xPC Target

For Use with Real-Time Workshop ®

Modeling

Simulation

Implementation

I/O Reference Guide

Version 2

The MathWorks

How to Contact The MathWorks:

`www.mathworks.com`	Web
`comp.soft-sys.matlab`	Newsgroup
`support@mathworks.com`	Technical support
`suggest@mathworks.com`	Product enhancement suggestions
`bugs@mathworks.com`	Bug reports
`doc@mathworks.com`	Documentation error reports
`service@mathworks.com`	Order status, license renewals, passcodes
`info@mathworks.com`	Sales, pricing, and general information
508-647-7000	Phone
508-647-7001	Fax
The MathWorks, Inc. 3 Apple Hill Drive Natick, MA 01760-2098	Mail

For contact information about worldwide offices, see the MathWorks Web site.

xPC Target I/O Reference Guide

Printing History:	November 2000	Online only	New for Version 1.1 (Release 12.0)
	June 2001	Online only	Revised for Version 1.2 (Release 12.1)
	September 2001	Online only	Revised for Version 1.3 (Release 12.1+)
	July 2002	Online only	Revised for Version 2 (Release 13)

Contents

1 RS232 I/O Support

2 GPIB I/O Support

3 CAN I/O Support

4 CAN I/O Support for FIFO

5 UDP I/O Support

6 Access IO

7 ADDI-DATA

8 Advantech

9 Apex

10 Burr-Brown

11 ComputerBoards (Measurement Computing)

12 Contec

13 Data Translation

14 Diamond

15 Gespac

16 Humusoft

17 Keithley

18 National Instruments

19 Real Time Devices

20 SBS Technologies

21 Softing

22 United Electronic Industries (UEI)

23 Versalogic

24 Miscellaneous I/O Blocks

1

RS232 I/O Support

xPC Target interfaces the target PC to an RS-232 device using either the COM1 or COM2 port. This chapter includes the following sections:

Introduction to RS-232 Drivers

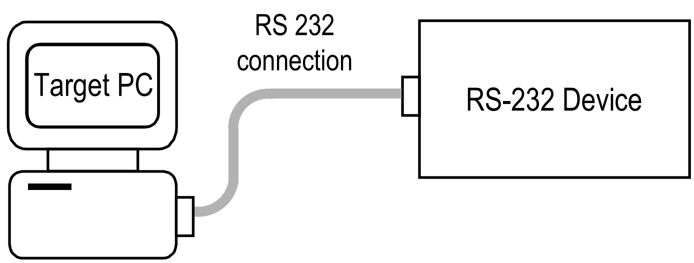
xPC Target uses a model for supporting RS-232 I/O that includes both Simulink blocks for the I/O drivers, and MATLAB structures for sequencing messages and commands. This section includes the following topics:

- **Hardware Connections for RS-232** — Connect the target PC to an RS-232 device.
- **Simulink Blocks for RS-232** — Add setup, send, send/receive, and receive blocks to your Simulink model.
- **MATLAB Message Structures for RS-232** — Create message structures to sequence instructions to and from the RS-232 device.
- **Host and Target PC Communication** — Consider limitations to using RS-232 for I/O on the target PC when using RS-232 communication between the host PC and target PC.

Hardware Connections for RS-232

xPC Target supports serial communication with the COM1 and COM2 ports on the target PC.

Your target applications can use these RS-232 ports as I/O devices. The target PC is connected to an RS-232 device with a null modem cable.



Simulink Blocks for RS-232

To support the use of RS-232, the xPC Target I/O library includes a set of RS-232 driver blocks. These driver blocks can be added to your Simulink model to provide inputs and outputs using one or more of the RS-232 ports:

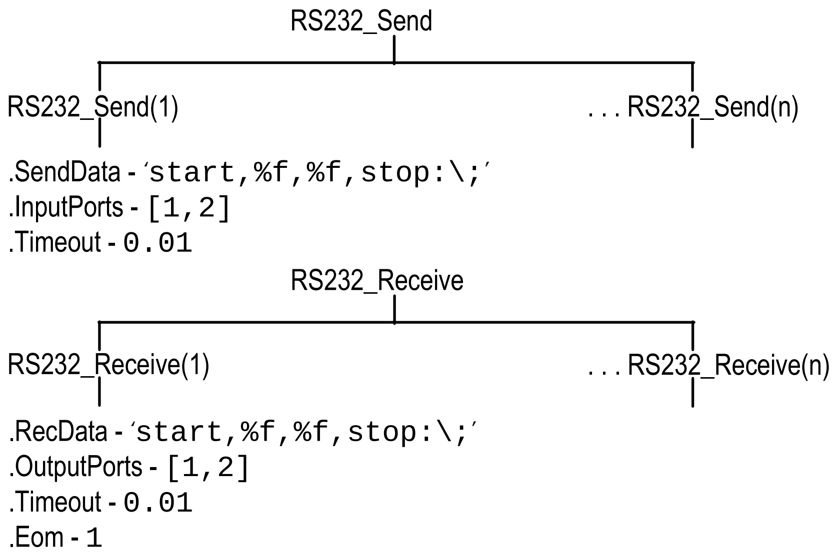
- **RS-232 Setup** — One setup block is needed for each RS-232 port you use in your model. The setup block does not have any inputs or outputs, but sends the initialization and termination messages.
- **RS-232 Send/Receive (Synchronous Mode)** — Send/Receive blocks have inputs and outputs from your Simulink model, and sequence both the send and receive messages.
- **RS-232 Send (Asynchronous Mode)** — Send blocks have inputs from your Simulink model, and sequences the send messages.
- **RS-232 Receive (Asynchronous Mode)** — Receive blocks have output from your Simulink model, and sequences the receive messages.

MATLAB Message Structures for RS-232

Communication is through a series of messages passed back and forth between the target PC and the RS-232 device. To accomplish this, the messages sent to the RS-232 device must be in a format that the device understands. Likewise, the target PC must know how to interpret the data returned from the RS-232 device.

xPC Target uses MATLAB structures to create messages and map the input and output ports on the RS-232 driver blocks to the data written and read from the RS-232 devices. The RS-232 Setup block executes the messages in the initialization structure after downloading the target application. The RS-232 Send/Receive, RS-232 Send, and, RS-232 Receive blocks repeat the execution of the messages in the send/receive, send, and receive structures during each sample interval. When the target application stops running, the RS-232 Setup block executes the messages in the termination structure.

Below is an example of the send and receive message structure for asynchronous communication. In this example, an external RS-232 device requires a string with two floating-point numbers. The numbers are entered from the Simulink model to the first and second input ports of the RS-232 Send driver block. The RS-232 device sends back two floating-point numbers that are passed to the outputs of the RS-232 Receive driver block.



For more information on this example, see "Creating RS-232 Message Structures (Asynchronous)" on page 1-22.

Host and Target PC Communication

If the host PC and target PC are connected using serial communication, one COM port on the target PC is dedicated for communication with the host PC. You cannot use this COM port in your block diagram as an I/O device.

For example, if the target PC uses COM1 for the communication with the host PC, COM1 cannot be used by your block diagram. If you try to use COM1 as an I/O device in your block diagram, an error message is displayed. The error message appears when you attempt to build and download the target application. In this example, it would be necessary for you to use COM2 as an I/O device in your block diagram.

If you are using TCP/IP as your host PC to target PC communications protocol, then you can use any of the COM ports for RS-232 I/O.

Note COM1 and COM3 share interrupt line 4. Similarly, COM2 and COM4 share interrupt line 3. To provide maximum performance, the COM port interrupt line on your target PC used for serial communication is disabled while real-time tasks that include RS-232 blocks are executing. This also means that when COM1 is disabled, COM3 is also disabled since they both share the same interrupt line. For this case, you would have to use either COM2 or COM4 as your RS-232 I/O device.

RS-232 Synchronous Mode

Use synchronous mode when you need to receive a response before continuing with other computations. In synchronous mode, data is sent to an external device and the driver block waits for a response. In other words, the I/O driver *blocks* or stops execution of the target application until an answer is received from the external device or it reaches a time-out. This section includes the following topics:

- **Notes For RS-232 Synchronous Mode** — Overview for RS-232 communication with xPC Target blocks
- **Adding RS-232 Driver Blocks (Synchronous)** — Add the setup, send, and receive blocks you need to your Simulink model for RS-232 communicating.
- **Creating RS-232 Message Structures (Synchronous)** — Create the initialize, send/receive, and termination message structures you need in the MATLAB workspace.

Notes For RS-232 Synchronous Mode

For the example in this section, assume an external device (RS-232 device) includes a D/A conversion module with four independent channels and an output voltage range of -10 to 10 volts. Also assume that the external device outputs a new voltage if it receives a serial string with a value to identify the D/A channel and the voltage value.

Use a Constant block as an input to the Send/Receive block to select the D/A channel, and a Signal Generator block as a source for voltage values. Also, set up the message structures to receive a confirmation message from the external module after the target PC sends a message string to the device.

In the synchronous mode the data is sent to the external device and the block waits until a response (for example, data) is received from the device before the execution of the block is considered to be complete. In other words, the I/O driver will *block* until an answer is received from the external device or it reaches a time-out.

When it is necessary to receive a response before continuing with other computations, the Synchronous Mode is used which implies that the Send & Receive block is placed in your model. This block includes both input and output lines.

Adding RS-232 Driver Blocks (Synchronous)

You add RS-232 driver blocks to your Simulink model when you want to use the serial ports on the target PC for I/O.

After you create a Simulink model, you can add xPC Target driver blocks and define the initialization, send/receive, and termination message structures.

1 In the MATLAB command window, type

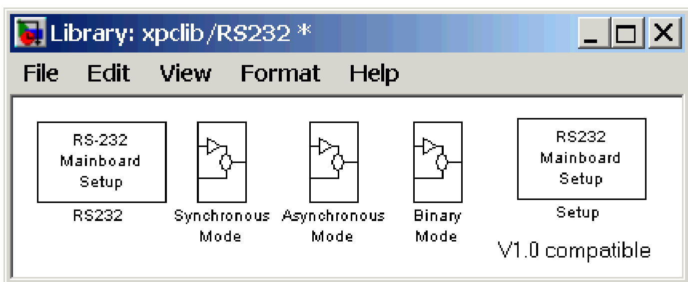
```
xpclib
```

The xPC Target driver block library opens.

2 Double-click the RS-232 group block.

A window with blocks for RS-232 drivers opens.

Note This library contains two setup blocks. The second block is included for compatibility with xPC Target Version 1.0.

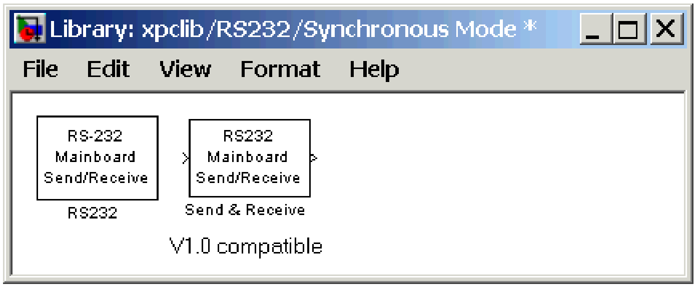


Alternatively, you could access the xPC Target block library from the Simulink Library Browser. In the Simulink window, and from the **View** menu, click **Show Library Browser**. In the left pane, double-click **xPC Target**, and then click **RS-232**.

3 Drag-and-drop an RS-232 Setup block to your Simulink model.

4 In the Library window, double-click the RS-232 Synchronous mode group block. The library window with blocks for RS-232 synchronous communication opens.

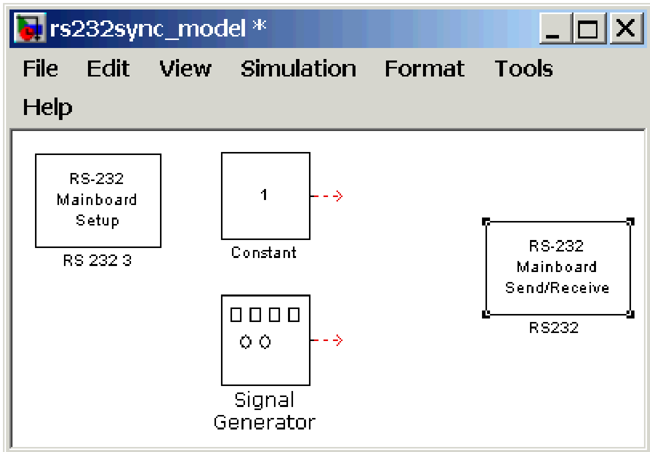
Note This library contains two Setup and Receive blocks. The second block is included for compatibility with xPC Target version 1.0.



5 Drag-and-drop an RS-232 Send/Receive block to your Simulink model.

6 Add a Signal Generator, and a Constant block.

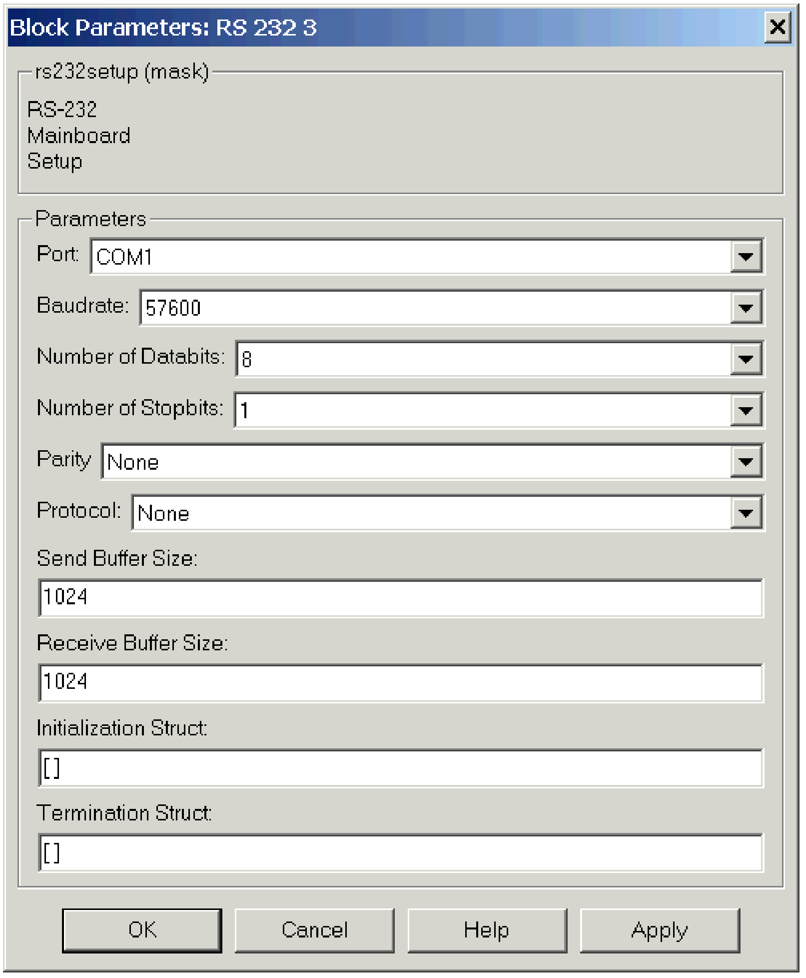
Your model should look similar to the figure shown below. Notice, the inputs on the RS-232 Send/Receive block are not defined or visible. The inputs are defined in a MATLAB message structure, and visible only after you load that structure into the MATLAB workspace and update your Simulink model.



7 Double-click the RS-232 Setup block. Enter values to configure the COM1 port on the target PC.

For example, if the target PC is connected to COM1, and serial communication is set to 5760 baud, 8 databits, and 1 stopbit, your **Block Parameter** dialog box should look similar to the figure shown below.

Note If you are not using an initialization or termination structure, in the **Initialization Struct** and **Termination Struct** boxes, enter the empty matrix `[]`.



For more information on entering the block parameters, see "RS-232 Setup Block" on page 1-27. For the procedure to create the initialization and termination structures, see "RS-232 MATLAB Structure Reference" on page 1-32.

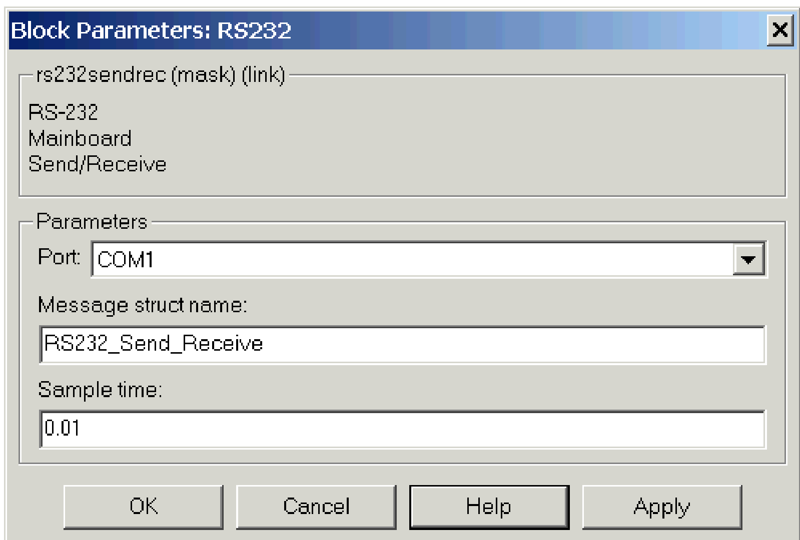
8 Click **OK**. The **Block Parameters** dialog box closes.

9 Double-click the RS-232 Send/Receive block. The **Block Parameters** dialog box opens.

10 From the Port list, select either `COM1` or `COM2`. For this example, select `COM1`. In the **Message Struct Name** box, enter the name for the MATLAB structure this block uses to send messages to the COM1 port. The name of the message structure is not the name of the M-file script, but the name of the structure created with the script.

In the **Sample Time** box, enter the sample time or a multiple of the sample time you entered in the Receive block.

Your **Block Parameter** dialog box should look similar to the figure shown below.



For information on entering the block parameters, see “RS-232 Send/Receive Block (Synchronous)” on page 1-29. For the procedure to create the send/receive structure, see “RS-232 MATLAB Structure Reference” on page 1-32.

11 Click **OK**. The **Block Parameters** dialog box closes.

Your next task is to create the MATLAB message structures that the RS-232 driver blocks use to sequence commands to the RS-232 device. See "Creating RS-232 Message Structures (Synchronous)" on page 1-12.

Creating RS-232 Message Structures (Synchronous)

RS-232 drivers use MATLAB structures to send and receive messages and map the input and output ports on the RS-232 driver blocks to the data written and read from the RS-232 devices.

After you add an RS-232 Setup and RS-232 Send/Receive block to your Simulink model, you can create the message structures to communicate with the RS-232 devices. You need to create and load these structures into the MATLAB workspace before you build your target application. The easiest way to create these structures is using an M-file and load that M-file into the MATLAB workspace.

1 In the MATLAB command window, and from the **File** menu, point to **New**, and then click **M-file**.

   A MATLAB text editor window opens.

2 Enter the initialization, send/receive, and termination messages. Each message is an element in a MATLAB structure array. For information and examples of this structure, see "RS-232 MATLAB Structure Reference" on page 1-32.

   For example, you could have an external RS-232 device with an D/A module that wants a string in the format `'identifier, channel, value;\n'`. Identifier is any string. Channel is an integer value between 1 and 2, defining which D/A channel to update. Value is a floating-point value indicating the new voltage for the D/A output.

   Additionally, when the external device receives a legal string, it accepts the string as an input message, and returns the message `'noerror;\n'`. This message is provided as a confirmation. As an example, you could type the following

   Note Field names in the structures are case sensitive.

   ```
   RS232_Send_Receive(1).SendData = 'da_1234,%d,%f,;\n';
   ```

```
RS232_Send_Receive(1).InputPorts = [1 2];
RS232_Send_Receive(1).RecData = 'noerror\n';
RS232_Send_Receive(1).Timeout = 0.01;
RS232_Send_Receive(1).EOM = 1;
```

3 From the **File** menu, click **Save As**. In the **Save as file** dialog box, enter the name of the M-file script. For example, enter

```
RS232Sync_Messages.m
```

4 Close the text editing window.

5 In the MATLAB command window, type the name of the M-file script you created with the RS-232 structures. For example, type

```
RS232Sync_Messages
```

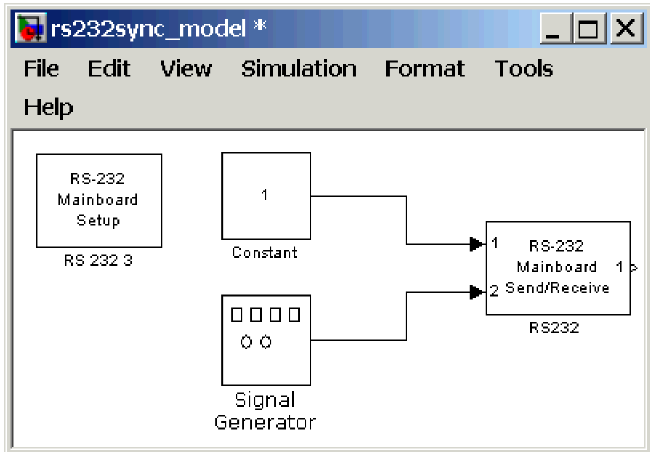
MATLAB loads and runs the M-file script to create the message structures in the MATLAB workspace needed by the RS-232 driver blocks.

6 Open your Simulink model, or press **Ctrl+D**.

Simulink updates the RS-232 driver blocks with the information from the structures. For example, Simulink adds inputs and outputs defined in the structures to the blocks.

7 Connect the input and output ports on the RS-232 driver blocks to other blocks in your Simulink model.

You model should look similar to the figure shown below.



8 Set the `PreLoadFcn` for your Simulink model to load the message structures when you open your model. For example, if you saved the message structures in the M-file `RS232Sync_messages`, type

```
set_param(gcs, 'PreLoadFcn','RS232Sync_messages.m')
```

Note If you do not manually load the message structures before opening your Simulink model, or have the message structures automatically loaded with the model, the port connections to the RS-232 driver break.

Your next task is to build and run the target application. However, the example above only illustrates how to set up the dialog entries when using the Send & Receive block. Without an external RS-232 device to receive the messages, and return a reply `"no error\n"`, this model cannot run successfully on your target PC. It will *block* and wait for a reply each time the application sends a message.

RS-232 Asynchronous Mode

Use asynchronous mode when you do not need a response before continuing with other computations. You can achieve faster sample rates with the asynchronous mode since neither the Send or Receive blocks wait for a reply. As a result, the asynchronous mode blocks do not *block* as do the synchronous mode blocks. The application updates the outputs only when the entire package of data is received from the external device. This section includes the following topics:

- **Notes for RS-232 Asynchronous Mode** — Overview for RS-232 communications with xPC Target blocks
- **Adding RS-232 Driver Blocks (Asynchronous)** — Add the setup, send, and receive blocks you need to your Simulink model for RS-232 communication
- **Creating RS-232 Message Structures (Asynchronous)** — Create the initialize, send/receive, and termination message structures you need in the MATLAB workspace
- **Building and Running the Target Application (Asynchronous)** — Run a real-time application with RS-232 communication

Notes for RS-232 Asynchronous Mode

For the example in this section, two asynchronous mode blocks illustrate how you can test RS-232 I/O on the target PC in a simple loop-back test. This simple but effective test lets you check that the RS-232 Send and RS-232 Receive blocks work correctly with your system using minimal hardware.

In this loop-back test, you use the COM1 port for sending signals and the COM2 port for receiving signals. A null modem serial cable connects COM1 to COM2 so that any messages sent from the target PC through COM1 are received by COM2 on the same target PC.

Use a Sine Wave block as an input to an RS-232 Send block that you connect to the COM1 port. Connect the COM2 port to an RS-232 Receive block. The signal received from this block is then passed through a Gain block of -1.

In the asynchronous mode, data is sent without waiting for response data to be received. The Send block completes execution immediately upon completing the Send transfer. The Receive block completes execution upon completing the Receive transfer or when no more data is ready to be retrieved.

For sending data in asynchronous mode, the RS232 Send block is used. This block only has input lines for the data that will be sent. For receiving data, the Receive block must be used. This block only has output lines for the data that will be received. Outputs are updated only when the entire package of data is received from the external device.

Faster sample rates can be achieved with the asynchronous mode since neither the Send or Receive blocks wait for a reply. As a result, the asynchronous mode blocks do not block as do the synchronous mode blocks.

Adding RS-232 Driver Blocks (Asynchronous)

You add RS-232 driver blocks to your Simulink model when you want to use the serial ports on the target PC for I/O.

After you create a Simulink model, you can add xPC Target driver blocks and define the initialization, send, receive, and termination message structures.

1 In the MATLAB command window, type

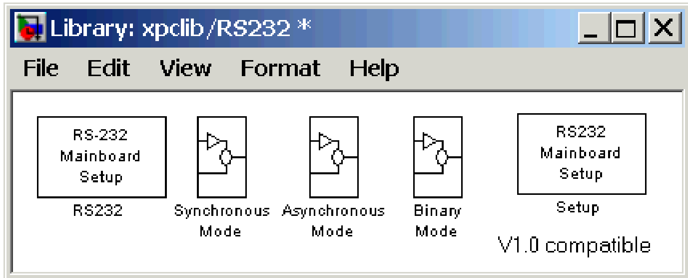
```
xpclib
```

The xPC Target driver block library opens.

2 Double-click the RS-232 group block.

A window with blocks for RS-232 drivers opens.

Note This library contains two setup blocks. The second block is included for compatibility with xPC Target Version 1.0.

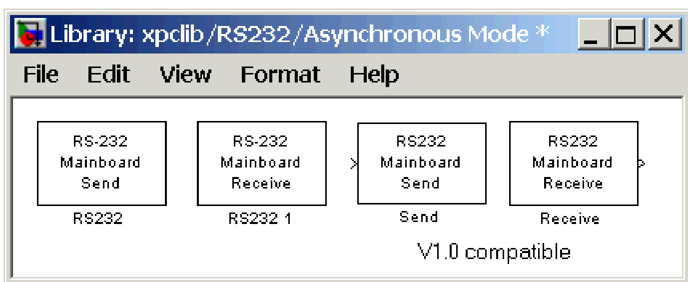


Alternatively, you could access the xPC Target block library from the Simulink Library Browser. In the Simulink window, and from the **View** menu, click **Show Library Browser**. In the left pane, double-click **xPC Target**, and then click **RS-232**.

3 Drag-and-drop a RS-232 Setup block to your Simulink model.

4 In the Library window, double-click the RS-232 Synchronous mode group block. The library window containing blocks for RS-232 Synchronous communication opens.

Note This library contains two send and two receive blocks. The second block is included for compatibility with xPC Target Version 1.0.

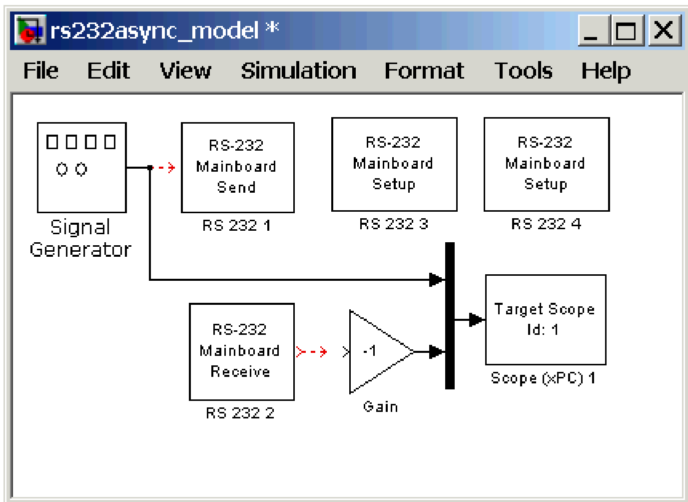


Alternatively, you could access the xPC Target block library from the Simulink Library Browser. In the Simulink window, and from the **View** menu, click **Show Library Browser**. In the left pane, double-click **xPC Target**, and then click **RS-232**.

