Message 2 is transported by using the memory of the processor board to controller 'CAN 1' via the gateway block. Controller 'CAN 1' transmits message 1 on bus 1. 'CAN 4' receives message 2.

Related topics

Basics

Guidelines for Selecting the Gateway Method....43

References

RTICAN Gateway (GW)....138

Switching the CAN Platform

Introduction

Provides information on reusing CAN models on different dSPACE CAN boards.

Where to go from here

Information in this section

How to Switch the CAN Platform

Objective

dSPACE offers several boards with RTI CAN functionality, and you can reuse your models on different platforms. For example, you can create your CAN models in the office using a DS2211 or DS4302. After completing and testing the models, you might want to use them on another platform such as MicroAutoBox II or MicroLabBox.

RTI CAN Blockset lets you convert the source platform model into the target platform model with the `rtican_boardtypeset` platform conversion function.

Number of boards/modules

The number of CAN controllers specified for the source platform is compared with the number on the target platform during model conversion. If the values are not compatible, conversion is aborted with an error message.

boardcnt parameter To calculate the number of available CAN controllers on the target, you can enter the number of available CAN boards or modules on the target platform using the (optional) `boardcnt` parameter of the `rtican_boardtypeset` platform conversion function.

Example of using the boardcnt parameter For example, to switch from a DS4302 platform to MicroAutoBox II with only one CAN module, execute the `rtican_boardtypeset` function like this:

```
rtican_boardtypeset(gcs, 'DS1401', 1)
```

The function calculates the maximum available CAN controllers on MicroAutoBox II as 2 (1 module with 2 CAN controllers). If the DS4302 model (source) contains more than 2 RTICAN_CONTROLLER_SETUP blocks, the function is aborted with an error code.

If you do not specify the `boardcnt` parameter, the function calculates the maximum available CAN controllers on MicroAutoBox II as 12 (6 modules, each providing 2 CAN controllers). A warning message informs you that there might be an invalid module/controller combination.

Method

To switch the CAN platform

1. Open the target RTI CAN Blockset in the MATLAB Command Window. For example, type **`RTI1401`**, open the MicroAutoBox II blockset and open the RTI CAN Blockset.
2. Open the model to be converted.

   Note

   If you instantiated the RTI CAN blocks in the model to be converted from a user-defined library (and not from the RTI CAN Blockset), you have to open the user-defined library to apply the `rtican_boardtypeset` (refer to rtican_boardtypeset Platform Conversion Function on page 50) conversion function to the library.

3. In the MATLAB Command Window, enter the `rtican_boardtypeset` (refer to rtican_boardtypeset Platform Conversion Function on page 50) conversion function with the appropriate parameters.

   After conversion, the converted model is opened.
4. Rename and store the converted model. Otherwise, the unconverted (original) model is overwritten.

Result

The `rtican_boardtypeset` conversion function validates the settings of the source platform and converts them into valid settings that depend on the target platform type:

- All the RTICAN_CONTROLLER_SETUP blocks are switched to the new platform.

- All configuration data is assigned to the new platform, for example, the bit timing configuration and the processor clock.

Note

- If you use, for example, a C252 transceiver with a DS4302 as the source platform, the transceiver is switched to ISO11898 (standard transceiver) since only DS4302 supports the C252 transceiver.
- If the baud rate exceeds the specific limitations of the target board, the baud rate is set to the limits and a warning message is generated. The name of the converted RTICAN_CONTROLLER_SETUP block is displayed so that you can find it in the model.

- The RTI CAN blocks of the source platform are renamed according to the target platform, and according to the RTI naming convention.

Related topics

References

Problems in Connection with Platform Switching 52
rtican_boardtypeset Platform Conversion Function 50

rtican_boardtypeset Platform Conversion Function

Purpose

To switch the CAN platform.

Description

RTI CAN Blockset lets you convert the source platform model into the target platform model with the `rtican_boardtypeset` platform conversion function.

For instructions, refer to How to Switch the CAN Platform on page 48.

Syntax

```
Error = rtican_boardtypeset(model, boardType [,boardCnt])
```

Input parameters

The following input parameters are available:

Parameter	Description
`model`	Lets you specify the name or handle of the model
`boardType`	Lets you specify the target board type (as a string) The valid target board type values are: ▪ `DS1202` (for MicroLabBox)

Parameter	Description
	▪ `DS1401` (for MicroAutoBox II) **Tip** The following tokens are also valid for MicroAutoBox II: ▪ `MABX` ▪ `CAN_TYPE1` ▪ `DS2202` ▪ `DS2210` ▪ `DS2211` ▪ `DS4302`
`boardCnt`	Lets you specify the number of boards or modules (MicroAutoBox II) on the target platform to calculate the number of available target CAN controllers. The parameter is optional. For details on this parameter, refer to Number of boards/modules on page 48.

Return parameters

The following values are returned by the function:

Value	Description
0	Model conversion was successful.
1	RTI CAN blocks were switched to a CAN interface that does not support the board number or controller number assigned to the RTI CAN blocks. The generated model is executable, but several warning messages might be generated. Refer to Problems in Connection with Platform Switching on page 52. Check whether the calculated settings meet your requirements.
2	Conversion process aborted because of an invalid board/controller combination. Refer to Problems in Connection with Platform Switching on page 52.
3	A severe error occurred. For example, you tried to convert a locked model (library) or the parameter settings of the function are invalid.

Related topics

HowTos

How to Switch the CAN Platform 48

References

Problems in Connection with Platform Switching 52

Problems in Connection with Platform Switching

Introduction

To perform platform switching, the CAN configuration of the source and target platform must be compatible.

Configurations that do not match

Mismatched configurations can lead to conversion problems. All known cases are listed below:

Case	Behavior or Result
The number of CAN controllers in the source model exceeds the number of controllers in the target model.	This is error code 2. The conversion process exits with an error message, for example: "The model is configured for 2 CAN controllers. The target RTI platform supports maximum 1 CAN controller(s)." "As a result the model cannot be converted successfully."
The source model contains more CAN controllers than the target platform has on one board, but the target can handle multiple boards or modules. For example, you use MicroAutoBox II or DS1007 with several DS4302s as targets.	This is error code 1. The conversion tries to use the next available CAN board and the first free CAN controller on it. If it is successful, the source CAN controller configuration is switched to the target configuration. The function returns "1". One of the following warning message is generated: "The model contains a CAN board/controller combination of B1, C3. This is not supported by the target board type "MABX"!" "The converting process calculates the other probably valid combination M2, C1!" "Please check your model now for usability of this new combination M2, C1!" "If this combination is invalid please close the model without saving to restore all made changes!"
If you specified the `boardCnt` parameter with the correct value, no message is shown. The conversion function calculates a valid combination with the limitation settings made with the `boardCnt` parameter.	This is error code 2. If no valid combination was found, an error message is generated, for example: "Could not calculate a valid CAN board and controller configuration." "As result this model could not be converted successfully --> process aborted."
The source model uses a baud rate which is not supported by the target platform.	This is error code 1. The conversion function calculates the nearest valid baud rate for the target platform and sets all affected CAN controllers to this value. A warning message is generated, for example: "Invalid baud rate detected in DS4302CAN_SETUP_M1_C1 changed to valid minimum of 3.125 kBd automatically."
The source model uses a transceiver type which is not supported by the target platform.	This is error code 1. The transceiver is automatically switched to a valid transceiver, normally to ISO11898. A warning message is generated, for example: "Invalid transceiver settings detected in CAN_TYPE1_SETUP_M1_C1, changed to valid transceiver 'ISO11898' automatically."
The source model uses a sample mode which is not supported by the target platform.	This is error code 1. A sample mode of 3 is only supported on the DS4302. The setting is automatically switched to sample mode 1. A warning message is generated, for example: "Invalid sample mode settings detected in CAN_TYPE1_SETUP_M1_C1, changed to valid sample mode of '1' automatically."

Related topics

HowTos

References

RTI CAN Blocks

Where to go from here

Information in this section

RTICAN CONTROLLER SETUP

Purpose

To define the global settings that apply to the CAN hardware such as the board/module type, board/module number, controller and baud rate.

Where to go from here

Information in this section

Block Description (RTICAN CONTROLLER SETUP)

Illustration

RTICAN
CONTROLLER SETUP
Group ID: RTICANx

Purpose

To define the global settings that apply to the CAN hardware such as the board/module type, board/module number, controller and baud rate.

Description

You can use either the standard configuration or the advanced configuration options.

Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- Each CAN controller must be specified by a separate RTICAN CONTROLLER SETUP block in the model.
- (Relevant for I/O boards with CAN interface in a dSPACE modular system only)
  If several I/O boards of the same type, e.g., DS2202, are connected to different PHS buses of a multiprocessor system, identical board numbers are assigned to the boards. Each I/O board requires its own RTICAN CONTROLLER SETUP block. To distinguish the Setup blocks, the Group ID number is set automatically for each block. The ID is only set to identify the blocks and cannot be edited.

Dialog pages

The dialog settings can be specified on the following pages:

- **Unit Page** (refer to Unit Page (RTICAN CONTROLLER SETUP) on page 59)
- **Data File Support Page** (refer to Data File Support Page (RTICAN CONTROLLER SETUP) on page 64)
- **Options Page** (refer to Options Page (RTICAN CONTROLLER SETUP) on page 67)
- **RX Service Page** (refer to RX Service Page (RTICAN CONTROLLER SETUP) on page 69)
- (Relevant for MicroLabBox and MicroAutoBox II equipped with the DS1513 I/O board only) **Partial Networking Page** (refer to Partial Networking Page (RTICAN CONTROLLER SETUP) on page 71)
- **Advanced Configuration Dialog** (refer to Advanced Configuration Dialog (RTICAN CONTROLLER SETUP) on page 73)

Example

For an example on how to use signal data typing, see Example of Using CAN Signal Data Typing on page 76.

Related topics

Examples

Example of Using CAN Signal Data Typing....76

References

Advanced Configuration Dialog (RTICAN CONTROLLER SETUP)....73
Data File Support Page (RTICAN CONTROLLER SETUP)....64
Options Page (RTICAN CONTROLLER SETUP)....67
Partial Networking Page (RTICAN CONTROLLER SETUP)....71
RX Service Page (RTICAN CONTROLLER SETUP)....69
Unit Page (RTICAN CONTROLLER SETUP)....59

Unit Page (RTICAN CONTROLLER SETUP)

Purpose

To set up a CAN controller.

Dialog settings

CAN controller specification

- Module/Board type:

  Lets you select the module/board type:

dSPACE Hardware	Module/Board Type
DS2202	DS2202
DS2210	DS2210
DS2211	DS2211
DS4302	DS4302
▪ MicroAutoBox II ▪ MicroLabBox	CAN Type 1

- Module/Board number:

  Lets you specify the module/board number:

dSPACE Hardware	Module/Board Number
▪ DS2202 ▪ DS2210 ▪ DS2211 ▪ DS4302 ▪ MicroAutoBox II	The valid numbers are 1 … 16.
MicroLabBox	The valid number is 1.

- Controller number:

  Lets you select the number of the CAN controller. The valid numbers are:

dSPACE Hardware	Controller Number
▪ DS2202 ▪ DS2210 ▪ DS2211 ▪ MicroAutoBox II ▪ MicroLabBox	1 … 2
DS4302	1 … 4

For the mapping of CAN controller numbers to CAN signals and pins of dSPACE hardware, refer to the topics in the table below:

dSPACE Hardware		Topic with Mapping Information
DS2202		Signal Mapping to I/O Pins (PHS Bus System Hardware Reference 🕮)
DS2210		Signal Mapping to I/O Pins (PHS Bus System Hardware Reference 🕮)
DS2211		Signal Mapping to I/O Pins (PHS Bus System Hardware Reference 🕮)
DS4302		Signal Mapping to I/O Pins (PHS Bus System Hardware Reference 🕮)
MicroAutoBox II	1401/1507	Signal Mapping to I/O Pins (1401/1507) (refer to Interfaces (MicroAutoBox II Hardware Reference 🕮))
	1401/1511	Signal Mapping to I/O Pins (1401/1511) (refer to Interfaces (MicroAutoBox II Hardware Reference 🕮))
	1401/1511/1514	Signal Mapping to I/O Pins (1401/1511/1514) (refer to Interfaces (MicroAutoBox II Hardware Reference 🕮))
	1401/1513	Signal Mapping to I/O Pins (1401/1513) (refer to Interfaces (MicroAutoBox II Hardware Reference 🕮))
	1401/1513/1514	Signal Mapping to I/O Pins (1401/1513/1514) (refer to Interfaces (MicroAutoBox II Hardware Reference 🕮))
MicroLabBox		Signal Mapping to I/O Pins (refer to CAN Connector (Sub-D) (MicroLabBox Hardware Installation and Configuration 🕮))

(Relevant for MicroAutoBox II only) Mapping between module/controller numbers in RTI and interface connector pin names:

Combination of ...		Name in Pin Description[1]
Module Number[2] ...	Controller Number[2] ...	
1	1	CAN 1
1	2	CAN 2
2	1	CAN 3
2	2	CAN 4
3	1	CAN 5
3	2	CAN 6

[1] As used in the MicroAutoBox II Hardware Reference 🕮 documents

[2] As used by RTI

(Relevant for MicroLabBox only) Mapping between module/controller numbers in RTI and interface connector pin names:

Combination of ...		Name in Pin Description
Module Number[1] ...	**Controller Number[1] ...**	
1	1	CAN 1
1	2	CAN 2

[1] As used by RTI

- Group ID: Shows the Group ID of the RTICAN CONTROLLER SETUP block. The Group ID is used to distinguish several RTICAN CONTROLLER SETUP blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own RTICAN Controller Setup block.
- Transceiver type:

  Lets you select the transceiver type for your CAN bus.

dSPACE Hardware	Transceiver Type
▪ DS2202 ▪ DS2210 ▪ DS2211	ISO11898
DS4302	The following transceiver types are supported: ▪ ISO11898 ▪ RS485 ▪ C252 ▪ Piggyback[1] **Note** The RTI CAN Blockset does not support transceiver types with different modes, for example single-wire and two-wire operation. Nevertheless, such transceiver types can be applied to the DS4302, but additional user-written S-functions are required to implement the communication between the piggyback module and the CAN controller.
MicroAutoBox II	The following transceiver types are supported: ▪ ISO11898 ▪ ISO11898-6[2), 3)]
MicroLabBox	The following transceiver types are supported: ▪ ISO11898 ▪ ISO11898-6[2)]

[1] If none of the above transceivers matches your application or if a TJA1041 transceiver is used, "piggyback" must be selected as the transceiver type.
[2] Selecting the ISO11898-6 transceiver type is required to perform partial networking.
[3] Supported only by MicroAutoBox II with DS1513 I/O board.

- Bus configuration register:

  Lets you configure the bus configuration register.

dSPACE Hardware	Bus Configuration Register
▪ DS2202 ▪ DS2210 ▪ DS2211 ▪ MicroAutoBox II ▪ MicroLabBox	Disabled because the dSPACE hardware does not support piggyback.
DS4302	Enabled only if "piggyback" is selected in the **Transceiver type** list. In the bus configuration register, you configure the CAN controller for the physical/electrical characteristics of the CAN bus according to the piggyback. Enter the value for the bus configuration register of the CAN controller in hexadecimal format. If a TJA1041 transceiver is used, enter **0x40** for the bus configuration register.

- Termination resistance:

  Lets you specify the termination resistance for the CAN bus.

dSPACE Hardware	Termination Resistance
▪ DS2202 ▪ DS2210 ▪ DS2211	Select **120 Ω**[1] or **off**.
DS4302	The selectable termination resistance depends on the transceiver selected in the **Transceiver Type** list: ▪ C252 transceiver: Select 1.6 kΩ[2] or 10 kΩ[2]. Refer to Line Termination (PHS Bus System Hardware Reference 🕮). ▪ For all other transceiver types: Select **120 Ω**[1] or **off**.
MicroAutoBox II	The options for 120 Ω and "off" are enabled. ▪ MicroAutoBox II 1401/1513 and 1401/1512/1513 variants: Select **120 Ω**[1] or **off**. ▪ All other MicroAutoBox II variants: There is no termination resistance installed on MicroAutoBox II. This means that there is no connection between CAN-Low and CAN-High. This setting therefore has no effect on the hardware.
MicroLabBox	Select **120 Ω** or **off**.

[1] The termination resistance of 120 Ω is installed on the dSPACE hardware. If you select this termination resistance, the CAN bus is physically terminated with the 120-Ω resistor at the CAN controllers end (side).

[2] The termination resistance is installed on the dSPACE hardware.

Standard configuration Lets you set up the CAN controller by specifying the baud rate.

When you select the standard configuration, a configuration for the bit timing registers of the CAN controller is automatically determined.

Tip

No detailed hardware information is required for the standard configuration.

Baudrate Lets you enter the baud rate in kBd.

The following table shows the minimum baud rate:

dSPACE Hardware	Minimum Baud Rate [Bd]
▪ DS2202 ▪ DS2210 ▪ DS2211 ▪ DS4302 ▪ MicroAutoBox II	10 k
MicroLabBox	20 k

The following table shows the maximum baud rate:

dSPACE Hardware	Maximum Baud Rate [Bd]
▪ DS2202 ▪ DS2210 ▪ DS2211 ▪ MicroAutoBox II	1 M
DS4302	The maximum baud rate depends on the chosen transceiver type: ▪ 1 M (ISO11898 and piggyback) ▪ 500 k (RS485) ▪ 125 k (C252)
MicroLabBox	10 M

Advanced configuration Disables the baud rate setting of the standard configuration, and lets you specify advanced configuration settings.

Note

Detailed knowledge of the CAN controller hardware and the CAN bus hardware is required for the advanced configuration.

Edit Lets you edit advanced configuration settings in the Advanced Configuration Dialog (RTICAN CONTROLLER SETUP).

Related topics

References

Block Description (RTICAN CONTROLLER SETUP)..... 57

Data File Support Page (RTICAN CONTROLLER SETUP)

Purpose

To enable data file support and configure the data file(s) to be used.

Description

You can configure TX, RX, RQ and RM message blocks via a data file. Enabling data file support enables the corresponding controls in the dialogs of the TX, RX, RQ and RM message blocks.

For details on data files, refer to Configuring CAN Messages via Data Files on page 32.

Data file path After you save the model, the path of each referenced data file is stored relative to the model root path. This allows you to copy an entire model to another folder without having to reload the data file. The data file list, however, displays the file's absolute path.

Note

- Data file support does not detect whether another data file was specified or the data file was modified. If the new file contains message modifications for specific message blocks, you must reload the message definitions of those blocks.
- You can specify your own signal labels. They are not overwritten by RTI CAN Blockset. Exception: if you change the signal name, the new signal name overwrites the current label.
- The automatic reload generates a log file in the MATLAB working folder.
- For data file formats other than *DBC*, you must write your own M files or MEX files.

Dialog settings

Use data file(s) Lets you enable all the controls for data file support. The supported default file format is the one used by CANalyzer database (DBC) files from Vector Informatik GmbH. In addition, an interface file must be specified to read the data file.

Data file list Lists all the added data files. All actions such as Remove or Reload affect the currently selected data file. The file path shown is always the actual absolute path, derived from the relative path. If the path name is longer

than 40 characters, only parts of it are displayed. Until the model is saved, the absolute path is stored since the relative path is not yet defined.

Add data file Lets you add a data file to the data file list. If the data file is already part of the list, no action is performed. The number of data files is only limited by the memory size provided by MATLAB/Simulink.

Note

If you add a DBC file with Motorola formats, some restrictions apply:

- Although the Motorola "forward" format is correctly transferred, DBC files saved with the Motorola "forward" format appear as Motorola "backwards" in the dialogs.
- CAN signals imported from DBC files in Motorola format are not checked by RTI CAN for signal overlapping and signal boundaries.

Replace data file Lets you replace the data file currently selected by another one. If the current data file is assigned to at least one message block, you get a warning to avoid unintended disconnection of the blocks. If you click OK, the data file is replaced, and all the assigned message blocks are reloaded immediately.

Note

You can no longer restore the old configuration after it has been replaced. You can restore it only by closing the model without saving it. Alternatively, you can add the old data file to the list and reassign it to the message blocks concerned.

Remove data file Lets you remove the currently selected data file. Before the data file is removed, you get a message that warns you against unintentional disconnection of message blocks assigned to the data file. Afterwards, you have to parameterize the message blocks manually. The Use data file option is disabled automatically.

If you use multiple data files and remove one, the CAN controller and data file-dependent message blocks are disconnected from that data file. If you open a block, the data file list is switched to "no file selected". This indicates that the block is not configured. You can select another data file or configure the block manually. The data file list is not disabled until the last data file is removed.

Note

- If you do not open the affected blocks the old configuration is still valid and is not removed. A "Reload data file" action does not affect the blocks.
- If you add a data file again after it was removed, the unopened blocks are automatically reconnected to the data file, regardless of whether or not you saved the model.