5 Drag-and-drop the RS-232 Send and RS-232 Receive blocks into your Simulink model.

6 Add a Signal Generator, Gain, and xPC Target Scope block.

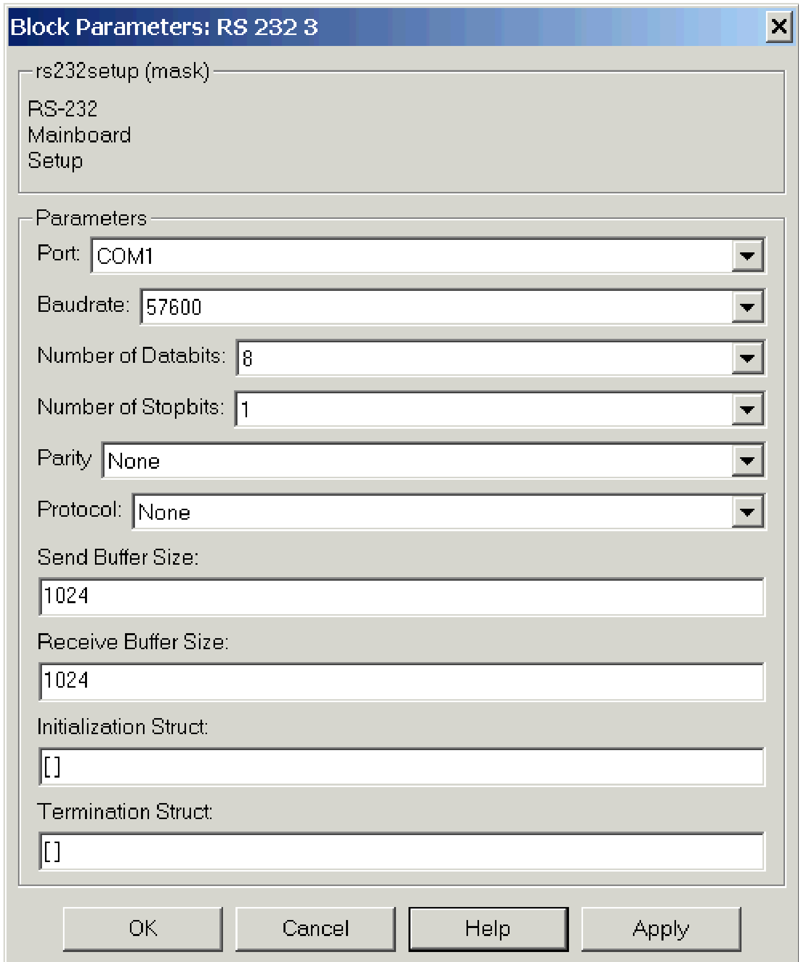
Your model should look similar to the figure below. Notice, you cannot connect to the inputs on the RS-232 Send block and the outputs on the RS-232 Receive block, because they are not defined or visible. The inputs and outputs are defined in a MATLAB massage structure, and visible only after you load that structure into the MATLAB workspace and update your Simulink model.



7 Double-click the first RS-232 Setup block. Enter values to configure the COM1 port on the target PC.

For example, if the COM1 and COM2 ports of the target are connected with a RS-232 null modem cable and setting serial communication to 57600 baud, 8 databits, and 1 stopbit. Your Block Parameter dialog box should look similar to the figure shown below.

Note If you are not using an initialization or termination structure, in the **Initialization Struct** and **Termination Struct** boxes, enter the empty matrix `[]`.



For more information on entering the block parameters, see “RS-232 Setup Block” on page 1-27. For the procedure to create the initialization and termination structures, see “RS-232 MATLAB Structure Reference” on page 1-32.

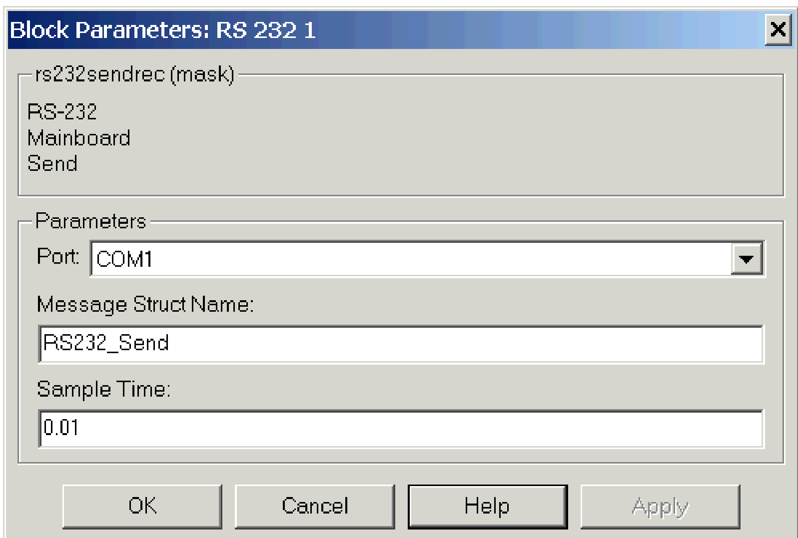
8 Click **OK**. The **Block Parameters** dialog box closes.

9 Repeat the previous setup for the second RS-232 Setup block and the COM2 port. Use the same Baudrate, Databits, Stopbits, Parity, and Protocol that you entered in the first RS-232 Setup block.

10 Double-click the Send block. The **Block Parameters** dialog box opens.

11 From the **Port** list, select either `COM1` or `COM2`. For this example, select `COM1`. In the **Message struct name** box, enter the name for the MATLAB structure this block uses to send messages to the COM1 port. In the **Sample Time** box, enter the sample time or a multiple of the sample time you entered in the RS-232 Receive block.

Your **Block Parameters** dialog box should look similar to the figure shown below.



For information on entering the block parameters, see “RS-232 Send Block (Asynchronous)” on page 1-30. For the procedure to create the send structure, see “RS-232 MATLAB Structure Reference” on page 1-32.

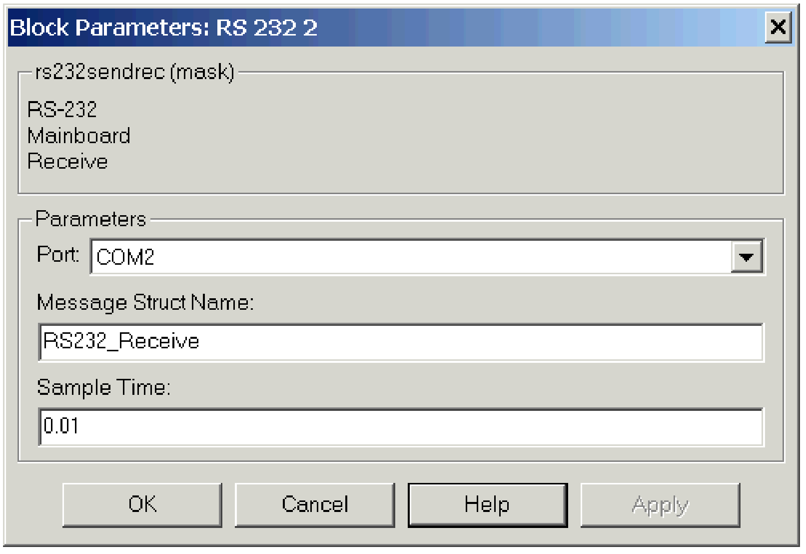
12 Click **OK**. The **Block Parameters** dialog box closes.

13 Double-click the RS-232 Send

14 . The **Block Parameters** dialog box opens.

15 From the **Port** list, select either `COM1` or `COM2`. For this example, select `COM2`. In the **Message Struct Name** box, enter the name for the MATLAB structure this block uses to receive messages from the COM2 port. In the **Sample Time** box, enter the sample time or a multiple of the sample time you entered in the RS-232 Send block.

Your **Block Parameters** dialog box should look similar to the figure shown below.



For information on entering the block parameters, see “RS-232 Receive Block (Asynchronous)” on page 1-31. For the procedure to create the send structure, see “RS-232 MATLAB Structure Reference” on page 1-32.

16 Click **OK**. The **Block Parameters** dialog box closes.

17 Double-click the Signal Generator block, and enter parameters. For example, from the **Wave Form** list, select, `sine`. In the **Amplitude** and **Frequency** boxes enter `1`. From the **Units** list, select `Hertz`. Click **OK**.

18 Double-click the Gain block, and enter parameters. For example, in the Gain box, enter -1. Click **OK**.

Your next task is to create the MATLAB message structures that the RS-232 driver blocks use to sequence commands to the RS-232 device. See “Creating RS-232 Message Structures (Synchronous)” on page 1-12.

Creating RS-232 Message Structures (Asynchronous)

RS-232 drivers use MATLAB structures to send and receive messages and map the input and output ports on the RS-232 driver blocks to the data written and read from the RS-232 devices in synchronous mode.

After you add RS-232 Setup, Asynchronous Send, and Asynchronous Receive block to your Simulink model, you can create the message structures to communicate with the RS-232 devices. You need to create and load these structures into the MATLAB workspace before you build your target application. The easiest way to create these structures is to use an M-file and load that M-file into the MATLAB workspace.

1 In the MATLAB command window, and from the **File** menu, point to **New**, and then click **M-file**.

   A MATLAB text editor window opens.

2 Enter the initialization, send, receive, and termination messages. Each message is an element in a MATLAB structure array with a series of fields For information and examples of these fields, see “RS-232 MATLAB Structure Reference” on page 1-32.

   For example, if you want to send and receive two floating-point numbers, type the following.

   Note Field names in the structures are case sensitive.

```
RS232_Send(1).SendData = 'start,%f,%f,stop;\r';
RS232_Send(1).InputPorts = [1];
RS232_Send(1).Timout = 0.01;

RS232_Receive(1).RecData = 'start,%f,%f,stop;\r';
RS232_Receive(1).OutputPorts = [1];
```

```
RS232_Receive(1).Timout = 0.01;
RS232_Receive(1).EOM 1;
```

Note If you do not manually load the message structures before opening your Simulink model, or have the message structures automatically loaded with the model, the port connections to the RS-232 blocks break.

3 From the **File** menu, click **Save As**. In the **Save As File** dialog box, enter the name of the M-file script. For example, enter

```
RS232Async_Messages.m
```

4 Close the text editing window.

5 In the MATLAB command window, type the name of the M-file script you created with the RS-232 structures. For example, type

```
RS232Async_Messages
```

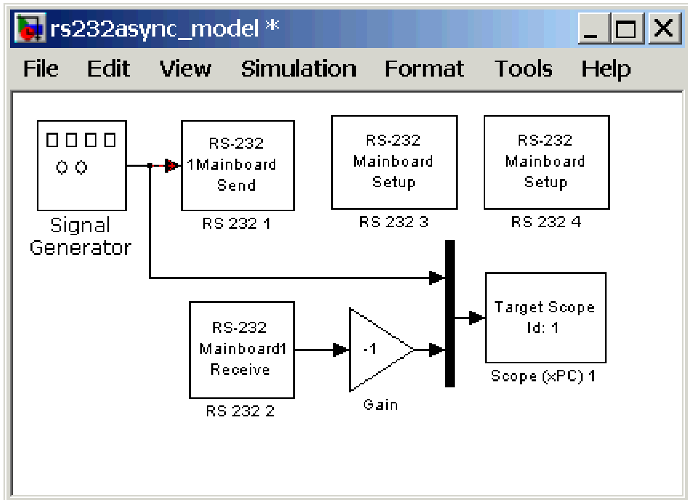
MATLAB loads and runs the M-file script to create the message structures in the MATLAB workspace needed by the RS-232 driver blocks.

6 Open your Simulink model, or press **Ctrl+D**.

Simulink updates the RS-232 driver blocks with the information from the structures. For example, Simulink adds the inputs and outputs defined in the structures to the blocks.

7 Connect the input and output ports on the RS-232 driver blocks to other blocks in your Simulink model.

Your model should look similar to the figure shown below.



8 Set the pre-load function for your Simulink model to load the message structures when you open the model. For example, if you saved the message structures in the M-file `RS232async_messages`, type

```
set_pram(gcs, 'PreLoadFcn','RS232async_messages')
```

Note If you do not manually load the message structures before opening your Simulink model, or have the message structures automatically loaded with the model, the port connections to the RS-232 blocks breaks.

Your next task is to build and run the target application.

Building and Running the Target Application (Asynchronous)

xPC Target and Real-Time Workshop create C code from your Simulink model. You can then use a C compiler to create executable code that runs on the target PC.

After you have added the RS-232 blocks for asynchronous mode to your Simulink model, and created and loaded the RS-232 structures into the MATLAB workspace, you can build your target application.

Note You cannot use a serial port to communicate between the host PC and target PC with this example. Using a serial port would disable the COM port and the example would not operate.

---

1 In the Simulink window, and from the **Tools** menu, point to **Real-Time Workshop**, and then click **Build Model**.

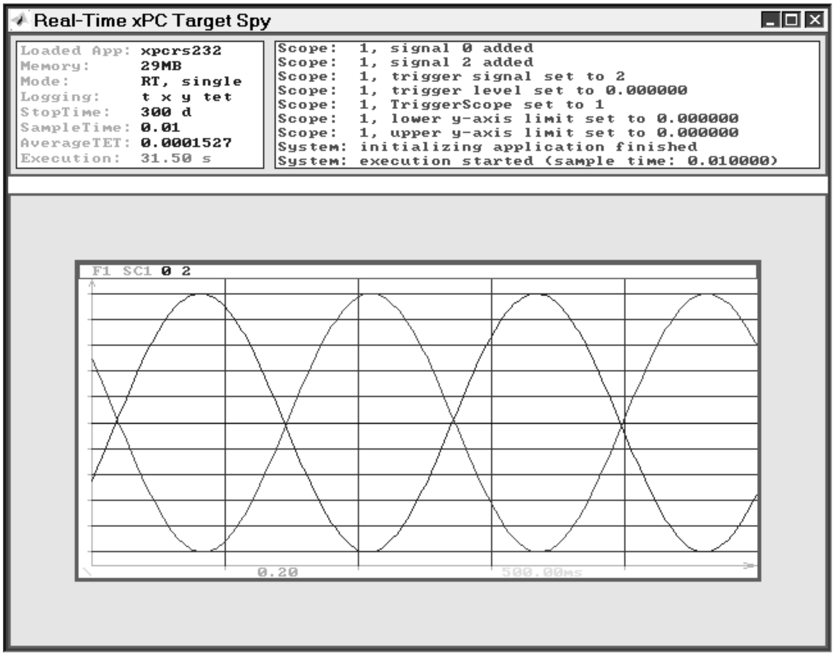
2 In the MATLAB command window, type

   ```
   +tg or tg.start or start(tg)
   ```

   The target application begins running in real time.

   For each sample period, the RS-232 messages you entered in the RS-232 send and receive message structures are executed.

   In this example, the target PC displays the inverted waveform. The RS-232 Send and RS-232 Receive blocks require a minimum delay or one sample to send the data and also receive it. When running at faster sample rates, several sample intervals may elapse while one set of data is transmitted since RS-232 communication is not particularly fast. The sample delay just described is not visible in this example.



This example can be extended for multiple D/A channels by simply adding more input signals and modifying the format string to have additional `'%f'format specifiers.`

Note This example requires that you are not using host PC to target PC communication using a serial port since that would disable that COM port and the example would not operate.

RS-232 Simulink Block Reference

xPC Target supports RS-232 communication with driver blocks in your Simulink model and message structures in the MATLAB workspace.

This section includes the following topics:

- **RS-232 Setup Block** — Sends the initialize and termination messages. You need one Setup block for each RS-232 port you use in your model.
- **"RS-232 Send/Receive Block (Synchronous)"** — Sequences the send and receive messages for synchronous serial communication.
- **RS-232 Send Block (Asynchronous)** — Sequences the send messages.
- **RS-232 Receive Block (Asynchronous)** — Sequences the receive messages.

RS-232 Setup Block

The **Block Parameters** dialog box for the RS-232 Setup block contains the following fields.

Parameter	Description
Port	From the list, select `COM1`, `COM2`, `COM3`, or `COM4`. This is the serial connection the target PC uses to communicate with the RS-232 device.
Baudrate	From the list, select `115200`, `57600`, `38400`, `19200`, `9600`, `4800`, `2400`, `1200`, `300`, or `110`.
Number of Databits	From the list, select either `7` or `8`.
Number of Stopbits	From the list, select `1` or `2`.
Parity	From the list, select `None`, `Odd`, or `Even`.
Protocol	From the list, select `None` or `XOnXOff`. If your serial device does not support hardware handshaking, or your application software requires XOn/XOff handshaking, you might need to select XOn/XOff

Parameter	Description
Send Buffer Size	Enter the size, in bytes, of the send buffer.
Receive Buffer Size	Enter the size, in bytes, of the receive buffer. The **Send Buffer Size** and **Receive Buffer Size** must be large enough to hold the data to be sent or received during each model step. It is important to be aware that the buffers must be large enough to also store old data from a prior model step in the event that the entire data transmission was not completed during the prior step.
Initialization Structure	Enter the name of the structure containing the initialization messages and the expected acknowledgements when the model is initialized. If you are not using initialization messages, enter an empty matrix in this box. For information on creating this structure, see “Creating RS-232 Message Structures (Synchronous)” on page 1-12 and “Creating RS-232 Message Structures (Asynchronous)” on page 1-22
Termination Structure	Enter the name of the structure containing the termination messages and expected acknowledgements when the model is terminated. If you are not using termination messages, enter an empty matrix in this box.

The RS232 Setup block defines the number of `databits`, `baudrate`, `protocol`, and so on, for each COM port used in your Simulink model. Each model that uses RS232 I/O must have one RS232 Setup block for each COM port in the model. The RS232 Setup block does not have any inputs or outputs.

If your host PC and target PC are connected using serial communication, one COM port on your target PC is dedicated for communication with your host PC. You cannot use this COM port in your block diagram as an I/O device. For example, if the target PC uses COM1 for the communication with the host PC, COM1 cannot be used by your block diagram. An error message displays if you use COM1 as an I/O device in your block diagram. The error message appears when you attempt to build and download the target application. In this example, it would be necessary for you to use COM2 as an I/O device in your block diagram. If you are using TCP/IP as your host PC to target PC communications protocol, then you can use any COM ports with RS232 I/O drivers in your block diagram.

Note COM1 and COM3 share interrupt line 4. Similarly, COM2 and COM4 share interrupt line 3. To provide maximum performance, the COM port interrupt line on your target PC used for serial communication is disabled while real-time tasks that include RS232 blocks are executing. This also means that when COM1 is disabled, COM3 is also disabled since they both share the same interrupt line. For this case, you would use either COM2 or COM4 as your RS232 I/O device.

RS-232 Send/Receive Block (Synchronous)

The **Block Parameters** dialog box for the Synchronous Send & Receive block contains the following fields.

Parameter	Description
Port	From the list, select `COM1`, `COM2`, `COM3`, or `COM4`. This list allows you to define which COM port is used to send and receive the data. The model must contain one Setup block for each COM port you use to send and receive data. Otherwise, an error message is displayed. Note that data is sent and received on the same COM port.

Parameter	Description
Message Structure Name	Enter the name of the MATLAB structure this block uses to send and receive messages and data to an RS-232 device. For information to create this structure, see "Creating RS-232 Message Structures (Synchronous)" on page 1-12.
Sample Time	This entry allows you to define the sample time of the block. Since this block waits for data to be received from the RS-232 external device before the block is finished executing, small sample times are not suitable with synchronous mode. You must allow sufficient time for both the RS232 send and the RS232 receive operations to be completed. The smallest sample time depends on the following. • Amount of data being sent • Amount of data being received • Selected baud rate • Response time of the external device

RS-232 Send Block (Asynchronous)

The **Block Parameters** dialog box for the Asynchronous Send block contains the following fields.

Parameter	Description
Port	This list allows you to define which COM port is used for sending data. The model must contain one RS232 Setup block to configure its COM port. Otherwise, an error message is displayed.

Parameter	Description
Message Structure Name	Enter the name of the MATLAB structure this block uses to send messages and data to an RS-232 device. For information to create this structure, see "Creating RS-232 Message Structures (Synchronous)" on page 1-12.
Sample Time	This entry allows you to define the sample time of the block. Because the block does not wait until data is received from the external RS-232 device, you can set sample times to small values.

RS-232 Receive Block (Asynchronous)

The **Block Parameters** dialog box for the Asynchronous Receive block contains the following fields.

Parameter	Description
Port	This list allows you to define which COM port the data is used to send and receive data. The model must contain one RS232 Setup block for the same COM port. Otherwise, an error message is displayed.
Message Structure Name	Enter the name of the MATLAB structure this block uses to receive messages and data from an RS-232 device. For information on creating this structure, see "Creating RS-232 Message Structures (Asynchronous)" on page 1-22.
Sample Time	This entry allows you to define the sample time of the block. Because the block does not wait until data is received from the external RS-232 device, you can set sample times to small values.

RS-232 MATLAB Structure Reference

You do not use all message fields in all messages. For example, a message to send data would not use the message field`.RecData`, but would use the field `.SendData`. However, knowing the possible message fields will be helpful when you are creating any of the message structures. This section contains the following topics:

- **RS-232 Send/Receive Message Structure (Synchronous) —** Description of the message fields for the send/receive structure associated with RS-232 asynchronous mode and the RS-232 Send/Receive block.
- **RS-232 Send Message Structure (Asynchronous) —** Description of the message fields for the send structure associated with RS-232 synchronous mode and the RS-232 Send block.
- **RS-232 Receive Message Structure (Asynchronous) —** Description of the message fields for the receive structure associated with RS-232 synchronous mode and the RS-232 Receive block.
- **Supported Data Types for Message Fields —** List of supported data types and the format you use to indicate those types in message fields.

RS-232 Send/Receive Message Structure (Synchronous)

Below are descriptions of the possible message fields for the send /receive structures with asynchronous mode. The order of the fields does not matter. However, the field names are case sensitive.

Message Field	Description
SendData	Data and format sent to the RS-232 device. Default value = `''`.
InputPorts	Number of input ports for the driver block. Data from the input ports is sent to the RS-232 device with the message field`.SendData`. Default value = `[]`. The highest number you enter determines the number of input ports on the driver block. For example, the following message creates two input ports on the driver block, `RS232_Send_Receive(1).InputPorts= [1 2];`
RecData	Data and format received from the RS-232 device. Default value = `''`. The format of this statement is very similar to a scanf statement. The read data is mapped to the output ports defined in the message field `.OutputPorts`. If a negative output port is given, the data is read in, but not sent to any output port.
OutputPorts	Number of output ports from the driver block. Data received from a RS-232 device is sent to the output ports with the message field `.ReceiveData`. Default value = `[]`. The highest number you enter determines the number of output ports on the driver block. For example, to use output ports 1 and 2 on the driver block, type `RS232_Send_Receive.OutputPorts = [1 2];`

Message Field	Description
Timeout	Time, in seconds, the driver block waits for data to be returned. Default value = `0.049`.
EOM	Number of characters you use to indicate the end of a message.

RS-232 Send Message Structure (Asynchronous)

Below is a description of the possible message fields for the send structure. with synchronous mode. The order of the message fields does not matter. However, the field names are case sensitive.

Message Field	Description
SendData	Data and format sent to the RS-232 device. Default value = `''`
InputPorts	Number of input ports for the driver block. Data from the input ports is sent to the RS-232 device with the message field `.SendData`. Default value = `[]`. The highest number you enter determines the number of input ports on the driver block. For example, the following message creates two input ports on the driver block, `RS232_Send_Receive(1).InputPorts= [1 2];`
Timeout	Time, in seconds, the driver block waits for data to be returned. Default value = `0.049`.
EOM	Number of characters you use to indicate the end of a message.

RS-232 Receive Message Structure (Asynchronous)

Below are descriptions of the possible message fields for the receive message Structures with synchronous mode.

Message Fields	Description
RecData	Data and format received from the RS-232 device. Default value = `''`. The format of this statement is very similar to a scanf statement. The read data is mapped to the output ports defined in the message field `.OutputPorts`. If a negative output port is given, the data is read in but not sent to any output port
OutputPorts	Number of output ports from the driver block. Data received from a RS-232 device is sent to the output ports with the message field `.ReceiveData`. Default value = `[]`. The highest number you enter determines the number of output ports on the driver block. For example, to use output ports 1 and 2 on the driver block. `RS232_Send_Receive.OutputPorts = [1 2];`
Timeout	Time, in seconds, the driver block waits for data to be returned. Default value = `0.049`.
EOM	End of message character.

Supported Data Types for Message Fields

The following table lists the supported data types for the RS-232 message fields.

Format	Description
`%c and %C`	Single character and wide character
`%d or %I`	Signed decimal integer

Format	Description
`%u`	Unsigned decimal integer
`%o`	Unsigned octal integer
`%x` or `%X`	Unsigned hexadecimal integer using `'abcdef'` or `'ABCDEF'` for the hexadecimal digits.
`%e` or `%E`	Exponential format using `e` or `E`
`%f`	Floating point
`%g`	Signed value printed in `f` or `e` format depending on which is smaller
`%G`	Signed value printed in `f` or `E` format depending on which is smaller

RS-232 Binary Mode

Use RS232 Binary Mode when you want to transfer raw data. The format of this data is either a custom format or is an image of the bytes as they are stored in memory. This section includes the following topics:

- **RS232 Binary Mode I/O** — When to use RS232 Binary Mode
- **RS232 Binary Mode I/O** — How to select drivers from the xPC Target block library
- **RS232 Binary Receive Block** — Explanation of Block Parameters, inputs and outputs
- **RS232 Binary Send Block** — Explanation of Block Parameters and input
- **Example Using RS232 Binary Mode I/O** — Simulink model using xPC Target driver blocks

RS232 Binary Mode I/O

The Binary mode drivers operate in asynchronous mode. In other words, they do not wait until an entire packet of data is received, but receive as many bytes as available and then go on to the next block. When an entire packet has been received, the block outputs the new data. Sent data is also handled similarly. The Send block instructs the RS232 hardware to send a certain number of bytes, but does not wait for these bytes to actually be sent.

The RS232 Binary Mode infrastructure also includes blocks to Pack and Unpack any data received. This translates the raw bytes into signals that Simulink can understand.

The functioning of these blocks is identical to the corresponding blocks in the UDP section of the xPC Target block library. The RS232 Binary Pack and Unpack blocks are actually references to these blocks. For information about UDP and the functionality of these blocks, see Chapter 5, “UDP I/O Support.”

Using RS232 Binary Mode

To use the RS232 Binary Mode blocks, you must first insert exactly one RS232 Setup block for each COM port into your model. The setup for this block is exactly the same as it is for text-based I/O, except that you do not need Initialization or Termination structures. In the dialog box, set both these fields to the empty matrix ([]).

The RS232 Binary Mode blocks may be found in the RS232 section of the xPC Target Block Library. Use the following procedure to access these blocks:

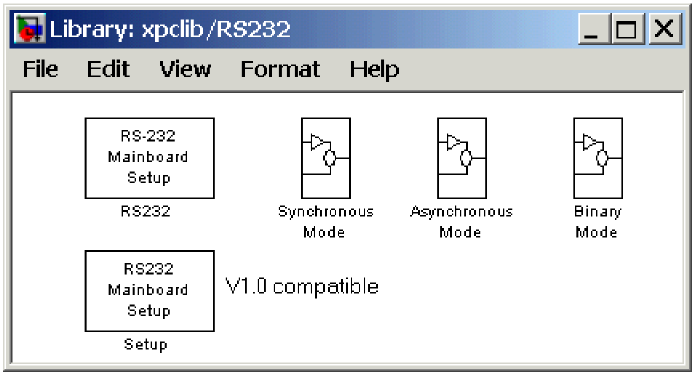
1 In the MATLAB command window, type

```
xpclib
```

The xPC Target Block Library open.

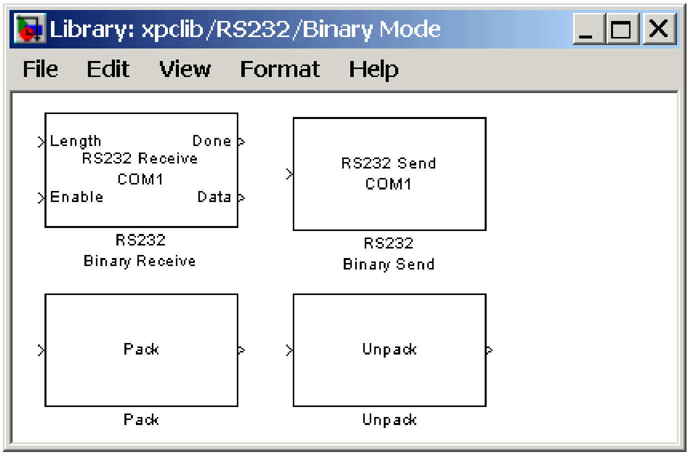
2 Double-click the group block **RS232**.

The **Library: xpclib/RS232** library opens.



3 Double-click the group block **Binary Mode**.

The **Library: xpclib/RS232/Binary Mode** library opens.