Reload data file Lets you reload the selected data file and all the parameters of TX, RX, RM and RQ messages that are connected to the RTICAN CONTROLLER SETUP block and the data file. A log file (`rtican_reload_<modelname>_<date and time>.log`) is automatically generated in the MATLAB working directory. It contains the following:

- Name of the reloaded data file
- Date and time of the reload process
- The number of blocks affected by the reload process
- The name of the blocks that caused the connections in the model to be opened
- The name of the blocks whose inport or outport configuration was changed by the reload. The blocks are shown only if they have not caused the connections between the blocks to be opened.

Note

Close all open message dialogs before you reload a data file.

Reload All Data Files Lets you automatically reload all the configured data files of the data file list. The log file created after the reload summarizes the complete reload for all files. If an error occurs, the complete reload is stopped, and you get a warning.

Note

Close all open message dialogs before you reload data files.

Interface file Lets you specify an interface file. You can either enter the file name directly in the edit field or select one by clicking the **Browse** button. The default entry in the edit field is `rtican_dbc_file_parser` for DBC data files.

Related topics

Basics

Configuring CAN Messages via Data Files....32

References

RTICAN CONTROLLER SETUP....57

Options Page (RTICAN CONTROLLER SETUP)

Purpose

To specify the settings for the acceptance mask and the signal data typing.

Description

The CAN controller features acceptance filtering on incoming messages. You can specify an acceptance mask for both message formats, STD and XTD. A mask classifies the identifier bits as "must match" or "don't care". Combined with the arbitration field containing the identifier of the message to be received, the mask allows you to switch off the reception of specific messages.

Dialog settings

CAN signal data typing The frame enables you to set the data type for RX, TX and RM blocks. Select one of the following options and assign the property to all the blocks stated above related to the RTICAN CONTROLLER SETUP block:

- Data type is always double: This is the default setting. All the signals made available by the RTICAN block configured for this controller have the data type Double.
- Data type is derived from the CAN signal specification: Depending on the signal length, data type, factor and offset the minimum and maximum signal values are calculated. The data types used on the signal outports are set according to these minimum/maximum values.
- Data type is derived from minimum/maximum CAN signal value: the CANalyzer data file format allows you to specify a minimum/maximum CAN signal value. These minimum/maximum signal values are used to determine the matching data type. See Example of Using CAN Signal Data Typing on page 76.
- Set signal data types: Click the button to assign the selected data type to all RX, TX and RM assigned to this RTICAN CONTROLLER SETUP block.

Note

- The **data type is derived from minimum/maximum CAN signal value:** this is only available if the **Use data file** checkbox in the **Data file support** frame is selected. If you clear this checkbox in a message block dialog and you have used this data typing option, the signal data types remain the same until you select another option in this dialog, because the minimum and maximum values are already stored in the block after reading the DBC file.
- Data types other than Double are only used for CAN signals if the value of the CAN signal is always an integer. This precondition is met if the factor and offset are integer values.
- The **data type is derived from the minimum/maximum CAN signal value** option is applied only to RX, TX and RM blocks on which the **Load parameters from data file** option is enabled.
- All TX, RX, and RM blocks with a deactivated **Use signal names on outports** checkbox always use Double data type. Refer to Options Page (RTICAN Transmit (TX)) on page 86, Options Page (RTICAN Receive (RX)) on page 99 and Options Page (RTICAN Remote (RM)) on page 116.

The following table shows the available data types:

Range	Integer	Data Type
0 … 1	✓	Boolean
–∞ … ∞	Don´t care	Double
–128 … 127	✓	Int8
0 … 255	✓	UInt8
–32768 … 32767	✓	Int16
0 … 65535	✓	UInt16
–2147483648 … 2147483647	✓	Int32
0 … 4294967295	✓	UInt32
Don´t care	–	Double

Use multiple message access mode Lets you select several RX and TX blocks for the same CAN message. The option is a precondition for multiple data file access. If you enable multiple message access, the error checks for ambiguous message configuration are disabled.

For details on data files, refer to How to Use Multiple Data Files on page 33.

Acceptance mask settings Lets you specify the masking feature for message filtering. Specifying a mask can be used to temporarily switch off the reception

of a certain class of identifiers. Mark the **use acceptance mask for** checkbox to enable the edit fields for the Acceptance mask and the Arbitration field.

Use acceptance mask for Lets you select an entry for the acceptance mask is generated in the TRC file. Choose whether you want to specify the acceptance mask for the STD or the XTD format. Click **hex** or **bin** to select the numerical format for the mask.

Acceptance mask Lets you enter the acceptance mask, using the corresponding format.

Arbitration field Lets you enter the identifier to be received, using the corresponding format.

Related topics

Examples

Example of Using CAN Signal Data Typing..... 76

References

RTICAN CONTROLLER SETUP..... 57

RX Service Page (RTICAN CONTROLLER SETUP)

Purpose

To enable and specify the RX service support for RTICAN RX blocks.

Tip

If RX service support is enabled, the number of RX blocks in a model is not limited.

Description

The RX service support uses a structure located on the master processor to receive messages. An interrupt can be triggered every time a message is received via the RX service. Refer to RTICAN Interrupt on page 120.

You can define a message filter to select which messages are received with the RX service and which with the common receive mode. You have to specify a message filter using a bitmask.

Dialog settings

STD/XTD Messages frame Specifies the RX service parameters for STD/XTD support.

- **Enable RX service for CAN STD/XTD Messages**: Lets you enable the service for standard/extended messages.

- **Binary representation of message filter:** Lets you specify the bitmask that you want to use as the message filter. A message passes the filter and is received via the RX service support only if it matches the bitmask.

  You can assign one of the following values to each bit position:

Value	Filter Behavior
0	Value must be matched
1	Value must be matched
X	Don't care

  For example, if you specify **`XXXXXXXX100`** as the bitmask, all the CAN message IDs that feature "**`100`**" in the 3 least significant bits are received. All the other bits of the identifier are ignored and set to "don´t care".

  The default bitmask is **`XXXXXXXXXXX`**, which means that all messages are received via RX service.

  Tip

  You can specify bitmasks *without spaces* although they have more than 8 signs. The spaces are automatically included when you leave the edit field.

- **Queue size:** Lets you enter the maximum number of CAN messages that you expect to receive. The number of CAN messages specifies the memory allocated on the master processor which is used to queue CAN messages.

  Note

  If more CAN messages than the specified **Queue size** are received in a sample step, the oldest CAN messages are lost. You should therefore specify the queue size so that no CAN messages are lost.

  Example:

  A CAN controller is configured to use the baud rate 500 kBd. The slowest RX block assigned to this CAN controller is sampled every 10 ms. At the specified baud rate, a maximum of about 46 CAN messages (STD format) might be received during two consecutive sample steps. To ensure that no CAN message is lost, set the queue size to 46.

Related topics

Basics

Using RX Service Support 25

References

RTICAN CONTROLLER SETUP 57

Partial Networking Page (RTICAN CONTROLLER SETUP)

Purpose

To enable partial networking, and to define messages for transceiver wake-up.

Note

Partial networking is possible for the following dSPACE real-time hardware only:

- MicroAutoBox II equipped with the DS1513 I/O board
- MicroLabBox

Basics on CAN partial networking

Principle of partial networking With CAN partial networking, selected ECUs in a network can be set to sleep mode or shut down if they do not have to run continuously. Wake-up messages then activate specific ECUs as and when required, and for as long as required.

Specifying wake-up messages The RTI CAN Blockset lets you specify the CAN partial networking wake-up messages by filtering message IDs and message data:

- Filtering message IDs: You can define a message filter to select the messages to use as wake-up messages. The filter uses a bitmask which represents the message. A message passes the filter and is used as wake-up message only if it matches the bitmask.
- Filtering message data: You can mask the data bytes of incoming wake-up messages to determine whether they are valid wake-up messages.

Switching the CAN transceiver to sleep mode The CAN transceiver of the dSPACE real-time hardware is switched to sleep mode via the real-time application.

Tip

(MicroAutoBox II only) You can stop and power down the MicroAutoBox II with the DS1401_POWER_DOWN block from the DS1401 MicroAutoBox II Base Board II library. To set MicroAutoBox II to sleep mode, KL15 must be disconnected from the power supply. For further information, refer to DS1401_POWER_DOWN (MicroAutoBox II RTI Reference 🕮).

This is not possible for MicroLabBox.

Waking up dSPACE real-time hardware You can use partial networking messages to wake up dSPACE real-time hardware after its CAN transceiver is switched to sleep mode.

- (Relevant for MicroAutoBox II only) If a MicroAutoBox II's CAN transceiver is woken up via a partial networking message, the MicroAutoBox II behaves as if it was powered up manually. Depending on where the real-time application is loaded (flash memory or RAM), the MicroAutoBox II starts the application or waits for further input.

- (Relevant for MicroLabBox only) Unlike MicroAutoBox II, MicroLabBox cannot be powered down and then woken up via partial networking messages. However, the CAN transceiver of MicroLabBox can be set to sleep mode, and then woken up via partial networking messages later on.

CAN controller partial networking frame

Enable Lets you enable or disable CAN partial networking for the transceiver.

- Partial networking is possible only if the correct ISO11898-6 transceiver type is selected.
- Partial networking is supported only for the following baud rates:
  - 125 kbit/s
  - 250 kbit/s
  - 500 kbit/s
  - 1000 kbit/s

  Other baud rates can be used for normal CAN operation, but detecting wake-up messages for partial networking is supported only for the baud rates listed above.

Refer to Unit Page (RTICAN CONTROLLER SETUP).

CAN controller wake-up mask settings frame

Identifier format (Available only if partial networking is enabled) Lets you select the STD or EXT format for the identifier mask used for CAN partial networking.

Identifier mask (Available only if partial networking is enabled) Lets you specify the bitmask that you want to use as the wake-up message filter. Each bit position can be assigned 0 (must be matched), 1 (must be matched), or X (don't care). A message is used as a partial networking wake-up message only if it matches the bitmask.

You can input bitmasks with more than 11 bits even if you selected the standard identifier format for filtering wake-up messages. RTI CAN then uses bits 10 ... 0 only, and further bits are ignored. If you input bitmasks with fewer bits than required by the selected identifier format, the required bits are set to X.

Wake-up mask extensions frame

Data mask used (Available only if partial networking is enabled) Lets you enable a data mask for filtering valid wake-up messages.

A data mask makes it possible to wake up a group (or several groups) of nodes simultaneously via a single wake-up message (see example below).

Data length (Available only if Data mask used is selected) Lets you select how many data bytes are expected in the data length code (DLC) of a wake-up message.

Messages with a different data length are ignored.

Data mask (Available only if Data mask used is selected) Lets you specify the data mask value in hexadecimal notation. The data mask is used for masking the relevant data bytes (as specified in the Data length field) of the wake-up

messages to determine valid wake-up messages. A logical AND operation is used between the data mask and the message.

Example:

You are working with CAN wake-up messages with a data length of 1 byte (**Data length** = 1). Each wake-up message can wake up several groups of network nodes simultaneously. A bit value of '1' at a certain bit position wakes up the related node group. For example, a wake-up message with the data field '11100010' wakes up group1, group2, group3, and group7. The specified **Data mask** is '0xA8' (which corresponds to the binary value '10101000').

When the wake-up message with data field '11100010' is received, it is evaluated against the specified data mask. Since there are two matching bits (bit 1 and bit 3), the wake-up message is valid and is used to wake up the node.

The table below illustrates this example:

	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	Group 7	Group 8
Mask	1	0	1	0	1	0	0	0
Received message	1	1	1	0	0	0	1	0
Wake-up result	✓ [1)]	-	✓ [1)]	-	-	-	-	-

[1)] Nodes of this group are woken up.

Related topics

Basics

Basics on ISO11898-6 Support Blocks..... 170

Advanced Configuration Dialog (RTICAN CONTROLLER SETUP)

Access

This dialog opens when you select advanced configuration and click the **Edit** button in the **Unit Page** of the RTICAN CONTROLLER SETUP on page 57 block.

Purpose

To specify the baud rate and either show all valid controller settings or enter the BTR values.

Dialog settings

The resulting controller settings are displayed in the Controller settings selection group.

Baudrate Lets you enter the baud rate in kBd.

The following table shows the minimum baud rate:

dSPACE Hardware	Minimum Baud Rate [Bd]
▪ DS2202 ▪ DS2210 ▪ DS2211 ▪ DS4302 ▪ MicroAutoBox II	10 k
MicroLabBox	20 k

The following table shows the maximum baud rate:

dSPACE Hardware	Maximum Baud Rate [Bd]
▪ DS2202 ▪ DS2210 ▪ DS2211 ▪ MicroAutoBox II	1 M
DS4302	The maximum baud rate depends on the chosen transceiver type: ▪ 1 M (ISO11898 and piggyback) ▪ 500 k (RS485) ▪ 125 k (C252)
MicroLabBox	10 M

External Clock (available only for DS2211 and MicroAutoBox II) Lets you select the CAN controller frequency used for calculating the BTR values. The available frequencies are 24 MHz, 36 MHz, and 64 MHz (MicroAutoBox II only). The related BTR values are calculated automatically.

Note

- You can have bit timing values calculated for 24 MHz and 36 MHz. RTI CAN Blockset calculates corresponding bit timing values for both frequencies. However, during board initialization, only the bit timing values that correspond to the highest possible frequency are loaded to the hardware.
- (Relevant for the DS2211 only) You can select 24 MHz or 36 MHz. It depends on the board version which clock frequency the CAN controller supports. Refer to DS2211: Selecting the CAN Controller Frequency on page 22.
- (Relevant for MicroAutoBox II only) You can select 24 MHz, 36 MHz, or 64 MHz. It depends on the board version which clock frequency the CAN controller supports. Refer to MicroAutoBox II: Selecting the CAN Controller Frequency on page 23.

Show all valid settings Disables the settings from **Enter BTR values**. The required CAN controller frequency is calculated as high as possible according to the selected sample mode and baud rate. Bit timing values are also calculated. The **Controller settings selection** frame lists all the possible combinations of

BTRO, BTR1, SP, SJW, TSEG1, TSEG2, and BRP values that are possible for the calculated frequency. Select one of the combinations. BTR values for the alternative CAN controller frequency are calculated if you use a DS2211 or MicroAutoBox II.

- **1 sample per bit**: Each bit is sampled only once.
- **3 samples per bit**: The controller samples each bit three times and uses the majority to decide if the bit is recessive or dominant.

Note

The 3-sample mode is supported only for the DS4302.

Enter BTR values Disables the settings of **Show all valid settings**, and enables the **BTR0** and **BTR1** edit fields. A baud rate and bit timing values (for SP, SJW, TSEG1, TSEG2, BRP and SMPL) are calculated according to the specified BTR0 and BTR1 values.

- **BTR0**: Lets you enter a value for Bit Timing Register 0. The value must be 8 bits long and hexadecimal.
- **BTR1**: Lets you enter a value for Bit Timing Register 1. The value must be 8 bits long and hexadecimal.

Controller settings selection Displays the controller frequency and the specified or calculated combinations for BTR0, BTR1, SP, SJW, TSEG1, TSEG2, BRP and SMPL.

- **Frequency:** (determined automatically) Displays the frequency at which the CAN controller runs.
- **Legend**: Click this button to get a list explaining the abbreviations BTR0, BTR1, ...
- **BTR0, BTR1, ...**: List of the possible combinations for BTR0, BTR1, SP, SJW, TSEG1, TSEG2, BRP, and SMPL. Click the **Legend** button to get a list explaining the abbreviations. If **Show all valid settings** is selected, you have to select one combination.

Note

The block appears in the TRC file only if receive acceptance masks have been specified. These masks can then be modified by using the dSPACE experiment software during a real-time simulation.

Related topics

References

RTICAN CONTROLLER SETUP 57

Example of Using CAN Signal Data Typing

Deriving the data type

The following example shows you how the data type is derived from the CAN signal specification.

A signal is defined by the following structure:

```
struct(char *SgnName,
          int SgnStartBit,
          int SgnLength,
          int SgnType,
          int ModeValue,
          int DataType,
          char *SLDataType,
          int ByteLayout,
          float Factor,
          float Offset,
          float Min,
          float Max,
          char *SgnComment,
          char *SgnUnit);
```

The minimum and maximum values of a possible signal are calculated via the bit length, factor and offset:

- `SgnStartBit = 8`
- `SgnLength = 16`
- `SgnType = signed integer`

This results in a minimum value of –32768 and a maximum value of 32767.

Example 1

Suppose the offset is –0.01 and the factor 0.03. The resulting minimum value is -983.05 (-32768 · 0.03 - 0.01) and the maximum value is 983. Because the offset and the factor are both of Float data type, the resulting data type is set to Double.

Example 2

Suppose the offset and the factor are both integers, the offset 100 and the factor 1. The resulting minimum value is -32668 (-32768 · 1 + 100) and the maximum value 32867 (32768 · 1 + 100). The number exceeds the 16-bit range valid for Int16. Therefore, the data type is set to Int32.

Related topics

References

Options Page (RTICAN CONTROLLER SETUP)..... 67

Controller Selection Dialog

Controller Selection Dialog

Access

The dialog is opened under the following conditions:

- More than one RTICAN CONTROLLER SETUP block is defined in your model. In this case, automatic assignment is not possible.
- More than one RTICAN CONTROLLER SETUP block is defined in your model. You select **Select CAN controller** on the **Unit** page of the block.

Note

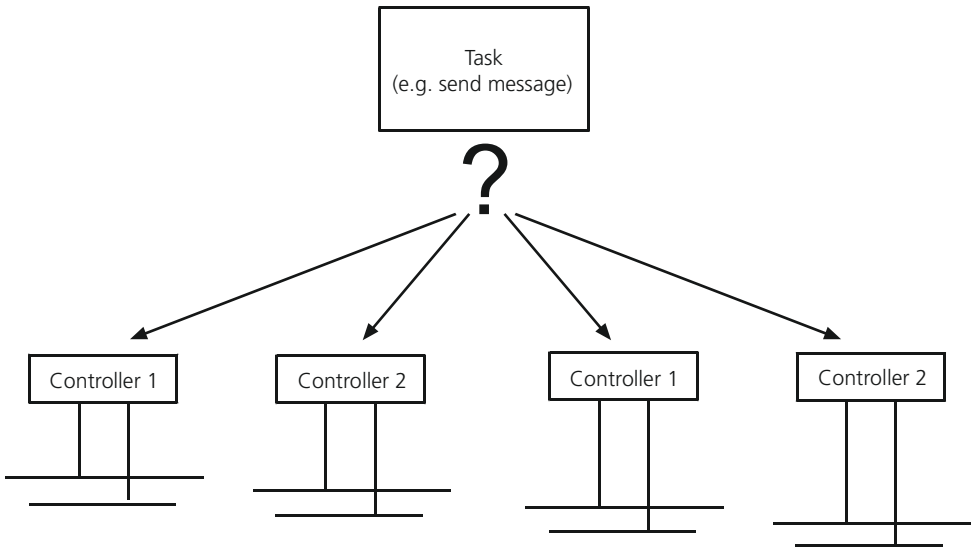
- The **Controller Selection** dialog is related to the most recent RTICAN CONTROLLER SETUP block. The block name is displayed in the dialog caption.
- Each CAN controller must be configured by a separate RTICAN CONTROLLER SETUP block in the model.

Purpose

To assign the selected RTICAN block to a specific CAN controller.

Description

If there is more than one RTICAN CONTROLLER SETUP block in your model, tasks such as sending or receiving messages must be assigned to a specific CAN controller. See the following illustration.



CAN bus lines

Usually, the assignment of a task such as sending or receiving messages to a specific CAN controller is done automatically by the dSPACE CAN software. If automatic assignment is not possible or not desired, you can change the assignment.

Dialog settings

Available CAN Controller Lists all the CAN controllers that are currently specified in your model. The list contains entries for the CAN controller's board/module type, board number and controller number.

- Controller list field: Select the RTICAN CONTROLLER SETUP block to which you want to assign the RTICAN block. Click OK.

  If none of the displayed CAN controllers is suitable, click Cancel and define a new suitable RTICAN CONTROLLER SETUP block.

Related topics

References

RTICAN CONTROLLER SETUP 57

RTICAN Transmit (TX)

Purpose

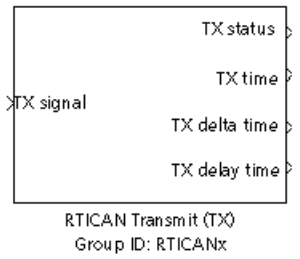
To encode and transmit a CAN message with a particular identifier. The message signals are delivered to the block via the signal inport.

Where to go from here

Information in this section

Block Description (RTICAN Transmit (TX))

Illustration



Purpose

To encode and transmit a CAN message with a particular identifier. The message signals are delivered to the block via the signal inport.