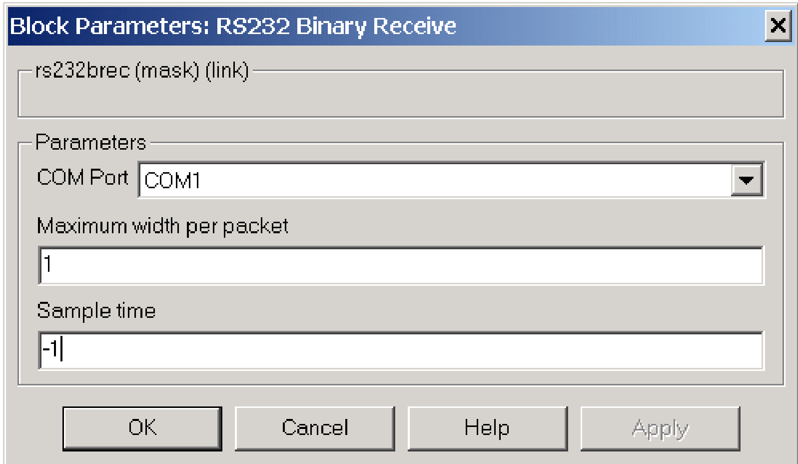


4 Drag-and-drop any of these blocks into your Simulink model.

RS232 Binary Receive Block

The RS232 Binary Receive Block was designed with generality in mind. To this end, it supports reception of variable length packets. A packet can be split up between two different RS232 Binary Receive blocks, say for a fixed length header followed by a variable length body. However, the maximum possible length of a packet has to be specified in the block, and the output from the block is a vector whose width is equal to this maximum length.

If you drop a block into your model and double-click it, a Block Parameters dialog box opens where you can modify the parameters for this block.



Block Parameters

COM Port — From the list, select, `COM1`, `COM2`, `COM3`, or `COM4`. This is the RS232 port you want to receive data from. An RS232 Setup block must also exist for the same COM port in your model.

Maximum width per packet — Enter a value that Simulink and Real-Time Workshop use to allocate memory for the received data. This is also the width of the data output. In case the actual data is less wide than the maximum, the first few bytes of the output vector are the real data and the remaining bytes are undefined.

Sample time — Specifies how often the block has to be executed. In the example dialog box shown above, the setting of `-1` specifies an inherited sample time, either from the base sample time of the model, or from the block that the output of this block goes to.

Block Inputs

The RS232 Binary Receive Block has two input ports.

- **First input port** — This port is labeled **Length**, and is the size of the packet it will receive. This value should be less than or equal to the Maximum packet length parameter. The effect of changing the **Length** input during reception of one packet is undefined.

- **Second input port** — This port is labeled **Enable**, and turns the block on or off. If the Enable input is nonzero, the block attempts to receive data, otherwise it simply does nothing.

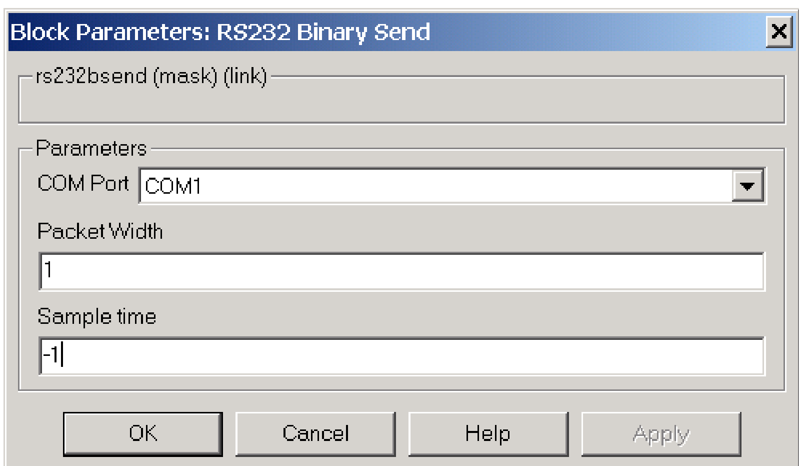
Block Outputs

The RS232 Binary Receive block has two output ports:

- **First output port** — This port is labeled **Done**, and is a function call output. This output issues a function call as soon as the block has completed receiving one packet. This may be used to drive a function call subsystem to switch. For example, to switch from a "header-receive" block to a "body-receive" block.
- **Second output port** — This port is labeled **Data**, and is the data output port. The data is a vector of `uint8`s, and is a vector of width equal to that specified in the **Maximum width per packet** parameter. If the **Length** input is less than this width, the first number of bytes equal to **Length** are the real data and the rest is garbage.

RS232 Binary Send Block

If you drop a block into your model and double-click it, a Block Parameters dialog box opens where you can modify the parameters for this block.



Block Parameters

COM Port — From the list, select, COM1, COM2, COM3, or COM4. This is the port you want to use to send the data.

Packet width — Enter the width of the incoming data. This value is a constant, unlike the Receive block.

Sample time — Enter the frequency this data is sent.

Block Input

Input Port — This port represents the data to be transmitted. The data should be a vector of type uint8 and of a **Packet width** specified in the dialog box.

RS232 Binary Pack Block

Same as UDP Pack block. See “UDP Pack Block” on page 5-8.

RS232 Binary Unpack Block

Same as UDP Unpack block. See “UDP Unpack Block” on page 5-10.

Example Using RS232 Binary Mode I/O

To understand the flexibility provided by the RS232 Binary Receive block, we present an example of how this block can be used. The model that implements this setup is provided with xPC Target. To access this model, type

```
xpcrs232bindemo
```

at the command prompt. This will open the model, which is essentially self-documenting. Open each subsystem in the model to see what that part is supposed to accomplish.

Here, we describe an example of a messaging protocol that the model has to conform to.

The protocol consists of a one-byte header, followed by a variable length body. The header can have only two legal values, 12 and 17. If the header is 12, the body is 6 bytes long, and consists of a uint16 followed by an int32 (in terms of MATLAB data types). If the header is 17, the body is 4 bytes long, and consists of a uint16 followed by an int16.

The model receives one header byte at a time, rejecting any invalid ones. As soon as a valid header byte is received, the execution switches to the body block, where the proper number of bytes is received. The data is then appropriately decoded and displayed on an xPC Target Scope of type target. The model should serve as an example of how this is done.

The basic algorithm is to receive a header byte and then compare it to the list of known headers (12 and 17). The body length is set appropriately depending on the header, and the “Done” function-call output of the header block is used to trigger functioning of the body block (via the “distributor” function call subsystem).

2

GPIB I/O Support

xPC Target interfaces the target PC to a GPIB instrument bus using an external GPIB controller from National Instruments. This external controller is connected to the target PC with a serial cable. This chapter includes the following sections:

Introduction to GPIB Drivers

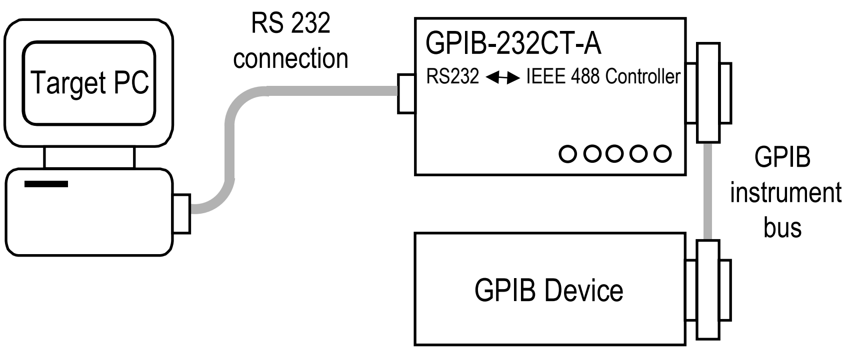
xPC Target uses a model for I/O that includes both Simulink blocks, for the I/O drivers, and MATLAB structures for sequencing messages and commands. This model provides increased flexibility and control over using only Simulink blocks in your model. The topics in this section are

- **Hardware Connections for GPIB** — Connect the target PC to a GPIB-232CT-A controller from National Instruments.
- **Simulink Blocks for GPIB** — Add setup, send, and receive blocks to your Simulink model.
- **MATLAB Message Structures for GPIB** — Create message structures to sequence instructions to and from the GPIB controller.

Hardware Connections for GPIB

xPC Target supports connecting to a GPIB instrument bus with a GPIB-232CT-A controller from National Instruments.

One end of the controller is connected to either the COM1 or COM2 port on the target PC with a null modem cable. The other end is connected to the GPIB instrument bus with a standard GPIB connector and cable.



Simulink Blocks for GPIB

To support the use of GPIB, the xPC Target I/O library includes a set of GPIB driver blocks. These driver blocks can be added to your Simulink model to provide inputs and outputs to devices on a GPIB instrument bus:

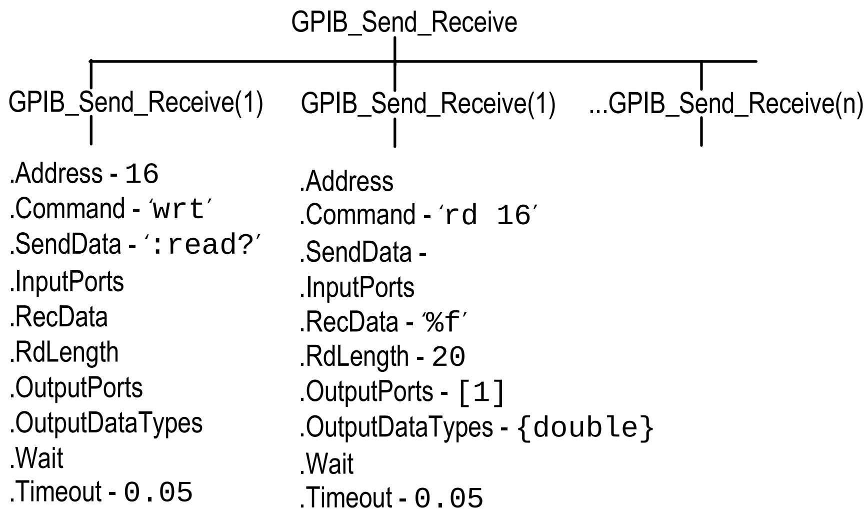
- **GPIB Setup** — One setup block is needed for each GPIB controller. The setup block does not have any inputs or outputs, but sends the initialization and termination messages.
- **GPIB Send/Receive** — The send/receive block has inputs and outputs from your Simulink model, and sequences both the send and receive messages.

MATLAB Message Structures for GPIB

Communication is through a series of messages passed back and forth between the target PC and the GPIB controller. To accomplish this, the messages sent to the GPIB controller must be in a format that the controller understands. Likewise, the target PC must know how to interpret the data returned from the GPIB controller.

xPC Target uses MATLAB structures to create messages and map the input and output ports on the GPIB driver blocks to the data written and read from the GPIB devices. The GPIB Setup block executes the messages in the initialization structure after downloading the target application. The GPIB Send/Receive block repeats the execution of the messages in the send/receive structure during each sample interval. When the target application stops running, the GPIB Setup block executes the messages in the termination structure.

Below is an example of a send/receive message structure. The first message writes a command to instruct the GPIB device to acquire a single data value, while the second message sends a command to read that value and output the result to the output port line coming from a GPIB driver block.



Currently, only two limitations exist. xPC Target does not support string data types as input and output data types. Also, you must know the size and order of data returned from a read command.

For more information on this example, see “Creating GPIB Message Structures” on page 2-10

Using GPIB Drivers

xPC Target uses a combination of Simulink blocks and MATLAB structures to support GPIB communication from your target application and target PC. The topics in this section are

- **Adding GPIB Driver Blocks** — Add the setup and send/receive blocks you need to add to your Simulink model for GPIB communication.
- **Creating GPIB Message Structures** — Create the initialize, send/receive, and termination message structures you need in the MATLAB workspace.

This section uses an example of a multimeter attached to a GPIB bus with an address of 16. This multimeter needs the initialization command

```
:conf:volt:dc
```

to set the device to read DC voltages, and needs the command

```
:read?
```

`during each sample interval` to read one voltage value

Adding GPIB Driver Blocks

The GPIB driver blocks initialize and communicate directly with the GPIB controller. The GPIB controller then communicates with the GPIB devices on the instrument bus.

After you create a Simulink model, you can add GPIB driver blocks and define the initialization, send/receive, and termination message structures.

1. In the MATLAB command window, type

   ```
   xpclib
   ```

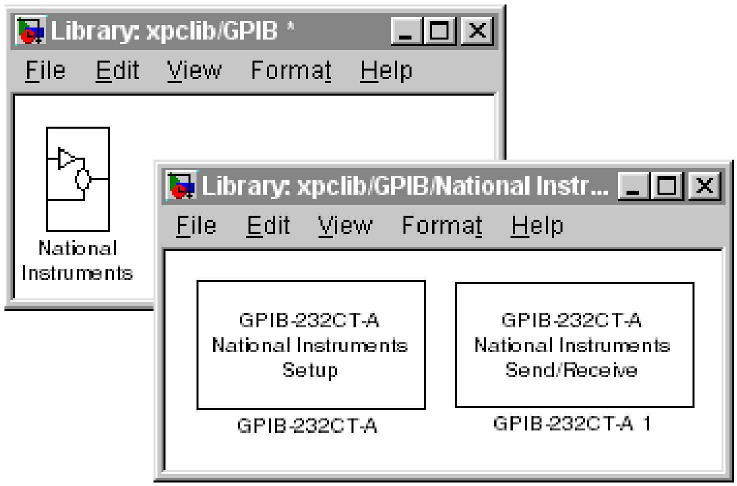
   The xPC Target driver block library opens.

2. Double-click the **GPIB** group block.

   A manufacturers window opens. Currently xPC target only supports GPIB communication with a National Instruments controller.

3 Double-click the National Instruments group block.

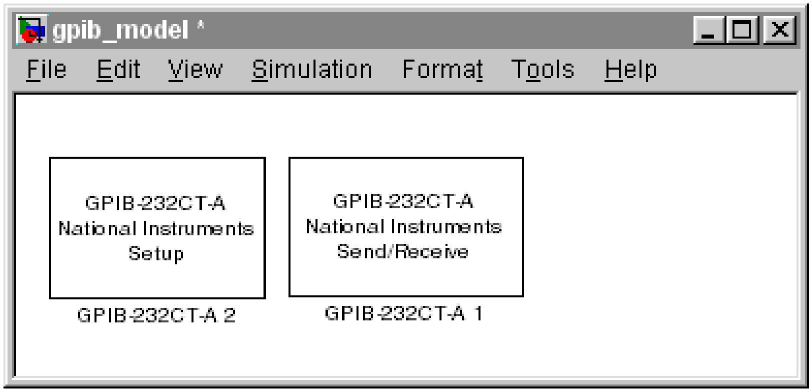
A window with blocks for GPIB drivers opens.



Alternatively, you could access the xPC Target block library from the Simulink Library Browser. In the Simulink window, and from the **View** menu, click **Show Library Browser**. In the left pane, double-click **xPC Target**, double-click **GPIB**, and then click **National Instruments**.

4 Drag-and-drop a GPIB Setup block and a GPIB Send/Receive block to your Simulink model.

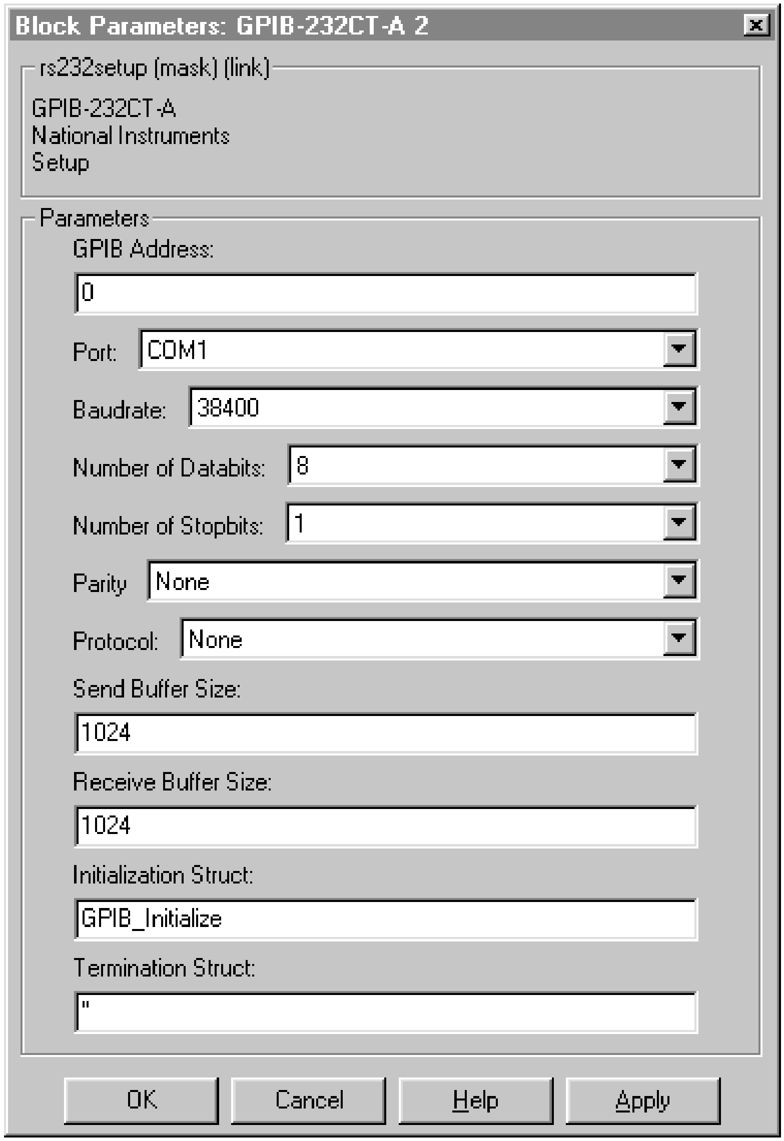
Your model should look similar to the figure below. Notice, the input and output ports are not defined or visible on the blocks. The inputs and outputs are defined in a MATLAB message structure, and visible only after you load that structure into the MATLAB workspace and update your Simulink model.



5 Double-click the GPIB Setup block. Enter values that correspond to the DIP switch settings you set on the GPIB-232CT-A controller. In the **Initialization Struct** box, enter the name for the MATLAB structure this block uses to send initialization messages to the GPIB device.

Note If you are not using an initialization or termination structure, enter two single quotes.

For example, if the target PC is connected to COM1, and you set the switches on the controller to 38400 baud, 8 databits, and 1 stopbit, your **Block Parameter** dialog box should look similar to the figure shown below.



For more information on entering the block parameters, see “PC-DIO-24” on page 18-7. For the procedure to create the initialization structure, see

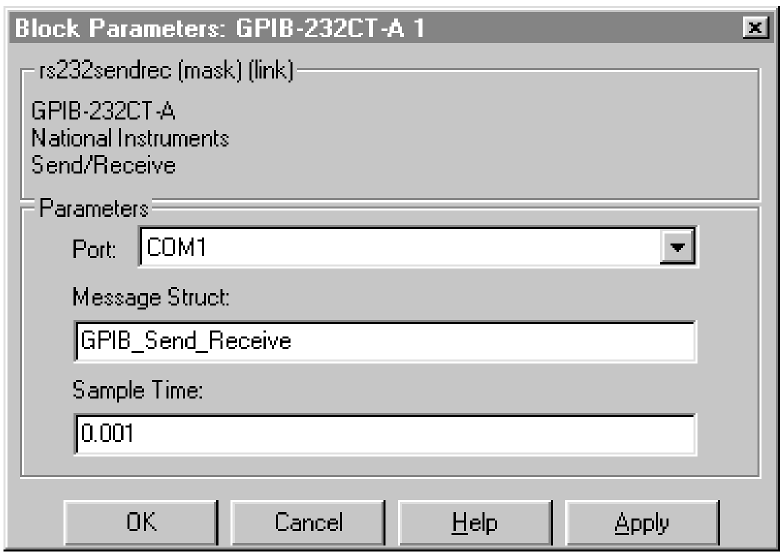
"Creating GPIB Message Structures" on page 2-10.

6 Click **OK**. The Block Parameters dialog box closes.

7 Double-click the GPIB Send/Receive block. The Block Parameters dialog opens.

8 From the **Port** list, select either `COM1` or `COM2`. This is the port on the target PC connected to the GPIB controller. In the **Message Struct Name** box, enter the name for the MATLAB structure this block uses to send and receive messages to the GPIB device. In the **Sample Time** box, enter the same sample time or multiple of the sample time you entered for the step size in the **Simulation Parameters** dialog box.

Your **Block Parameter** dialog box should look similar to the figure shown below.



For more information on entering the block parameters, see "PC-DIO-24" on page 18-7.

9 Click **OK**. The **Block Parameters** dialog box closes.

Your next task is to create the MATLAB message structures that the GPIB driver blocks use to sequence commands to the GPIB controller. See "Creating GPIB Message Structures" on page 2-10.

Creating GPIB Message Structures

GPIB drivers use MATLAB structures to send and receive messages, and map the input and output ports on the GPIB driver blocks to the data written and read from the GPIB devices.

After you add GPIB driver blocks to your Simulink model, you can create the message structures to communicate with the GPIB controller. You need to create and load these structures into the MATLAB workspace before you build your target application. The easiest way to create these structures is to create an M-file and load that M-file into the MATLAB workspace.

1 In the MATLAB command window, and from the **File** menu, point to **New**, and then click **M-file**.

   A MATLAB text editor window opens.

2 Enter the initialization and send/receive messages. Each message is an element in a MATLAB structure array with a series of fields. For information and examples of these fields, see "GPIB Initialization and Termination Message Structures" on page 2-17 and "GPIB Send/Receive Message Structure" on page 2-18.

   As an example, if you have a multimeter attached to a GPIB bus that has an address of 16, needs the initialization command `:conf:volt:dc` to set the device to read DC voltages, and uses the command `:read?` to read one voltage value, you could type the following.

   Note Field names in the structures are case-sensitive.

```
GPIB_Initialize(1).Command = 'wrt 16';
GPIB_Initialize(1).SendData = ':conf:volt:dc';

GPIB_Send_Receive(1).Address= 16;
GPIB_Send_Receive(1).Command = 'wrt 16';
GPIB_Send_Receive(1).SendData = ':read?';
GPIB_Send_Receive(1).Timeout = 0.05;
```

```
GPIB_Send_Receive(2).Command = 'rd 16';
GPIB_Send_Receive(2).RecData = '%f';
GPIB_Send_Receive(2).RdLength = 20;
GPIB_Send_Receive(2).OutputPorts = [1];
GPIB_Send_Receive(2).OutputDataTypes = {'double'};
GPIB_Send_Receive(2).Timeout = 0.15;
```

This example did not need a termination structure. But if it did, the format of the structure is the same as the initialization structure. For example, a termination structure could have a message with the `.Command` and `.SendData` fields.

```
GPIB_Termination(1).Command
GPIB_Termination(1).SendData
```

3 From the **File** menu, click **Save As**. In the **Save As File** dialog box, enter the name of the M-file script. For example, enter

```
GPIB_Messages.m
```

4 Close the text editing window.

5 In the MATLAB command window, type the name of the M-file script you created with the GPIB structures. For example, type

```
GPIB_Messages
```

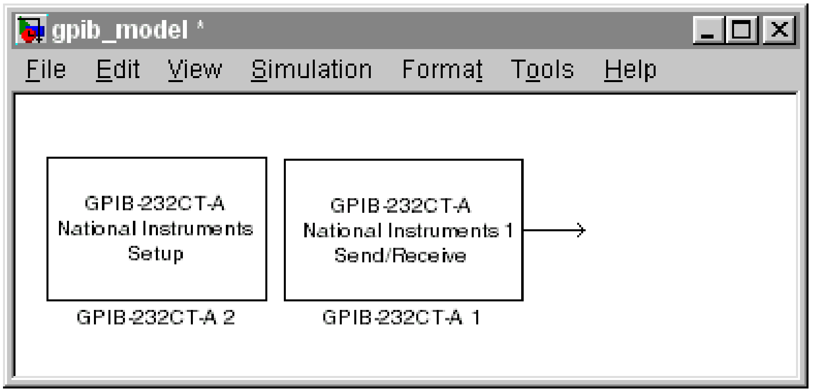
MATLAB loads and runs the M-file script to create the message structures in the MATLAB workspace needed by the GPIB driver blocks.

6 Open your Simulink model, or press **Ctrl+D**.

The GPIB driver blocks are updated with the information from the structures. For example, inputs and outputs defined in the structures are now visible on the driver blocks.

7 Connect the input and output ports on the RS-232 driver blocks to other blocks in your Simulink model.

Your model should look similar to the figure shown below.



8 Set the `PreLoadFcn` for your Simulink model to load the message structures when you open the model. For example, if you saved the message structures in the M-file `GPIB_messages`, type

```
set_param(gcs, 'PreLoadFcn','GPIB_messages.m')
```

Note If you do not manually load the message structures before opening your Simulink model, or have the message structures automatically loaded with the model, the port connections to the GPIB driver blocks break.

Your next task is to build the target application and download it to the target PC.

GPIB Simulink Block Reference

The GPIB-232CT-A is a GPIB controller external to the target PC. It is connected to the target PC with an RS-232 cable.

xPC Target supports this controller with two driver blocks:

- GPIB-232CT-A Setup Block
- GPIB-232CT-A Send/Receive Block

Board Characteristics

Board name	GPIB-232CT-A
Manufacturer	National Instruments
Bus type	N/A
Access method	RS232
Multiple block instance support	No
Multiple board support	Yes

GPIB-232CT-A Setup Block

The setup block parameters must be set to match the jumper settings on the GPIB-232CT-A controller.

Driver Block Parameters

Parameter	Description
GPIB id	Enter the identification number for the GPIB controller. When the GPIB-232CT-A is turned on, the identification number is set to `0`.
Port	From the list, select `COM1`, `COM2`, `COM3`, or `COM4`. This is the serial connections the target PC uses to communicate with the GPIB-232CT-A controller.

Parameter	Description
Baudrate	From the list, select 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200, 600, or 300.
Number of Databits	From the list, select 8 or 7.
Number of Stopbits	From the list, select 1 or 2.
Parity	From the list, select None, Odd, or Even.
Protocol	From the list, select None or XOn/XOff. If your serial device does not support hardware handshaking, or your application soft war re requires XOn/XOff handshaking, you might need to select XOn/XOff.
Send Buffer Size	Enter the buffer size in bytes.
Receive Buffer Size	Enter the buffer size in bytes.
Initialization Struct	Enter the name of the structure containing the initialization information. For example, enter `GPIB_Initialize` If you are not using initialization messages, enter two single quotes in this box. For information on creating this structure, see "Creating GPIB Message Structures" on page 2-10.
Termination Struct	Enter the name of the structure containing the termination information.

GPIB-232CT-A Send/Receive Block

Driver Block Parameters

Parameter	Description
Port	From the list, select COM1, COM2, COM3, or COM4. Serial connection on the target PC to send and receive data
Message StructName	Enter the name of the MATLAB structure containing the messages to be sent to the GPIB controller.
Sample Time	Enter the base sample time or a multiple of the base sample time you entered in the Simulations Parameter dialog box.

GPIB MATLAB Structure Reference

You do not use all message fields in all messages. For example, a message to send data would not use the message field `.RecData`, but would use the field `.SendData`. However, knowing the possible message fields will be helpful when you are creating any of the message structures.

This section includes the following topics:

- **GPIB Initialization and Termination Message Structures** — Description of the message fields for the initialization and termination structures associated with the GPIB Setup block.
- **GPIB Send/Receive Message Structure** — Description of the message fields for the send/receive structure associated the GPIB Send/Receive block.
- **Shortcuts and Features for Messages** — Shortcuts to using the GPIB `wrt` and `rd` commands.
- **Supported Data Types for Message Fields** — List of supported data types and the format you use to indicate those types in message fields.

GPIB Initialization and Termination Message Structures

The format for the initialization and termination structures are similar to the send/receive structure except for a few differences:

The initialization and termination structures do not need to receive or send information through driver block ports on your Simulink model. Therefore, the initialization and termination structures do not use the message fields `.InputPorts`, `.OutputPorts`, `.RecData`, and `.OutputDataTypes`.

Below is a description of the possible message fields for the initialization and termination structures. The order of the message fields does not matter. However, the field names are case sensitive.