Note

- One RTICAN CONTROLLER SETUP block must already be in your model if you want to use the RTICAN Transmit TX block.
- Use a separate RTICAN Transmit TX block for each message to be sent, and for each controller that is about to send it.
- Signals in Motorola format are not checked for signal overlapping and signal boundaries within the CAN message. dSPACE therefore recommends to use the Motorola format only in conjunction with CANalyzer database files.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the Controller Selection Dialog may open. See Controller Selection Dialog on page 77 for details.
- The block has a varying number of inports and outports, depending on its configuration made on the Options Page.

I/O characteristics

The following table shows the characteristics of the block's input in Simulink. For details on the configuration, refer to Options Page (RTICAN Transmit (TX)).

Simulink Inport	Description
TX signal	Signal(s) of the TX message.
Delay TX command	Specifies the value that is used to calculate a delay time that must pass before the message is sent. Specifying a delay time allows you to distribute CAN messages evenly in time to avoid message bursts. The inport value must be an integer and is multiplied with the time required for a single CAN message to be sent at the specified baud rate and identifier format.
TX loop back	Activates or deactivates the loop back feature: ▪ "0" or smaller: TX loop back feature is deactivated ▪ "1" or higher: TX loop back feature is activated
Enable/Trigger port	Activates or deactivates the execution of the block.

The following table shows the characteristics of the block's output in Simulink. For details on the configuration, refer to Options Page (RTICAN Transmit (TX)).

Simulink Outport	Description
TX status	Provides status information related to the previous block execution: ▪ "0": Message not sent ▪ "1": Message sent
TX time	Provides the point in time (in seconds) the previous message was sent.
TX delta time	Provides the time difference (in seconds) between the sending times of the previous two messages.

Simulink Outport	Description
TX delay time	Provides the time difference (in seconds) between the point of time the TX block was executed and the point of time the message was sent successfully by the CAN controller.

Dialog pages

The following pages and dialogs are available:

- **Message Page** (refer to Message Page (RTICAN Transmit (TX)) on page 81)
- **Message Composition Page** (refer to Message Composition Page (RTICAN Transmit (TX)) on page 84)
- **Options Page** (refer to Options Page (RTICAN Transmit (TX)) on page 86)
- **Custom Encoding Page** (refer to Custom Encoding Page (RTICAN Transmit (TX)) on page 89)
- **Message Selection Dialog** (refer to Message Selection Dialog (RTICAN Transmit (TX)) on page 90) (only if data file support is enabled)

Related topics

References

Message Page (RTICAN Transmit (TX))

Purpose

To specify the settings for the CAN controller and the data file support.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- **Board/Module type**: Displays the board/module type on which the CAN support is built.
- **Board/Module number**: Displays the board/module number.
- **Controller number**: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click **Select CAN controller**.
- **Select CAN controller**: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Data file support (Enabled only if data file support is enabled on the Data File Support Page (RTICAN CONTROLLER SETUP)) Lets you parameterize the message block by hand or by loading parameters from a data file.

- Parameterize block by hand: Lets you parameterize the block by hand via the Signal editor on the Message Composition Page. Loading parameters from a data file is disabled.
- Load parameters from data file: Lets you use definitions from a data file. The data file list displays all the data files that were specified on the Data File Support Page (RTICAN CONTROLLER SETUP). If only one data file is specified the list is disabled. During compilation of the model it is checked whether the TX block is unique for each CAN message and controller used within the model. If you have enabled the Multiple message access option this check is disabled.

  Selecting the Load parameters from data file option enables the Load message button. Click this button to open the Message Selection Dialog, which lets you select a message for this RTICAN Transmit (TX) block.

TX message parameters Lets you define the basic message properties.

- Message name: (Enabled only if Parameterize block by hand is selected) Lets you enter a name for the message. This name is for the user only and has no impact on the CAN protocol.
- Use as block name: Lets you specify whether to use the message name as the block name in your model.

Note

Duplicate names in the same subsystem are not allowed. Therefore, Simulink automatically adds numbers to the name to create unique names in the subsystem.

- Message identifier: (Enabled only if Parameterize block by hand is selected) Lets you specify the identifier format, the numeric format for the identifier and the identifier's value. The value is entered in the edit field. Valid ranges are 0 … 2047 for the STD format, and 0 … 536870895 for the XTD format.

  Lets you specify:
  - The identifier format
  - The numeric format of the identifier
  - The identifier value. The value is entered in the edit field. The following table lists the valid ranges:

Identifier Format	Valid Range
STD	0 … 2047[1)]
XTD	0 … 536870895

[1)] Message identifiers in the range 2032 … 2047 may lead to malfunction if one of the CAN controllers on the connected CAN bus does not support that range. In this case, use only the identifier range 0 … 2031.

- **Message length**: Enabled only if **Parameterize block by hand** is selected) Lets you select the number of data bytes the message includes.

  Note

  If you select 0 byte, the message cannot include any data and the **Message Composition Page** is disabled. All the signal inports of the block are removed.

- **Block sample time**: Lets you specify the block sample time. The block sample time defines the time interval for the Simulink block to be executed. The default is –1, which means the inherited sample time. 0.0 is for the base sample time. Other negative values are not allowed.
- **Start time**: Lets you specify the start time, which is the time delay before the first message is sent. Specify a start time to avoid CAN bus conflicts, which may occur if all the CAN bus members send their messages simultaneously at the start of the real-time simulation. Valid values are in the range 0 … 420 seconds.

  The start time takes effect only in combination with the **Auto repeat time** feature.
- **Auto repeat time**: Lets you specify a time interval for automatically repeating message transmission, independently of new data. The valid values are in the range 0 … 100 seconds.

  Note

  - Do not specify an **Auto repeat time** if you want to define a message as a *trigger message* in your model.
  - To stop the transmission of a message for which you specified an **Auto repeat time**, you have to stop the running application using dSPACE experiment software.

- **Timeout**: Lets you specify a timeout for messages that cannot be sent successfully. After the specified time, the message is canceled. This is useful to avoid blocking the bus. Valid values are in the range 0 … 100 seconds.

Comments Lets you enter comments on your message for documentation purposes.

Related topics

References

RTICAN Transmit (TX) 79

Message Composition Page (RTICAN Transmit (TX))

Purpose

To specify the transmit message.

Dialog settings

Message Displays the most important information from the **Message Page**: the name, identifier and length of the message.

Message Composition Lets you define how the message is composed of individual signals.

- **List signals for mode**: Lets you select whether to display all the signals in the list or only the signals active for a specific mode. The entries in the list are "all modes" and all mode values already specified for mode-dependent signals.
- **Signal list**: Lists all the signals selected in the **List signals for mode** list and their settings.

  The signal settings are displayed in the following columns:
  - Inp (Input number)
  - Outp (Output number)
  - Signal name
  - SB (Start bit)
  - SL (Signal length)
  - ST (Signal type)
  - MV (Mode value)
  - DT (Data type)
  - BL (Byte layout)
  - Factor
  - Offset
- **New**: Lets you create a new signal. The **Signal editor** is enabled to specify the signal.
- **Copy**: (Enabled only if a signal is selected in the list) Lets you copy the signal selected in the list. The **Signal editor** is enabled to let you modify the copied signal.

  The start bit of the copied signal is calculated automatically depending on the last signal of the list. If you specified a signal name ending with a number, the number is automatically increased and the signal is added to the end of the list.

  To copy signals conveniently, click **Copy** and **Apply** alternately.
- **Delete**: (Enabled only if a signal was selected in the list) Lets you delete the selected signal.
- **Comments**: (Enabled only if a signal was selected in the list) Lets you open a dialog to enter comments on the selected signal for documentation purposes.

  If data file support is enabled, the **Comments** button is also enabled to let you read the signal comments stored in the data file. In this case, the signal comments are read-only.

Signal editor The signal editor lets you define the signals within a CAN message.

You can specify the following signal properties:

- **Signal name**: Lets you specify the name of the signal as a string.

  Note

  - All the signal names in the same message must be unique.
  - You can specify your own signal labels. These are not overwritten by the RTI CAN Blockset.

    Exception: if you change the signal name, the new signal name overwrites the current label.

- **Start bit**: Lets you specify the start bit at which the signal should begin within the message. Only integer values in the range 0 … (8 · n-1) are valid, where n is the number of bytes in the message.
- **Signal length**: Lets you specify the length of the signal in bits. Only integer values in the range 1 … (8 · n–k) ≤ 32 are valid, where n is the number of bytes in the message and k is the start bit.
- **Signal type**: Lets you specify whether the signal should be a standard signal, a mode signal or a mode-dependent signal by selecting the appropriate entry from the list.

  Note

  Only one mode signal is allowed in the same message. To define mode-dependent signals, the message must already contain a mode signal.

- **Mode value**: (Enabled only if **mode-dependent** is selected in the **Signal type** list) Lets you enter the value of the mode signal for which the selected mode-dependent signal must become active. Valid values are in the range 0 … (2^n–1), where n is the length of the mode signal.
- **Data type**: (Disabled if **mode** is selected in the **Signal type** list) Lets you select one of the following data types for the selected signal:
  - unsigned int
  - signed int
  - float 32 (IEEE)

    If you select float 32 (IEEE), the **Signal length** value is set to 32 and the edit field is disabled.

  Note

  If you use non byte-aligned start bits with CANalyzer or CANoe, data conversion may not work correctly. RTICAN lets you set the start bit in the range 0 … (8 · n–1) of the 32 bits, but CANalyzer or CANoe need the start bit on the first bit of the word.

- **Byte layout**: Lets you select one of the following byte layouts for the selected signal:
  - little endian
  - big endian
  - motorola backwards
- **Factor**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the value of the factor for converting the Simulink float format to the desired data format. All float values are valid, with the exception of zero.

  The **Factor** value is used as follows:

  "BlockOutput = (BlockInput – Offset) / Factor", where BlockOutput is the message on the CAN bus, and BlockInput the Simulink signal.
- **Offset**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the value of the offset for converting the Simulink float format to the desired data format. All float values are valid.

  The **Offset** value is used as follows:

  "BlockOutput = (BlockInput – Offset) / Factor", where BlockOutput is the message on the CAN bus, and BlockInput the Simulink signal.
- **Physical unit**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the physical unit of the signal (for documentation purposes) as a string. The physical unit has no impact on the CAN protocol or the message.
- **OK**: Lets you activate all the settings made in the **Signal editor**.
- **Cancel**: Lets you ignore the settings made in the **Signal editor**.

Related topics

References

RTICAN Transmit (TX)..... 79

Options Page (RTICAN Transmit (TX))

Purpose

To specify the block inports and outports, and the initial and termination message.

Dialog settings

Message Displays message information from the **Message Page**: the message name, identifier and length.

Inport and Outport settings Lets you enable further block inports and outports.

- **Use signal names on inports**: Lets you create separate inports for each defined signal. The inports are labeled according to the signal name. Deselecting this checkbox collects all signals in one vectorized inport. In this

case, the Inp numbers from the **Signal list** are used to specify the appropriate signal.

Note

Enable this option to support signal-dependent data types at the signal ports. Refer to CAN signal data typing on page 67.

- **Enable TX status outport**: Lets you enable the **TX status** block outport.
  The status information provided by the block outport relates to the previous block execution. Possible output values:

Value	Description
0	Message not sent
1	Message sent

- **Enable TX time outport**: Lets you enable the **TX time block** outport. The block outport delivers the point in time (in seconds) the previous message was sent.
- **Enable TX delta time outport**: Lets you enable the **TX delta time** block outport. The outport delivers the time difference (in seconds) between the sending times of the previous two messages.
- **Enable TX delay time outport**: Lets you enable the **TX delay time** block outport. The block outport returns the time difference (in seconds) between the point in time the block was executed and the point in time the message was sent successfully by the CAN controller.
- **Enable Delay TX command inport** (Not available if **Auto repeat time** is selected on the **Message Page** (refer to Message Page (RTICAN Transmit (TX)) on page 81)): Lets you enable the **Delay TX command** block inport. The block inport value is used to calculate a delay time that must pass before the message is sent. Specifying a delay time allows you to distribute CAN messages evenly in time to avoid message bursts. The inport value must be an integer and is multiplied with the time required for a single CAN message to be sent at the specified baud rate and identifier format.

  Tip

  Example:
  Suppose an STD message (length: 120 bits) and a 500 kBd baud rate. The time needed for sending this message is 240 µs (120 bit/500 kBd). The delay time (integer) must therefore be a multiple of 240 µs.
  The **Delay TX command** inport value therefore is multiplied with 240 µs.

- **Use enable/trigger port**: Lets you specify whether to use the **RTICAN TX** block with a trigger or enable port. If you use a trigger port, you can select the trigger source (rising, falling, either or a function call).
- **Enable TX loop back inport**: Lets you make CAN messages sent by one CAN controller available to the same CAN controller. The message must be received by an additional **RTICAN Receive (RX)** block for this purpose. You can activate the loop back feature by setting the inport value to '1' or higher. A value of '0' or smaller deactivates the loop back feature.

Preconditions for using the TX loop back feature:

- RX service support must be enabled for the relevant message identifier format on the RX Service Page (RTICAN CONTROLLER SETUP) on page 69.
- An RTICAN Receive (RX) block using the same CAN controller, the same message identifier and the same identifier format. RX service support must be enabled for this RX block.

Note

- The TX loop back inport is not part of the CAN protocol. It is available only if the corresponding RX block uses the RX service to acquire CAN data.
- The CAN message is available only if the message was sent successfully, i.e., the acknowledge bit was sent.
- The CAN message is available on the same CAN controller only if the message was received by other CAN controllers in the network. A delay time is therefore inevitable because RTI CAN blockset checks if the message was successfully received by at least one CAN controller. In standard applications, the delay time should not be longer than twice the sample time used on the corresponding RX block on the same CAN controller.

Initial message The initial message is sent once before the first execution step of the real-time application.

- Set message contents to: Lets you specify the message content. Enter the content as a hexadecimal number. The value must match the message length specified on the Message Page.

Note

Initial messages are not considered by the loop back feature. Even if the feature is enabled, no initial message is written to the receive buffer on the same CAN controller. Initial messages are therefore not received by this CAN controller.

Termination message The termination message is sent once when the real-time application is stopped.

- Set message contents to: Lets you specify the message contents. Enter the contents as a hexadecimal number. The value must match the message length specified on the Message Page.

Encoding options Bits that do not belong to a signal are usually set to "0". You can switch the unused signal bits to "1".

- Set unused bits to "1": Lets you set the unused signal bits "1".

Related topics

References

RTICAN Transmit (TX)..... 79

Custom Encoding Page (RTICAN Transmit (TX))

Purpose

To specify custom encoding of the CAN message bytes.

Description

You can view and change the CAN message bytes after the standard encoding sequence is applied. You can access the unchanged CAN data bytes via a C function that you must implement. To define the vectors to be used by the RTICAN message block, you have to specify an optional inport and/or outport width.

For details on custom encoding/decoding, refer to Implementing Custom Message Encoding and Decoding on page 37.

Optional block inport You can specify the width of an optional block inport. The inport lets you control the encoding function via a signal (or signal vector) from the Simulink model. For example, you can connect the inport to a reference signal for CRC checksum comparison.

Optional block outport You can specify the width of an optional block outport. The outport provides access to the data calculated via the encoding custom function. This allows you, for example, to get a calculated CRC checksum directly from the TX block.

Note

- Do not define encoding/decoding functions with the same name. This also applies if the C function code is placed in different C files. Do not use the same functions with different APIs.
- In the user makefile, add the name of the C file to link and compile the encoding/decoding functions.
  RTI automatically generates the user makefile. For details, refer to User Makefile (USR.MK File) (RTI and RTI-MP Implementation Reference 🕮).

Dialog settings

Message Displays the name, identifier, and length of the current message from the Message Page.

Enable customized message encoding Lets you enable custom encoding. If enabled, the generated code contains a call to the function specified in the Custom encoding function name field.

Custom encoding interface function Lets you enter the name of the encoding interface function. Use the name in the C file that implements custom encoding.

Enable encoding data inport Lets you specify the inport width in the range 1 … 64.

Enable encoding data outport Lets you specify the output width in the range 1 … 64.

Custom encoding interface Displays the C function specified by the above settings. The parameters displayed depend on the selected options:

- The first parameter is a pointer to the signal raw data field, implemented as an 8-byte vector. The vector length does not depend on the message data length. Unused bytes are set to 0.
- Between the first and the last parameter, the optional inports and outports are displayed as pointers to a vector of the Double data type.
- The last parameter is a pointer to the board-specific CAN message structure.

  For details, refer to:
  - DS2202: ds2202_canMsg (DS2202 RTLib Reference 🕮)
  - DS2210: ds2210_canMsg (DS2210 RTLib Reference 🕮)
  - DS2211: ds2211_canMsg (DS2211 RTLib Reference 🕮)
  - DS4302: ds4302_canMsg (DS4302 RTLib Reference 🕮)
  - MicroAutoBox II: can_tp1_canMsg (MicroAutoBox II RTLib Reference 🕮)
  - MicroLabBox: can_tp1_canMsg (MicroLabBox RTLib Reference 🕮)

Related topics

References

RTICAN Transmit (TX) 79

Message Selection Dialog (RTICAN Transmit (TX))

Access

The dialog opens if Load parameters from data file is selected on the Message Page and the Load message button was clicked.

Purpose

To link the block to a message definition in a data file.

The data file must have been previously specified on the Data File Support Page (RTICAN CONTROLLER SETUP).

Dialog settings

dec Displays the message identifiers in decimal numeric format.

hex Displays the message identifiers in hexadecimal numeric format.

Message list Lists all the messages defined in the specified data file. The identifier format, identifier, message name, message length, signal count and message comments are displayed.

To sort the messages in ascending or descending order, click the headers of the message names or identifiers. You can also select a message and enter numbers or characters. If you enter them at a rate faster than 1.5 seconds, they are concatenated to one word/number. The selection bar switches to the next matching entry. A new word begins if you pause for at least 1.5 seconds.

Select one message and click the OK button. To exit the dialog without selecting a message, click Cancel.

Related topics

References

RTICAN Transmit (TX) 79

RTICAN Receive (RX)

Purpose

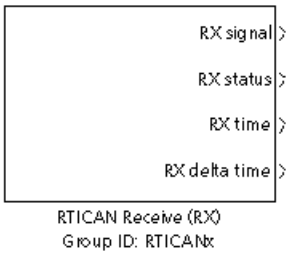
To receive and decode a CAN message with a particular identifier.

Where to go from here

Information in this section

Block Description (RTICAN Receive (RX))

Illustration



Purpose

To receive and decode a CAN message with a particular identifier.

Note

- To use the RTICAN Receive (RX) block, the RTICAN CONTROLLER SETUP block must already be in your model.
- Use one RTICAN Receive (RX) block for each message to be received.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the Controller Selection Dialog may open. See Controller Selection Dialog on page 77 for details.
- Signals in Motorola format are not checked for signal overlapping and signal boundaries within the CAN message. dSPACE therefore recommends using the Motorola format only in conjunction with CANalyzer database files.
- The number of block inports and outports depends on its configuration made on the Options Page.

Description

The RTICAN Receive (RX) block reads one CAN message from the CAN controller per sample step. This message is the message that was received last by the controller.

Note

Further CAN messages received by the CAN controller since the last model sample step are ignored by the RTICAN Receive (RX) block. To avoid this, the RTICAN Receive (RX) block must be executed more frequently.

I/O characteristics

The following table shows the characteristics of the block's input in Simulink. For details on the configuration, refer to Options Page (RTICAN Receive (RX)) on page 99.

Simulink Inport	Description
Enable/Trigger port	Activates or deactivates the execution of the block.

The following table shows the characteristics of the block's output in Simulink. For details on the configuration, refer to Options Page (RTICAN Receive (RX)) on page 99.

Simulink Outport	Description
RX signal	Provides the signal(s) of the RX message.
RX status	Provides status information related to the current execution of the block: ▪ "0": No new data available ▪ "1": New data available
RX time	Provides the point in time (in seconds) when the message was received.

Simulink Outport	Description
RX delta time	Provides the time difference (in seconds) between the points of time the current message and the previous message were received.

Dialog pages

The following pages and dialogs are available:

- **Message Page** (refer to Message Page (RTICAN Receive (RX)) on page 94)
- **Message Composition Page** (refer to Message Composition Page (RTICAN Receive (RX)) on page 96)
- **Options Page** (refer to Options Page (RTICAN Receive (RX)) on page 99)
- **Custom Decoding Page** (refer to Custom Decoding Page (RTICAN Receive (RX)) on page 100)
- **Message Selection Dialog** (refer to Message Selection Dialog (RTICAN Receive (RX)) on page 101) (only if data file support is enabled)

Related topics

References

Message Page (RTICAN Receive (RX))

Purpose

To specify the settings for the CAN controller and the data file support.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- **Board/Module type**: Displays the board/module type on which the CAN support is built.
- **Board/Module number**: Displays the board/module number.
- **Controller number**: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click **Select CAN controller**.
- **Select CAN controller**: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Data file support (Enabled only if data file support is enabled on the **Data File Support Page (RTICAN CONTROLLER SETUP)**) Lets you parameterize the message block by hand or by loading parameters from a data file.