Message Fields	Description
Address	Sets the GPIB address for the device being accessed and defines the keyword ADDR. Default value = `[]`.
Command	GPIB command sent to a GPIB device. Default value = `''`.
SendData	Data sent with the GPIB command. Default value = `''`.
RdLength	Defines the length of the acknowledge string, in bytes, from the GPIB controller.
Ack	The expected acknowledge string from the controller as a result of an initialization or termination message. If this value is set, you need to set the time-out value. If no string is defined, then no acknowledge is expected.
Timeout	Time, in seconds, allowed for the GPIB controller to respond to a message and send back an acknowledge string. Default value = `0.049` seconds. If the time-out value is exceeded, a time-out error is reported.

GPIB Send/Receive Message Structure

Below is a description of the possible fields for the send/receive message structure. The order of the message fields in a message does not matter. However, the field names are case sensitive.

Message Fields	Description

Address	Sets the GPIB address for the device being accessed. After the GPIB address is set, the remaining messages use this address value until another message changes the address value. Default value = `0`.
	
	The keyword ADDR is equal to the value in the message field `.Address`. You can use this keyword in the message fields `.Command` or `.SendData` to replace the numerical value of the GPIB address. For example, you can write
	
	`GPIB_Send_Receive(1).Command='wrt 16';`
	
	Or you can write
	
	`GPIB_Send_Receive(1).Address = 16;`
	`GPIB_Send_Receive(1).Command='wrt ADDR';`

Command	GPIB command sent to a GPIB device. Default value = `''`.
SendData	Data sent with the GPIB command. Default value = `''`.

InputPorts	Defines the input ports for the driver block. Data from the input ports is sent to the GPIB device with the message fields `.Command` and `.SendData`. Default value = `[]`. The highest number you enter determines the number of input ports on the driver block.
	
	For example, the following message creates two input ports on the driver block, and passes data from the input ports to the `read` command.
	
	`GPIB_Send_Receive(1).Command = 'rd #%d %d';`
	`GPIB_Send_Receive(1).InputPorts= [1 2];`

Message Fields	Description
	The first port is used to dynamically provide the length of the receive string, while the second port provides the value of the GPIB device.

RecData	Format of the data received from the GPIB device. Default value = `''`. The format of this statement is very similar to a scanf statement. The read data is mapped to the output ports defined in the field `.OutputPorts`. If a negative output port is given, the data is read in, but not sent to any output port
	
	For example, to read from a GPIB device with an address of 16, one floating- point number with a maximum number of bytes of 20, and send the data to the first driver block output, type the following
	
	`GPIB_Send_Receive(1).Command = 'rd #20 16';`
	`GPIB_Send_Receive(1).RecData = '%f';`
	`GPIB_Send_Receive(1).OutputPorts = [1];`

RdLength	Defines the length of the data, in bytes, received with the read command and defines the keyword LENGTH. Default value = `0`.
OutputPorts	Defines the output ports from the driver block. Data received from a GPIB device with the `read` command is sent to the output ports. Default value = `[].` The highest number you enter determines the number of output ports on the driver block. For example, to use output ports 1 and 2 on the driver block, type `GPIB_Send_Receive.OutputPorts = [1 2];`

Message Fields	Description
OutputData Types	Defines the data types for the output ports on the driver block. Default value = `[]` If this value is not define, and there are output ports, the default type is `double`. Also, if there are more output ports than output data types listed, the default type for the undefined ports is `double`.
Wait	The amount of time, in seconds, to wait before executing the next message. This value is limited to 50 milliseconds. Default value = `0`.
Timeout	Time, in seconds, the driver block waits for data to be returned Default value = `0.049`.

Shortcuts and Features for Messages

xPC Target defines the abbreviations `wrt` and `rd` to make message writing easier with GPIB commands. When the message interpreter sees the statements:

- `Structure_name(index).'wrt'`, it is replaced with `Structure_name(index).'wrt ADDR'`. For example, you could write

  ```
  GPIB_Initialize(1).Command = 'wrt 8';
  ```

  or you could write

  ```
  GPIB_Initialize(1).Address = 8;
  GPIB_Initialize(1).Command = 'wrt ;
  ```

  The following message fields, with the keyword ADDR, use the address value `8` defined in the message field `.Address`.

- `Structure_name (index).Command = 'rd'`, it is replaced with `Structure_name(index).Command = 'rd #LENGTH ADDR'`. For example, you could write

  ```
  GPIB_Initialize(1).Command = 'rd #10 8';
  ```

  or you could write

  ```
  GPIB_Initialize(1).Address = 8;
  GPIB_Initialize(1).RdLength = 10
  GPIB_Initialize(1).Command = 'wrt ;
  ```

If you enter numerical values in the `wrt` and `rd` commands, then the command uses those values instead of the values in the variables ADDR and LENGH. For example, the following message uses the GPIB address 10 even though the value for ADDR is defined as 8.

```
GPIB_Initialize(1).Address = 8;
GPIB_Initialize(1).Command = 'wrt 10';
```

Changes to the Read Command — When a GPIB `rd` command is sent to the GPIB controller, the controller responds with the data and length of data. To make using this command easier, the xPC Target diver block, discards the length of data information. For example, using the normal GPIB `rd` command, you could write

```
GPIB_Message(1).Command = 'rd #20 16';
```

```
GPIB_Message(1).RecData = '%f%d';
GPIB_Message(1).OutputPorts = [1 -1];
```

The code %d reads the length of data and the -1 discards the length. Using the modified xPC Target rd command, you would write

```
GPIB_message(1).Command = 'rd #20 16';
GPIB_message(1).RecData = '%f';
GPIB_message(1).OutputPorts = [1];
```

Automatic Addition of Escape Characters — The message interpreter automatically places the correct escape characters at the end of the message fields .Command, .SendData, and .Ack. However, if you add the escape characters, then the message interpreter does not add additional characters.

The escape characters that are: \\, \a, \b, \f, \r,\t, \v, \', \'', and \n.

For example, you can write

```
GPIB_Message.Command = 'wrt 16\n';
GPIB_Message.SendData = ':conf:volt:dc\r';
GPIB_Message.Ack = '10\n\r';
```

or you can write the following, and the appropriate escape characters are added.

```
GPIB_Message.Command = 'wrt 16';
GPIB_Message.SendData = ':conf:volt:dc';
GPIB_Message.Ack = '10';
```

Supported Data Types for Message Fields

The following table lists the supported data types for the message fields `.SendData` and `.Recdata`.

Format	Description
`%c` and `%C`	Single character and wide character
`%d` or `%I`	Signed decimal integer
`%u`	Unsigned decimal integer
`%o`	Unsigned octal integer
`%x` or `%X`	Unsigned hexadecimal integer using `'abcdef'` or `'ABCDEF'` for the hexadecimal digits.
`%e` or `%E`	Exponential format using `e` or `E`
`%f`	Floating point
`%g`	Signed value printed in `f` or `e` format depending on which is smaller
`%G`	Signed value printed in `f` or `E` format depending on which is smaller

3

CAN I/O Support

This chapter includes the following sections:

Model Execution Driven by CAN-Messages (Interrupt Capability of CAN Receive Blocks) (p. 3-53)	In certain application it is necessary that the model (target application) execution is driven by the pace of an incoming CAN message.
Defining Initialization and Termination CAN Messages (p. 3-56)	The CAN Setup driver blocks for all supported CAN boards allow the definition of CAN-messages to be sent out during initialization and termination of the target application.

Introduction

xPC Target offers support to connect a target PC to a CAN network using the CAN driver blocks provided by the xPC Target I/O block library for I/O device drivers for the CAN-AC2-ISA and CAN-AC2-PCI boards from Softing GmbH (Germany). The CAN driver library allows xPC Target applications to connect to any CAN fieldbus network for I/O communication or real-time target-to-target communication. This topics in this section are

- xPC Target CAN Library
- CAN-AC2
- CAN-AC2-PCI
- CAN-AC2-104
- Selecting a CAN Library
- CAN Library Property Values

xPC Target CAN Library

The drivers support CAN specification 2.0A and 2.0B and use the dynamic object mode of the CAN-AC2 firmware to achieve maximum real-time performance.

The library supports the following CAN-boards from Softing GmbH, Germany.

Board Name	Form factor	Identifier Range	Multiple Board Support
CAN-AC2	ISA	Standard (& Extended with piggyback module)	No
CAN-AC2-PCI	PCI	Standard & Extended	Yes (up to 3)
CAN-AC2-104	PC/104	Standard & Extended	Yes (up to 3)

For more information on the board specifications visit `www.softing.com`.

The xPC Target CAN library intentionally restricts its support for Softing boards with two CAN ports (boards with one channel would be available as well). This is because the two port versions allow checking the correct functioning of the board and drivers by just connecting the first CAN port to the second CAN port. This forms a loop-back without having the need to connect the board to a 'real' CAN-network. The xpcdemos directory `xpcdemos` contains simple loop-back test models to test the ISA, PCI and PC/104 boards. Type the following commands to open the corresponding test models.

Model name (command)	For board
xpccanisa	CAN-AC2
xpccanpci	CAN-AC2-PCI
xpccanpc104	CAN-AC2-104

The size of the driver code of the CAN boards supported by the xPC Target block library is significant and because not all xPC Target applications will use CAN, the CAN library code is not linked by default when building a target application. This makes target applications smaller if no CAN-communication functionality is needed. If the model to be built contains CAN driver blocks, the corresponding CAN-library support has to be enabled prior to the initiation of the build process. This has to be done in the xPC Target setup environment either using the xpcsetup-GUI or the corresponding command-line functions. See chapter 2 below for further information.

For each CAN-board three driver blocks are provided. These are: A Setup block, which defines the type of physical connection (baud rate and so forth). Exactly one instance of the setup block has to be defined in a model for each physically installed CAN-board. A Send block, which transmits (sends) the data entering the block's input ports to the connected CAN-network. One or more instances of the Send-block can be used in a model. A Receive block, which retrieves (reads) CAN-messages received by the board and outputs the data at the corresponding output ports. One or more instances of the Receive block can be used in a model.

The maximum size of the data frame of a CAN-message is 8 bytes. This is the same size as the C data type 'double' uses on PC-compatible systems. At the same time, the double data type is the default data type used for Simulink signals. Therefore the CAN data frame within a Simulink model can be easily represented by a scalar Simulink signal even if the data frame normally has nothing in common with a double floating point value. The xPC Target CAN library provides a Utility sublibrary which offers bit-packing and bit-unpacking blocks. These blocks are used to pack data types other than doubles into 64 bits (8 bytes or a double) as well as for the opposite operation.

This will be discussed in greater details further below. What is important for now, is, that CAN data frames are represented by Simulink signals of data type double.

All drivers for the supported CAN-boards program the boards for the so-called “dynamic object mode”. This is one of three modes the CAN-board firmware from Softing can operate in. For a more detailed discussion of the three modes see the board’s user manual. The dynamic object mode is best suited for real-time environments where each component of the application has to have deterministic time behavior. This is the case for xPC Target and that is the main reason why this mode has been chosen over the other two modes, which are FIFO and static object mode.

The following paragraph summarizes the differences between the three supported Softing boards.

CAN-AC2

This is the CAN-board for the ISA-Bus offering two CAN ports (Highspeed). In its standard hardware configuration it uses the Philips PCA 82C200 CAN controller, which supports Standard Identifiers only. Piggyback modules are available (one for each port) which replace the Philips CAN controllers by the Intel 82527 CAN-controllers. The Intel controllers support both Standard and Extended Identifiers. The board is a memory-mapped device and uses a 16 kilobyte big address range between 640k Byte and 1M Byte. We do not recommend this board for new projects, use the CAN-AC2-PCI which is described below instead. We will freeze the driver code for this board with the release of R12. Softing has confirmed that no new firmware versions are planned for this board either.

CAN-AC2-PCI

This is the CAN-board for the PCI-bus offering two CAN ports. The CAN-controllers used on the board are the SJA1000 from Philips. In its standard hardware configuration the board is designed for both Standard and Extended identifiers for Highspeed CAN. Piggyback modules are available (one for each port) which add Lowspeed CAN support in order to switch between Highspeed and Lowspeed CAN controlled by the driver block. The board is a memory mapped PCI device, which uses 64k Bytes of address space. The address space is assigned automatically by the PCI BIOS of the target PC and lies usually in the range between 2G bytes and 4G bytes. Any new projects where a desktop PC is used as the target system should use this board and not the ISA board described above.

CAN-AC2-104

This is the CAN-board for the PC/104-bus offering two CAN ports. The CAN-controllers used on the board are the SJA1000 from Philips. The board offers both Standard and Extended identifiers for Highspeed CAN. A Lowspeed CAN hardware extension is not available. The board is both I/O-mapped and memory-mapped. The I/O-mapped area uses a 3 bytes big address range and the memory-mapped area uses a 16k bytes big address range between 640k bytes and 1M bytes.

Selecting a CAN Library

Before you can build a target application using CAN driver blocks, you need to select the correct CAN library. The different CAN libraries are listed and selected in the xPC Target environment setup. The xPC Target environment contains a property that allows you to control the configuration and behavior of the supported CAN boards.

It is assumed that the xPC Target environment is already set up and working correctly for models that do not use CAN drivers. If you have not already done so, please confirm that you are able to build and run a target application that does not include any CAN blocks.

You can view CAN driver settings using the function `getxpcenv` or using the xPC Target Setup window.

1 In the MATLAB command window, type

   ```
   xpcsetup
   ```

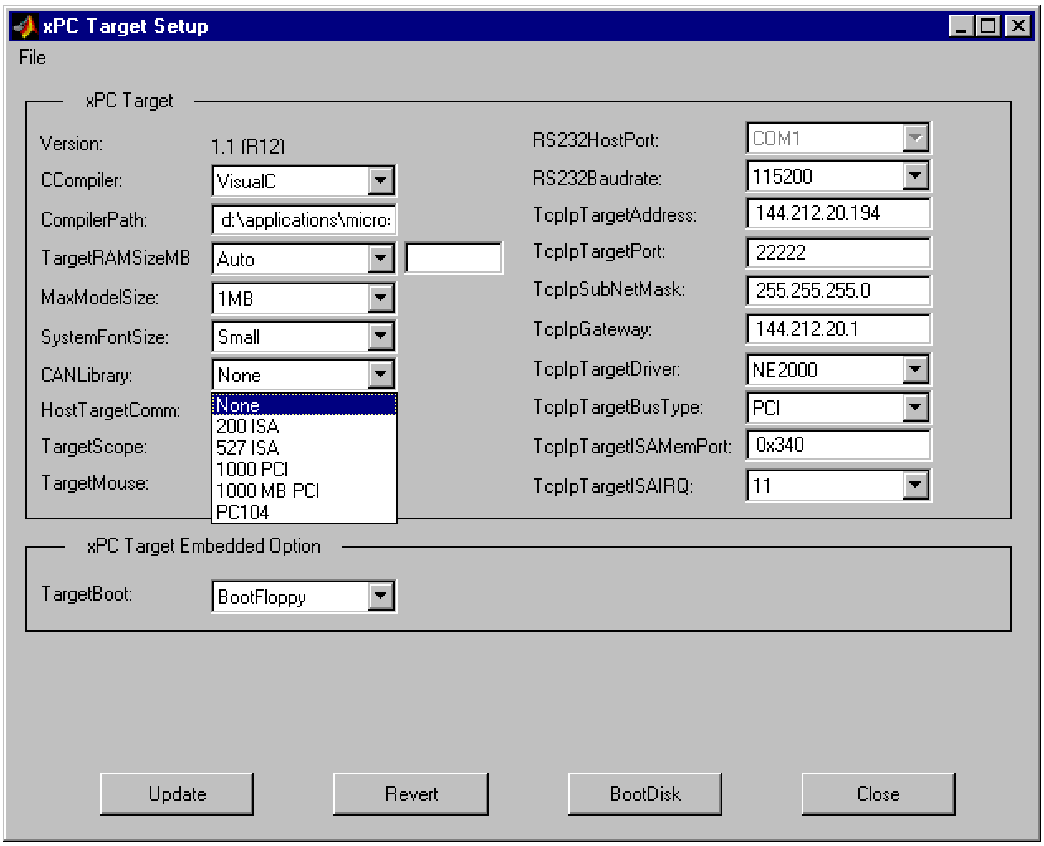
   The xPC Target Setup dialog box opens.

2 From the **CANLibrary** list, select `200ISA`, `527ISA`, `1000PCI`, or `1000MB PCI`

   The default value for the **CAN Library** list is none, which indicates that CAN devices are not enabled. If your are using CAN devices, use the **CANLibrary** list to select your CAN controller.

   Definition of the correct CAN library in the setup environment is crucial.

- If no CAN-library is defined, the target application build process will error out during the linking stage reporting several “unresolved external” errors.
- If the wrong CAN-library is defined (mismatch between actual installed physical board and CAN-library in the environment setup) the build process succeeds but downloading the firmware during application initialization on the target will error out.



3 After changing any selections in the xPC Target Setup window, be sure to update the xPC Target environment by clicking the **Update** button.

For more detailed information about using the xPC Target Setup window, see Chapter 8, "Software Environment."

Alternatively, you can select the CAN Library using the corresponding command-line functions. See "Changing Environment Properties with a Command-Line Interface" on page 8-15.

CAN Library Property Values

The following table shows which CAN Library property value depending on the used board or boards.

Board	**CAN-Library Property Value**
CAN-AC2 (ISA) with Philips PCA 82C200 (Standard)	'200 ISA'
CAN-AC2 (ISA) with Intel 82527 (Standard & Extended)	'527 ISA'
CAN-AC2-PCI with Philips SJA 1000	'1000 PCI' or '1000 MB PCI'
CAN-AC2-104 with	'PC104'

* the setting '1000 MB PCI' is the same as '1000 PCI' and is still supported in order to provide backward compatibility to version 1.0 of xPC Target.

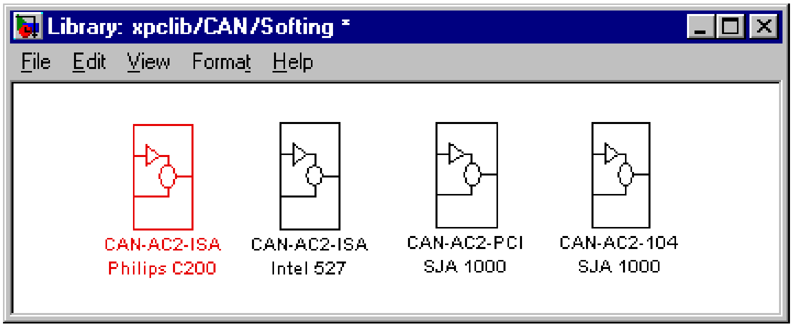
The environment setup allows to set one of the above property values. This does imply that you cannot mix different types of Softing CAN-boards in the same target system. For example you cannot use a CAN-AC2 (ISA) board together with a CAN-AC2-PCI board in the same target PC (desktop). On the other hand the xPC Target CAN-drivers support multiple boards of the same type (CAN-AC2-PCI and CAN-AC2-104, but not CAN-AC2 (ISA)) in the same target PC. For more information see the board specific driver block description.

After having chosen the right CAN-library push the "Update"-button to make the current setting the actual setting. If this is the only property you have changed in the environment setup the regeneration of the target boot floppy disk is not necessary.

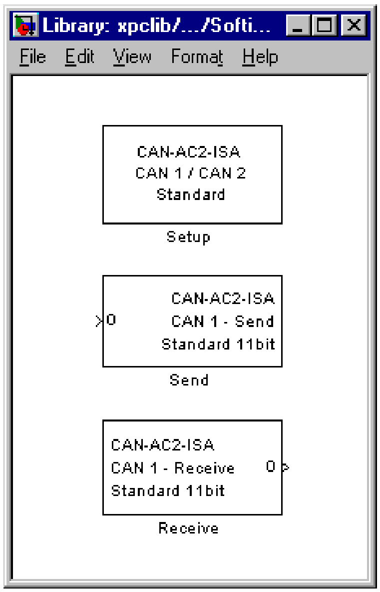
After this, close the xpcsetup-GUI. You are now ready to create the first target application using CAN driver blocks

CAN Driver Blocks for the CAN-AC2 (ISA) with Philips PCA 82C200 CAN-Controller

The driver blocks described here support the CAN-AC2 (ISA) without piggyback modules. The Philips PCA 82C200 chip is used as the CAN controller in this configuration and supports the Standard identifier range only. The driver block set for this board is found in the xPC Target I/O block library in the group CAN/Softing.

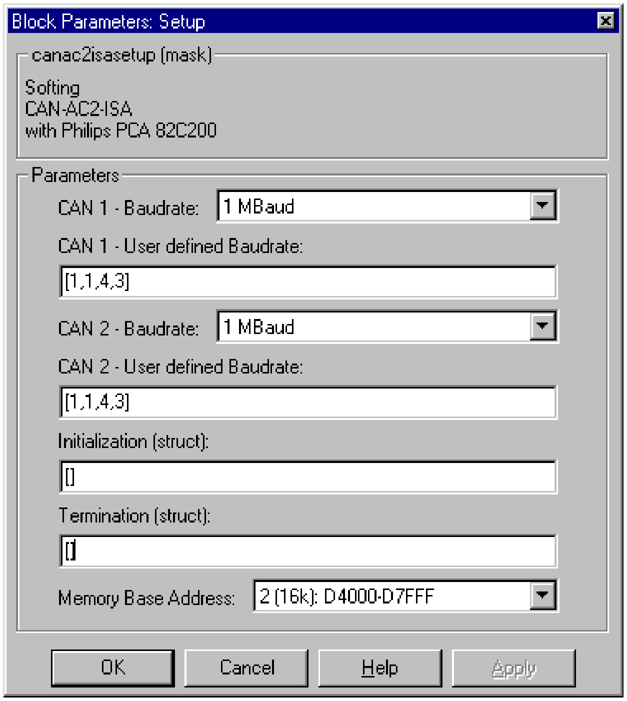


The first block group highlighted above contains the three available CAN blocks: Setup, Send, and Receive.



Setup Driver Block

The Setup block is used to define general settings of the plugged-in CAN board. Because the CAN driver blocks for this ISA board only support a single physical board for each target system, this block has to be used exactly once (one instance) in a model.



The dialog box of the Setup block lets you define the following settings.

CAN 1 - Baud rate — The first control (pop-up menu) lets you define the most common baud rates for CAN port 1. If special timing is necessary (baud rate), the value “User defined” can be selected. In this case the second control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 2 - Baud rate — The third control (pop-up menu) lets you define the most common baud rates for CAN port 2. If special timing is necessary (baud rate), the value "User defined" can be selected. In this case the fourth control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

Initialization and Termination — The fifth and sixth control (edit fields) can be used to define CAN messages sent during initialization and termination of the Setup block.

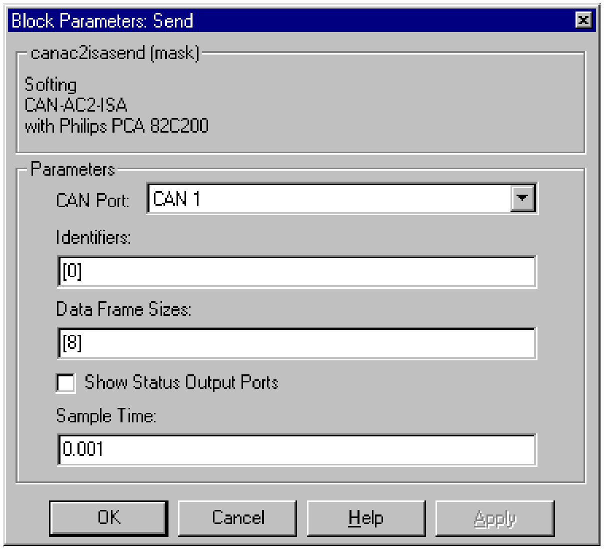
Memory base address — The seventh control (pop-up menu) is used to define the memory base address of the board. The address range used by the board has to be set by hardware jumpers on the board itself. Refer to the Softing user manual on how set the various address ranges. The setting in the dialog box has to correspond to the jumper setting otherwise the board cannot be accessed. The available address ranges (memory base address) in the pop-up menu are those supported by the board. Because the xPC Target kernel only reserves a sub range (C0000 – DC000) of the 640 kilobyte to 1 megabyte address range for memory mapped devices, the valid settings when used within a xPC target systems are:

```
1 (16k): D0000-D3FFF
2 (16k): D4000-D7FFF
```

The board allows to activate proper termination for each of the two CAN ports separately by means of hardware jumpers. Refer to the Softing user manual on how the jumpers have to be set. Both CAN ports have to be terminated properly when you use the provided loop-back model in order to test the board and drivers.

Send Driver Block

The Send driver block is used to transmit data to a CAN network from within a block model.



The dialog box of the block lets you define the following settings.

CAN port — The first control (pop-up menu) is used to select at which CAN port the CAN message will be sent out.

Identifiers — The second control (edit field) is used to define the identifiers of the CAN-messages sent out by this block. It has to be a row vector where the elements define a set of Standard identifiers. Each element has to be in the range between 0 and 2031. The number of identifiers for each CAN port in a model per physical CAN-board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of inputs ports of the block. The block icon displays the selected identifier at each input port. Each input port accepts the data frame to be sent along with the CAN-message. The signal entering each input port has to be a scalar of type double representing the maximum size of 8 bytes of a CAN-message data frame.

Data frame sizes — The third control (edit field) is used to define the data frame size for each identifier (CAN-message) in bytes. It has to be a row vector where the elements define a set of data frame sizes. Each element has to be in the range between 1 and 8. If the data frame sizes for all identifiers defined in the control above have to be the same, the size can be provided as a scalar only and scalar expansion applies. If the sizes are different for at least two identifiers (CAN-messages) one size element has to be provided for each identifier defined in the control above. Therefore the length of the two vectors have to be the same.

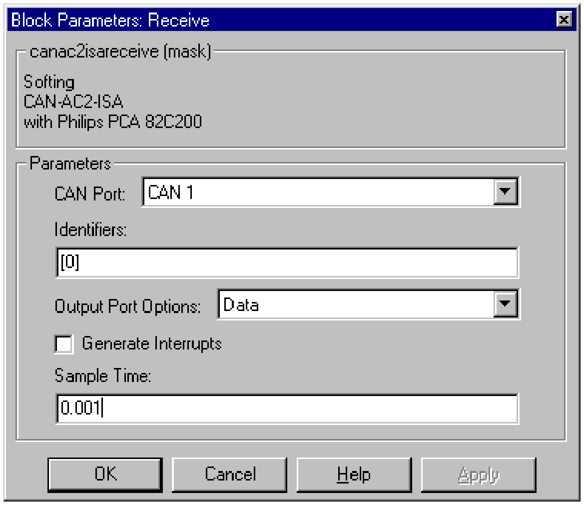
Show status output ports — The fourth control (check box) lets you enable status output ports for each identifier (CAN-message). If the check box is checked the block shows as many output ports as input ports. The data type of each output port is a double and the value is identical to the return argument of function CANPC_write_object(…) described in the Softing user manual. Refer to the manual for more information.

Sample time — The fifth control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

You can use as many instances of the Send block in the model as needed. For example by using two instances of the block with different sample times, CAN-messages can be sent out at different rates. Or you can use multiple instances to structure your model more efficiently.

Receive Driver Block

The Receive driver block is used to retrieve data from a CAN-network to be used within a block model.



The dialog box of the block lets you define the following settings.

CAN port — The first control (pop-up menu) is used to select from which CAN port, the CAN messages will be retrieved from.

Identifiers — The second control (edit field) is used to define the identifiers of the CAN-messages retrieved by this block. It has to be a row vector where the elements define a set of Standard identifiers. Each element has to be in the range between 0 and 2031. The number of identifiers for each CAN port in a model per physical CAN board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of output ports of the block. The block icon displays the selected identifier at each output port. Each output port will output the data frame being retrieved along with the CAN-message. The signal leaving each output port is a scalar of type double representing the maximum size of 8 bytes of a CAN message data frame.

Output port options — The third control (pop-up menu) lets you define which type of retrieved data is output at each output port. Three different types of data can be output, which are data frame, status and timestamp. The status information is of type double and is identical to the return value of function CANPC_read_rcv_data(…) described in the Softing user manual. Refer to the manual for more information. The timestamp information is of type double and outputs the latest time at which a CAN message with the corresponding identifier has been received. This time information in seconds (with a resolution of 1 microsecond) can be used to implement time-out-logic within your model.

The pop-up menu lets you select which output information is output at each output port of the block. If **Data** is selected each output port signal is a scalar only. If **Data – Status** is selected each output port signal is a vector with two elements, where the first element contains the data frame and the second element the status information. If **Data – Status – Timestamp** is selected each output port signal is a vector with three elements, where the first element contains the data frame, the second element the status information, and the third element the timestamp.