- Parameterize block by hand: Lets you parameterize the block by hand, using the Signal editor on the Message Composition Page. The controls for loading parameters from a data file are disabled.
- Load parameters from data file: Lets you use definitions from a data file. The data file list displays all the data files that were specified on the Data File Support Page (RTICAN CONTROLLER SETUP). If only one data file is specified the list is disabled. During compilation of the model it is checked whether the RX block is unique for each CAN message and controller used within the model. If you have enabled the Multiple message access option this check is disabled.

  Selecting the Load parameters from data file option enables the Load message button. Click this button to open the Message Selection Dialog, which lets you select a message for this RTICAN Receive (RX) block.

RX message parameters Lets you define the basic properties for the message.

- Message name: Can only be edited when Parameterize block by hand is selected. Lets you enter a name for the message. This name is for the user only and has no impact on the CAN protocol.
- Use as block name: The specified message name is used as the block name in your model.

  Note

  Duplicate names in one subsystem are not allowed.

- Message identifier: (Enabled only if Parameterize block by hand is selected) Lets you specify the identifier format, the numeric format for the identifier and the identifier's value. The value is entered in the edit field. Valid ranges are 0 … 2047 for the STD format, and 0 … 536870895 for the XTD format.

  Lets you specify:
  - The identifier format
  - The numeric format of the identifier
  - The identifier value. The value is entered in the edit field. The following table lists the valid ranges:

Identifier Format	Valid Range
STD	0 … 2047[1)]
XTD	0 … 536870895

  [1)] Message identifiers in the range 2032 … 2047 may lead to malfunction if one of the CAN controllers on the connected CAN bus does not support that range. In this case, use only the identifier range 0 … 2031.

- Message length: Only enabled when Parameterize block by hand is selected. Lets you select the number of data bytes the message includes.

  Message length: Enabled only if Parameterize block by hand is selected) Lets you select the number of data bytes the message includes.

- **Block sample time**: Specifies the block sample time. The block sample time defines the time interval for the Simulink block to be executed. The default is –1, which means the inherited sample time. 0.0 is for the base sample time. Other negative values are not allowed.

Comments Lets you enter comments on your message for documentation purposes.

Related topics

References

RTICAN Receive (RX) 92

Message Composition Page (RTICAN Receive (RX))

Purpose

To specify the receive message.

Dialog settings

Message Displays the most important information – name, identifier and length of the message – from the **Message Page**.

Message composition Lets you define how the message is composed of individual signals.

- **List signals for mode**: Lets you select whether to display all the signals in the list or only the signals active for a specific mode. The entries in the list are "all modes" and all mode values already specified for mode-dependent signals.
- **Signal list**: Lists all the signals selected in the **List signals for mode** list and their settings.

  The signal settings are displayed in the following columns:
  - Inp (Input number)
  - Outp (Output number)
  - Signal name
  - SB (Start bit)
  - SL (Signal length)
  - ST (Signal type)
  - MV (Mode value)
  - DT (Data type)
  - BL (Byte layout)
  - Factor
  - Offset
- **New**: Lets you create a new signal. The **Signal editor** is enabled to specify the signal.

- Copy: (Enabled only if a signal is selected in the list) Lets you copy the signal selected in the list. The Signal editor is enabled to let you modify the copied signal.

  The start bit of the copied signal is calculated automatically depending on the last signal of the list. If you specified a signal name ending with a number, the number is automatically increased and the signal is added to the end of the list.

  To copy signals conveniently, click Copy and Apply alternately.
- Delete: (Enabled only if a signal was selected in the list) Lets you delete the selected signal.
- Comments: (Enabled only if a signal was selected in the list) Lets you open a dialog to enter comments on the selected signal for documentation purposes.

  If data file support is enabled, the Comments button is also enabled to let you read the signal comments stored in the data file. In this case, the signal comments are read-only.

Signal editor The signal editor lets you define the signals within a CAN message.

You can specify the following signal properties:

- Signal name: Lets you specify the name of the signal as a string.

  Note

  - All the signal names in the same message must be unique.
  - You can specify your own signal labels. These are not overwritten by the RTI CAN Blockset.

    Exception: if you change the signal name, the new signal name overwrites the current label.

- Start bit: Lets you specify the start bit at which the signal should begin within the message. Only integer values in the range 0 … (8 · n-1) are valid, where n is the number of bytes in the message.
- Signal length: Lets you specify the length of the signal in bits. Only integer values in the range 1 … (8 · n–k) ≤ 32 are valid, where n is the number of bytes in the message and k is the start bit.
- Signal type: Lets you specify whether the signal should be a standard signal, a mode signal or a mode-dependent signal by selecting the appropriate entry from the list.

  Note

  Only one mode signal is allowed in the same message. To define mode-dependent signals, the message must already contain a mode signal.

- Mode value: (Enabled only if mode-dependent is selected in the Signal type list) Lets you enter the value of the mode signal for which the selected mode-dependent signal must become active. Valid values are in the range 0 … (2^n–1), where n is the length of the mode signal.

- **Data type**: (Disabled if **mode** is selected in the **Signal type** list) Lets you select one of the following data types for the selected signal:
  - unsigned int
  - signed int
  - float 32 (IEEE)

    If you select float 32 (IEEE), the **Signal length** value is set to 32 and the edit field is disabled.

Note

If you use non byte-aligned start bits with CANalyzer or CANoe, data conversion may not work correctly. RTICAN lets you set the start bit in the range 0 … (8 · n–1) of the 32 bits, but CANalyzer or CANoe need the start bit on the first bit of the word.

- **Byte layout**: Lets you select one of the following byte layouts for the selected signal:
  - little endian
  - big endian
  - motorola backwards
- **Factor**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the value of the factor for converting the Simulink float format to the desired data format. All float values are valid, with the exception of zero.

  The **Factor** value is used as follows:

  "BlockOutput = BlockInput · Factor + Offset", where BlockInput is the message on the CAN bus, and BlockOutput the Simulink signal.
- **Offset**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the value of the offset for converting the Simulink float format to the desired data format. All float values are valid.

  The **Offset** value is used as follows:

  "BlockOutput = BlockInput · Factor + Offset", where BlockInput is the message on the CAN bus, and BlockOutput the Simulink signal.
- **Physical unit**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the physical unit of the signal (for documentation purposes) as a string. The physical unit has no impact on the CAN protocol or the message.
- **OK**: Lets you activate all the settings made in the **Signal editor**.
- **Cancel**: Lets you ignore the settings made in the **Signal editor**.

Related topics

References

RTICAN Receive (RX) 92

Options Page (RTICAN Receive (RX))

Purpose

To specify the block outports.

Dialog settings

Message Displays the most important information – the name, identifier and length of the message – from the **Message Page**.

Outport settings Lets you enable further block outports.

- **Use signal names on outports**: Lets you create separate block outports for each defined signal. The outports are labeled according to the appropriate signal name. Deselecting this checkbox collects all signals in one vectorized outport. In this case, the Outp numbers from the Signal list are used to specify the appropriate signal.

  Note

  Enable this option to support signal-dependent data types at the signal ports. Refer to CAN signal data typing on page 67.

- **Enable RX status outport**: Lets you enable the RX status outport. The port delivers the status information related to the current execution of the block. There are two possible types of status information: "0" for no new data available, and "1" for new data available.
- **Enable RX time outport**: Lets you enable the RX time outport. The port delivers the point in time (in seconds) when the message was received.
- **Enable RX delta time outport**: Lets you enable the RX delta time outport. The port delivers the time difference (in seconds) between the points of time the current message and the previous message were received.

  Note

  Do not use the RX delta time to determine the time difference if RX service support is enabled.

- **Enable/Trigger port**: Lets you use the **RX block** with a trigger or enable port. If you use a trigger inport, you can select the trigger source (rising, falling, either or a function call).

RX service support Lets you acquire CAN data via a structure on the master processor. This structure is filled with new CAN data by the RTICAN RX service. With RX service support enabled, the number of RX blocks for which RX service support is enabled is unlimited.

To use RX service support with a specific RX block, you must also enable RX service support for the related CAN controller; refer to RX Service Page (RTICAN CONTROLLER SETUP) on page 69.

For basics on RX service support, refer to Using RX Service Support on page 25.

Related topics

References

RTICAN Receive (RX) 92

Custom Decoding Page (RTICAN Receive (RX))

Purpose

To specify custom decoding of the received CAN message bytes.

Description

You can view and change the CAN message bytes before the standard decoding sequence is applied. You can access the unchanged CAN data bytes via a C function that you must implement. To define the vectors to be used by the RTICAN message block, you have to specify an optional inport and/or outport width.

For details on custom encoding/decoding, refer to Implementing Custom Message Encoding and Decoding on page 37.

Optional block inport You can specify the width of an optional block inport. The inport lets you control the decoding function via a signal (or signal vector) from the Simulink model. For example, you can connect the inport to a reference signal for CRC checksum comparison.

Optional block outport You can specify the width of an optional block outport. The outport provides access to the data calculated via the custom decoding function. This allows you, for example, to get a calculated CRC checksum directly from the RX block.

Note

- Do not define encoding/decoding functions with the same name. This also applies if the C function code is placed in different C files. Do not use the same functions with different APIs.
- In the user makefile, add the name of the C file to link and compile the encoding/decoding functions.
  RTI automatically generates the user makefile. For details, refer to User Makefile (USR.MK File) (RTI and RTI-MP Implementation Reference 🕮).

Dialog settings

Enable customized message decoding Lets you enable custom message decoding. If enabled, the generated code contains a call to the function specified in the Custom decoding function name field.

Custom decoding function name Lets you enter the name of the decoding interface function. Use the name in the C file that implements custom decoding.

Enable decoding data inport Lets you specify the inport width in the range 1 … 64.

Enable decoding data outport Lets you specify the output width in the range 1 … 64.

Custom decoding interface Displays the C function specified by the above settings. The parameters displayed depend on the selected options:

- The first parameter is a pointer to the signal raw data field, implemented as an 8-byte vector. The vector length does not depend on the message data length. Unused bytes are set to 0.
- Between the first and the last parameter, the optional inports and outports are displayed as pointers to a vector of the Double data type.
- The last parameter is a pointer to the board-specific CAN message structure.

  For details, refer to:
  - DS2202: ds2202_canMsg (DS2202 RTLib Reference 🕮)
  - DS2210: ds2210_canMsg (DS2210 RTLib Reference 🕮)
  - DS2211: ds2211_canMsg (DS2211 RTLib Reference 🕮)
  - DS4302: ds4302_canMsg (DS4302 RTLib Reference 🕮)
  - MicroAutoBox II: can_tp1_canMsg (MicroAutoBox II RTLib Reference 🕮)
  - MicroLabBox: can_tp1_canMsg (MicroLabBox RTLib Reference 🕮)

Related topics

References

RTICAN Receive (RX) 92

Message Selection Dialog (RTICAN Receive (RX))

Access

The dialog opens if Load parameters from data file is selected on the Message Page and the Load message button was clicked.

Purpose

To link the block to a message definition in a data file.

The data file must have been previously specified on the Data File Support Page (RTICAN CONTROLLER SETUP).

Dialog settings

dec Displays the message identifiers in decimal numeric format.

hex Displays the message identifiers in hexadecimal numeric format.

Message list Lists all the messages defined in the specified data file. The identifier format, identifier, message name, message length, signal count and message comments are displayed.

To sort the messages in ascending or descending order, click the headers of the message names or identifiers. You can also select a message and enter numbers or characters. If you enter them at a rate faster than 1.5 seconds, they are concatenated to one word/number. The selection bar switches to the next matching entry. A new word begins if you pause for at least 1.5 seconds.

Select one message and click the OK button. To exit the dialog without selecting a message, click Cancel.

Related topics

References

RTICAN Receive (RX) 92

RTICAN Request (RQ)

Purpose

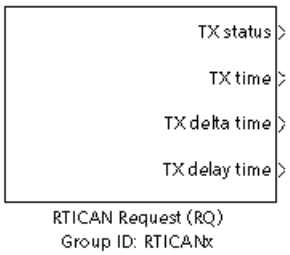
To request data from another bus node with a particular identifier. This message does not include any data, and the RTR (Remote Transmission Request) bit is set.

Where to go from here

Information in this section

Block Description (RTICAN Request (RQ))

Illustration



Purpose

To request data from another bus node with a particular identifier. This message does not include any data, and the RTR (Remote Transmission Request) bit is set.

Note

- The RTICAN CONTROLLER SETUP block must already be in your model if you want to use the Request block.
- Use one RTICAN Request (RQ) block for each request message to be sent.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the Controller Selection Dialog may open. See Controller Selection Dialog on page 77 for details.
- The number of block inports and outports depends on the configuration made on the Options Page.

I/O characteristics

The following table shows the characteristics of the block's input in Simulink. For details on the configuration, refer to Options Page (RTICAN Request (RQ)) on page 107.

Simulink Inport	Description
Enable/Trigger port	Activates or deactivates the execution of the block.

The following table shows the characteristics of the block's output in Simulink. For details on the configuration, refer to Options Page (RTICAN Request (RQ)) on page 107.

Simulink Outport	Description
TX status	Provides status information related to the previous block execution: ▪ "0": Message not sent ▪ "1": Message sent
TX time	Provides the point in time (in seconds) the previous message was sent.
TX delta time	Provides the time difference (in seconds) between the sending times of the previous two messages.
TX delay time	Provides the time difference (in seconds) between the point of time the TX block was executed and the point of time the message was sent successfully by the CAN controller.

Dialog pages

The following pages and dialogs are available:

- Message Page (refer to Message Page (RTICAN Request (RQ)) on page 105)
- Options Page (refer to Options Page (RTICAN Request (RQ)) on page 107)
- Message Selection Dialog (refer to Message Selection Dialog (RTICAN Request (RQ)) on page 108) (only if data file support is enabled)

Related topics

References

Message Page (RTICAN Request (RQ))

Purpose

To specify the settings for the CAN controller and the data file support.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- Board/Module type: Displays the board/module type on which the CAN support is built.
- Board/Module number: Displays the board/module number.
- Controller number: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click Select CAN controller.
- Select CAN controller: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Data file support (Enabled only if data file support is enabled on the Data File Support Page (RTICAN CONTROLLER SETUP)) Lets you parameterize the message block by hand or by loading parameters from a data file.

- Parameterize block by hand: Lets you parameterize the block by hand. The controls for loading parameters from a data file are disabled.
- Load parameters from data file: Lets you use definitions from a data file. The data file list displays all the data files that were specified on the Data File Support Page (RTICAN CONTROLLER SETUP). If only one data file is specified the list is disabled. During compilation of the model it is checked whether the RTICAN Request (RQ) block is unique for each CAN message and controller used within the model. If you have enabled the Multiple message access option this check is disabled.

  Selecting the Load parameters from data file option enables the Load message button. Click this button to open the Message Selection Dialog.

RQ message parameters Lets you define the basic message properties.

- Message name: (Enabled only if Parameterize block by hand is selected) Lets you enter a name for the message. This name is for the user only and has no impact on the CAN protocol.

- **Use as block name**: Lets you specify whether to use the message name as the block name in your model.

  Note

  Duplicate names in the same subsystem are not allowed. Therefore, Simulink automatically adds numbers to the name to create unique names in the subsystem.

- **Message identifier**: (Enabled only if **Parameterize block by hand** is selected) Lets you specify the identifier format, the numeric format for the identifier and the identifier's value. The value is entered in the edit field. Valid ranges are 0 … 2047 for the STD format, and 0 … 536870895 for the XTD format.

  Lets you specify:

  - The identifier format
  - The numeric format of the identifier
  - The identifier value. The value is entered in the edit field. The following table lists the valid ranges:

Identifier Format	Valid Range
STD	0 … 2047[1)]
XTD	0 … 536870895

    [1)] Message identifiers in the range 2032 … 2047 may lead to malfunction if one of the CAN controllers on the connected CAN bus does not support that range. In this case, use only the identifier range 0 … 2031.

- **Block sample time**: Lets you specify the block sample time. The block sample time defines the time interval for the Simulink block to be executed. The default is –1, which means the inherited sample time. 0.0 is for the base sample time. Other negative values are not allowed.
- **Start time**: Lets you specify the start time, which is the time delay before the first message is sent. Specify a start time to avoid CAN bus conflicts, which may occur if all the CAN bus members send their messages simultaneously at the start of the real-time simulation. Valid values are in the range 0 … 420 seconds.

  The start time takes effect only in combination with the **Auto repeat time** feature.
- **Auto repeat time**: Lets you specify a time interval for automatically repeating message transmission, independently of new data. The valid values are in the range 0 … 100 seconds.

  Note

  - Do not specify an **Auto repeat time** if you want to define a message as a *trigger message* in your model.
  - To stop the transmission of a message for which you specified an **Auto repeat time**, you have to stop the running application using dSPACE experiment software.

- **Timeout**: Lets you specify a timeout for messages that cannot be sent successfully. After the specified time, the message is canceled. This is useful to avoid blocking the bus. Valid values are in the range 0 … 100 seconds.

Comments Lets you enter comments on your message for documentation purposes.

Related topics

References

RTICAN Request (RQ)..... 103

Options Page (RTICAN Request (RQ))

Purpose

To specify the outport for the request message.

Dialog settings

Message Displays the most important information – the name and identifier – from the **Message Page**.

Outport settings Lets you enable further block outports.

- **Enable RQ status outport**: Lets you enable the **RQ status** block outport.

  The status information provided by the block outport relates to the previous block execution. Possible output values:

Value	Description
0	Message not sent
1	Message sent

- **Enable RQ time outport**: Lets you enable the **RQ time block** outport. The block outport delivers the point in time (in seconds) the previous message was sent.
- **Enable RQ delta time outport**: Lets you enable the **RQ delta time** block outport. The outport delivers the time difference (in seconds) between the sending times of the previous two messages.
- **Enable RQ delay time outport**: Lets you enable the **RQ delay time** block outport. The block outport returns the time difference (in seconds) between the point in time the block was executed and the point in time the message was sent successfully by the CAN controller.

- **Use enable/trigger port**: Lets you use the **RTICAN RQ** block with a trigger or enable port. If you use a trigger port, you can select the trigger source (rising, falling, either or a function call).

Related topics

References

RTICAN Request (RQ)..... 103

Message Selection Dialog (RTICAN Request (RQ))

Access

The dialog opens if **Load parameters from data file** is selected on the **Message Page** and the **Load message** button was clicked.

Purpose

To link the block to a message definition in a data file.

The data file must have been previously specified on the **Data File Support Page (RTICAN CONTROLLER SETUP)**.

Dialog settings

dec Displays the message identifiers in decimal numeric format.

hex Displays the message identifiers in hexadecimal numeric format.

Message list Lists all the messages defined in the specified data file. The identifier format, identifier, message name, message length, signal count and message comments are displayed.

To sort the messages in ascending or descending order, click the headers of the message names or identifiers. You can also select a message and enter numbers or characters. If you enter them at a rate faster than 1.5 seconds, they are concatenated to one word/number. The selection bar switches to the next matching entry. A new word begins if you pause for at least 1.5 seconds.

Select one message and click the **OK** button. To exit the dialog without selecting a message, click **Cancel**.

Related topics

References

RTICAN Request (RQ)..... 103

RTICAN Remote (RM)

Purpose

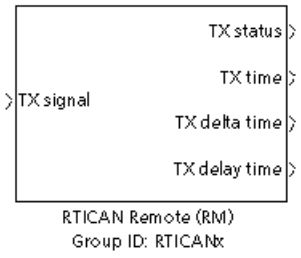
To encode and transmit a CAN message with a particular identifier. The message signals are delivered to the block via the Signal inport. This message is sent only if the CAN controller receives a corresponding request message (same identifier and same identifier format).

Where to go from here

Information in this section

Block Description (RTICAN Remote (RM))

Illustration



Purpose

To encode and transmit a CAN message with a particular identifier. The message signals are delivered to the block via the Signal inport. This message is sent only if the CAN controller receives a corresponding request message (same identifier and same identifier format).

Note

- The RTICAN CONTROLLER SETUP must already be in your model if you want to use the Remote block.
- Use one Remote block for each remote message to be sent. The model must not contain more than 10 RTICAN Remote (RM) blocks.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the Controller Selection Dialog may open. See Controller Selection Dialog on page 77 for details.
- Signals in Motorola format are not checked for signal overlapping and signal boundaries within the CAN message. dSPACE therefore recommends to use the Motorola format only in conjunction with CANalyzer database files.
- The number of block inports and outports depends on the configuration made on the Options Page.

I/O characteristics

The following table shows the characteristics of the block's input in Simulink. For details on the configuration, refer to Options Page (RTICAN Remote (RM)) on page 116.

Simulink Inport	Description
TX signal	Signal(s) of the TX message.
Enable/Trigger port	Activates or deactivates the execution of the block.

The following table shows the characteristics of the block's output in Simulink. For details on the configuration, refer to Options Page (RTICAN Remote (RM)) on page 116.

Simulink Outport	Description
TX status	Provides status information related to the previous block execution: ▪ "0": Message not sent ▪ "1": Message sent
TX time	Provides the point in time (in seconds) the previous message was sent.
TX delta time	Provides the time difference (in seconds) between the sending times of the previous two messages.
TX delay time	Provides the time difference (in seconds) between the point of time the RM message was written to the CAN controller and the point of time the RM message was sent successfully by the CAN controller.

Dialog pages

The following pages and dialogs are available:

- **Message Page** (refer to Message Page (RTICAN Remote (RM)) on page 111)
- **Message Composition Page** (refer to Message Composition Page (RTICAN Remote (RM)) on page 113)
- **Options Page** (refer to Options Page (RTICAN Remote (RM)) on page 116)
- **Custom Encoding Page** (refer to Custom Encoding Page (RTICAN Remote (RM)) on page 117)
- **Message Selection Dialog** (refer to Message Selection Dialog (RTICAN Remote (RM)) on page 119) (only if data file support is enabled)

Related topics

References

Message Page (RTICAN Remote (RM))

Purpose

To specify the settings for the CAN controller and the data file support.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- **Board/Module type**: Displays the board/module type on which the CAN support is built.
- **Board/Module number**: Displays the board/module number.
- **Controller number**: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click **Select CAN controller**.
- **Select CAN controller**: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Data file support (Enabled only if data file support is enabled on the **Data File Support Page (RTICAN CONTROLLER SETUP)**) Lets you parameterize the message block by hand or by loading parameters from a data file.

- **Parameterize block by hand**: Lets you parameterize the block by hand, using the **Signal editor** on the **Message Composition Page**. Loading parameters from a data file is disabled.

- **Load parameters from data file**: Lets you use definitions from a data file. The data file list displays all the data files that were specified on the **Data File Support Page (RTICAN CONTROLLER SETUP)**. If only one data file is specified the list is disabled. During compilation of the model it is checked whether the **RM** block is unique for each CAN message and controller used within the model. If you have enabled the **Multiple message access** option this check is disabled.

  Selecting the **Load parameters from data file** option enables the **Load message** button. Click this button to open the **Message Selection Dialog**, which lets you select a message for this Remote block.

RM message parameters Lets you define the basic message properties.

- **Message name**: (Enabled only if **Parameterize block by hand** is selected) Lets you enter a name for the message. This name is for the user only and has no impact on the CAN protocol.
- **Use as block name**: Lets you specify whether to use the message name as the block name in your model.

Note

Duplicate names in the same subsystem are not allowed. Therefore, Simulink automatically adds numbers to the name to create unique names in the subsystem.

- **Message identifier**: (Enabled only if **Parameterize block by hand** is selected) Lets you specify the identifier format, the numeric format for the identifier and the identifier's value. The value is entered in the edit field. Valid ranges are 0 … 2047 for the STD format, and 0 … 536870895 for the XTD format.

  Lets you specify:
  - The identifier format
  - The numeric format of the identifier
  - The identifier value. The value is entered in the edit field. The following table lists the valid ranges:

Identifier Format	Valid Range
STD	0 … 2047[1)]
XTD	0 … 536870895

  [1)] Message identifiers in the range 2032 … 2047 may lead to malfunction if one of the CAN controllers on the connected CAN bus does not support that range. In this case, use only the identifier range 0 … 2031.

- **Message length**: Enabled only if **Parameterize block by hand** is selected) Lets you select the number of data bytes the message includes.

Note

If you select 0 byte, the message cannot include any data and the **Message Composition Page** is disabled. All the signal inports of the block are removed.

- **Block sample time**: Lets you specify the block sample time. The block sample time defines the time interval for the Simulink block to be executed. The default is –1, which means the inherited sample time. 0.0 is for the base sample time. Other negative values are not allowed.

Comments Lets you enter comments on your message for documentation purposes.

Related topics

References

RTICAN Remote (RM)..... 109

Message Composition Page (RTICAN Remote (RM))

Purpose

To specify the remote message.

Dialog settings

Message Displays the most important information from the **Message Page**: name, identifier and length of the message.

Message composition Lets you define how the message is composed of individual signals.

- **List signals for mode**: Lets you select whether to display all the signals in the list or only the signals active for a specific mode. The entries in the list are "all modes" and all mode values already specified for mode-dependent signals.
- **Signal list**: Lists all the signals selected in the **List signals for mode** list and their settings.

  The signal settings are displayed in the following columns:
  - Inp (Input number)
  - Outp (Output number)
  - Signal name
  - SB (Start bit)
  - SL (Signal length)
  - ST (Signal type)
  - MV (Mode value)
  - DT (Data type)
  - BL (Byte layout)
  - Factor
  - Offset
- **New**: Lets you create a new signal. The **Signal editor** is enabled to specify the signal.

- Copy: (Enabled only if a signal is selected in the list) Lets you copy the signal selected in the list. The Signal editor is enabled to let you modify the copied signal.

  The start bit of the copied signal is calculated automatically depending on the last signal of the list. If you specified a signal name ending with a number, the number is automatically increased and the signal is added to the end of the list.

  To copy signals conveniently, click Copy and Apply alternately.
- Delete: (Enabled only if a signal was selected in the list) Lets you delete the selected signal.
- Comments: (Enabled only if a signal was selected in the list) Lets you open a dialog to enter comments on the selected signal for documentation purposes.

  If data file support is enabled, the Comments button is also enabled to let you read the signal comments stored in the data file. In this case, the signal comments are read-only.

Signal editor The signal editor lets you define the signals within a CAN message.

You can specify the following signal properties:

- Signal name: Lets you specify the name of the signal as a string.

  Note

  - All the signal names in the same message must be unique.
  - You can specify your own signal labels. These are not overwritten by the RTI CAN Blockset.

    Exception: if you change the signal name, the new signal name overwrites the current label.

- Start bit: Lets you specify the start bit at which the signal should begin within the message. Only integer values in the range 0 … (8 · n-1) are valid, where n is the number of bytes in the message.
- Signal length: Lets you specify the length of the signal in bits. Only integer values in the range 1 … (8 · n–k) ≤ 32 are valid, where n is the number of bytes in the message and k is the start bit.
- Signal type: Lets you specify whether the signal should be a standard signal, a mode signal or a mode-dependent signal by selecting the appropriate entry from the list.

  Note

  Only one mode signal is allowed in the same message. To define mode-dependent signals, the message must already contain a mode signal.

- Mode value: (Enabled only if mode-dependent is selected in the Signal type list) Lets you enter the value of the mode signal for which the selected mode-dependent signal must become active. Valid values are in the range 0 … (2^n–1), where n is the length of the mode signal.

- **Data type**: (Disabled if **mode** is selected in the **Signal type** list) Lets you select one of the following data types for the selected signal:
  - unsigned int
  - signed int
  - float 32 (IEEE)

    If you select float 32 (IEEE), the **Signal length** value is set to 32 and the edit field is disabled.

Note

If you use non byte-aligned start bits with CANalyzer or CANoe, data conversion may not work correctly. RTICAN lets you set the start bit in the range 0 … (8 · n–1) of the 32 bits, but CANalyzer or CANoe need the start bit on the first bit of the word.

- **Byte layout**: Lets you select one of the following byte layouts for the selected signal:
  - little endian
  - big endian
  - motorola backwards
- **Factor**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the value of the factor for converting the Simulink float format to the desired data format. All float values are valid, with the exception of zero.

  The **Factor** value is used as follows:

  "BlockOutput = (BlockInput – Offset) / Factor", where BlockOutput is the message on the CAN bus, and BlockInput the Simulink signal.
- **Offset**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the value of the offset for converting the Simulink float format to the desired data format. All float values are valid.

  The **Offset** value is used as follows:

  "BlockOutput = (BlockInput – Offset) / Factor", where BlockOutput is the message on the CAN bus, and BlockInput the Simulink signal.
- **Physical unit**: (Disabled if **mode** is selected in the **Signal type** list) Lets you specify the physical unit of the signal (for documentation purposes) as a string. The physical unit has no impact on the CAN protocol or the message.
- **OK**: Lets you activate all the settings made in the **Signal editor**.
- **Cancel**: Lets you ignore the settings made in the **Signal editor**.

Related topics

References

RTICAN Remote (RM) 109

Options Page (RTICAN Remote (RM))

Purpose

To specify the block inports and outports, and the initial and termination message.

Dialog settings

Message Displays the most important information from the **Message Page**: the name, identifier and length of the message.

Inport and outport settings Lets you enable further block inports and outports.

- **Use signal names on inports**: Lets you create separate inports for each defined signal. The inports are labeled according to the signal name. Deselecting this checkbox collects all signals in one vectorized inport. In this case, the Inp numbers from the **Signal list** are used to specify the appropriate signal.

  Note

  Enable this option to support signal-dependent data types at the signal ports. Refer to CAN signal data typing on page 67.

- **Enable RM status outport**: Lets you enable the **RM status** block outport.

  The status information provided by the block outport relates to the previous block execution. Possible output values:

Value	Description
0	Message not sent
1	Message sent

- **Enable RM time outport**: Lets you enable the **RM time block** outport. The block outport delivers the point in time (in seconds) the previous message was sent.
- **Enable RM delta time outport**: Lets you enable the **RM delta time** block outport. The outport delivers the time difference (in seconds) between the sending times of the previous two messages.
- **Enable RM delay time outport**: Lets you enable the **RM delay time** block outport. The block outport returns the time difference (in seconds) between the point in time the block was executed and the point in time the message was sent successfully by the CAN controller.
- **Use enable/trigger port**: Lets you use the **RTICAN Remote (RM)** block with a trigger or enable port. If you use a trigger inport, you can select the trigger source (rising, falling, either or a function call).

Initial message The initial message is sent once before the first execution step of the real-time application.

- **Set message contents to**: Lets you specify the message content. Enter the content as a hexadecimal number. The value must match the message length specified on the **Message Page**.

Termination message The termination message is sent once when the real-time application is stopped.

Note

If the Termination message checkbox is selected, the RM message is set to the specified contents when the simulation of the real-time application is stopped. Other CAN nodes can always retrieve the termination message by means of request messages, even if the simulation is stopped.

- Set message contents to: Lets you specify message contents. Enter the contents as a hexadecimal number. The value must match the message length specified on the Message Page.

Encoding options Bits that do not belong to a signal are usually set to "0". You can switch the unused signal bits to "1".

- Set unused bits to "1": Lets you set the unused signal bits "1".

Related topics

References

RTICAN Remote (RM) 109

Custom Encoding Page (RTICAN Remote (RM))

Purpose

To specify custom encoding of the CAN message bytes.

Description

You can view and change the CAN message bytes after the standard encoding sequence is applied. You can access the unchanged CAN data bytes via a C function that you must implement. To define the vectors to be used by the RTICAN message block, you have to specify an optional inport and/or outport width.

For details on custom encoding/decoding, refer to Implementing Custom Message Encoding and Decoding on page 37.

Optional block inport You can specify the width of an optional block inport. The inport lets you control the encoding function via a signal (or signal vector) from the Simulink model. For example, you can connect the inport to a reference signal for CRC checksum comparison.

Optional block outport You can specify the width of an optional block outport. The outport provides access to the data calculated via the encoding custom function. This allows you, for example, to get a calculated CRC checksum directly from the TX block.

Note

- Do not define encoding/decoding functions with the same name. This also applies if the C function code is placed in different C files. Do not use the same functions with different APIs.
- In the user makefile, add the name of the C file to link and compile the encoding/decoding functions.
  RTI automatically generates the user makefile. For details, refer to User Makefile (USR.MK File) (RTI and RTI-MP Implementation Reference 🕮).

Dialog settings

Message Displays the name, identifier and length of the current message from the Message Page.

Enable customized message encoding Lets you enable custom encoding. If enabled, the generated code contains a call to the function specified in the Custom encoding function name field.

Custom encoding interface function Lets you enter the name of the encoding interface function. Use the name in the C file that implements custom encoding.

Enable encoding data inport Lets you specify the inport width in the range 1 … 64.

Enable encoding data outport Lets you specify the output width in the range 1 … 64.

Custom encoding interface Displays the C function specified by the above settings. The parameters displayed depend on the selected options:

- The first parameter is a pointer to the signal raw data field, implemented as an 8-byte vector. The vector length does not depend on the message data length. Unused bytes are set to 0.
- Between the first and the last parameter, the optional inports and outports are displayed as pointers to a vector of the Double data type.
- The last parameter is a pointer to the board-specific CAN message structure.
  For details, refer to:
  - DS2202: ds2202_canMsg (DS2202 RTLib Reference 🕮)
  - DS2210: ds2210_canMsg (DS2210 RTLib Reference 🕮)
  - DS2211: ds2211_canMsg (DS2211 RTLib Reference 🕮)
  - DS4302: ds4302_canMsg (DS4302 RTLib Reference 🕮)
  - MicroAutoBox II: can_tp1_canMsg (MicroAutoBox II RTLib Reference 🕮)
  - MicroLabBox: can_tp1_canMsg (MicroLabBox RTLib Reference 🕮)

Related topics

References

RTICAN Remote (RM) 109

Message Selection Dialog (RTICAN Remote (RM))

Access

The dialog opens if **Load parameters from data file** is selected on the **Message Page** and the **Load message** button was clicked.

Purpose

To link the block to a message definition in a data file.

The data file must have been previously specified on the **Data File Support Page** (**RTICAN CONTROLLER SETUP**).

Dialog settings

dec Displays the message identifiers in decimal numeric format.

hex Displays the message identifiers in hexadecimal numeric format.

Message list Lists all the messages defined in the specified data file. The identifier format, identifier, message name, message length, signal count and message comments are displayed.

To sort the messages in ascending or descending order, click the headers of the message names or identifiers. You can also select a message and enter numbers or characters. If you enter them at a rate faster than 1.5 seconds, they are concatenated to one word/number. The selection bar switches to the next matching entry. A new word begins if you pause for at least 1.5 seconds.

Select one message and click the **OK** button. To exit the dialog without selecting a message, click **Cancel**.

Related topics

References

RTICAN Remote (RM)..... 109

RTICAN Interrupt

Purpose

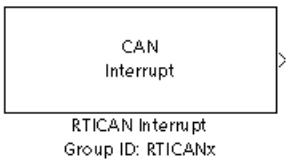
To define for which events the CAN controller sends an interrupt to the master processor on the dSPACE real-time hardware. These interrupts are then available as trigger sources for Simulink function-call subsystems.

Where to go from here

Information in this section

Block Description (RTICAN Interrupt)

Illustration



Purpose

To define for which events the CAN controller sends an interrupt to the master processor on the dSPACE real-time hardware. These interrupts are then available as trigger sources for Simulink function-call subsystems.

Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- Several instances of this block are allowed in the model, with the following restrictions: only two block instances (one for STD, the other for XTD) are allowed for one message identifier and one interrupt type. Only one block is allowed to have the Bus Off interrupt selected.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the Controller Selection Dialog may open. See Controller Selection Dialog on page 77 for details.

Dialog pages

The dialog settings are available on the **Unit Page** (refer to Unit Page (RTI CAN Interrupt) on page 121).

Related topics

References

Unit Page (RTI CAN Interrupt)

Purpose

To specify the CAN interrupts.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- **Board/Module type**: Displays the board/module type on which the CAN support is built.
- **Board/Module number**: Displays the board/module number.
- **Controller number**: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click **Select CAN controller**.
- **Group ID**: Shows the Group ID of the **RTICAN CONTROLLER SETUP** block. The Group ID is used to distinguish several **RTICAN CONTROLLER SETUP** blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own **RTICAN CONTROLLER SETUP** block.
- **Select CAN controller**: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Interrupt type Lets you specify the event for which an interrupt is to be sent.

Interrupt Type	Description
RX Interrupt	Lets you specify whether an interrupt is to be sent after a specified RX message is received. The identifier must be specified in the **Message identifier** group. The model must contain a matching RX message block.
TX Interrupt	Lets you specify whether an interrupt is to be sent after a specified TX message was sent successfully. The identifier must be specified in the **Message identifier** group. The model must contain a matching TX message block.

Interrupt Type	Description
RM Interrupt	Lets you specify whether an interrupt is to be sent after a specified remote message was sent successfully. The identifier must be specified in the **Message identifier** group. The model must contain a matching RM message block.
RQ-RX Interrupt	Lets you specify whether an interrupt is to be sent after the requested message was received successfully. The identifier must be specified in the **Message identifier** group. The model must contain a matching RQ message block and a matching RX message block.
RQ-TX Interrupt	Lets you specify whether an interrupt is to be sent after a specified request message was sent successfully. The identifier must be specified in the **Message identifier** group. The model must contain a matching RQ message block.
RX Service Interrupt	Lets you specify whether an interrupt is to be sent after a message is received by the RX service. For this type of interrupt, you do not need to select an identifier in the **Message identifier** group. You have to define only the identifier format. Depending on the specified RX service filter, the identifier is calculated automatically and written to the Interrupt block. If no valid identifier can be calculated, a message box informs you about an erroneous RX service mask configuration.
Bus Off Interrupt	Lets you specify whether an interrupt is to be sent when the CAN controller goes into the bus off state.
Wake-up Interrupt (DS4302 only)	Lets you specify whether an interrupt is to be sent when a wake-up signal is detected by the C252 transceiver. **Note** ▪ The wake-up interrupt is not triggered when a transition from sleep to normal mode is executed manually. ▪ The wake-up interrupt is only triggered for the first wake-up signal. All following wake-up signals do not trigger an interrupt. To enable the wake-up interrupt again, you have to switch the C252 transceiver to normal mode and then back again to sleep mode.
Bus Failure Interrupt (DS4302 only)	Lets you specify whether an interrupt is to be sent when a bus failure is detected by the C252 transceiver. The following errors cause an interrupt: ▪ CAN-H wire interrupted ▪ CAN-L wire interrupted ▪ CAN-H shorted to battery ▪ CAN-L shorted to battery ▪ CAN-H shorted to ground ▪ CAN-L shorted to ground

Interrupt Type	Description
	▪ CAN-L mutually shorted to CAN-H

Message identifier Lets you specify an identifier for the actual message, according to the selected message type for which an interrupt is to be sent.

Specify the identifier format, the numeric format for the identifier and the identifier's value. The value is entered in the edit field. Valid ranges are 0 … 2047 for the STD format, and 0 … 536870895 for the XTD format.

Related topics

References

RTICAN Interrupt 120

RTICAN Status

Purpose

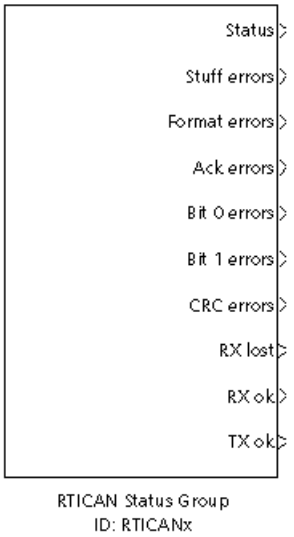
To get status information on the CAN controller and the CAN bus.

Where to go from here

Information in this section

Block Description (RTICAN Status)

Illustration



Purpose

To get status information on the CAN controller and the CAN bus.

Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- Only one Status block per CAN controller is allowed in the model.
- To avoid computation overhead, it is recommended you disable all outputs that are not needed.
- At least one option of the block must be checked.
- Disabled outports are removed from the block.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the **Controller Selection Dialog** may open. See Controller Selection Dialog on page 77 for details.

I/O characteristics

The following table shows the characteristics of the block's output in Simulink:

Simulink Outport	Range	Simulink Data Type	Description
Status	0 ... 2	uint8	Provides status information of the CAN controller's Error Management Logic (EML): ▪ "0" (error active): The CAN controller is active. ▪ "1" (error warn): Before turning to the error passive state, the controller sets an error warn (EWRN) bit. ▪ "2" (bus off): The CAN controller disconnects itself from the bus. To recover, an external action is required (bus off recovery).
Stuff errors	0 … 2^{32}–1	uint32	Provides the number of stuff bit errors. In a CAN bit stream, a *stuff bit* is inserted after 5 consecutive bits of the same value. This stuff bit is complementary to the other 5 bits. If the CAN controller detects a deviation from this pattern, it sets a stuff bit error.
Format errors	0 … 2^{32}–1	uint32	Provides the number of format errors. In a CAN message, certain bits always need to be recessive (= 0). If any of these bits is dominant (= 1), a *format error* is set. There are several fields that contain such bits, such as the ACK delimiter, the CRC delimiter and the EOF field.
Ack(nowledge) errors	0 … 2^{32}–1	uint32	Provides the number of acknowledge errors. Each time a receiver receives a CAN message, the receiver acknowledges this message by sending an answer to the sender. If the sender does not receive this acknowledgement, it will set an *Ack error*.