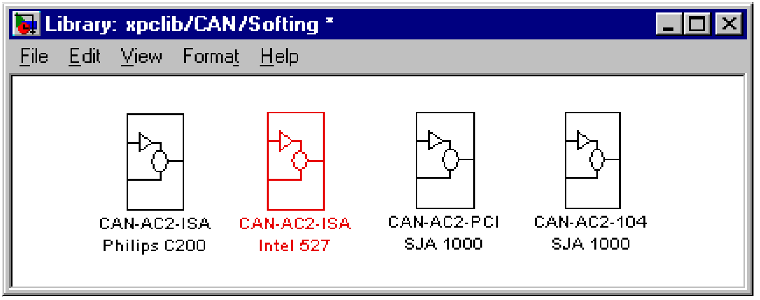
Generate interrupts — The fourth control (check box) lets you define if the CAN messages defined in this instance of the block will initiate an interrupt from the CAN board each time they are received. If checked this allows driving the model (target application) execution controlled by CAN messages.

Sample time — The fifth control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

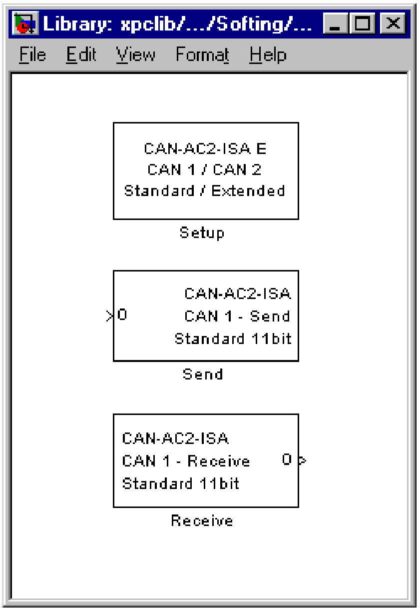
You can use as many instances of the Receive block in the model as needed. For example by using two instances of the block with different sample times, CAN messages can be retrieved at different rates. Or you can use multiple instances to structure your model more efficiently.

CAN Driver Blocks for the CAN-AC2 (ISA) with Intel 82527 CAN-Controller

The driver blocks described here support the CAN-AC2 (ISA) with piggyback modules. The Intel 82527 chip is used as the CAN-controller in this configuration and supports both Standard and Extended identifier ranges in parallel. The driver block set for this board is found in the xPC Target I/O block library in the group CAN/Softing.

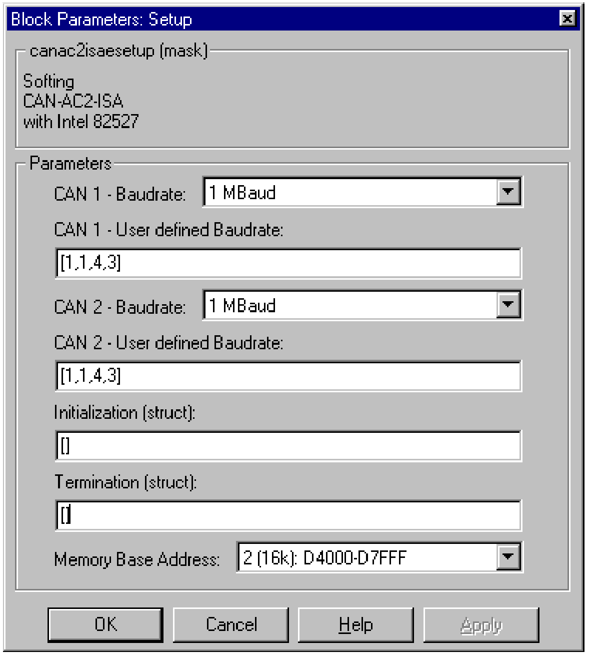


The second block group highlighted above contains the three available CAN blocks: Setup, Send, and Receive.



Setup Driver Block

The Setup block is used to define general settings of the plugged-in CAN board. Because the CAN driver blocks for this board only supports a single physical board for each target system, this block has to be used exactly once (one instance) in a model. The dialog box of the Setup block lets you define the following settings.



CAN 1 - Baud rate — The first control (pop-up menu) lets you define the most common baud rates for CAN port 1. If special timing is necessary (baud rate), the value “User defined” can be selected. In this case the second control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning:

```
    [ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 2 - Baud rate — The third control (pop-up menu) lets you define the most common baud rates for CAN port 1. If special timing is necessary (baud rate), the value “User defined” can be selected. In this case the fourth control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning

```
    [ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

Initialization and Termination — The fifth and sixth control (edit fields) can be used to define CAN messages sent during initialization and termination of the Setup block.

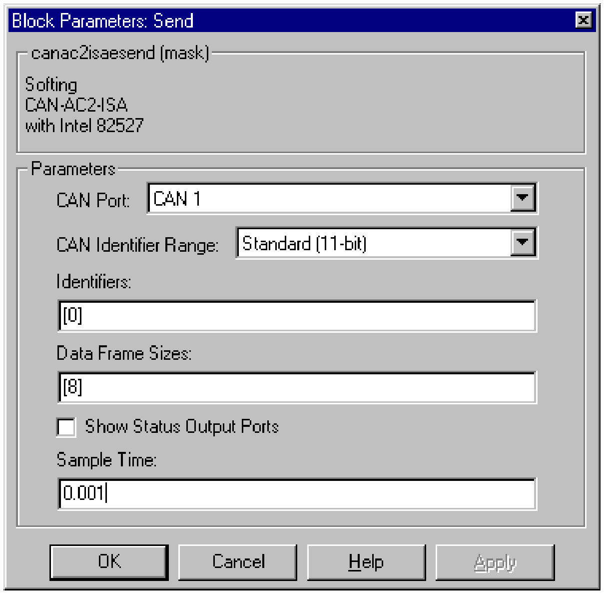
Memory base address — The seventh control (pop-up menu) is used to define the memory base address of the board. The address range used by the board has to be set by hardware jumpers on the board itself. Refer to the Softing user manual on how set the various address ranges. The setting in the dialog box has to correspond to the jumper setting otherwise the board cannot be accessed. The available address ranges (memory base address) in the pop-up menu are those supported by the board. Because the xPC Target kernel only reserves a sub range (C0000 – DC000) of the 640 kilobyte to 1 megabyte address range for memory mapped devices, the valid settings when used within a xPC target systems only are:

```
1 (16k): D0000-D3FFF
2 (16k): D4000-D7FFF
```

The board allows activating proper termination for each of the two CAN ports separately by means of hardware jumpers. Refer to the Softing user manual on how the jumpers have to be set. Both CAN ports have to be terminated properly when you use the provided loop-back model in order to test the board and drivers.

Send Driver Block

The Send driver block is used to transmit data to a CAN-network from within a block model.



The dialog box of the block lets you define the following settings.

CAN port — The first control (pop-up menu) is used to select at which CAN port the CAN message will be sent out.

CAN identifier range — The second control (pop-up menu) is used to select the identifier range of the CAN messages sent out by this block instance. If an application makes use of mixed Standard and Extended identifier ranges, at least two instances of this block have to be used, each defining the corresponding identifier range.

Identifiers — The third control (edit field) is used to define the identifiers of the CAN-messages sent out by this block. It has to be a row vector where the elements define a set of either Standard or Extended identifiers. Each element has to be in the range between 0 and 2031 for Standard identifiers or 0 and (2^29)-1 for Extended identifiers. The number of identifiers for each CAN port in a model per physical CAN-board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of inputs ports of the block. The block icon displays the selected identifier at each input port. Each input port accepts the data frame to be sent along with the CAN-message. The signal entering each input port has to be a scalar of type double representing the maximum size of 8 bytes of a CAN-message's data frame.

Data frame sizes — The fourth control (edit field) is used to define the data frame size for each identifier (CAN-message) in bytes. It has to be a row vector where the elements define a set of data frame sizes. Each element has to be in the range between 1 and 8. If the data frame sizes for all identifiers defined in the control above have to be the same, the size can be provided as a scalar only and scalar expansion applies. If the sizes are different for at least two identifiers (CAN-messages) one size element has to be provided for each identifier defined in the control above. Therefore the length of the two vectors have to be the same.

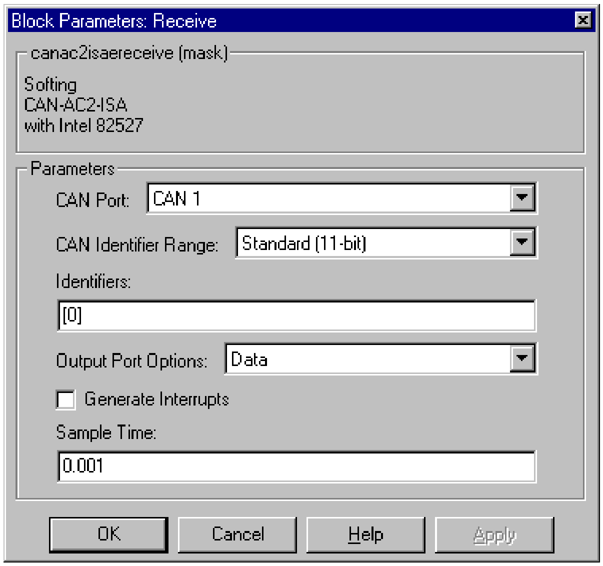
Show status: Output ports — The fourth control (check box) lets you enable status output ports for each identifier (CAN message). If the check box is checked, the block shows as many output ports as input ports. The data type of each output port is a double and the value is identical to the return argument of function CANPC_write_object(…) described in the Softing user manual. Refer to the manual for more information.

Sample time — The fifth control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

You can use as many instances of the Send block in the model as needed. For example by using two instances of the block, different sample times at which CAN messages are sent out can be defined. Or you can use multiple instances to structure your model more efficiently.

Receive Driver Block

The Receive driver block is used to retrieve data from a CAN-network to be used within a block model.



The dialog box of the block lets you define the following settings.

CAN port — The Receive driver block is used to retrieve data from a CAN network to be used within a block model. The first control (pop-up menu) is used to select from which CAN port, the CAN messages will be retrieved from.

CAN identifier range — The second control (pop-up menu) is used to select the identifier range of the CAN messages retrieved by this block instance. If an application makes use of mixed Standard and Extended identifier ranges, at least two instances of this block have to be used, each defining the corresponding identifier range.

Identifiers — The third control (edit field) is used to define the identifiers of the CAN messages retrieved by this block. It has to be a row vector where the elements define a set of either Standard or Extended identifiers. Each element has to be in the range between 0 and 2031 for Standard identifiers or 0 and 2^{29} - 1 for Extended identifiers. The number of identifiers for each CAN port in a model per physical CAN board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of output ports of the block. The block icon displays the selected identifier at each output port. Each output port will output the data frame being retrieved along with the CAN-message. The signal leaving each output port is a scalar of type double representing the maximum size of 8 bytes of a CAN-message's data frame.

Output port options — The fourth control (pop-up menu) lets you define which type of retrieved data is output at each output port. Three different types of data can be output, which are data frame, status and timestamp. The status information is of type double and is identical to the return value of function CANPC_read_rcv_data(…) described in the Softing user manual. Refer to the manual for more information. The timestamp information is of type double and outputs the latest time at which a CAN message with the corresponding identifier has been received. This time information in seconds (with a resolution of 1 microsecond) can be used to implement time-out-logic within your model.

The pop-up menu lets you select which output information is output at each output port of the block. If "Data" is selected each output port signal is a scalar only. If "Data – Status" is selected each output port signal is a vector with two elements, where the first element contains the data frame and the second element the status information. If "Data – Status – Timestamp" is selected each output port signal is a vector with three elements, where the first element contains the data frame, the second element the status information, and the third element the timestamp.

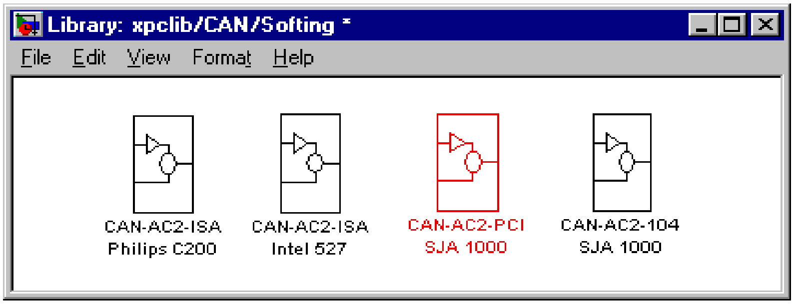
Generate interrupts — The fifth control (check box) lets you define if the CAN messages defined in this instance of the block will initiate an interrupt from the CAN board each time they are received. If checked this allows driving the model (target application) execution controlled by CAN messages.

Sample time — The sixth control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

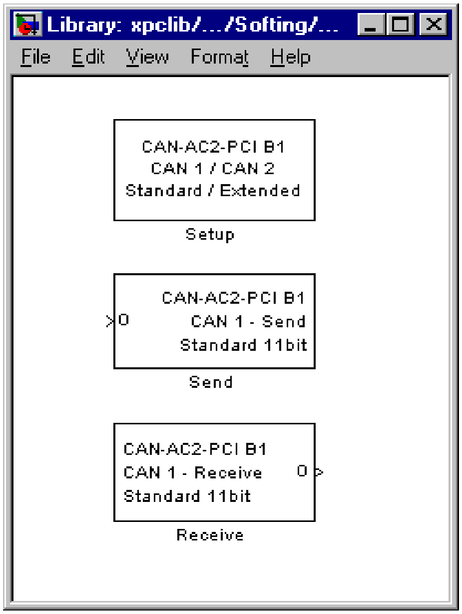
You can use as many instances of the Receive block in the model as needed. For example by using two instances of the block with different sample times, CAN messages can be retrieved at different rates. Or you can use multiple instances to structure your model more efficiently.

CAN Driver Blocks for the CAN-AC2-PCI with Philips SJA1000 CAN-Controller

The driver blocks described here support the CAN-AC2-PCI. The Philips SJA1000 chip is used as the CAN-controller in this configuration and supports both Standard and Extended identifier ranges in parallel. The driver block set for this board is found in the xPC Target I/O block library in the group CAN/Softing.

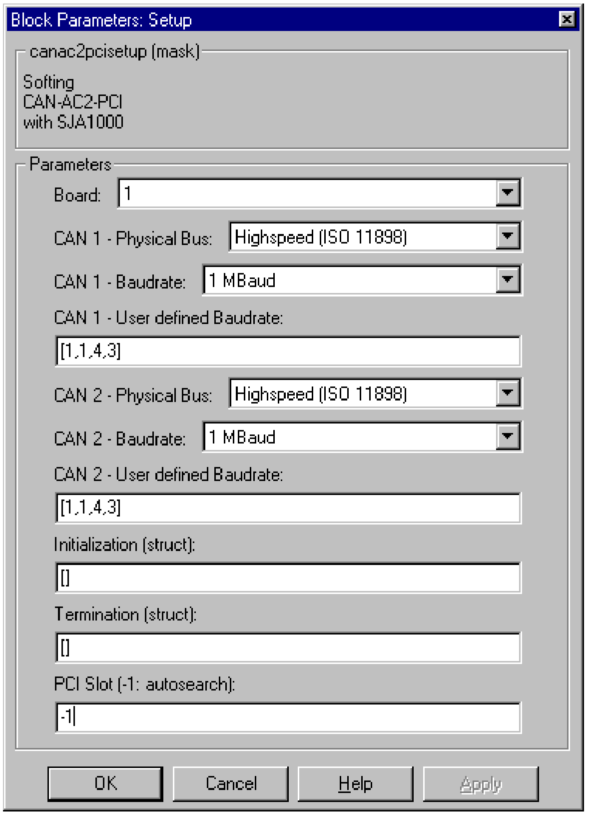


The third block group highlighted above contains the three available CAN blocks: Setup, Send, and Receive.



Setup Driver Block

The Setup block is used to define general settings of the plugged-in CAN board(s). The CAN driver blocks for this board support up to three boards for each target system what leads to the availability of up to six CAN ports. For each board in the target system exactly one Setup driver block has to be used in a model.



The dialog box of the Setup block lets you define the following settings.

Board — The first control (pop-up menu) lets you define which board is being accessed by this driver block instance. The board number (1…3) can be seen as a reference identifier in order to differentiate the boards if multiple boards are present in the target system. The physical board finally referenced by the board number depends on the PCI Slot edit field described further below. If just one board is present in the target system, board number 1 should be selected.

CAN 1 - Physical bus — The second control (pop-up menu) is used to define the physical CAN bus type of CAN port 1. In the board standard hardware configuration only Highspeed CAN is supported. By extending the board with Lowspeed CAN piggyback modules it is possible to additionally select Lowspeed CAN as the physical bus. The value of this control shouldn't be changed to Lowspeed if no module is present for the corresponding CAN port. If the module is present (see the Softing user manual on how to install the modules) you can select between Highspeed and Lowspeed CAN here.

CAN 1- Baud rate — The third control (pop-up menu) lets you define the most common baud rates for CAN port 1. If special timing is necessary (baud rate), the value “User defined” can be selected. In this case the fourth control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 2 - Physical bus — The fifth control (pop-up menu) is used to define the physical CAN bus type of CAN port 2. In the board’s standard hardware configuration only Highspeed CAN is supported. By extending the board with Lowspeed CAN piggyback modules it is possible to additionally select Lowspeed CAN as the physical bus. The value of this control shouldn’t be changed to Lowspeed if no module is present for the corresponding CAN port. If the module is present (see the Softing user manual on how to install the modules) you can select between Highspeed and Lowspeed CAN here.

CAN 2 - Baud rate — The sixth control (pop-up menu) lets you define the most common baud rates for CAN port 2. If special timing is necessary (baud rate), the value “User defined” can be selected. In this case the seventh control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

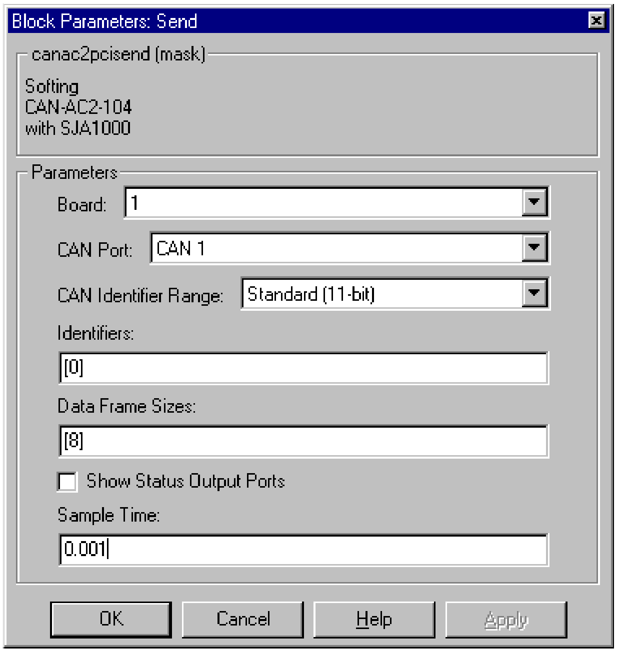
Initialization and Termination — The eighth and ninth control (edit fields) can be used to define CAN messages sent during initialization and termination of the Setup block.

PCI slot (-1: autosearch) — The tenth control (edit field) is used to define the PCI slot in which the referenced board (board number) resides. If only one board is present in the target system the value for this control should be –1 (autosearch). This value makes sure that the xPC Target kernel automatically finds the board independently of the PCI slot it is plugged into. If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. Use the xPC Target function 'getxpcpci' to query the target system for installed PCI boards and the PCI slots they are plugged into. For more information see 'help getxpcpci'.

The board allows activating proper termination for each of the two CAN ports separately by means of DIP-switches at the rear panel of the board. Refer to the Softing user manual on how the DIP-switches have to be set. Both CAN ports have to be terminated properly when you use the provided loop-back model in order to test the board and drivers.

Send Driver Block

The Send driver block is used to transmit data to a CAN-network from within a block model.



The dialog box of the block lets you define the following settings.

Board — The first control (pop-up menu) lets you define which physically present board is used to send out the CAN-messages defined by this block instance. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

CAN Port — The second control (pop-up menu) is used to select at which CAN port the CAN message will be sent out.

CAN Identifier range — The third control (pop-up menu) is used to select the identifier range of the CAN-messages sent out by this block instance. If an application makes use of mixed Standard and Extended identifier ranges, at

least two instances of this block have to be used, each defining the corresponding identifier range.

Identifiers — The fourth control (edit field) is used to define the identifiers of the CAN-messages sent out by this block. It has to be a row vector where the elements define a set of either Standard or Extended identifiers. Each element has to be in the range between 0 and 2031 for Standard identifiers or 0 and (2^29)-1 for Extended identifiers. The number of identifiers for each CAN port in a model per physical CAN board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of inputs ports of the block. The block icon displays the selected identifier at each input port. Each input port accepts the data frame to be sent along with the CAN message. The signal entering each input port has to be a scalar of type double representing the maximum size of 8 bytes of a CAN message's data frame.

Data frame size — The fifth control (edit field) is used to define the data frame size for each identifier (CAN-message) in bytes. It has to be a row vector where the elements define a set of data frame sizes. Each element has to be in the range between 1 and 8. If the data frame sizes for all identifiers defined in the control above have to be the same, the size can be provided as a scalar only and scalar expansion applies. If the sizes are different for at least two identifiers (CAN messages) one size element has to be provided for each identifier defined in the control above. Therefore the length of the two vectors have to be the same.

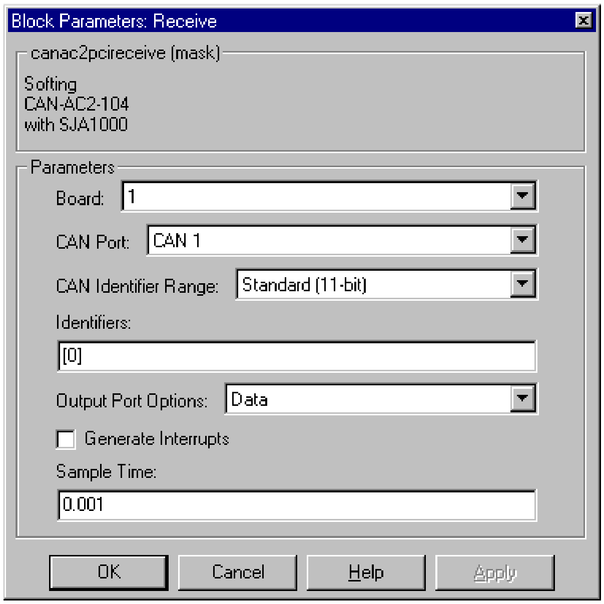
Show status output ports — The sixth control (check box) lets you enable status output ports for each identifier (CAN-message). If the check box is checked the block shows as many output ports as input ports. The data type of each output port is a double and the value is identical to the return argument of function CANPC_write_object(…) described in the Softing user manual. Refer to the manual for more information.

Sample time — The seventh control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

You can use as many instances of the Send block in the model as needed. For example by using two instances of the block, different sample times at which CAN-messages are sent out can be defined. Or you can use multiple instances to structure your model more efficiently.

Receive Driver Block

The Receive driver block is used to retrieve data from a CAN-network to be used within a block model. You can use as many instances of the Receive block in the model as needed. For example by using two instances of the block with different sample times, CAN messages can be retrieved at different rates. Or you can use multiple instances to structure your model more efficiently.



Board — The dialog box of the block lets you define the following settings.

The first control (pop-up menu) lets you define from which physically present board the CAN messages defined by this block instance are retrieved from. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

CAN Port — The second control (pop-up menu) is used to select from which CAN port, the CAN messages will be retrieved from.

CAN Identifier range — The third control (pop-up menu) is used to select the identifier range of the CAN-messages retrieved by this block instance. If an application makes use of mixed Standard and Extended identifier ranges, at

least two instances of this block have to be used, each defining the corresponding identifier range.

Identifiers — The fourth control (edit field) is used to define the identifiers of the CAN-messages retrieved by this block. It has to be a row vector where the elements define a set of either Standard or Extended identifiers. Each element has to be in the range between 0 and 2031 for Standard identifiers or 0 and 2^{29} - 1 for Extended identifiers. The number of identifiers for each CAN port in a model per physical CAN board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of output ports of the block. The block icon displays the selected identifier at each output port. Each output port will output the data frame being retrieved along with the CAN message. The signal leaving each output port is a scalar of type double representing the maximum size of 8 bytes of a CAN message's data frame.

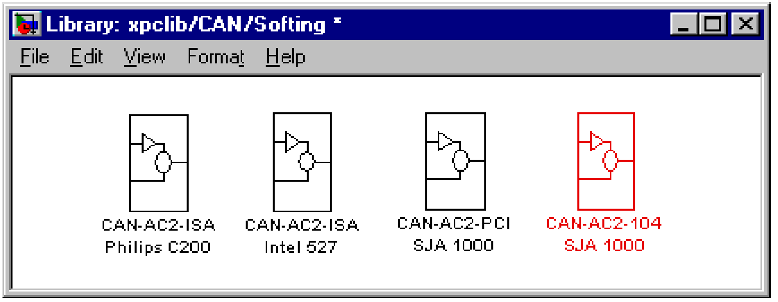
Output port options — The fifth control (pop-up menu) lets you define which type of retrieved data is output at each output port. Three different types of data can be output, which are data frame, status and timestamp. The status information is of type double and is identical to the return value of function CANPC_read_rcv_data (…) described in the Softing user manual. Refer to the manual for more information. The timestamp information is of type double and outputs the latest time at which a CAN message with the corresponding identifier has been received. This time information in seconds (with a resolution of 1 microsecond) can be used to implement time-out-logic within your model. The pop-up menu lets you select which output information is output at each output port of the block. If **Data** is selected each output port signal is a scalar only. If **Data – Status** is selected each output port signal is a vector with two elements, where the first element contains the data frame and the second element the status information. If **Data – Status – Timestamp** is selected each output port signal is a vector with three elements, where the first element contains the data frame, the second element the status information, and the third element the timestamp.

Generate interrupts — The sixth control (check box) lets you define if the CAN messages defined in this instance of the block will initiate an interrupt from the CAN board each time they are received. If checked this allows driving the model (target application) execution controlled by CAN messages.

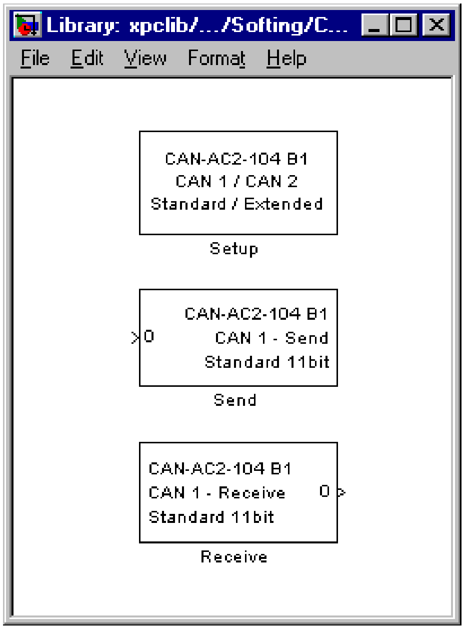
Sample time — The seventh control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

CAN Driver Blocks for the CAN-AC2-104 (PC/104) with Philips SJA1000 CAN-Controller

The driver blocks described here support the CAN-AC2-104 (PC/104). The Philips SJA1000 chip is used as the CAN-controller in this configuration and supports both Standard and Extended identifier ranges in parallel. The driver block set for this board is found in the xPC Target I/O block library in the group CAN/Softing.

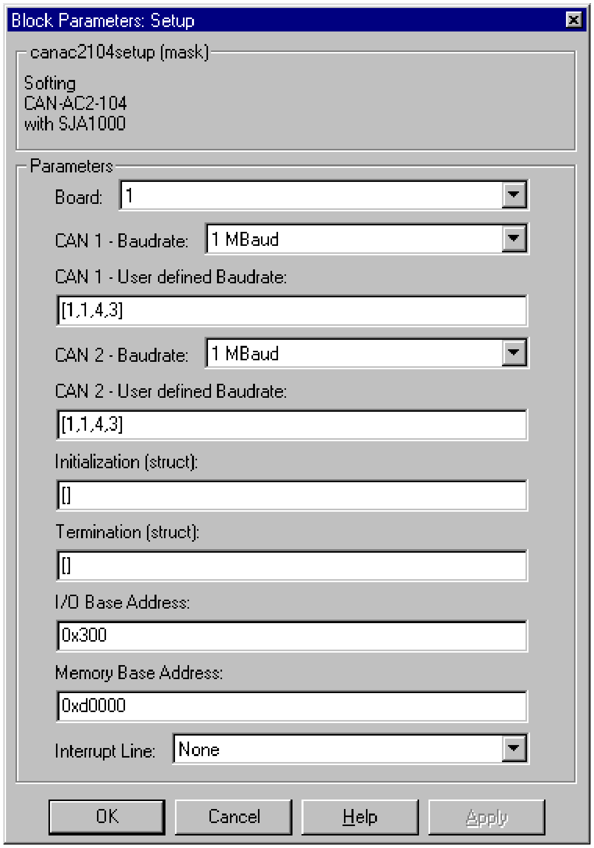


The fourth block group highlighted above contains the three available CAN blocks: Setup, Send, and Receive.



Setup Driver Block

The Setup block is used to define general settings of the stacked CAN board(s). The CAN driver blocks for this board support up to three boards for each target system what leads to the availability of up to six CAN ports. For each board in the target system exactly one Setup driver block has to be used in a model.



The dialog box of the Setup block lets you define the following settings.

Board — The first control (pop-up menu) lets you define which board is being accessed by this driver block instance. The board number (1…3) can be seen as a reference identifier in order to differentiate the boards if multiple boards are present in the target system. The physical board finally referenced by the board number depends on the I/O Base Address edit field described further below. If just one board is present in the target system, board number 1 should be selected.

CAN 1 - Baud rate — The second control (pop-up menu) lets you define the most common baud rates for CAN port 1. If special timing is necessary (baud rate), the value **CAN 1 - User defined baud rate** can be selected. In this case the third control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 1 - Baud rate — The fourth control (pop-up menu) lets you define the most common baud rates for CAN port 2. If special timing is necessary (baud rate), the value “User defined” can be selected. In this case the fifth control (edit field) is used to provide the four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

Initialization and **Termination** — The sixth and seventh control (edit fields) can be used to define CAN messages sent during initialization and termination of the Setup block.

I/O Base address — The eighth control (edit field) is used to define the I/O base address of the board to be accessed by this block instance. The I/O Base address is given by the DIP-switch setting on the board itself. The I/O address range is 3 bytes and is mainly used to transfer the information which memory base address the board should use. See the Softing user manual for this board on how the I/O base address can be set. The I/O base address entered in this control has to correspond with the DIP-switch setting on the board. If more than one board is present in the target system a different I/O base address has to be entered for each board. In this case the I/O base address itself defines which board is referenced by which board number.

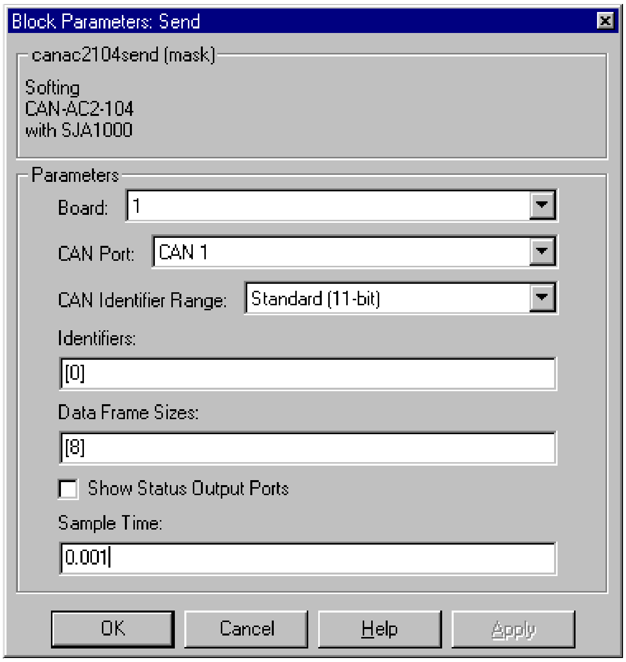
Memory base address — The ninth control (edit field) is used to define the memory base address of the board to be accessed by this block instance. The memory base address is a software setting only (no corresponding DIP-switch is found on the board). The memory address range is 64 kilobytes. If more than one board is present in the target system a different memory base address has to be entered for each board and you have to make sure that the defined address ranges do not overlap. Because the xPC Target kernel only reserves a subset of the address range between 640 kilobytes and 1megabytes for memory mapped devices the address ranges have to lie within the following range:

```
C0000    DC000
```

The board allows activating proper termination for each of the two CAN ports separately by means of jumpers found on the board. Refer to the boards user manual on how the DIP-switches have to be set. Both CAN ports have to be terminated properly when you use the provided loop-back model in order to test the board and drivers.

Send Driver Block

The Send driver block is used to transmit data to a CAN-network from within a block model.



The dialog box of the block lets you define the following settings.

Baud — The first control (pop-up menu) lets you define which physically present board is used to send out the CAN messages defined by this block instance. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

CAN Port — The second control (pop-up menu) is used to select at which CAN port the CAN message will be sent out.

CAN Identifier range — The third control (pop-up menu) is used to select the identifier range of the CAN-messages sent out by this block instance. If an application makes use of mixed Standard and Extended identifier ranges, at

least two instances of this block have to be used, each defining the corresponding identifier range.

Identifiers — The fourth control (edit field) is used to define the identifiers of the CAN messages sent out by this block. It has to be a row vector where the elements define a set of either Standard or Extended identifiers. Each element has to be in the range between 0 and 2031 for Standard identifiers or 0 and 2^{29} - 1 for Extended identifiers. The number of identifiers for each CAN port in a model per physical CAN-board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of inputs ports of the block. The block icon displays the selected identifier at each input port. Each input port accepts the data frame to be sent along with the CAN message. The signal entering each input port has to be a scalar of type double representing the maximum size of 8 bytes of a CAN message's data frame.

Data frame sizes — The fifth control (edit field) is used to define the data frame size for each identifier (CAN-message) in bytes. It has to be a row vector where the elements define a set of data frame sizes. Each element has to be in the range between 1 and 8. If the data frame sizes for all identifiers defined in the control above have to be the same, the size can be provided as a scalar only and scalar expansion applies. If the sizes are different for at least two identifiers (CAN messages) one size element has to be provided for each identifier defined in the control above. Therefore the length of the two vectors have to be the same.

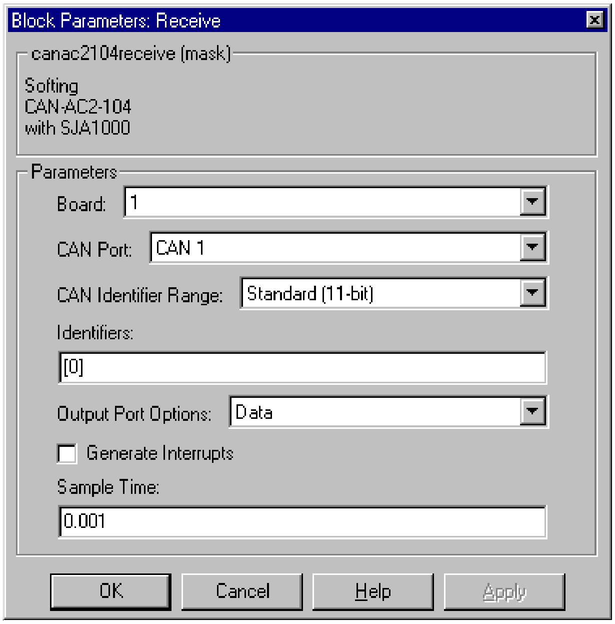
Show status output ports — The sixth control (check box) lets you enable status output ports for each identifier (CAN message). If the check box is checked the block shows as many output ports as input ports. The data type of each output port is a double and the value is identical to the return argument of function CANPC_write_object(…) described in the Softing user manual. Refer to the manual for more information.

Sample time — The seventh control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

You can use as many instances of the Send block in the model as needed. For example by using two instances of the block, different sample times at which CAN messages are sent out can be defined. Or you can use multiple instances to structure your model more efficiently.

Receive Driver Block

The Receive driver block is used to retrieve data from a CAN-network to be used within a block model. You can use as many instances of the Receive block in the model as needed. For example by using two instances of the block with different sample times, CAN messages can be retrieved at different rates. Or you can use multiple instances to structure your model more efficiently.



The dialog box of the block lets you define the following settings.

Board — The first control (pop-up menu) lets you define from which physically present board the CAN messages defined by this block instance are retrieved from. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

CAN Port — The second control (pop-up menu) is used to select from which CAN port, the CAN messages will be retrieved from.

CAN Identifier range — The third control (pop-up menu) is used to select the identifier range of the CAN messages retrieved by this block instance. If an application makes use of mixed Standard and Extended identifier ranges, at

least two instances of this block have to be used, each defining the corresponding identifier range.

Identifiers — The fourth control (edit field) is used to define the identifiers of the CAN messages retrieved by this block. It has to be a row vector where the elements define a set of either Standard or Extended identifiers. Each element has to be in the range between 0 and 2031 for Standard identifiers or 0 and 2^{29} - 1 for Extended identifiers. The number of identifiers for each CAN port in a model per physical CAN board cannot exceed 200 (restriction of the firmware's dynamic object mode). The number of elements defined here, define at the same time the number of output ports of the block. The block icon displays the selected identifier at each output port. Each output port will output the data frame being retrieved along with the CAN message. The signal leaving each output port is a scalar of type double representing the maximum size of 8 bytes of a CAN-message's data frame.

Output port options — The fifth control (pop-up menu) lets you define which type of retrieved data is output at each output port. Three different types of data can be output, which are data frame, status and timestamp. The status information is of type double and is identical to the return value of function CANPC_read_rcv_data(…) described in the Softing user manual. Refer to the manual for more information. The timestamp information is of type double and outputs the latest time at which a CAN message with the corresponding identifier has been received. This time information in seconds (with a resolution of 1 microsecond) can be used to implement time-out-logic within your model.

The pop-up menu lets you select which output information is output at each output port of the block. If "Data" is selected each output port signal is a scalar only. If "Data – Status" is selected each output port signal is a vector with two elements, where the first element contains the data frame and the second element the status information. If "Data – Status – Timestamp" is selected each output port signal is a vector with three elements, where the first element contains the data frame, the second element the status information, and the third element the timestamp.

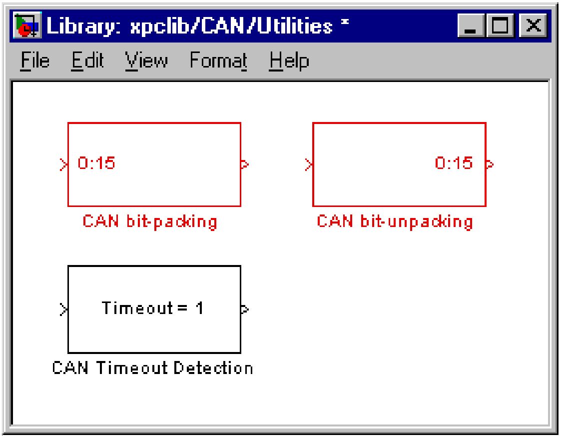
Generate interrupts — The sixth control (check box) lets you define if the CAN messages defined in this instance of the block will initiate an interrupt from the CAN board each time they are received. If checked this allows driving the model (target application) execution controlled by CAN messages.

Sample time — The seventh control (edit field) defines the sample time at which the Send block is executed during a model (target application) run.

Constructing and Extracting CAN Data Frames

CAN data frames have a maximum size of 8 bytes (64 bits). For the CAN driver blocks found in the xPC Target I/O block library, Simulink signals of data type double are used to propagate data frames as an entity. But in most applications the data frame content does not consist of 64-bit floating point values, instead they are constructed from one or more smaller data type entities like signed and unsigned integers of various size.

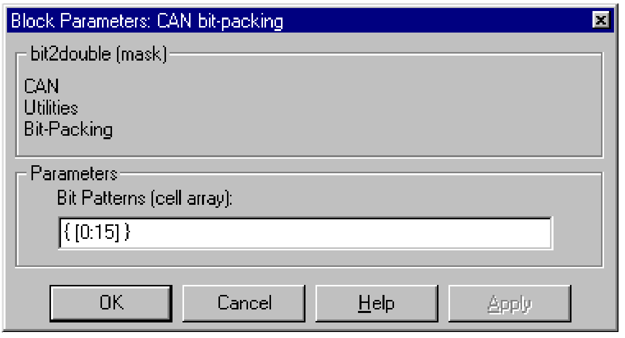
In order to simplify the construction and extraction of data frames for the user, the xPC Target I/O library contains two utility blocks (found in subgroup CAN/Utilities) which allow bit-packing (construction) and bit-unpacking (extraction) of data frames in a very flexible way.



The main purpose of the two blocks is to be used in conjunction with CAN Send and Receive driver blocks, but they can be used as well for other types of data manipulation. Their functionality is entirely independent of any CAN driver blocks or CAN library.

CAN Bit-Packing Block

This block is used to construct CAN data frames and its output port is normally connected to an input port of a CAN Send driver block. The block has one output port of data type double (a scalar) which represents the data frame entity constructed by the signals entering the block at its input ports. The number of input ports and the data type of each input port depends on the setting in the blocks dialog box.



The dialog box contains one single control (edit field) which lets you define the bit patterns in a flexible way. The data type entered in the control has to be a MATLAB cell array vector. The number of elements in the cell array define the number of input ports shown by this block instance. The cell array elements have to be of type double array and define where each bit of the incoming value (data typed input port) comes to lie at what position in the outgoing double value (data frame).

From a data type perspective (input ports) the block behaves like a Simulink sink block and therefore the data types of the input ports are inherited from the driving blocks.

The sample time of the block is also inherited from the driving blocks. Therefore no explicit sample time has to be provided in the block's dialog box.

The functionality of the block is easiest explained by means of an example.

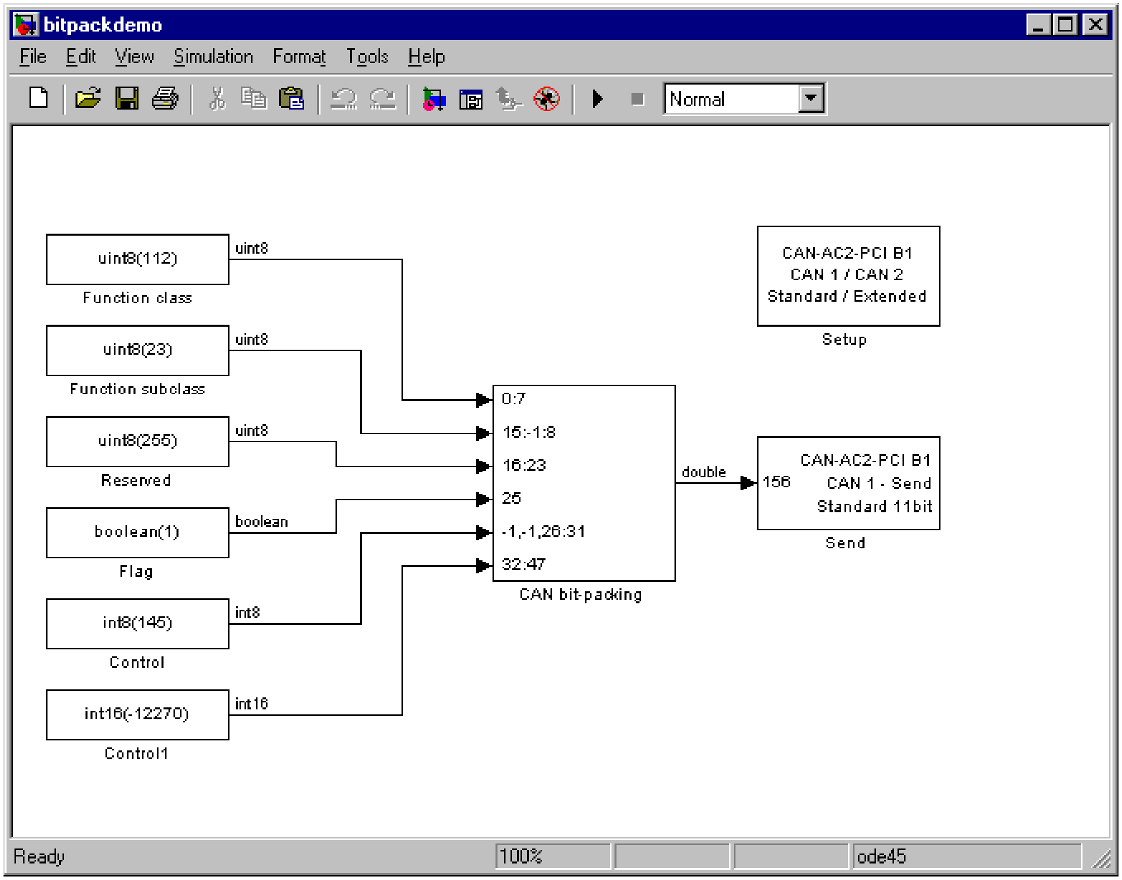
We assume that a node on the CAN network needs to receive a CAN message with identifier 156 having the following data frame content. The data frame has to be 6 bytes long.

Byte 0	Function class of type uint8
Byte 1	Function subclass of type uint8 with reversed bit order
Byte 2	Reserved, all bits have to be 1
Byte 3	Bit 0 has to be 0, Bit 1 has to be a boolean (flag), bits 2 to 7 have to be bit 2 to 7 of an incoming int8 value (control)
Byte 4 and 5	Value of type int16

The bit pattern cell array, which bit-packs the data frame according to the above specification, can look as follows.

```
{ [0:7] , [15:-1:8] , [16:23] , [25] , [-1,-1,26:31] , [32:47] }
```

And the Simulink model simulating the needed behavior would be as show below.



Let us analyze the model.

The first input is the Function class of type uint8, which has an example value of 112. This value has to become byte 0 (bits 0 to 7) of the data frame. Therefore the first bit (element 1 of double array [0:7]) has to get bit 0 of the data frame, the second bit 1 and so on. It is easiest to define this mapping by the MATLAB colon operator:.

The second input is the Function subclass of type uint8, which has an example value of 23. This value has to become byte 1 (bits 8:15) of the data frame but in reversed bit order. Therefore the first bit (element 1 of double array [15:-1:8]) has to get bit 15, the second bit 14 and so on. It is easiest to define this mapping by the MATLAB colon operator: and an increment of –1.

The third input is only necessary because the reserved byte 2 has to have all bits set to 1. If a bit position in the outgoing data frame isn't referenced by a bit pattern array element, the bit will be by default 0, but there is no construct to have them set to 1 as the default. Therefore a uint8 constant with value 255 has to be externally brought in. The constant 255 has to get to bit position 16 to 23 (byte 2) of the outgoing data frame.

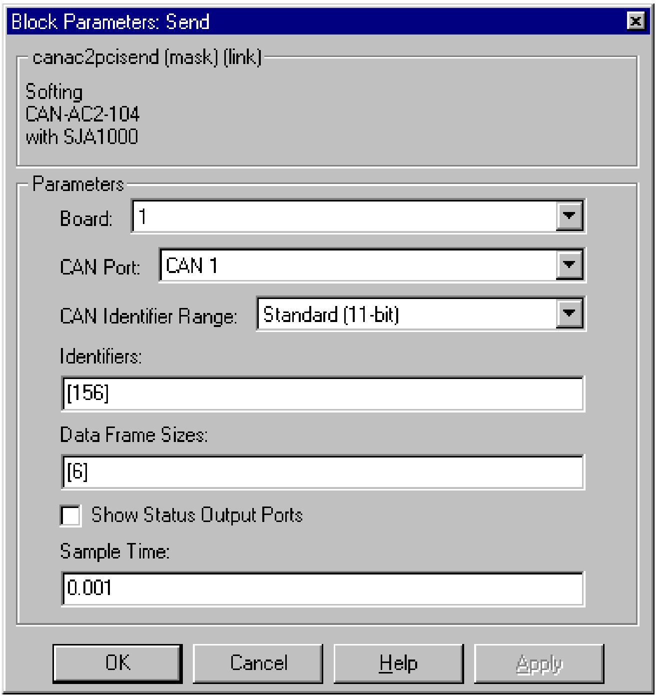
Because bit 0 of data frame byte 3 (bit 24) has to be 0 and 0 is the default bit value if not referenced by a bit pattern array element, no explicit action has to be taken here.

The fourth input is the Flag of type boolean, which has an example value of 1. This value has to become bit 1 of byte 3 (bit 25) of the data frame. Therefore the single bit (element 1 of double array [25]) has to get bit 25 of the data frame.

The fifth input is the Control of type int8, which has an example value of 121. But only bits 2 to 7 have to be mapped into the outgoing data frame or in other words bits 0 and 1 have to be thrown away. Because indexing of incoming values always starts with the first bit (bit 0) a special indexing value (-1) has to be used in order to skip bit 0 and 1 of the incoming int8 value. Bits 2 to 7 will be directly mapped to bit 2 to 7 of byte 3 (bits 26 to 31) of the outgoing data frame. This leads to the following bit pattern: [-1,-1,26:31]

The sixth input is the Value of type int16, which has an example value of –12270. This value has to become byte 4 and 5 (bits 32 to 47) of the outgoing data frame. Therefore the first bit (element 1 of double array [32:47]) has to get bit 32 of the data frame, the second bit 33 and so on. It is easiest to define this mapping by the MATLAB colon operator:.

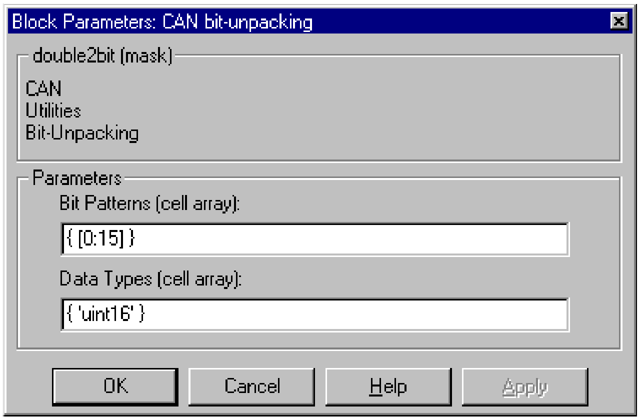
The output of the block then consists of a double value representing the packed data types within the first 6 bytes. The last two bytes are zero. This means that even in the case were less than 8 bytes are significant, the CAN data frame is always represented by a double value (8 bytes). The value of the constructed floating point double doesn't have any particular meaning but still can be inspected by a numerical display.



The data frame is then propagated to the CAN Send driver block and is sent out as part of a CAN-message having identifier 156. When looking at the Send block's dialog box, the data frame size is defined as 6 bytes. This makes sure that only the first 6 bytes of the incoming double value are transmitted as part of the CAN-message.

CAN Bit-Unpacking Block

This block is used to extract CAN data frames and its input port is normally connected to an output port of a CAN Receive driver block. The block has one input port of data type double (a scalar) which represents the data frame entity from which the signals are extracted and leaving the block at its output ports. The number of output ports and the data type of each output port depends on the settings in the blocks dialog box.



The dialog box contains two controls (edit fields).

The first lets you define the bit patterns in a flexible way. The data type entered in the control has to be a MATLAB cell array vector. The number of elements in the cell array define the number of output ports shown by this block instance. The cell array elements have to be of type double array and define where the bits of the incoming double value (data frame) come to lie at what position in the output port values (data typed).

From a data type perspective (output ports) the block behave like a Simulink source block and therefore the data types of the output ports have to be defined in the second control (edit field). The data type entered in that control has to be a MATLAB cell array vector of the same length as the bit pattern cell array. The cell array elements have to be of type char and define the data type of the corresponding output port. The following values are supported:

```
boolean, int8, uint8, int16, uint16, int32, and uint32
```

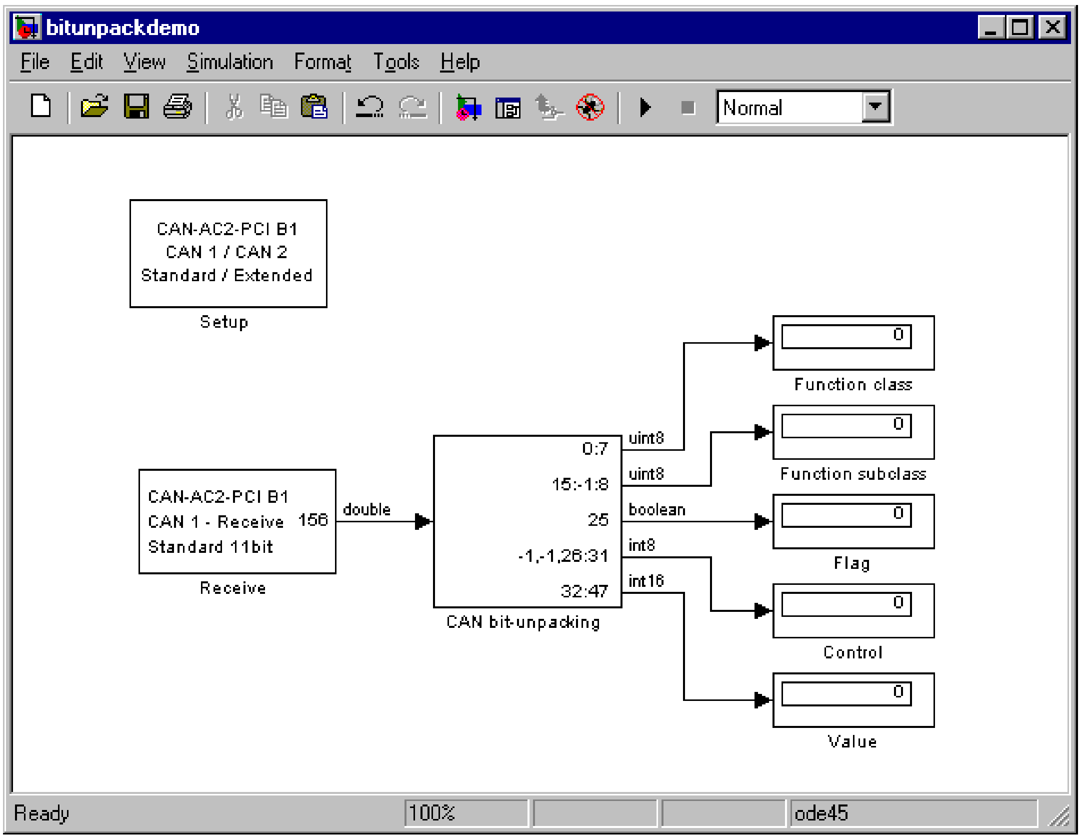
The sample time of the block is inherited from the driving block. Therefore no explicit sample time has to be provided in the block's dialog box.