Simulink Outport	Range	Simulink Data Type	Description
Bit 0 errors	0 … 2^{32}–1	uint32	Provides the number of bit 0 errors. Every CAN sender simultaneously reads back the message it just has sent. This allows the sender to check whether the message was sent correctly. If any of the bits in the sent bit stream was wrong, a *bit 0* or *bit 1 error* is set.
Bit 1 errors	0 … 2^{32}–1	uint32	Provides the number of bit 1 errors. Every CAN sender simultaneously reads back the message it just has sent. This allows the sender to check whether the message was sent correctly. If any of the bits in the sent bit stream was wrong, a *bit 0* or *bit 1 error* is set.
CRC errors	0 … 2^{32}–1	uint32	Provides the number of cyclic redundancy check (CRC) errors. If the CRC code calculated from the received message does not match the CRC field in the message, a *CRC error* is set.
RX lost	0 … 2^{32}–1	uint32	Provides the number of RX messages lost by the CAN controller: If a message cannot be stored in the buffer of the CAN controller, the message is lost and an *RX lost error* is detected.
RX ok	0 … 2^{32}–1	uint32	Provides the number of successfully received RX messages.

Additional output for a fault tolerant receiver/transceiver (DS4302 only):

Simulink Outport	Range	Simulink Data Type	Description
Status of fault tolerant receiver (DS4302 only)	0/1		Provides the following status information: ▪ "0": No error is detected. ▪ "1": A bus error occurs or the transceiver is in a low power mode and a wake-up occurs..
Fault tolerant transceiver errors (DS4302 only)			Provides the number of fault tolerant transceiver errors. The numerical value of the output is increased if one of the following CAN bus events occurs: ▪ CAN-H wire interrupted ▪ CAN-L wire interrupted ▪ CAN-H shorted to battery ▪ CAN-L shorted to battery ▪ CAN-H shorted to ground ▪ CAN-L shorted to ground ▪ CAN-L mutually shorted to CAN-H

Dialog pages

The following pages are available:

- Unit Page (refer to Unit Page (RTICAN Status) on page 127)
- (Relevant for I/O boards with CAN interface in a dSPACE modular system only) Options Page (refer to Options Page (RTICAN Status) on page 128)

Related topics

References

Unit Page (RTICAN Status)

Purpose

To specify the status and error messages to be output.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- Board/Module type: Displays the board/module type on which the CAN support is built.
- Board/Module number: Displays the board/module number.
- Controller number: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click Select CAN controller.
- Group ID: Shows the Group ID of the RTICAN CONTROLLER SETUP block. The Group ID is used to distinguish several RTICAN CONTROLLER SETUP blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own RTICAN CONTROLLER SETUP block.
- Select CAN controller: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Read CAN controller status information Lets you specify whether to use the available status information as outports of the block.

Related topics

References

Options Page (RTICAN Status)

Purpose

To specify the sample time for the RTICAN Status block.

Note

The page is available for I/O boards with CAN interface in a dSPACE modular system only.

Dialog settings

Sample time Lets you enter the sample time for the Status block in seconds. Enter –1 to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Note

If the model´s sample time is < 10 ms, you should set the sample time to the hundredfold value of the model's sample time. This avoids communication overload and loss of data.

Related topics

References

RTICAN Status 124
Unit Page (RTICAN Status) 127

RTICAN Go Bus Off

Purpose

To remove the CAN controller from the bus.

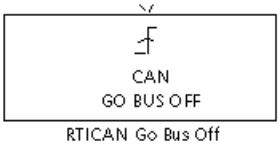
If you use several CAN controllers, the Go Bus Off allows you to switch between them.

Where to go from here

Information in this section

Block Description (RTICAN Go Bus Off)

Illustration



RTICAN Go Bus Off
Group ID: RTICANx

Purpose

To remove the CAN controller from the bus.

Note

- The RTICAN CONTROLLER SETUP block must already be in your model if you want to use the RTI CAN Go Bus Off block.
- Only one RTICAN Go Bus Off block with exactly the same Unit Specification is allowed in a model.
- When a CAN controller is switched to the bus-off state with the RTI CAN Go Bus Off block, the CAN controller status delivered by the RTICAN Status block is not switched to "2". The value of the CAN controller status is not affected by the RTICAN Go Bus Off block.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the Controller Selection Dialog may open. See Controller Selection Dialog on page 77 for details.

I/O characteristics

The following table shows the characteristics of the block's input in Simulink.

Simulink Inport	Description
Trigger port	Activates or deactivates the execution of the block.

Dialog pages

The following pages are available:

- Unit Page (refer to Unit Page (RTICAN Go Bus Off) on page 131)
- (Relevant for I/O boards with CAN interface in a dSPACE modular system only) Options Page (refer to Options Page (RTICAN Go Bus Off) on page 131)

Related topics

References

Unit Page (RTICAN Go Bus Off)

Purpose

To specify the CAN controller.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- Board/Module type: Displays the board/module type on which the CAN support is built.
- Board/Module number: Displays the board/module number.
- Controller number: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click Select CAN controller.
- Group ID: Shows the Group ID of the RTICAN CONTROLLER SETUP block. The Group ID is used to distinguish several RTICAN CONTROLLER SETUP blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own RTICAN CONTROLLER SETUP block.
- Select CAN controller: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Related topics

References

RTICAN Go Bus Off 129

Options Page (RTICAN Go Bus Off)

Purpose

To specify the trigger port.

Note

The page is available for I/O boards with CAN interface in a dSPACE modular system only.

Dialog settings

Trigger port Lets you select whether the trigger port is triggered by the rising and/or falling edge of the trigger signal or by a function call.

Related topics

References

RTICAN Go Bus Off..... 129

RTICAN Bus Off Recovery

Purpose

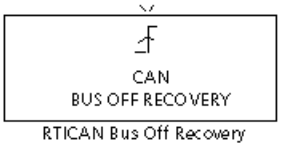
To perform a software reset of the CAN controller when the error status is bus off.

Where to go from here

Information in this section

Block Description (RTICAN Bus Off Recovery)

Illustration



RTICAN Bus Off Recovery
Group ID: RTICANx

Purpose

To perform a software reset of the CAN controller when the error status is bus off.

Note

- The RTICAN CONTROLLER SETUP block must already be in your model if you want to use RTICAN Bus Off Recovery block.
- Only one RTICAN Bus Off Recovery block with exactly the same Unit Specification is allowed in a model.
- If you have more than one RTICAN CONTROLLER SETUP block in your model and you open the block for the first time, the Controller Selection Dialog may open. See Controller Selection Dialog on page 77 for details.
- (Relevant for I/O boards with CAN interface in a dSPACE modular system only) The RTICAN Bus Off Recovery block does not appear in the TRC file.
- (Relevant for I/O boards with CAN interface in a dSPACE modular system only) For safety reasons, the software reset will not be performed if the RTICAN Bus Off Recovery block is triggered while the CAN controller is not in the bus off state.
- When bus off recovery is started, the bus off recovery sequence begins. This sequence resets the counters of the Error Management Logic (EML) of the CAN controller. After counting 128 sequences of 11 consecutive recessive bits on the CAN bus, the controller recovers and goes into the error active state again.

I/O characteristics

The following table shows the characteristics of the block's input in Simulink.

Simulink Inport	Description
Trigger port	Activates or deactivates the execution of the block.

Dialog pages

The following pages are available:

- Unit Page (refer to Unit Page (RTICAN Bus Off Recovery) on page 135)
- (Relevant for I/O boards with CAN interface in a dSPACE modular system only) Options Page (refer to Options Page (RTICAN Bus Off Recovery) on page 135)

Related topics

References

Unit Page (RTICAN Bus Off Recovery)

Purpose

To specify the CAN controller.

Dialog settings

CAN controller specification Displays to which CAN controller the block is currently assigned to.

- Board/Module type: Displays the board/module type on which the CAN support is built.
- Board/Module number: Displays the board/module number.
- Controller number: Displays the number of the CAN controller the block is currently assigned to. To change the assignment, click Select CAN controller.
- Group ID: Shows the Group ID of the RTICAN CONTROLLER SETUP block. The Group ID is used to distinguish several RTICAN CONTROLLER SETUP blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own RTICAN CONTROLLER SETUP block.
- Select CAN controller: The Controller Selection Dialog on page 77 is opened for you to assign the block to another CAN controller.

Related topics

References

RTICAN Bus Off Recovery..... 133

Options Page (RTICAN Bus Off Recovery)

Purpose

To specify the trigger port.

Note

The page is available for I/O boards with CAN interface in a dSPACE modular system only.

Dialog settings

Trigger port Lets you select whether the trigger port is triggered by the rising and/or falling edge of the trigger signal or by a function call.

Related topics

References

RTICAN Bus Off Recovery..... 133

RTI CAN Default Board

Purpose

To specify a default board for all RTICAN blocks.

Note

Do not copy the Default Board block into a model.

Library Page (Default Board)

Purpose

To specify the default board.

Note

Do not copy the Default Board block into a model.

Dialog settings

Specification Lets you specify the default board. The board type of all the blocks copied from the library will adapt themselves to the specified default board.

- **Default board**: Lets you select the board type to be used as the default board.

Related topics

References

RTI CAN Default Board..... 137

RTICAN Gateway (GW)

Purpose

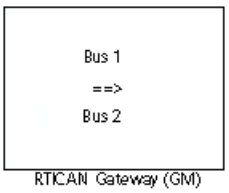
To exchange messages between two CAN buses.

Where to go from here

Information in this section

Block Description (RTICAN Gateway (GW))

Illustration



Purpose

To exchange messages between two CAN buses.

Note

- Both CAN controllers must be within the same model.
- The CAN controller combination is unique for each model and identifier format.
- The **RTICAN Gateway** (GW) block can transfer messages only between two buses on the same processor board.
- For successful access to the data of CAN controller 1, the Group ID of this CAN controller is used. To define the dependencies to the second CAN controller, a new structure field, named GroupID2, is implemented for the **RTICAN Gateway** (GW) block.
- A message can be transmitted to another CAN controller only if the gateway has already received the message. Therefore, no auto repeat time is required. No start time is implemented, because the gateway has to transfer the received messages to the other CAN bus.
- RX service must be enabled for both CAN buses.
  Refer to Using RX Service Support on page 25.
- If the CAN controllers handle both identifier formats (STD and XTD), you must implement two **RTICAN Gateway** (GW) blocks, one for each format.

For details on the communication between two CAN buses, refer to How to Gateway All Messages Using the RTICAN Gateway (GW) Block on page 45.

I/O characteristics

The following table shows the characteristics of the block's input in Simulink. For details on the configuration, refer to Options Page (RTICAN Receive (RX)) on page 99.

Simulink Inport	Description
Enable port	Allows you to switch the **RTICAN Gateway** (GW) block on and off by using a constant value or another Simulink signal.
Trigger port	Allows you to switch the **RTICAN Gateway** (GW) block on and off by using a pulse generator or another Simulink signal.

The following table shows the characteristics of the block's output in Simulink. For details on the configuration, refer to Options Page (RTICAN Receive (RX)) on page 99.

Simulink Outport	Description
RX counter (bus 1)	Provides the number of messages received by the **RTICAN Gateway** (GW) block and sent from CAN bus 1.
TX counter (bus 1)	Provides the number of messages transmitted by the **RTICAN Gateway** (GW) block to CAN bus 1.

Simulink Outport	Description
RX counter (bus 2)	Provides the number of messages received by the RTICAN Gateway (GW) block and sent from CAN bus 2.
TX counter (bus 2)	Provides the number of messages transmitted by the RTICAN Gateway (GW) block to CAN bus 1.

Dialog pages

The following dialog pages are available:

- Message Page (refer to Message Page (RTICAN Gateway (GW)) on page 140)
- Options Page (refer to Options (RTICAN Gateway (GW)) on page 141)
- Exclude Message List (refer to Exclude Message List (RTICAN Gateway (GW)) on page 142)
- Message Selection Dialog (refer to Message Selection Dialog (RTICAN Gateway (GW)) on page 143)

Related topics

Basics

Using RX Service Support....25

HowTos

How to Gateway All Messages Using the RTICAN Gateway (GW) Block....45

Message Page (RTICAN Gateway (GW))

Purpose

To configure the CAN controllers used, specify the data file and the sample time.

Dialog settings

CAN controller specification bus 1 Displays the settings for CAN bus 1 and lets you specify the CAN controller.

- Board type: Displays the board that the selected CAN controller of CAN bus 1 is associated with.
- Board/Module number: Displays the number of the board/module.
- Controller number: Displays the controller number of the selected Controller.
- Select CAN controller: Lets you select the CAN controller for CAN bus 1. Refer to Controller Selection Dialog on page 77.

CAN controller specification bus 2 Displays the settings for CAN bus 2 and lets you specify the CAN controller.

- Board type: Displays the board that the selected CAN controller of CAN bus 2 is associated with.

- **Board/Module number**: Displays the number of the board/module.
- **Controller number**: Displays the controller number of the selected Controller.
- **Select CAN controller**: Lets you select the CAN controller for CAN bus 2. Refer to **Controller Selection Dialog** on page 77.

Selected data file Lets you select the data file you want to use for message configuration for the **RTICAN Gateway** block for CAN bus 1.

Refer to **Data File Support Page** (refer to Data File Support Page (RTICAN CONTROLLER SETUP) on page 64).

Message identifier format (disabled if there is at least one message in the exclusion list) Lets you specify the identifier format (STD or XTD). Since the **RTICAN Gateway** (GW) block can transfer messages in only one identifier format (STD or XTD), you have to add one **RTICAN Gateway** (GW) block to your model for each format.

Block sample time Lets you specify the block sample time. It defines the time interval for the Simulink block to be executed. The default is –1, which means the inherited sample time 0.0 is the base sample time. Other negative values are not allowed.

Related topics

References

RTICAN Gateway (GW)..... 138

Options (RTICAN Gateway (GW))

Purpose

To specify the outports for both buses and the global settings.

Dialog settings

Settings for bus 1 Lets you specify different counters for CAN bus 1.

- **Enable RX counter outport**: Lets you specify an outport that returns the number of messages received by the RTICAN Gateway (GW) block and sent from CAN bus 1.
- **Enable TX counter outport**: Lets you specify an outport that delivers the number of messages transmitted by the RTICAN Gateway (GW) block to CAN bus 1. The checkbox is only enabled when the **Use the Gateway in both directions** option is selected.

Settings for bus 2 Lets you specify different counters for CAN bus 2.

- **Enable TX counter outport**: Lets you specify an outport that delivers the number of messages transmitted by the RTICAN Gateway (GW) block to CAN bus 2.

- **Enable RX counter outport**: Lets you specify an outport that delivers the number of messages received by the RTICAN Gateway (GW) block and sent from CAN bus 2. The checkbox is only enabled when the **Use the Gateway in both directions** option is selected.

Global settings for both sides Lets you specify settings that are valid for both CAN buses.

- **Use Gateway for both directions**: Lets you enable communication between the buses in both directions. Messages sent by bus 1 are transferred to bus 2 and vice versa. The checkbox also activates the settings that are valid for bidirectional bus communication (see above).
- **Use different message exclude lists**: Lets you use two exclusion files, one for each direction. The checkbox is activated when the **Gateway for both directions** option is selected.
- **Use enable port**: Lets you create an inport that allows you to switch the RTICAN Gateway (GW) block on and off by using a constant value or another Simulink signal.
- **Use trigger port**: Lets you create an inport that allows you to switch the RTICAN Gateway (GW) block on and off by using a pulse generator or another Simulink signal. Select the trigger type in the list below.

Related topics

References

RTICAN Gateway (GW)..... 138

Exclude Message List (RTICAN Gateway (GW))

Purpose

To exclude messages from being exchanged between the two buses.

Dialog settings

CAN Bus 1 to Bus 2 Lists all the messages that are not transferred to CAN bus 2. The list can only include messages of the selected database file.

CAN Bus 2 to Bus1 Lists all the messages that are not transferred to CAN bus 1. The CAN bus 2 messages must also be defined by using a database file.

Add message Opens the **Message Selection Dialog (RTICAN Gateway (GW))** that lets you select messages you want to add to the Exclude message list.

Delete message Lets you remove the selected message from the **Exclude message** list. The message is then transferred if the **RTICAN Gateway (GW)** block receives it.

Delete all Removes all messages from the corresponding **Exclude message** list. All the configured messages are then transferred.

Related topics

References

RTICAN Gateway (GW)..... 138

Message Selection Dialog (RTICAN Gateway (GW))

Purpose

To add messages to the Exclude message list.

The Message Selection dialog opens when you click Add message on the Exclude Message List (RTICAN Gateway (GW)) page.

Dialog settings

dec Displays the message identifiers in decimal numeric format.

hex Displays the message identifiers in hexadecimal numeric format.

Message list Displays all the messages configured in the selected database file. Messages already configured are removed from the list. The list shows the identifier, the name, the length, the signal count and comments of each message. Select a message by clicking the corresponding checkbox.

Related topics

References

RTICAN Gateway (GW)..... 138

C252 Support Blocks

Where to go from here

Information in this section

General Information on C252 Support Blocks

Basics on C252 Support Blocks

Introduction

The library extension contains blocks to use the C252 fault-tolerant transceiver.

Note

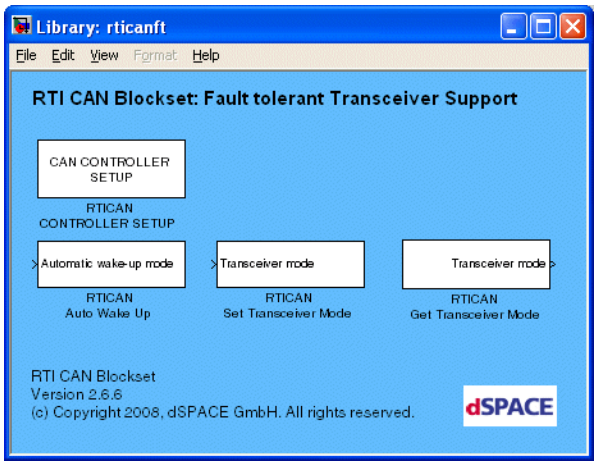
The C252 support blocks are available only for the DS4302.

Library access

The rticanft library provides the RTI blocks supporting the C252 transceiver.

To open the rticanft library:

1. Double-click Library Extensions in the RTICAN library.
2. Double-click rticanft in the RTICAN_Extensions library.



Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- The TJA1054 transceiver is pin and downwards compatible with the C252 transceiver. If the TJA1054 transceiver is on board the DS4302 and you want to use the fault tolerant transceiver functionality, you must select "C252" in the CAN CONTROLLER SETTUP block. Refer to Unit Page (RTICAN CONTROLLER SETUP) on page 59.

Demo models

For Simulink models that show how to use the RTI CAN Blockset, refer to the RTI demo library of the blockset. You will also find the model files in the `%ProgramData%\dSPACE\<InstallationGUID>\Demos\Rtican\` folder.

You can access the `%ProgramData%\dSPACE\<InstallationGUID>` folder via a shortcut in the Windows Start menu below dSPACE RCP and HIL <version>.

Related topics

References

RTICAN CONTROLLER SETUP 57

DS4302CAN_STM_Bx_Cy

Purpose

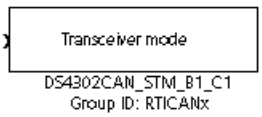
To switch the C252 transceiver mode from sleep to normal mode and vice versa.

Where to go from here

Information in this section

Block Description (DS4302CAN_STM_Bx_Cy)

Illustration



Purpose

To switch the C252 transceiver mode from sleep to normal mode and vice versa.

Note

The C252 transceiver is only available with DS4302. Therefore, the C252 RTI CAN Blockset is only executable when a DS4302 is used.

Description

The transceiver mode is only set if the inport value changes. For example, the transceiver is set to sleep mode by setting the inport value of the DS4302_STM_Bx_Cy block to 2. After a wake-up signal is detected, the transceiver is automatically set to normal mode. Although the inport value of the block is 2, the block does not switch the transceiver automatically to sleep mode. You first have to set the inport value to 1 and then back to 2 to switch into sleep mode.

The inport's value is rounded. Values smaller than 0 are set to 0 and values greater than 1 to 1.

Dialog pages

The following pages and dialogs are available:

- Unit Page (DS4302CAN_STM_Bx_Cy) on page 149
- Options Page (DS4302CAN_STM_Bx_Cy) on page 150

Related topics

References

Unit Page (DS4302CAN_STM_Bx_Cy)

Purpose

To specify the CAN controller.

Dialog settings

CAN controller specification Displays the CAN controller the block is currently connected to. In this group you can change the current settings for the CAN controller number.

- **Board type**: Displays the board type on which the CAN support is built.
- **Board/Module number**: Displays the board number or module number.
- **Controller number**: Displays the number of the CAN controller the block is currently connected to. Lets you change the connection by clicking the **Select CAN controller** button.
- **Group ID**: Shows the Group ID of the RTICAN CONTROLLER SETUP block. The Group ID is used to distinguish several RTICAN CONTROLLER SETUP blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own RTICAN CONTROLLER SETUP block.
- **Select CAN controller**: Lets you connect the block to a different CAN controller to the one it is currently connected to. The Controller Selection dialog appears for this purpose.

Related topics

References

Options Page (DS4302CAN_STM_Bx_Cy)

Purpose

To specify the sample time.

Dialog settings

Sample time Lets you enter the sample time for this block in seconds. Enter –1 to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Related topics

References

DS4302CAN_STM_Bx_Cy..... 148

DS4302CAN_GTM_Bx_Cy

Purpose

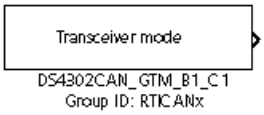
To read the mode of the C252 transceiver.

Where to go from here

Information in this section

Block Description (DS4302CAN_GTM_Bx_Cy)

Illustration



Purpose

To read the mode of the C252 transceiver.

Note

The C252 transceiver is only available with DS4302. Therefore, the C252 RTI CAN blockset is only executable when a DS4302 is used.

Description

The output of the block is a scalar value which is either 1 (normal mode) or 2 (sleep mode).

Dialog pages

The following pages and dialogs are available:

- Unit Page (DS4302CAN_GTM_Bx_Cy) on page 152
- Options Page (DS4302CAN_GTM_Bx_Cy) on page 153

Related topics

References

Options Page (DS4302CAN_GTM_Bx_Cy)..... 153
Unit Page (DS4302CAN_GTM_Bx_Cy)..... 152

Unit Page (DS4302CAN_GTM_Bx_Cy)

Purpose

To specify the CAN controller.

Dialog settings

CAN controller specification Displays the CAN controller the block is currently connected to. Lets you change the current settings for the CAN controller number.

- **Board type**: Displays the board type on which the CAN support is built.
- **Board/Module number**: Displays the board number or module number.
- **Controller number**: Displays the number of the CAN controller the block is currently connected to. Lets you change the connection by clicking the **Select CAN controller** button.
- **Group ID**: Shows the Group ID of the RTICAN CONTROLLER SETUP. The Group ID is used to distinguish several RTICAN CONTROLLER SETUP blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own RTICAN CONTROLLER SETUP block.
- **Select CAN controller**: Lets you connect the block to a different CAN controller to the one it is currently connected to. The Controller Selection Dialog on page 77 is opened for this purpose.

Related topics

References

DS4302CAN_GTM_Bx_Cy..... 151

Options Page (DS4302CAN_GTM_Bx_Cy)

Purpose

To specify the sample time.

Dialog settings

Sample time Lets you enter the sample time for this block in seconds. Enter –1 to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Note

It is not possible to detect whether the mode has been switched and immediately switched back during two sample hits of the read block.

Related topics

References

DS4302CAN_GTM_Bx_Cy 151

DS4302CAN_AWU_Bx_Cy

Purpose

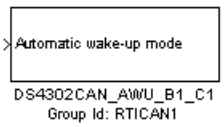
To wake up the C252 transceiver.

Where to go from here

Information in this section

Block Description (DS4302CAN_AWU_Bx_Cy)

Illustration



Purpose

To wake up the C252 transceiver.

Note

The C252 transceiver is only available with DS4302. Therefore, the C252 RTI CAN blockset is only executable when a DS4302 is used.

Description

It is possible to switch the transceiver from sleep to normal mode automatically when a wake-up signal is detected on the CAN bus. The value at the inport of the block controls the wake-up mode:

Inport Value	Meaning
0	No automatic wake up is executed
1	An automatic wake up is executed

The inport's value is rounded. Values smaller than 1 are set to 0 and values greater than 1 to 1.

Dialog pages

The following pages and dialogs are available:

- Unit Page (DS4302CAN_AWU_Bx_Cy) on page 155
- Options Page (DS4302CAN_AWU_Bx_Cy) on page 156

Related topics

References

Unit Page (DS4302CAN_AWU_Bx_Cy)

Purpose

To specify the CAN controller.

Dialog settings

CAN controller specification Displays the CAN controller the block is currently connected to. Lets you change the current settings for the CAN controller number.

- **Board type**: Displays the board type on which the CAN support is built.
- **Board/Module number**: Displays the board number or module number.
- **Controller number**: Displays the number of the CAN controller the block is currently connected to. Lets you change the connection by clicking the **Select CAN controller** button.
- **Group ID**: Shows the Group ID of the RTICAN CONTROLLER SETUP. The Group ID is used to distinguish several RTICAN CONTROLLER SETUP blocks. If you use several I/O boards, for example, in multiprocessor systems, each I/O board must be initialized by its own RTICAN CONTROLLER SETUP block.
- **Select CAN controller**: Lets you connect the block to a different CAN controller to the one it is currently connected to. The Controller Selection Dialog on page 77 is opened for this purpose.

Related topics

References

Options Page (DS4302CAN_AWU_Bx_Cy)

Purpose

To specify the sample time.

Dialog settings

Sample time Lets you enter the sample time for this block in seconds. Enter –1 to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Related topics

References

DS4302CAN_AWU_Bx_Cy..... 154

TJA1041 Support Blocks

Where to go from here

Information in this section

General Information on TJA1041 Support Blocks

Where to go from here

Information in this section

Basics on TJA1041 Support Blocks

Introduction

The library extension contains blocks to use the optional TJA1041 high-speed transceiver.

Note

The TJA1041 support blocks are available only for the DS4302.

dSPACE provides the optional TJA1041 that you can use as a custom transceiver for the DS4302. For a detailed description of the transceiver and the available transceiver modes, refer to the data sheet of the TJA1041 transceiver.

Limitations

There are some limitations when you use the optional TJA1041 transceiver:

- No wake-up interrupt is implemented.
- You have to enable **Automatic Wake Up** in the **DS4302_TJA1041_SST** on page 162 block before you build the model. You cannot enable automatic wake-up during run time.
- If the transceiver is in *power on / listen only* mode, the CAN controller does not send an acknowledge message to the transmitter. The transmitter therefore continues to send the message until it receives the acknowledge signal. This might cause a task overrun if an RX interrupt is configured for the CAN controller.

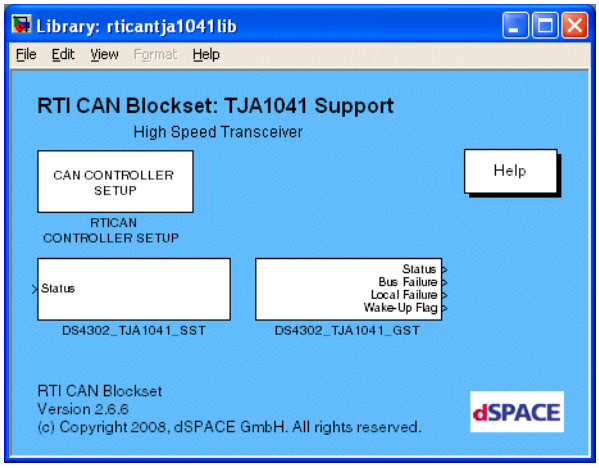
- If the transceiver is in *power on / listen only* mode, it is not able to send CAN messages. Automatic wake-up is not possible if the transceiver is in *power on / listen only* mode. Because no message is sent on the CAN bus by the transceiver in *power on / listen only* mode, CAN arbitration fails. The CAN controller changes to the BUS OFF state. It is not possible to set the BUS state automatically to BUS ON via an interrupt, because the reason for the BUS OFF state still remains. You must set the CAN controller to BUS ON after you have switched the transceiver state to normal, standby, or sleep mode.

Library access

The rticantja1041lib library provides the RTI blocks for the TJA1041 support.

To open the rticantja1041lib library:

1. Double-click Library Extensions in the RTICAN library.
2. Double-click rticantja1041lib in the RTICAN_Extensions library.



Note

One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.

Demo models

For Simulink models that show how to use the RTI CAN Blockset, refer to the RTI demo library of the blockset. You will also find the model files in the `%ProgramData%\dSPACE\<InstallationGUID>\Demos\Rtican\` folder.

You can access the `%ProgramData%\dSPACE\<InstallationGUID>` folder via a shortcut in the Windows Start menu below dSPACE RCP and HIL <version>.

Related topics

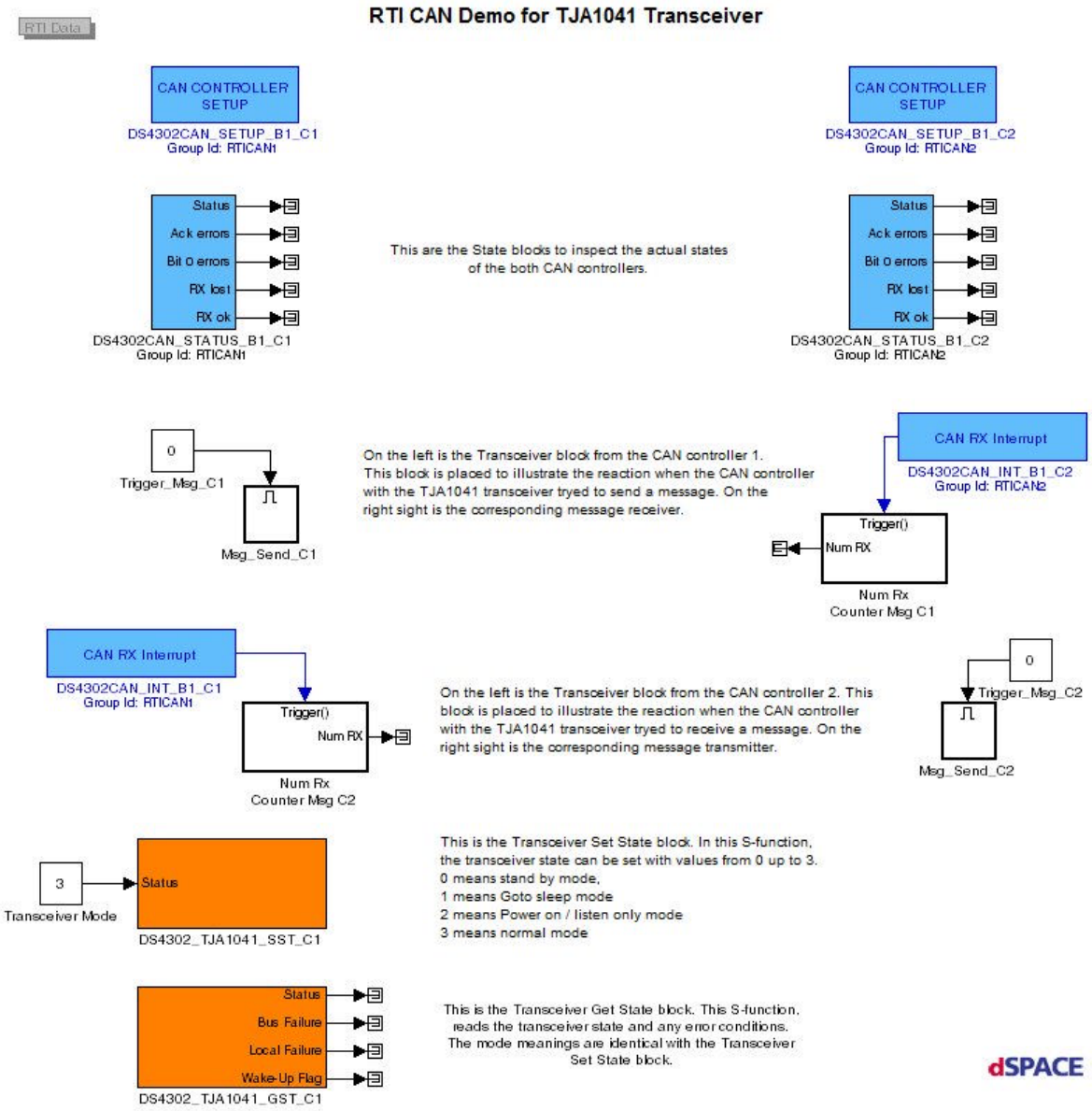
References

DS4302_TJA1041_SST 162
RTICAN CONTROLLER SETUP 57

Example of Using the TJA1041 Transceiver

Use case for the RTI CAN TJA1041 blockset

The demo application illustrates a typical use of the RTI CAN TJA1041 blockset. With the DS4302_TJA1041_SST block, you can set the transceiver mode to be used in accordance with the data sheet of the TJA1041. The new transceiver mode is available in the next sample step.



After building the demo model, you can run it using ControlDesk.

Note

Select the Automatic Wake Up option before you build the model.

Testing the functionality

Standard functionality To check the standard functionality (send and receive messages), enable Enable Send. The error counters of the two CAN controllers count the received messages. To stop the counters, select Disable Send.

Automatic Wake Up from Standby Mode Change the transceiver mode to standby mode by setting the numeric input to "0". The transceiver mode is changed and "0" is displayed in the corresponding display instrument. Select Enable Send to start message transfer. The transceiver mode is automatically switched to Normal Mode and "3" is displayed in the corresponding display instrument. Repeat the steps to check the settings of the second CAN controller.

Automatic Wake Up from Goto Sleep Mode Change the transceiver mode to goto sleep mode by setting the numeric input to "1". The transceiver mode is changed and "1" is displayed in the corresponding display instrument. Select Enable Send to start message transfer. The transceiver mode is automatically switched to normal mode and "3" is displayed in the corresponding display instrument. Repeat the steps to check the settings of the second CAN controller.

Power On / Listen Only functionality Change the transceiver mode to power on / listen only mode by setting the numeric input to "2". The transceiver mode is changed and "1" is displayed in the corresponding display instrument. Select Enable Send to start message transfer. The transceiver mode is not switched to normal mode and still "2" is displayed in the corresponding display instrument.

Error states The different error states (bus failure, local failure and wake-up flag) are only indicated if the Automatic Wake Up is disabled. The transition state is too short to catch any error pin state changes when the Automatic Wake Up is enabled.

Related topics

References

TJA1041 Support Blocks..... 157

DS4302_TJA1041_SST

Purpose

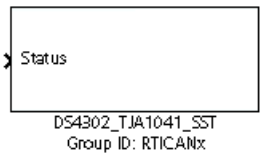
To set the transceiver mode of the optional TJA1041 transceiver.

Where to go from here

Information in this section

Block Description (DS4302_TJA1041_SST)

Illustration



Purpose

To set the transceiver mode of the optional TJA1041 transceiver.

Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- The TJA1041 support blocks are available only for the DS4302.
- Enable **Automatic Wake Up** before you compile the model. The feature cannot be changed during run time.

Description

The status of the transceiver can be set by using the status inport of the DS4302_TJA1041_SST block. For a detailed description of the different modes, refer to the data sheet of the TJA1041 transceiver.

I/O characteristics

The following values can be input, for example, with a constant block:

Simulink Input	Meaning
0	Standby mode; energy saving mode that allows only to monitor the CAN bus.
1	Goto sleep mode; sets the transceiver into sleep mode.
2	Power on / listen only mode; the transmitter is switched off and the bus still receives messages.
3	Normal mode; allows normal bidirectional bus communication.

Note

Values larger than "3" are reset to "3".

Dialog pages

The dialog settings can be specified on the Parameters page (refer to Parameters Page (DS4302_TJA1041_SST) on page 163).

Related topics

References

Parameters Page (DS4302_TJA1041_SST)..... 163

Parameters Page (DS4302_TJA1041_SST)

Purpose

To set the board, channel and sample time for the TJA1041 transceiver.

Dialog settings

Board-No Lets you specify the board number in the range 1 … 16.

Channel-No Lets you specify the channel number in the range 1 … 4.

Automatic Wake Up Lets you enable the *automatic wake-up feature*. Select the check box to switch the transceiver automatically from *standby mode* or *goto sleep mode* to *normal mode* if a message is on the bus or the model wants to send a message.

Note

The *automatic wake-up feature* does not switch *power on / listen only mode* to *normal mode*. The transceiver remains in *power on / listen only mode* until you select another mode that allows switching to *normal mode*.

Sample time Lets you enter the sample time for the block in seconds. Enter –1 to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Related topics

References

DS4302_TJA1041_SST..... 162

DS4302_TJA_1041_GST

Purpose

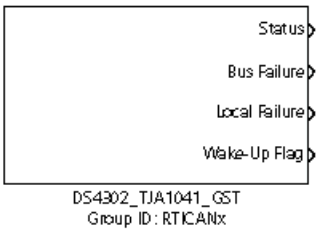
To read failure and status information of the optional TJA1041 transceiver.

Where to go from here

Information in this section

Block Description (DS4302_TJA_1041_GST)

Illustration



Purpose

To read failure and status information of the optional TJA1041 transceiver.

Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- The TJA1041 support blocks are available only for the DS4302.
- For a detailed description of the error pin, refer to the data sheet of the TJA1041.

Description

The DS4302_TJA1041_GST block reads the state of the transceiver ERR pin. Depending on the current transceiver mode the error is output via the Bus Failure, Local Failure or Wake-Up Flag port.

If no error occurs “0” is output.

I/O characteristics

The following values can be output:

Simulink Output	Meaning
Status	Outputs the different modes of the TJA1041 transceiver: ▪ "0" for the standby mode ▪ "1" for the goto sleep mode ▪ "2" for the power on / listen only mode ▪ "3" for the normal mode
Bus Failure	Outputs "1" if the transceiver is in normal mode and an error was detected on the bus.
Local Failure	Outputs "1" if the transceiver is in power on / listen only mode and an error was detected on the bus. Outputs "0" if no error was detected. The TJA1041 provides an additional meaning of this error state. If the transceiver is switched from goto sleep mode or standby mode to power on / listen only mode. This is not supported.
Wake-Up Flag	Outputs "1" if the transceiver is in standby mode or goto sleep mode and a wake up is detected. It is possible that the wake-up flag is set only for one sample step.

Dialog pages

The dialog settings are available on the Parameter page (refer to Parameters Page (DS4302_TJA1041_GST) on page 166).

Related topics

References

Parameters Page (DS4302_TJA1041_GST)..... 166

Parameters Page (DS4302_TJA1041_GST)

Purpose

To set the board, channel and sample time for the TJA1041 transceiver.

Dialog settings

Board-No Lets you specify the board number in the range 1 … 16.

Channel-No Lets you specify the channel number in the range 1 … 4.

Sample time Lets you enter the sample time for the block in seconds. Enter –1 to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Related topics

References

DS4302_TJA_1041_GST 165

ISO11898-6 Support Blocks

Where to go from here

Information in this section

General Information on ISO11898-6 Support Blocks

Basics on ISO11898-6 Support Blocks

Introduction

The library extension contains blocks to use the ISO11898-6 transceiver.

Note

The ISO11898-6 support blocks are available only for:
- MicroAutoBox II
- MicroLabBox

Limitations

There are some limitations when you use the optional ISO11898-6 transceiver:
- No wake-up interrupt is implemented.
- Partial networking is supported only for the following baud rates:
  - 125 kbit/s
  - 250 kbit/s
  - 500 kbit/s
  - 1000 kbit/s

  Other baud rates can be used for normal CAN operation, but detecting wake-up messages for partial networking is supported only for the baud rates listed above.
- You have to enable **Automatic Wake Up** on the **Parameters Page (RTI<xxxx>_ISO11898_6_SST)** before you build the model. You cannot enable automatic wake-up during run time.
- If the transceiver is in *power on / listen only* mode, the CAN controller does not send an acknowledge message to the transmitter. The transmitter therefore continues to send the message until it receives the acknowledge signal. This might result in a task overrun if an RX interrupt is configured for the CAN controller.
- If the transceiver is in *power on / listen only* mode, it is not able to send CAN messages. Automatic wake-up is not possible if the transceiver is in *power on / listen only* mode. Because no message is sent on the CAN bus by the transceiver in *power on / listen only* mode, CAN arbitration fails. The CAN controller changes to the BUS OFF state. It is not possible to set the BUS state automatically to BUS ON via an interrupt, because the reason for the BUS OFF state still remains. You must set the CAN controller to BUS ON after you have switched the transceiver state to normal, standby, or sleep mode.