The functionality of the block is easiest explained by means of an example. We take the same example as used above to demonstrate the functionality of the bit-packing block. But in this case, the data frame is sent by an external CAN node and is received by the target application running on an xPC Target system. Therefore the bit-unpacking block has to be used in order to extract the various data fields out of the entire data frame. Because the bit pattern definition of the packing and unpacking block are symmetric, the bit pattern definition could look exactly the same. There is one simple optimization possible: We don't have to extract byte 2 (reserved area), because its content is known. The bit pattern edit field can therefore look as follows,

```
{ [0:7] , [15:-1:8] , [25] , [-1,-1,26:31] , [32:47] }
```

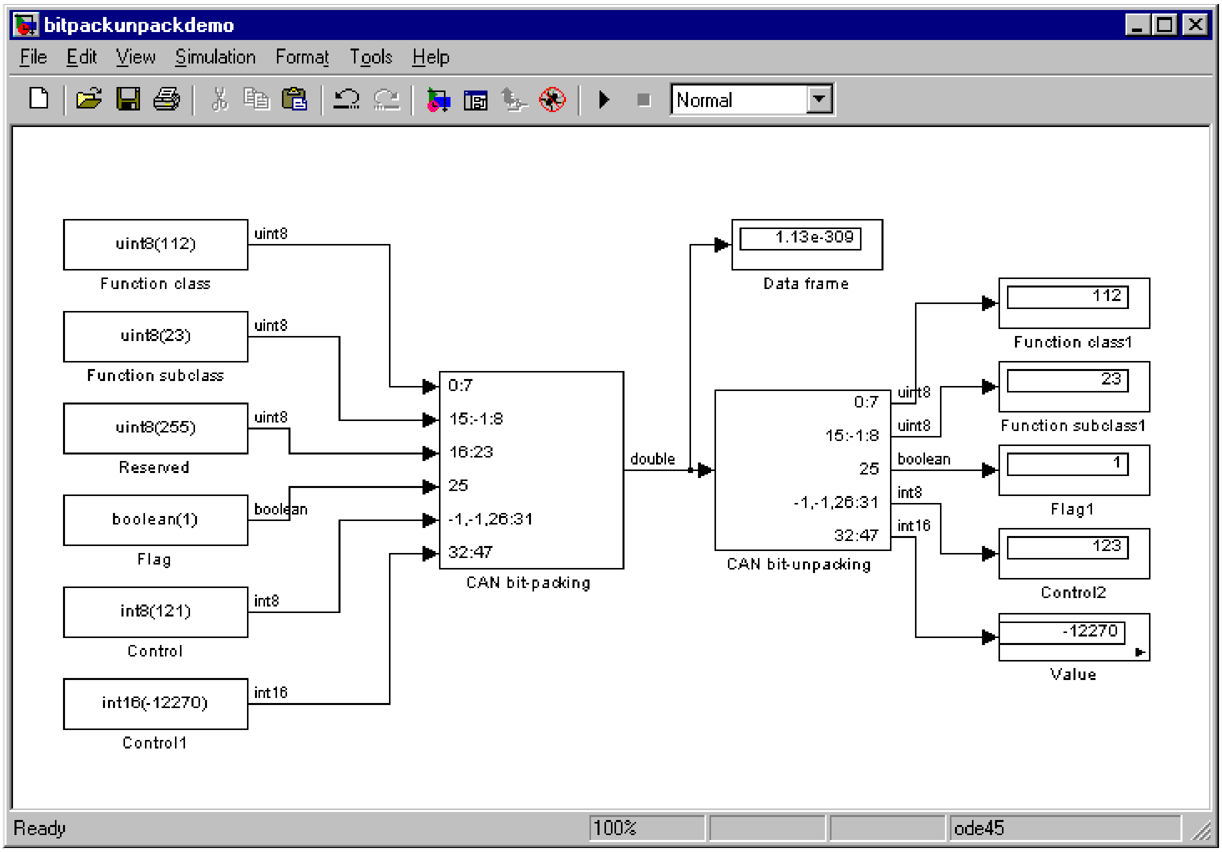
and the data type edit field as

```
{ 'uint8' , 'uint8' , 'boolean' , 'int8' , 'int16' }
```



This leads to the following Simulink model.

In many cases it makes sense to test the proper bit-packing and bit-unpacking operations in a Simulink model (simulation) before building the target application. Both blocks are working the same way either in Simulink or within the generated code. By combining the two models shown so far we get to a third one which can be used to simulate the behavior.

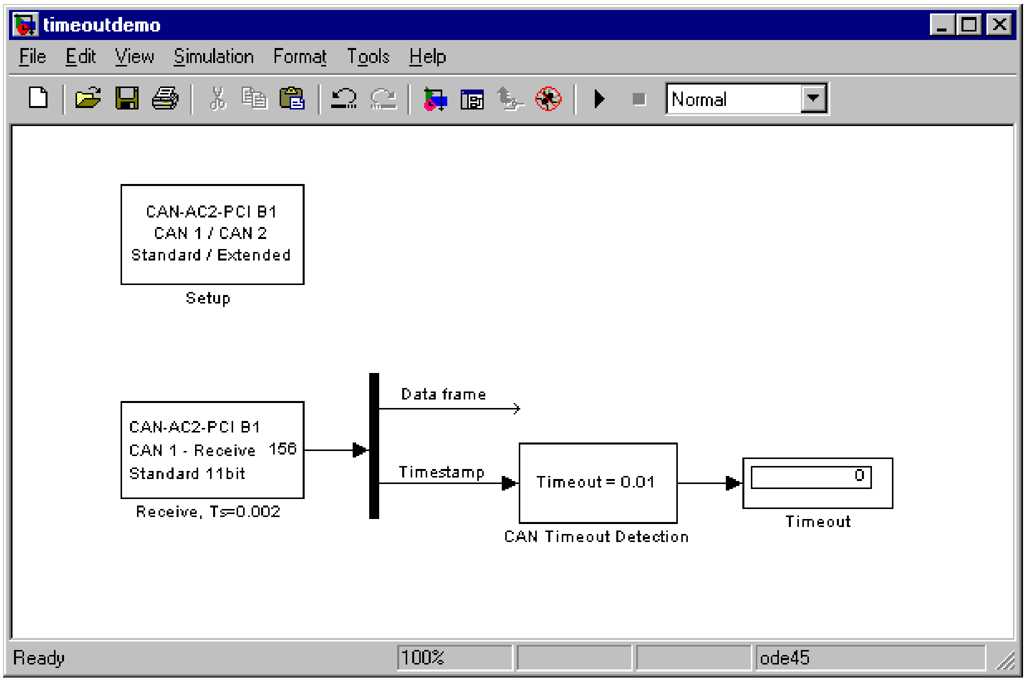


Detecting Time-outs When Receiving CAN Messages

The Receive driver blocks for all CAN boards allow to output the timestamp at which the latest corresponding CAN message has been received. This information can be used to detect if another CAN node is still alive and therefore is sending CAN messages or is no longer alive and special action has to be taken. Assume that we expect a CAN message from another CAN node every 2 ms. If no new message is received within 10 ms the other CAN node is being considered faulty and the Simulink model (target application) has to proceed accordingly.

The CAN blockset in the xPC Target I/O block library provides a utility block called CAN Time-out Detection. This is a simple graphical subsystem (inspect it by looking under its mask) which uses the timestamp information to calculate the time-out condition.

A Simulink model using this block in conjunction with a Receive block could look as follows.



The dialog box of the CAN Time-out Detection block has one edit field and lets you define the time-out value in seconds. The output of the block will be 0 if no time-out has been detected and 1 otherwise. See as well the loop-back example for the CAN-AC2-PCI and CAN-AC2-104 boards (xpccanpci and xpccanpc104) which make use of this utility block as well.

Model Execution Driven by CAN-Messages (Interrupt Capability of CAN Receive Blocks)

In certain application it is necessary that the model (target application) execution is driven by the pace of an incoming CAN message. The standard behavior of the xPC Target kernel is to drive the model (target application) in time monotonic fashion (time interrupt), but allows to replace the driving interrupt by any other hardware interrupt. Because the three supported CAN boards allow to fire a hardware interrupt upon reception of a specific CAN message, the timer interrupt line in the kernel can be replaced by the interrupt line assigned to a CAN board. This leads to a CAN message driven execution of the target application.

To set this up, two independent steps are necessary.

1 The timer interrupt line in the kernel setup has to be replaced by the board's hardware interrupt line.

2 The CAN Setup and CAN Receive blocks have to be properly set up.

Both steps are slightly different for each of the three supported CAN boards. Therefore the two steps are explained for each board type below.

CAN-AC2 (ISA)

The CAN-AC2 is an ISA-board, and the hardware interrupt line is set by means of hardware jumpers on the board. Refer to the Softing user manual of the board on how to set a certain interrupt line. Select an interrupt line, which is not used by any other hardware device in the xPC Target system (for example by the Ethernet card).

1 In the Simulink window, and from the **Tools** menu, point to **Real-Time Workshop**, and then click **Options**. Select the category "xPC Target code generations options" in the displayed dialog box. In the "Real-Time Interrupt Source" pop-up menu select the interrupt line number which you have set by the jumpers on the board. Close the dialog box and save the model.

2 Open the dialog box of the CAN Receive block in the model which defines the CAN message (identifier) to be used to fire the interrupt. Check the

Generate interrupts check box. Checking this box will declare all CAN-messages defined in this Receive block instance through their identifiers as messages, which will fire an interrupt. Or in other words it is not possible to define a single CAN message within the set of defined identifiers to be the only one to fire an interrupt. In most cases only the reception of one specific message is used to drive the application execution. Therefore use at least two instances of the Receive block. One to receive the CAN message, which drives the execution (Generate Interrupts checked) and the other for all other "normal" CAN-messages to be received (Generate Interrupts unchecked).

CAN-AC2-PCI

The CAN-AC2 is a PCI-board, and the hardware interrupt line is automatically assigned by the PCI BIOS during the boot-up of the target system. Use the xPC Target function 'getxpcpci' (see 'help getxpcpci') at the MATLAB command prompt to query the target system for installed PCI devices and the assigned resources. Write down the interrupt line number assigned to the CAN-AC2-PCI board.

1 In the Simulink window, and from the **Tools** menu, point to **Real-Time Workshop**, and then click **Options**. Select the category "xPC Target code generations options" in the displayed dialog box. In the "Real-Time Interrupt Source" pop-up menu select the interrupt line number which you have retrieved by calling 'getxpcpci'. Close the dialog box and save the model.

2 Open the dialog box of the CAN Receive block in the model which defines the CAN-message (identifier) to be used to fire the interrupt. Check the **Generate interrupts** check box. Checking this box will declare all CAN-messages defined in this Receive block instance through their identifiers as messages, which will fire an interrupt. Or in other words it is not possible to define a single CAN-message within the set of defined identifiers to be the only one to fire an interrupt. In most cases only the reception of one specific message is used to drive the application execution. Therefore use at least two instances of the Receive block. One to receive the CAN-message, which drives the execution (Generate Interrupts checked) and the other for all other "normal" CAN-messages to be received (Generate Interrupts unchecked).

CAN-AC2-104 (PC/104)

The CAN-AC2-104 is an ISA-board (PC/104), and the hardware interrupt line is set by means of a software setting within the CAN Setup driver block. Write down a free interrupt line, which is not used by any other hardware device in the xPC target system (for example by the Ethernet card).

1 In the Simulink window, and from the **Tools** menu, point to **Real-Time Workshop**, and then click **Options**. Select the category “xPC Target code generations options” in the displayed dialog box. In the “Real-Time Interrupt Source” pop-up menu select the interrupt line number which you have chosen.

2 In the model open the dialog box of the CAN Setup block for the CAN-AC2-104 board. Select the chosen interrupt line in the “Interrupt Line” pop-up menu and close the dialog box. Open the dialog box of the CAN Receive block in the model which defines the CAN-message (identifier) to be used to fire the interrupt. Check the “Generate interrupts” check box. Checking this box will declare all CAN-messages defined in this Receive block instance through their identifiers as messages, which will fire an interrupt. Or in other words it is not possible to define a single CAN-message within the set of defined identifiers to be the only one to fire an interrupt. In most cases only the reception of one specific message is used to drive the application execution. Therefore use at least two instances of the Receive block. One to receive the CAN-message, which drives the execution (Generate Interrupts checked) and the other for all other “normal” CAN-messages to be received (Generate Interrupts unchecked).

After having completed the two steps the model is ready to be built. After the downloading has succeeded and the target application execution has been started, the execution is now driven by the selected CAN-message(s). The execution time information displayed on the target screen is now directly dependent on the reception of the corresponding message. If no message is received the time will not advance. You should make sure, that the corresponding CAN-message on the other CAN node is only generated if the xPC target application is running, otherwise “unexpected interrupt” messages may be displayed on the target screen.

Defining Initialization and Termination CAN Messages

The CAN Setup driver blocks for all supported CAN boards allow the definition of CAN-messages to be sent out during initialization and termination of the target application (once at the beginning of each application run and once before an application run is stopped). The main purpose for sending out those messages is to initialize or terminate other CAN nodes on the network. This is for example the case for CANOpen or DeviceNet nodes. Even if xPC Target doesn't provide direct support of those CAN application layers, communication with those nodes can usually be done over 'standard' CAN messages as long as the nodes have been properly initialized. The initialization and termination fields of the Setup blocks are intended for this purpose.

The initialization and termination CAN-messages are defined by using MATLAB struct arrays with CAN specific field names. This is the same concept as used for the RS-232, GPIB and general Counter driver blocks found in the xPC Target I/O library. Refer to those driver blocks and their help for additional information about this basic concept.

The CAN Setup block specific field names are the following.

Port — Selects the CAN port over which the message has to be sent out. Valid values are either 1 or 2 (double).

Type — Defines if the message to be sent out is of type Standard or Extended. Valid values are either 'Standard' or 'Extended' (strings).

Identifier — Defines the identifier of the message. The value (scalar) itself has to be in the corresponding identifier range (Standard or Extended).

Data — Defines the data frame to be sent out along with the CAN message. The value has to be a row vector of type double with a maximum length of 8. Each element of the vector defines one byte, where the first element defines the data for byte 0 and the eight's element the data for byte 7. Each element can have a value between 0 and 255 (decimal). The data frame size is defined by the length of the row vector.

Pause — Defines the amount of time in seconds the Setup block waits after this message has been sent out and before the next message defined in the struct array is parsed and sent out as well. Valid values are in between 0 and 0.05 seconds. Some CAN nodes need some time to settle before they can accept the next message, especially when the just received message puts the node in a new operational mode. Use this field to define those necessary idle times.

Example

Let's consider an A/D converter module with a CANOpen interface. After the node has been powered up, the module is in pre-operational mode, which is common for CANOpen nodes. At least two initialization messages have to be sent to the node in order to get the module fully operational.

The first message puts the node from pre-operational into operational mode. The second message programs the module in such a sense, that each time the converted A/D value differs for more than 10 mVolts from the former conversion, a CAN-message is automatically sent out, with the converted value as the data frame.

After the target application has been started and the node is properly initialized, the node will automatically send out CAN message, which the xPC target application receives and then processes the contained frame data.

Before the target application execution is actually stopped, the module (node) has to be brought back into pre-operational mode. This is achieved by sending out one corresponding termination message.

The initialization and termination message struct for this example could look as follows.

```
% put node into operational mode
init(1).port=1;
init(1).type='Standard';
init(1).identifier=1536+11;
init(1).data=[hex2dec('22'),hex2dec('23'),hex2dec('64'),hex2dec(
    '00'),hex2dec('01')];
init(1).pause=0.02;

% program node to send CAN messages with converted A/D values
automatically
init(2).port=1;
init(2).type='Standard';
init(2).identifier=0;
init(2).data=[hex2dec('01'),11];
init(2).pause=0;

% put node back into pre-operational mode
term(1).port=1;
term(1).type='Standard';
```

4

CAN I/O Support for FIFO

This chapter includes the following sections:

Introduction

This chapter describes the alternative First In First Out (FIFO) CAN drivers provided with xPC Target. The standard CAN drivers, for the CAN boards from Softing GmbH, program the CAN board firmware to run in Dynamic Object Buffer (DOB) mode. This mode is best suited for real-time environments where it is mandatory that the driver latency time is time deterministic. Actually, running the firmware in Dynamic Object Buffer mode would always be the best choice, but this mode has the undesired side effect of high driver latency times.

- **Sending a CAN message** — When sending out a CAN message, the latency time is the time interval between the time accessing the board in order to provide all the information of the CAN message to be sent out and the time the board returns the acknowledgement that the information has been received by the firmware.
- **Receiving a CAN message** — When receiving a CAN message, the latency time is the time interval between the time accessing the board in order to ask for current data (object data) of a certain CAN identifier and the time the board returns the actual data and other information about the CAN message.

Disadvantages of Dynamic Object Buffer mode — These latency times are mainly defined by the reaction time of the board firmware. In the case of the Softing boards, the latency time is the same for sending and receiving, messages with a fixed value of about 40us. If your xPC Target application has to send and receive a larger number of CAN messages, the overall latency time can quickly become high and may make it impossible to run the application at the desired base sample time.

For example, assuming that a specific xPC Target application has to get data from 12 CAN identifiers and has to transmit data by using 8 CAN messages, the total number of CAN board read and write accesses adds up to 20. This results in a total CAN I/O latency time of

```
20*40us = 800us
```

With such an application, base sample times below 800us are impossible even if the dynamics of the corresponding Simulink model are simple and would only need 20us of computational time.

Advantages of Dynamic Object Buffer mode — However, even if the CAN I/O latency time in the Dynamic Object Buffer mode is high, the benefit of this mode is that the latency time stays constant almost independent of the traffic volume on the CAN network. This leads to the conclusion that the Dynamic Object Buffer mode is best suited for xPC Target applications which only have to deal with a smaller subset of all CAN messages going over the CAN network.

FIFO Mode Drivers for CAN Boards from Softing

The CAN boards from Softing support another mode called First In First Out (FIFO) mode. In this mode the Dynamic Object Buffer mode abstraction layer in the firmware is missing and the firmware plays the role of a slim interface between the receive and transmit FIFOs and the drivers in the application code. Because of this slimmer interface, the I/O latency times are considerably smaller. Writing to the transmit FIFO takes 4us per CAN message and reading one event (CAN message) from the receive FIFO takes 17us. Both of these latency times are smaller than the 40us for the Dynamic Object Buffer mode. While writing to the transmit FIFO is efficient, this is not the case for reading from the receive FIFO. Because the receive FIFO gets filled with all CAN messages (identifiers) going over the CAN network, there may be a lot of data (CAN messages) which have to be read out of the FIFO even if their data is not used in the target application. Because of the FIFO structure, all events (messages) have to be read until the message is returned which has to be propagated to the target application. The driver code for reading the receive FIFO is principally a while loop and this can add the problem of non-deterministic latency times.

The latency time issue in the xPC Target CAN FIFO drivers is resolved by defining a receive FIFO read depth which is a constant number during application execution. For example, if we assume a FIFO read depth of 5, each time the Read Receive FIFO driver block gets executed at the block sample time, the driver code reads and returns 5 events (messages) from the receive FIFO. This is independent of how may events the FIFO currently contains. There may be only two messages received in the FIFO and the third to fifth read attempt may just return the “No new event” code. But nevertheless, because the FIFO read latency does not exceed 17us independent of the event read out of the FIFO, the latency time gets deterministic and is the Read FIFO Depth multiplied by 17us. But again, the driver block returns all new events and therefore all CAN messages going over the network. If only a small subset of the CAN messages received has to be processed in the target application, the total latency may easily exceed the latency encountered when using the Dynamic Object Buffer mode driver scheme for the same application. There is another restriction specific to the FIFO mode concept. Using more than one Read Receive FIFO block in a Simulink model is not recommended, because a new event (message) read by one block instance cannot be read out again by another block instance (the event is no longer in the FIFO).Therefore the entire CAN receive part has to be concentrated in one Read Receive FIFO block in your model. For the write transmit FIFO side, this restriction does not apply. Here you can use as many instances as you want.

The Setup block for the CAN FIFO mode allows controlling the CAN acceptance filters of the CAN controller. The acceptance filter allows defining a range of CAN messages not to be forwarded to the receive FIFO. Filtering out unwanted CAN messages can drastically reduce the read receive FIFO latency time because the unwanted messages do not reach the receive FIFO. Unfortunately, the acceptance filter process uses binary evaluation, which does not allow filtering messages below and above a certain decimal range. Therefore the use of the acceptance filter does only resolve the problem for a small subset of CAN network applications. See “Acceptance Filters” on page 4-38 for more information on this.

Lets look again at our example of 12 messages to be received and 8 messages to be transmitted. If those 20 messages with their specific identifiers are the only messages going over the CAN network (100% usage ratio) the total latency time is

```
12*17us + 8*4us = 236us
```

This is a considerable smaller value than the 800us, which result when using the Dynamic Object Buffer mode drivers.

For the next case we assume that there are 12 additional messages going regularly over the network which have not to be processed by the target application. Additionally, we assume that those messages cannot be filtered out by the CAN controller acceptance filter. Then the total latency time increases to

```
12*17us +20*4us = 284us
```

There is no impact to the final result. That’s the trade-off. Therefore the FIFO mode drivers are best suited for either CAN network monitoring applications or low latency CAN applications where the ratio between the number of messages to be processed and the number of total messages going over the network is high.

Especially for monitor type of applications the FIFO mode drivers are well suited, because the FIFO mode can return additional information like the bus state or the reception of error frames. The Dynamic Object Buffer mode drivers do not allow querying such information.

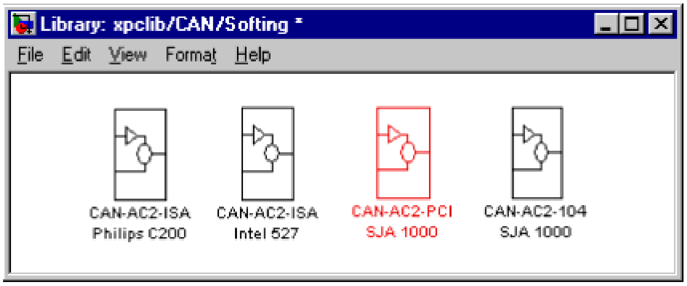
This documentation only covers the differences between the Dynamic Object Buffer mode drivers (which are the standard drivers), and the FIFO mode drivers introduced here. It is assumed that you are familiar with the Dynamic Object Buffer mode drivers and have successfully run one of the loop-back tests provided with xPC Target.

If you use the FIFO mode drivers in your model, you have to replace all Dynamic Object Buffer mode blocks (Setup, Send, Receive) by FIFO mode driver blocks. The CAN-AC2-xxx boards from Softing do not allow to run the two CAN ports in different modes. Therefore the mode has to be same for both ports, but you can use more than one CAN board and run the boards in different modes just by selecting the correct I/O driver blocks.

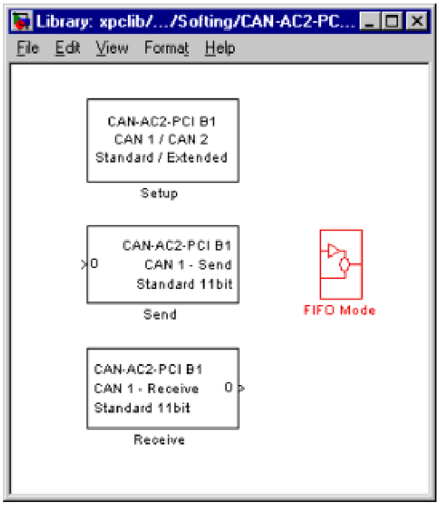
As mentioned in the standard CAN chapter we do not recommend using the CAN-AC2 (ISA) for any new projects. Instead use the CAN-AC2-PCI. As a consequence FIFO mode drivers are only provided for the CAN-AC2-PCI and the CAN-AC2-104 boards.

CAN FIFO Driver Blocks for the CAN-AC2-PCI with Philips SJA1000 CAN-Controller

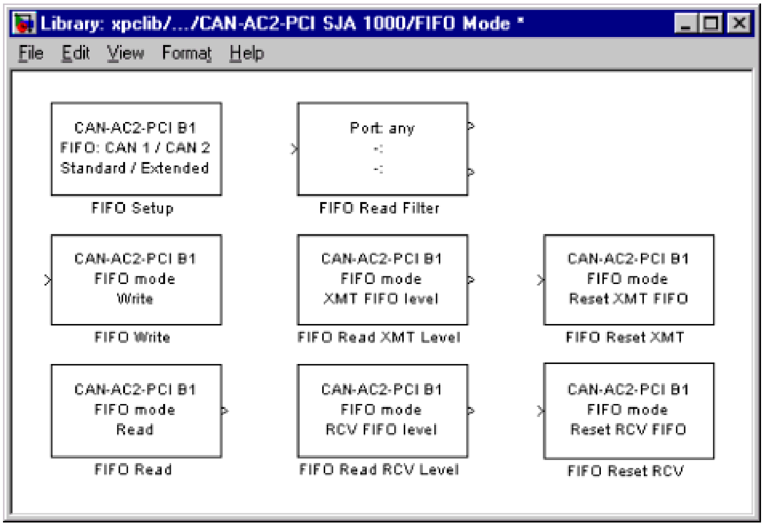
The driver blocks described here support the CAN-AC2-PCI using FIFO mode. The Philips SJA1000 chip is used as the CAN controller in this configuration and supports both Standard and Extended identifier ranges in parallel. The driver block set for this board is found in the xPC Target I/O block library in the group CAN/Softing.



The third block group highlighted above contains the FIFO mode sub group.

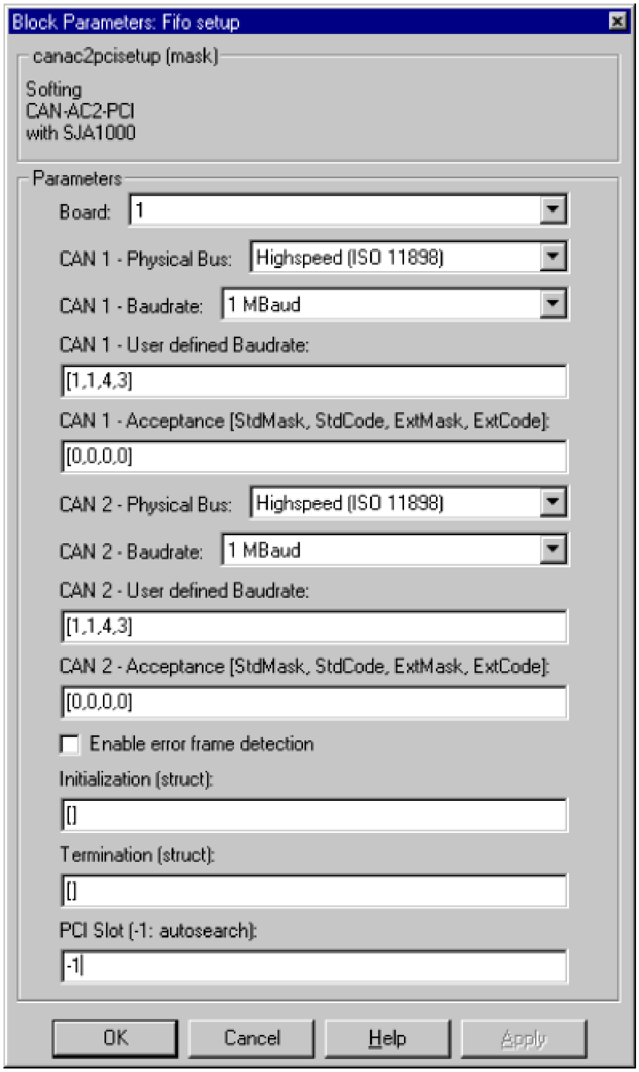


The highlighted group then contains all driver blocks available for FIFO Mode CAN.



FIFO Setup Driver Block

The Setup block is used to define general settings of the plugged-in CAN board(s). The CAN driver blocks for this board support up to three boards for each target system what leads to the availability of up to six CAN ports. For each board in the target system exactly one Setup block has to be used in a model. The dialog box of the Setup block lets you define the following settings.



Board — Defines which board is being accessed by this driver block instance. The board number (1...3) can be seen as a reference identifier in order to differentiate the boards if multiple boards are present in the target system. The physical board finally referenced by the board number depends on the PCI

Slot edit field described further below. If just one board is present in the target system, board number 1 should be selected.

CAN - Physical Bus — Defines the physical CAN bus type of the CAN port 1. In the board's standard hardware configuration, only High speed CAN is supported. By extending the board with Low speed CAN piggyback modules, it is possible to additionally select Low speed CAN as the physical bus. The value of this control shouldn't be changed to Low speed if no module is present for the corresponding CAN port. If the module is present (see the Softing user manual on how to install the modules), you can select between High speed and Low speed CAN.

CAN 1 - Baudrate — Defines the most common baud rates for CAN port 1. If special timing is necessary (baud rate), select **User defined**.

CAN 1 - User defined Baudrate — If you select User defined from the CAN-1 Baud rate list, enter the four values for the timing information. The vector elements have the following meaning.

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 2 - Physical Bus — Defines the physical CAN bus type of the CAN port 2. In the board's standard hardware configuration, only High speed CAN is supported. By extending the board with Low speed CAN piggyback modules, it is possible to additionally select Low speed CAN as the physical bus. The value of this control shouldn't be changed to Low speed if no module is present for the corresponding CAN port. If the module is present (see the Softing user manual on how to install the modules), you can select between High speed and Low speed CAN here.

CAN 2- Baudrate — Defines the most common baud rates for CAN port 2. If special timing is necessary (baud rate), select **User defined**.

CAN 1 - User defined Baudrate — If you select User defined from the CAN-2 Baud rate list, enter the four values for the timing information. The vector elements have the following meaning.

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 2 - Acceptance — Defines the acceptance filters for the CAN 1 port. Because the receive FIFO gets filled with any CAN messages going over the bus, the use of the CAN controller acceptance filters becomes important in order to filter out unwanted messages already at the controller level. This acceptance filter information is provided by a row vector with 4 elements, where the first two are used to define the acceptance mask and acceptance code for Standard identifiers and the latter two for Extended identifiers. The default value defined by the Setup block doesn't filter out any messages. For information on how to define the acceptance information in order to filter certain messages, see "Acceptance Filters" on page 4-38.