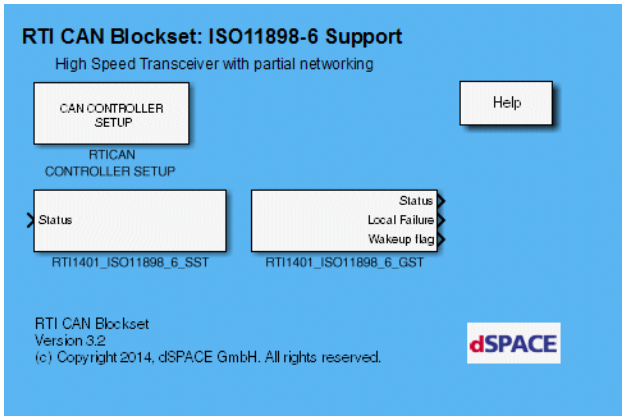
Library access

The rticaniso11898_6lib library (MicroAutoBox II) or rticaniso11898_6_1202lib library (MicroLabBox) provides the RTI blocks for the ISO11898-6 support.

To open the library:

1. Double-click **Library Extensions** in the RTICAN library.
2. Double-click **rticaniso11898_6lib** (MicroAutoBox II) or **rticaniso11898_6_1202lib** (MicroLabBox) in the **RTICAN_Extensions** library.

The illustration below shows the ISO11898-6 support blocks for MicroAutoBox II as an example:



Note

One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.

Demo models

For Simulink models that show how to use the RTI CAN Blockset, refer to the RTI demo library of the blockset. You will also find the model files in the `%ProgramData%\dSPACE\<InstallationGUID>\Demos\Rtican\` folder.

You can access the `%ProgramData%\dSPACE\<InstallationGUID>` folder via a shortcut in the Windows **Start** menu below **dSPACE RCP and HIL <version>**.

Related topics

References

Parameters Page (RTI<xxxx>_ISO11898_6_SST)..... 173
RTICAN CONTROLLER SETUP..... 57

RTI<xxxx>_ISO11898_6_SST

Purpose

To set the transceiver mode of the ISO11898-6 transceiver.

Where to go from here

Information in this section

Block Description (RTI<xxxx>_ISO11898_6_SST)

Illustration

The illustration below shows the block for MicroAutoBox II as an example:

Purpose

To set the transceiver mode of the ISO11898-6 transceiver.

Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- The ISO11898-6 support blocks are available only for:
  - MicroAutoBox II
  - MicroLabBox

Description

The status of the transceiver can be set by using the status inport of the RTI<xxxx>_ISO11898_6_SST block. For a detailed description of the different modes, refer to the data sheet of the transceiver.

I/O characteristics

The table below shows the characteristics of the block's input in Simulink:

Simulink Input	Description
0	Standby mode; energy saving mode that allows to monitor the CAN bus.
1	Sleep mode; sets the transceiver to the sleep mode.
2	Power on / listen only mode; the transmitter is switched off and the bus still receives messages.
3	Normal mode; allows normal bidirectional bus communication.

Note

- Values larger than "3" are reset to "3".
- Values smaller than "0" will be set to "0".

Dialog pages

The dialog settings can be specified on the Parameters page (refer to Parameters Page (RTI<xxxx>_ISO11898_6_SST) on page 173).

Related topics

References

Parameters Page (RTI<xxxx>_ISO11898_6_SST)

Purpose

To set the board, channel and sample time for the ISO11898-6 transceiver.

Dialog settings

Board - No Lets you specify the board number.

MicroAutoBox II: The valid numbers are 1 … 3.

MicroLabBox: The valid number is 1.

Channel - No Lets you specify the channel number in the range 1 … 2.

Automatic Wake-up Lets you enable the *automatic wake-up feature*. Select the check box to switch the transceiver automatically from *standby mode* or *sleep mode* to *normal mode* if a message is on the bus or the model wants to send a message.

Note

The *automatic wake-up feature* does not switch *power on / listen only mode* to *normal mode*. The transceiver remains in *power on / listen only mode* until you select another mode that allows switching to *normal mode*.

Partial networking enable (Relevant for MicroAutoBox II only) Lets you enable the usage of partial networking messages for waking-up MicroAutoBox II after its CAN transceiver is switched to sleep mode and after MicroAutoBox II is powered down via the **DS1401_POWER_DOWN** block of the DS1401 MicroAutoBox Base Board II library.

For further information, refer to DS1401_POWER_DOWN (MicroAutoBox II RTI Reference 🕮).

(Relevant for MicroLabBox only) Lets you enable the usage of partial networking messages for waking-up MicroLabBox's CAN transceiver after the transceiver was switched to sleep mode.

Sample time Lets you enter the sample time for the block in seconds. Enter **-1** to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Related topics

References

Block Description (RTI<xxxx>_ISO11898_6_SST)..... 172

RTI<xxxx>_ISO11898_6_GST

Purpose

To read failure and status information of the ISO11898-6 transceiver.

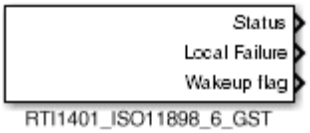
Where to go from here

Information in this section

Block Description (RTI<xxxx>_ISO11898_6_GST)

Illustration

The illustration below shows the block for MicroAutoBox II as an example:



Purpose

To read failure and status information of the ISO11898-6 transceiver.

Note

- One RTICAN CONTROLLER SETUP block must always be present in your model if you want to use any of the other RTI CAN blocks.
- The ISO11898-6 support blocks are available only for:
  - MicroAutoBox II
  - MicroLabBox

Description

The RTIxxxx_ISO11898_6_GST block reads the state of the transceiver ERR pin. Depending on the current transceiver mode, the error is output via the Local Failure or Wake-Up Flag outport.

If no error occurs "0" is output.

I/O characteristics

The table below shows the characteristics of the block's output in Simulink:

Simulink Output	Description
Status	Outputs the current state of the ISO11898-6 transceiver. The state can be one of the following: ▪ "0" for the standby mode ▪ "1" for the sleep mode ▪ "2" for the power on / listen only mode ▪ "3" for the normal mode
Local Failure	Outputs "1" if the transceiver is in power on / listen only mode and an error was detected. Outputs "0" if no error was detected.
Wake-Up Flag	Outputs "1" if the transceiver is in standby mode or goto sleep mode and a wake up is detected. **Note** The wake-up flag might be set for one sample step only.

Dialog pages

The dialog settings can be specified on the Parameters page (refer to Parameters Page (RTI<xxxx>_ISO11898_6_GST) on page 176).

Related topics

References

Parameters Page (RTI<xxxx>_ISO11898_6_GST)..... 176
Partial Networking Page (RTICAN CONTROLLER SETUP)..... 71

Parameters Page (RTI<xxxx>_ISO11898_6_GST)

Purpose

To set the board, channel and sample time for the ISO11898-6 transceiver.

Dialog settings

Board - No Lets you specify the board number.

MicroAutoBox II: The valid numbers are 1 … 3.

MicroLabBox: The valid number is 1.

Channel - No Lets you specify the channel number in the range 1 … 2.

Sample time Lets you enter the sample time for the block in seconds. Enter **–1** to keep the model's base sample time (inherited). The sample time determines how often the block's functions are executed. Use this parameter to minimize the execution time.

Related topics

References

Block Description (RTI<xxxx>_ISO11898_6_GST)..... 175

Appendix

Where to go from here

Information in this section

Basics on CAN

Where to go from here

Information in this section

Introduction to CAN

CAN

The Controller Area Network (CAN) is a field bus system. It is used for applications with widely distributed connection points (nodes) and wiring, such as automation. The system components are located in different places and need to be interconnected.

Communication via CAN

CAN is a serial communications protocol that supports distributed real-time control with a high level of security. In a CAN network, each signal is transmitted and received with minimum delay and without data loss. The range of applications for CAN extends from high-speed networks to low-cost multiplexed wiring. Connected via CAN, each bus member can transmit and receive the data it requires as and when needed.

Baud rate

The baud rate determines how many bits per second are transported via the bus. Different application systems have different rates. However, in all systems, the baud rate has to be uniform and fixed, which means that the same baud rate must be specified for each bus member.

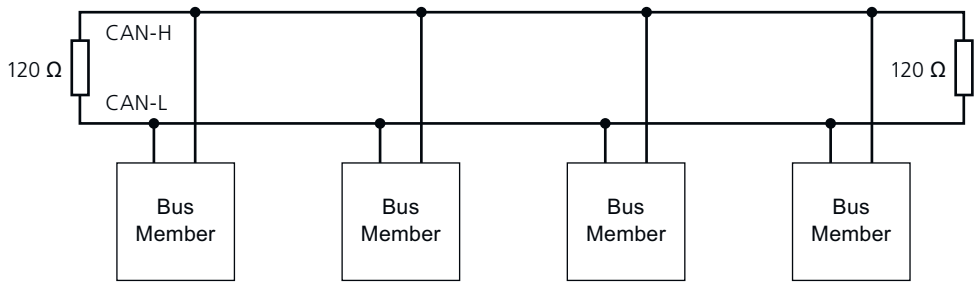
After you specify the baud rate, the dSPACE CAN software determines all the parameters for the bit timing on its own (see Initializing the CAN Controller on page 17).

CAN network

Each CAN network is ruled by two states: dominant and recessive. Bit value 0 represents the dominant state, and bit value 1 the recessive state.

Example

The illustration below shows a simple example of a CAN network. It complies with the ISO 11898 standard. The network consists of two wires (CAN-H, CAN-L). Each bus member is connected directly to the bus. A bus member is also called a CAN bus *node*.



Messages and Signals in a CAN Network

Introduction

Each bus member can transmit and receive data. Data for or from other bus members must be arranged in *messages*.

CAN message characteristics

A message (also called a frame) contains the data to be received and transmitted. It also contains all the information that is relevant to the transmission, such as the start bit, identifier bits, and control bits.

Message identifier The identifier of a message contains address information. It does not address individual bus members, but refers to the information it contains. As a result, a message can be received by any number of bus members. Each bus member can filter out messages of interest. The identifier is located in the arbitration field (see below). You have to specify the identifier value.

Message formats There are two message formats, which have identifiers of different lengths: the standard (STD) format (11-bit identifier), and the extended (XTD) format (29-bit identifier).

The STD format and the XTD format correspond to the CAN 2.0 A specifications and the CAN 2.0 B specifications, respectively. The following table shows the most important characteristics:

	STD	XTD
Specifications	CAN 2.0 A	CAN 2.0 B
Identifier length	11 bits	29 bits
Overhead	44 bits	64 bits
Data field	0 … 64 bits	0 … 64 bits
Max. frame length	108 bits[1)]	128 bits[1)]

[1)] CAN frames can be longer if stuff bits are added to the frame: If the frame contains five successive bits of the same level, a stuff bit is added to the frame for synchronization purposes. The maximum frame length can therefore be 135 bit (STD format) / 158 bit (XTD format). The frame receiver eliminates stuff bits.

The message format must be specified when you set up a message.

Message fields Each message is divided into fields, which are the same for both message formats. The table below lists the fields:

Message Field	Length	Description
Start of frame	1 bit (dominant)	Marks the beginning of the message. All stations synchronize to the leading edge of the start bit of the bus member that starts transmitting first.
Arbitration field	(depends on the Message formats)	Contains the message identifier used to arbitrate and prioritize a message (see CAN Bus Arbitration on page 184). The bits are transmitted starting with the most significant bit. The field ends with the RTR bit (remote transmission request). If the bit is dominant, the message is a transmit (TX) message, otherwise it is a remote (RM) message (see Defining CAN Messages on page 23).
Control field	4 bits (+ 2 reserved bits)	Contains information on the length of the following data field.
Data field	0 … 64 bits	Contains the data to be transmitted.
CRC field	(15 + 1) bits	Contains the cyclic redundancy check (CRC) bits. Each bit before the CRC field is divided by a polynomial. The remainder of this division is the CRC sequence transmitted in the CRC field. Each bus member uses the CRC sequence to check whether the message was transmitted correctly. The CRC field ends with the single recessive CRC delimiter bit.
Acknowledge Check (ACK) field	2 bits	Enables the sender to recognize whether the message was received successfully by at least one other bus member.
End of frame	7 bits (recessive)	Marks the end of the message.

The following illustration shows the field arrangement in a message:

Signals in a CAN message

Several signals can be included in a single CAN message. This is not part of the CAN protocol, but a user convention. The RTI CAN Blockset supports the composition of messages with one or more signals.

Signals must be incorporated into a message's data field, which is up to 8 bytes long. A signal is defined by its start bit in the data field and its length in bits. For example, three signals are incorporated into the data field: pressure (3 bits), temperature (2 bits), and position (4 bits). The pressure signal starts with bit 0 and a length of 3 bits, the temperature follows with start bit 4 and a length of 2 bits, and the position can start at bit 10 with a length of 4 bits.

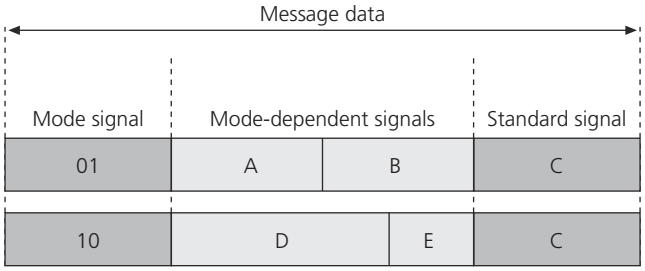
Note

Not all bits of the data field have to be used by the signal or arranged in a specific order. However, the signals must not overlap.

Mode signals and mode-dependent signals

Some CAN applications work with mode signals and mode-dependent signals. These signal types are a user convention and not part of the CAN protocol. The dSPACE CAN software supports mode signals and mode-dependent signals.

A mode signal can switch the coding and meaning of mode-dependent signals in a message. This means that mode-dependent signals can change their start bits and signal lengths, and their encoding/decoding schemes, according to the mode value. A mode signal is included in a message as a standard signal. Using mode signals therefore allows the transmission of several signals with the same message identifier.



Related topics

Basics

CAN Bus Arbitration....184
Defining CAN Messages....23

CAN Bus Arbitration

Introduction

Each bus member needs to transmit messages as and when required. When a serial bus such as CAN is used, this requires specific bus arbitration.

Arbitration principle

The CAN bus uses the CSMA/CR protocol (Carrier Sense Multiple Access with Collision Resolution). Physically, dominant and recessive bits are used: If two bits meet on the bus (collision), the dominant bit is transported and the recessive bit is "switched off."

Each bus member is always the transmitter and receiver at the same time. If a member wants to transmit, it must first determine whether the bus is free. If so, it can start the transmission.

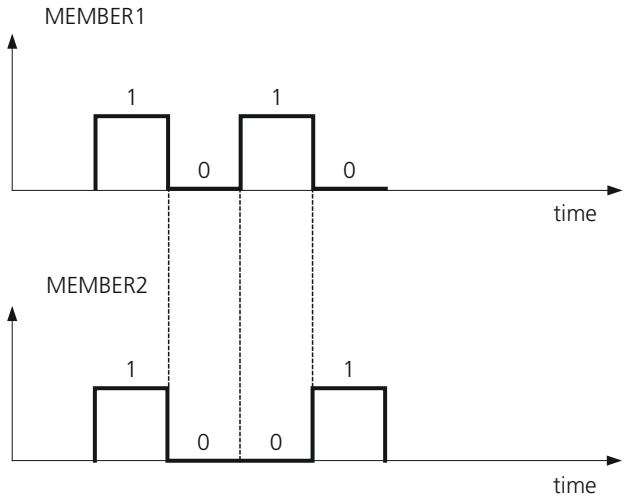
Avoiding message collisions on the bus

Because each message has a protracted run time, it is possible for messages to collide on the bus if bus members start message transmission at the same time. In a CAN network, the message identifier is used to arbitrate the bus: Because each CAN member constantly monitors the bus, the collision is detected when a member transmits a recessive bit but detects a dominant bit on the bus. It therefore stops the transmission and the CAN member with the most leading dominant bits wins the arbitration. The lower the value of an identifier, the higher its priority.

CAN bus arbitration ensures that the message with the highest priority is transmitted with minimum time delay. Important messages are therefore recommended to have high priorities (meaning a small identifier value).

Example

In the following example the message of member 1 has the identifier 1010... The identifier of the member 2 message is 1001...



The first two identifier bits are identical. The third bit of the member 2 message is dominant (bit value 0), but the third bit of the member 1 message is recessive (bit value 1). The recessive bit then turns to 0. When member 1 reads back what it sent, it recognizes that it has "lost" bus allocation and therefore stops transmission.

Related topics

Basics

Messages and Signals in a CAN Network 181

Mailbox Interface

Introduction

A mailbox interface handles the communication between the CAN controller and the CAN bus. It decouples the processes on the bus and on the controller.

Setting up a mailbox

To send or receive a message, the mailbox must first be set up. When dSPACE CAN software is used, this is done implicitly when a message is configured.

Mailboxes for sending messages

Mailboxes 1 … 13 are used to send messages. The messages are allocated to mailboxes dynamically during run time.

Mailboxes for receiving messages

Mailboxes 14 and 15 are used to receive messages. The format of mailbox 15 can be specified to receive messages in STD or XTD format. Setting the mailbox 15 format automatically sets the format of mailbox 14 to the other value.

For mailbox 15, you can use acceptance masks (Message 15 Mask Register) for message filtering to receive only messages corresponding to the specified mask.

Limitations

Limited Number of CAN Messages

Limitation

When you implement CAN communication with RTI CAN Blockset or with RTLib's CAN access functions, the number of CAN messages in an application is limited.

This applies to the following message types:

- Transmit (TX) messages
- Receive (RX) messages
- Request (RQ) messages

  An RQ message and the corresponding RX message are interpreted as a single (RQ) message. You cannot enable RX service support for the corresponding RX message.
- Remote (RM) messages

The sum of these messages is n_{sum}:

$n_{sum} = n_{TX} + n_{RX} + n_{RQ} + n_{RM}$

Maximum number of CAN messages

The sum of the above messages n_{sum} in one application must always be smaller than or equal to the maximum number of CAN messages n_{max}:

$n_{sum} \leq n_{max}$; $n_{RM} \leq 10$

n_{max} in one application depends on:

- Whether you implement CAN communication with RTI CAN Blockset or with RTLib's CAN access functions.
- Whether you use RX service support.

The maximum number of CAN messages n_{max} is listed in the table below:

Platform	**n_{max} with RTLib**	**n_{max} with RTI CAN Blockset**							
		RX Service Support Disabled				**RX Service Support Enabled**			
		1 [1]	**2** [1]	**3** [1]	**4** [1]	**1** [1]	**2** [1]	**3** [1]	**4** [1]
DS2202 (2 CAN controllers)	100	98	96	–	–	96 [2]	92 [2]	–	–
DS2210 (2 CAN controllers)	100	98	96	–	–	96 [2]	92 [2]	–	–
DS2211 (2 CAN controllers)	100	98	96	–	–	96 [2]	92 [2]	–	–
MicroAutoBox II [3] (2 CAN controllers per CAN_Type1)	100	98	96	–	–	96 [2]	92 [2]	–	–

Platform	n_{max} with RTLib	n_{max} with RTI CAN Blockset							
		RX Service Support Disabled				RX Service Support Enabled			
		1 [1]	2 [1]	3 [1]	4 [1]	1 [1]	2 [1]	3 [1]	4 [1]
MicroLabBox (2 CAN controllers)	100	98	96	–	–	96 [2]	92 [2]	–	–
DS4302 (4 CAN controllers)	200	198	196	194	192	196 [2]	192 [2]	188 [2]	184 [2]

[1] Number of CAN controllers used in the application

[2] It is assumed that RX service support is enabled for all the CAN controllers used, and that both CAN message identifier formats (STD, XTD) are used.

[3] Depending on the variant, the MicroAutoBox II contains up to three CAN_Type1 modules, each with 2 CAN controllers. The values in the list apply to a single CAN_Type1 module.

Ways to implement more CAN messages

There are two ways to implement more CAN messages in an application.

Using RX service support If you use RTI CAN Blockset's RX service support, the number of receive (RX) messages n_{RX} in the equations above applies only to RTICAN Receive (RX) blocks for which RX service support is not enabled.

The number of RTICAN Receive (RX) blocks for which RX service support is enabled is unlimited. Refer to Using RX Service Support on page 25.

Using the RTI CAN MultiMessage Blockset To implement more CAN messages in an application, you can also use the RTI CAN MultiMessage Blockset. Refer to the RTI CAN MultiMessage Blockset Tutorial 🕮.

Maximum number of CAN subinterrupts

The number of available CAN subinterrupts you can implement in an application is limited:

Platform	Available CAN Subinterrupts
DS2202 (2 CAN controllers)	15
DS2210 (2 CAN controllers)	15
DS2211 (2 CAN controllers)	15
MicroAutoBox II [1] (2 CAN controllers per CAN_Type1)	15
MicroLabBox (2 CAN controllers)	15
DS4302 (4 CAN controllers)	31

[1] Depending on the variant, the MicroAutoBox II contains up to 3 CAN_Type1 modules, each with 2 CAN controllers. The values in the list apply to a single CAN_Type1 module.