Enable error frame detection — If the CAN controller should detect Error frames and forward these to the Receive FIFO, check this box. Checking this box makes sense for monitoring applications where you want to be informed about all events going over the bus. For low latency time applications, checking this box may increase the FIFO Read driver block latency time because the receive FIFO gets filled with additional events.

Initialization (struct) and **Termination (struct)** — Define the CAN messages sent during initialization and termination of the Setup block. For more information, see the standard CAN driver documentation in See "Defining Initialization and Termination CAN Messages" on page 3-56.

PCI Slot (-1: autosearch) — Defines the PCI slot in which the referenced board (board number) resides. If only one CAN board is present in the target system, the value for this control should be `1` for auto search. This value makes sure that the xPC Target kernel automatically finds the board independently of the PCI slot it is plugged into. If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. Use the xPC Target function `getxpcpci` to query the target system for installed PCI boards and the PCI slots they are plugged into. For more information see 'help getxpcpci'.

The board allows activating proper termination for each of the two CAN ports separately by means of DIP-switches at the rear panel of the board. Refer to the Softing user manual on how the DIP-switches have to be set. Both CAN ports have to be terminated properly where you use the provided loop-back model in order to test the board and drivers.

FIFO Write Driver Block

The FIFO Write driver block is used to write CAN messages into the transmit FIFO. The firmware running in FIFO mode processes the information found in the transmit FIFO and finally puts the constructed CAN messages onto the bus.

The block has one input port of type double. At this port, all necessary information has to be provided in order to construct valid CAN messages to be written into the transmit FIFO. For each CAN message, 5 elements have to be passed, which are

```
Port
Identifier
Identifier type
Data frame size
Data
```

Port — The value can be either 1 (port 1) or 2 (port 2) and defines at which port the CAN message is sent out from.

Identifier — This is the identifier of the CAN message to be sent out. If it is a Standard CAN message the valid range is 0 to 2047. If the CAN message is extended, the range is 0 to 2^{29}-1.

Identifier type — The value can be either 0 (Standard identifier range) or 1 (Extended identifier range) and defines the identifier type of the outgoing CAN message.

Data frame size — The value can be in the range of 0 to 8 and defines the data frame size in bytes of the outgoing CAN message

Data — This is the data for the data frame itself and is defined as a double value (8 bytes). The CAN packing block is used to construct the data as a double value.

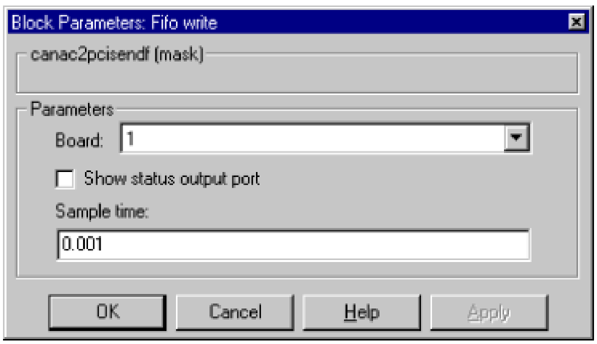
Because all this information can be dynamically changed in FIFO mode during application execution, the information is provided at the block input instead of using block parameters. In order to be able to transmit more than one CAN message per block instance, a matrix signal is used as a container for all information.

The dimension of the matrix signal entering the block has to be n*5, where n is the number of CAN messages to be sent out by this block instance. Therefore each row of the matrix signal defines one CAN message and each row combines the 5 elements of information defined above (in this order).

For more on how to construct the correct matrix signal for the FIFO write block, see See "Examples" on page 4-40.

For certain applications it may be necessary to make the writing of a CAN message into the transmit FIFO dependent on the model dynamics. For this case, the matrix signal can also be of dimension n*6 instead of n*5. In this case, the sixth column defines if the corresponding CAN message is written into the transmit FIFO (value 1) or not (value 0).

The dialog box of the block lets you define the following settings.



Board — Define which physically present board is used to send out the CAN messages defined by this block instance. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, you should select 1.

Show status output port — Check this box to enable the status output port. If the box is unchecked, the block does not have an output port. If enabled, a port is shown. The signal leaving the block is a vector of type double where the number of elements depends on the signal dimension of the block input port. There is one element for each CAN message written into the transmit FIFO and the value is identical to the return argument of function CANPC_send_data(...) described in the Softing user manual. Refer to that manual for more information.

Sample time — Defines the sample time at which the FIFO Write block is executed during a model (target application) run.

You can use as many instances of the FIFO Write block in the model as needed. For example by using two instances of the block, different sample times at which CAN messages are sent out can be defined. Or you can use multiple instances to structure your model more efficiently.

FIFO Read Driver Block

The FIFO Read driver block is used to read CAN messages out of the receive FIFO. The firmware running in FIFO mode puts received events (CAN messages) into the receive FIFO from where the FIFO Read driver reads it out.

The FIFO Read driver block has at least one output port of type double. The signal of this port is a matrix of size m*6, where m is the FIFO Read depth defined in the block dialog box (see below). For example, if the FIFO read depth is 5, then the matrix signal of port 1 has size 5*6. Therefore, one row for each event is read out of the receive FIFO (no new message is considered as an event as well). For information on how to extract data from the matrix signal, see See “Examples” on page 4-40.

Each row with its 6 elements containing all the information defining a CAN message. These are

```
Port
Identifier
Event type
Data frame size
Timestamp
Data
```

Port — The value will be either 1 (port 1) or 2 (port 2) and reports at which port the CAN message was received.

Identifier — This is the identifier of the CAN message being received. If it is a Standard CAN message the range is 0 to 2047, if is an extended CAN message, the range is 0 to 2^{29}-1.

Event type — This value defines the type of event read out of the receive FIFO. The following values are defined from the Softing user manual.

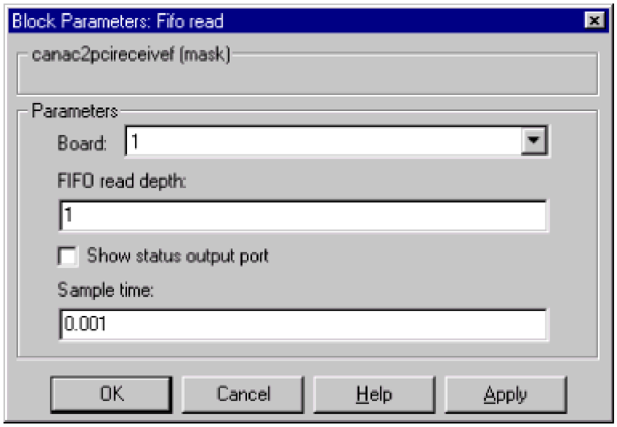
```
0 No new event
1 Standard data frame received
2 Standard remote frame received
3 Transmission of a standard data frame is confirmed
4 -
5 Change of bus state
6 -
7 -
8 Transmission of a standard remote frame is confirmed
9 Extended data frame received
10 Transmission of an extended data frame is confirmed
11 Transmission of an extended remote frame is confirmed
12 Extended remote frame received
13 -
14 -
15 Error frame detected
```

Data frame size — If a data frame has been received, the length of the data in bytes is reported by this element. Possible values are 0 to 8.

Timestamp — This element reports the time at which the event was received. The resolution of the timestamp counter is 1us.

Data — This is the data of the data frame itself and is returned as a double value (8 bytes). The CAN unpacking block is used to extract the data out of the double value.

The dialog box of the block lets you define the following settings.



Board — Defines which physically present board is used to send out the CAN messages defined by this block instance. For more information about the meaning of the board number see the Setup driver block described above. If one board is present in the target system, select board number 1.

FIFO read depth — Defines the number of receive FIFO read attempts. Each time the block gets executed it reads this fixed amount of events (CAN messages) which lead to a deterministic time behavior independent of the number of events currently stored in the receive FIFO. The Read depth (m) defines at the same time the size of the matrix signal (m*6) leaving the first output port. If no event is currently stored in the receive FIFO, the FIFO will be read anyway, but the Event type will be reported as 0 (No new event).

Show status output port — Check this box to enable the Status output port. If the box is unchecked (disabled) the block has one output port for the events. If enabled, a second port is shown. The signal leaving that port is a vector of type double with two elements.

```
[Number of lost messages (events), Bus state]
```

The first element returns the current value of the lost messages counter. The receive FIFO can store up to 255 events. If the receive FIFO is not regularly accessed for reading events, the FIFO gets filled and the lost messages counter starts to count up. This is an indicator that events (messages) will be unavoidably lost. The second element returns the current bus state. Possible values are:

```
0 Error active
1 Error passive
2 Bus off
```

Sample time — The fourth control (edit field) defines the sample time at which the FIFO Read block is executed during a model (target application) run.

It is strongly recommended that you only use one instance of this block per physical CAN board in your model. Otherwise you may get the unwanted behavior that one instance would read events while they have to be processed by blocks connected to the other, second instance.

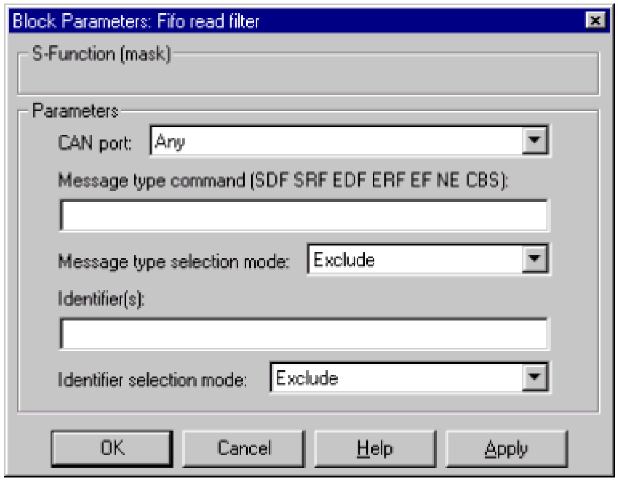
FIFO Read Filter Block

This is a utility block for the CAN FIFO driver block set, but does not actually access the CAN board or any other hardware device. This block is usually connected to the first output port of the FIFO Read driver block and allows filtering events out of the event matrix which is the signal leaving the FIFO Read driver block.

The block code walks through the rows of the incoming event matrix signal and looks out for matching events according to the criteria defined in the block dialog box. If it matches, the entire event information (row) is written to the block first output port. If more than one row matches the criteria, the later event overwrites the earlier event.

The block has one input port and two output ports. The input port is of type double and accepts a matrix signal of size m*6. The two output ports are of type double as well. The first outputs is a row vector (1*6), the filtered event and the second outputs a scalar value which reports the number of matching events the filter block has processed.

The dialog box of the block lets you define the following settings.



CAN port — Defines the filter criterion for the CAN port. From the list, select `Any`, `1`, or `2`.

Message type command — Defines the filter criterion for the event types. This entry can consist of a concatenation of space delimited keywords which are:

```
SDF    Standard data frame
SRF    Standard remote frame
EDF    Extended data frame
ERF    Extended remote frame
EF     Error frame
NE     No new event
CBS    Change of bus state
```

Message type selection mode — Defines how the event type (message type) entered in the control above is treated. If you select `Include`, the event type criterion is the sum of the concatenated keywords. If you select `Exclude`, the event type criterion is equal to all event types minus the sum of the concatenated keywords.

Identifier(s) — Defines the filter criterion for the CAN message identifiers. A set of identifiers can be provided as a row vector.

Identifier selection mode — Defines how the identifier criterion entered in the control above is treated. If you select Include, the identifier criterion is the sum of all specified identifiers. If you select Exclude, the identifier criterion is equal to all identifiers minus the specified identifiers.

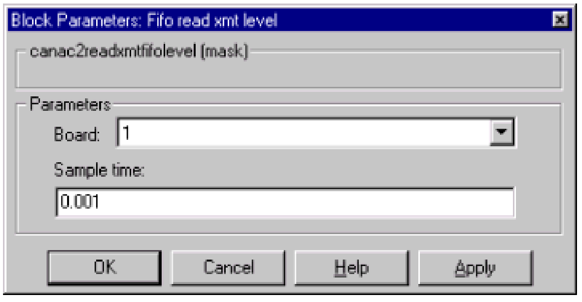
You can use as many instances of this block in your model as needed. Usually, you connect several instances in parallel to the output of the FIFO Read driver block in order to filter out particular messages or events. For more information on how to do this, see See "Examples" on page 4-40.

FIFO Read XMT Level Driver Block

The FIFO Read XMT Level driver block is used to read the current number of CAN messages stored in the transmit FIFO to be processed by the firmware. The transmit FIFO can store up to 255 messages. If it is full and a FIFO write driver block tries to add another messages to the transmit FIFO, the passed messages are lost. You can use this driver block to check for this condition and take appropriate action. For example, you could stop the execution or wait for a non-full transmit FIFO.

The block has a single output port of type double returning a scalar value containing the current transmit FIFO level (number of messages to be processed).

The dialog box of the block lets you define the following settings.



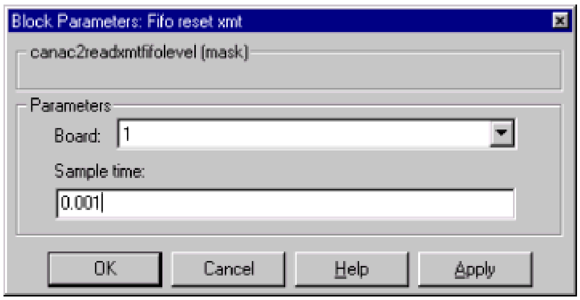
Board — Defines which physically present board is accessed to read the current transmit FIFO level. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Sample time — Defines the sample time at which the FIFO Read XMT Level driver block is executed during a model (target application) run.

FIFO Reset XMT Driver Block

The FIFO Reset XMT driver block is used to reset the transmit FIFO. This will delete all messages currently stored in the transmit FIFO and reset the level counter to 0. As an example, you can use this driver block to reset the transmit FIFO after having detected a fault condition.

The block has a single input port of type double. If a scalar value of 1 is passed, the transmit FIFO gets reset,. If a scalar value of 0 is passed, no action takes place.



The dialog box of the block lets you define the following settings.

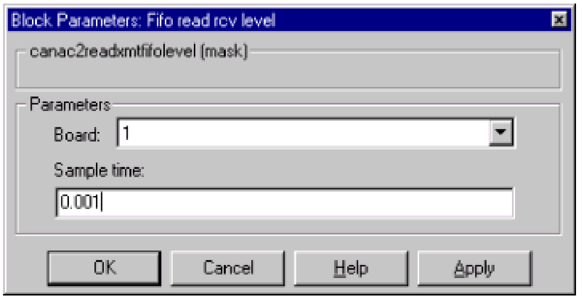
Board — Defines which physically present board is accessed to reset the transmit FIFO. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Sample time — Defines the sample time at which the FIFO Reset XMT driver block is executed during a model (target application) run.

FIFO Read RCV Level Driver Block

The FIFO Read RCV level driver block is used to read the current number of CAN messages stored in the receive FIFO. The receive FIFO can store up to 255 events (messages). If it is full and no FIFO read driver block attempts to read the stored events, new incoming events are lost what is reflected by the lost message counter counting up. You can use this driver block to check for this condition and take appropriate action, like stopping the execution or resetting the receive FIFO.

The block has a single output port of type double returning a scalar value containing the current receive FIFO level (number of messages to be processed).



The dialog box of the block lets you define the following settings.

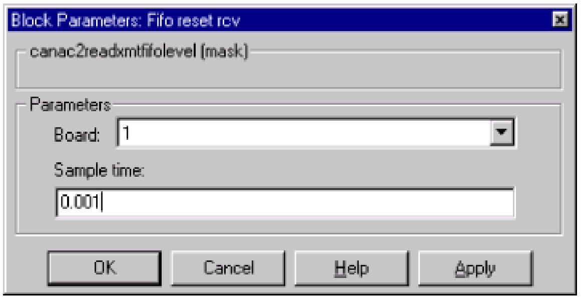
Board — Defines which physically present board is accessed to read the current receive FIFO level. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Sample time — Defines the sample time at which the FIFO Read RCV Level driver block is executed during a model (target application) run.

FIFO Reset RCV Driver Block

The FIFO Reset RCV driver block is used to reset the receive FIFO. This will delete all messages currently stored in the receive FIFO and reset the level counter to 0. As an example, you can use this driver block to reset the receive FIFO after having detected a fault condition.

The block has a single input port of type double. If a scalar value of 1 is passed, the transmit FIFO gets reset. If a scalar value of 0 is passed, no action takes place.



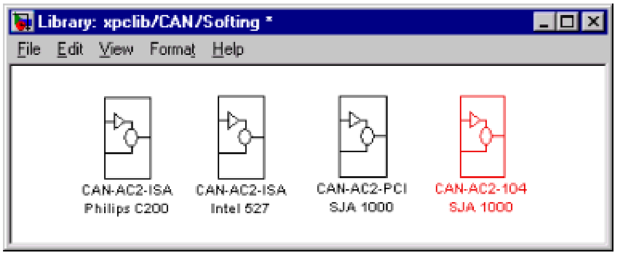
The dialog box of the block lets you define the following settings.

Board — Defines which physically present board is accessed to reset the receive FIFO. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

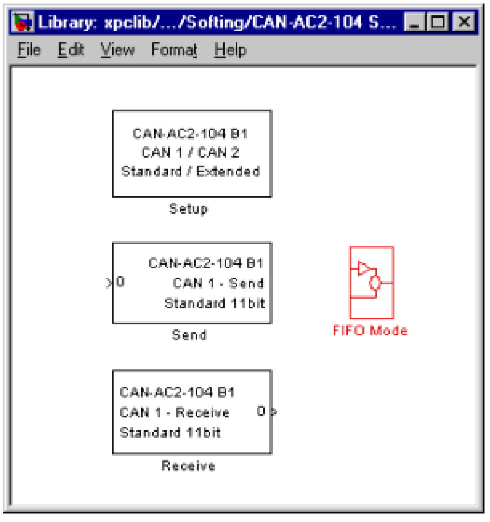
Sample time — Defines the sample time at which the FIFO Reset RCV driver block is executed during a model (target application) run.

CAN FIFO Driver Blocks for the CAN-AC2-104 with Philips SJA1000 CAN-Controller

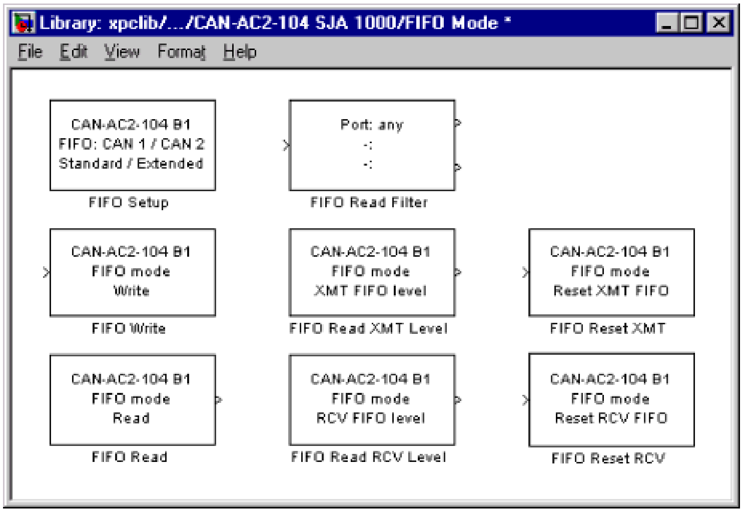
The driver blocks described here support the CAN-AC2-104 (PC/104) using FIFO mode. The Philips SJA1000 chip is used as the CAN controller in this configuration and supports both Standard and Extended identifier ranges in parallel. The driver block set for this board is found in the xPC Target I/O block library in the group CAN/Softing.



The fourth block group highlighted above contains the FIFO Mode sub group.

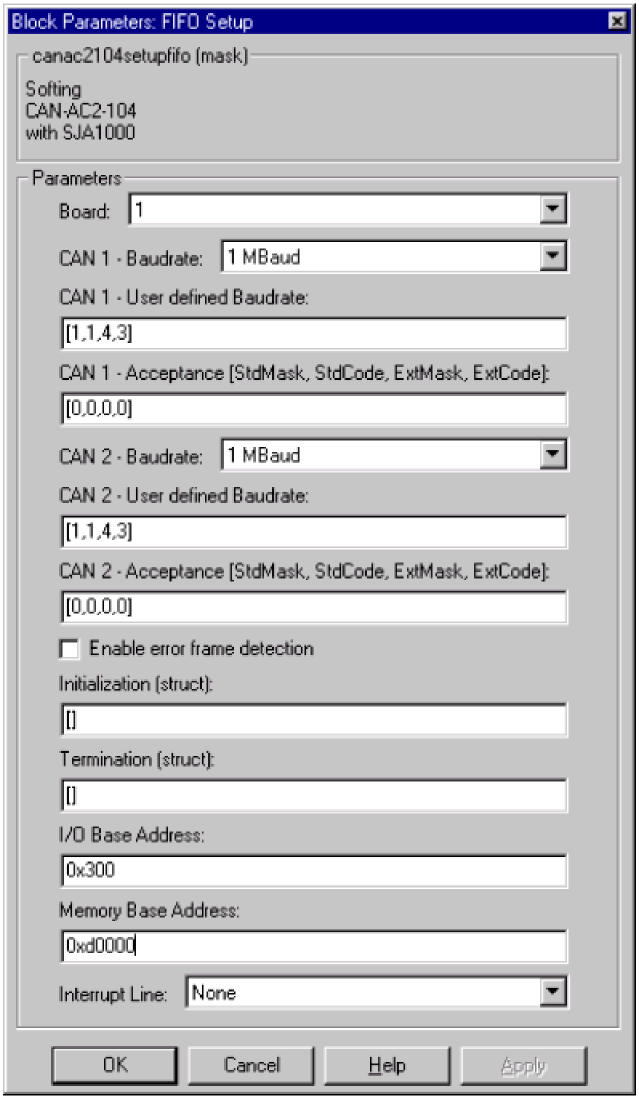


The highlighted group then contains all driver blocks available for FIFO Mode CAN.



FIFO Setup Driver Block

The Setup driver block is used to define general settings of the plugged-in CAN board(s). The CAN driver blocks for this board support up to three boards for each target system what leads to the availability of up to six CAN ports. For each board in the target system exactly one Setup block has to be used in a model. The dialog box of the Setup block lets you define the following settings.



Board — Define which board is being accessed by this driver block instance. If multiple boards are present in the target system, the board number (1, 2 or3) can be seen as a reference identifier in order to differentiate the boards. The physical board finally referenced by the board number depends on the PCI Slot edit field described further below. If just one board is present in the target system, board number 1 should be selected.

CAN 1 - Baudrate — Defines the most common baud rates for CAN port 1. If special timing is necessary (baud rate), you can select `User defined`.

CAN 1 - User defined Baudrate — If you selected `User defined` from the CAN 1 - Baud rate list, enter four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 1 - Acceptance — Defines the acceptance filters for CAN port 1. Because the receive FIFO gets filled with any CAN messages going over the bus, the use of the CAN controller acceptance filters becomes important in order to filter out unwanted messages already at the controller level. This acceptance filter information is provided by a row vector with 4 elements, where the first two are used to define the acceptance mask and acceptance code for Standard identifiers and the latter two for Extended identifiers. The default value defined by the Setup block doesn't filter out any messages. For information how to define the acceptance information in order to filter certain messages, see See "Acceptance Filters" on page 4-38.

CAN 2 - Baudrate — Defines the most common baud rates for CAN port 2. If special timing is necessary (baud rate), You can select `User defined`.

CAN 2- User defined Baudrate — If you selected `User defined` from the CAN 1 - Baud rate list, enter four values for the timing information. The vector elements have the following meaning

```
[ Prescaler, Synchronisation-Jump-Width, Time-Segement-1,
Time-Segment-2 ]
```

For more information about these values see the Softing user manual for this board.

CAN 2 Acceptance — Defines the acceptance filters for CAN port 2. Because the receive FIFO gets filled with any CAN messages going over the bus, the use of the CAN controller acceptance filters becomes important in order to filter out unwanted messages already at the controller level. This acceptance filter information is provided by a row vector with 4 elements, where the first two are used to define the acceptance mask and acceptance code for Standard identifiers and the latter two for Extended identifiers. The default value defined by the Setup block doesn't filter out any messages. For information how to define the acceptance information in order to filter certain messages, see See "Acceptance Filters" on page 4-38.

Enable error frame detection — Defines if the CAN controller should detect Error frames and forward these to the Receive FIFO. Checking this box makes sense for monitoring applications where you want to be informed about all events going over the bus. For low latency time applications, checking this box may increase the FIFO Read driver block latency because the receive FIFO gets filled with additional events.

Initialization (struct) and **Termination (struct)** — Define CAN messages sent during initialization and termination of the Setup block. For more information, see Chapter 3, "CAN I/O Support".

I/O Base address — Defines the I/O base address of the board to be accessed by this block instance. The I/O base address is given by the DIP-switch setting on the board itself. The I/O address range is 3 bytes and is mainly used to transfer the information which memory base address the board should use. See the Softing user manual for this board on how the I/O base address can be set. The I/O base address entered in this control has to correspond with the DIP-switch setting on the board. If more than one board is present in the target system a different I/O base address has to be entered for each board. In this case the I/O base address itself defines which board is referenced by which board number.

Memory base address — Defines the memory base address of the board to be accessed by this block instance. The memory base address is a software setting only (no corresponding DIP-switch is found on the board). The memory address range is 64k bytes. If more than one board is present in the target system a different memory base address has to be entered for each board and you have to make sure that the defined address ranges do not overlap. Because the xPC Target kernel only reserves a subset of the address range between 640k bytes and 1M bytes for memory mapped devices the address ranges have to lie within the following range:

```
C0000    DC000
```

The board allows activating proper termination for each of the two CAN ports separately by means of DIP-switches at the rear panel of the board. Refer to the Softing user manual on how the DIP-switches have to be set. Both CAN ports have to be terminated properly where you use the provided loop-back model in order to test the board and drivers.

FIFO Write Driver Block

The FIFO Write driver block is used to write CAN messages into the transmit FIFO. The firmware running in FIFO mode will then process the information found in the transmit FIFO and finally put the constructed CAN messages onto the bus.

The block has one input port of type double. At this port all necessary information has to be provided in order to construct valid CAN messages to be written into the transmit FIFO. For each CAN message 5 elements have to be passed, which are

```
Port
Identifier
Identifier type
Data frame size
Data
```

Port — The value can be either 1 (port 1) or 2 (port 2) and defines at which port the CAN message will be sent out from.

Identifier — This is the identifier of the CAN message to be sent out. If it is a Standard CAN message the valid range is 0 to 2047, if extended the range is 0 tp 2^{29}-1.

Identifier type — The value can be either 0 (Standard identifier range) or 1 (Extended identifier range) and defines the identifier type of the outgoing CAN message.

Data frame size — The value can be in the range of 0 to 8 and defines the data frame size in bytes of the outgoing CAN message

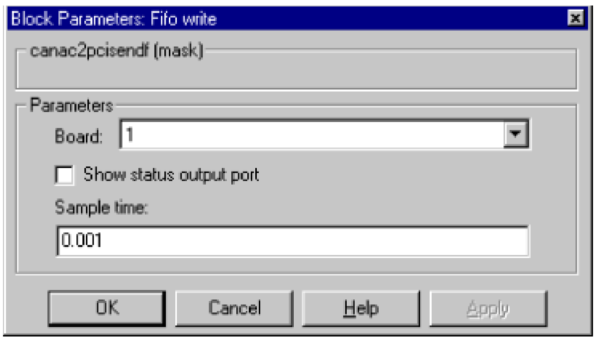
Data — This is the data for the data frame itself and is defined as a double value (8 bytes). The CAN packing block is used to construct the data as a double value.

Because all this information can be dynamically changed in FIFO mode during application execution, the information is provided at the block input instead of using the block parameters. In order to be able to transmit more than one CAN message per block instance a matrix signal is used as a container for all information.

The dimension of the matrix signal entering the block has to be n*5, where n is the number of CAN messages to be sent out by this block instance. Therefore, each row of the matrix signal defines one CAN message and each row combines the 5 elements of information defined above (in this order).

For more information on how to construct the correct matrix signal for the FIFO write block, see “Examples” on page 4-40.

For certain applications it may be necessary to make the writing of a CAN message into the transmit FIFO dependent on the model dynamics. For this, the matrix signal can also be of dimension n*6 instead of n*5. In this case, the sixth column defines if the corresponding CAN message is written into the transmit FIFO (value 1) or not (value 0).



The dialog box of the block lets you define the following settings.

Board — Defines which physically present board is used to send out the CAN messages defined by this block instance. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Show status output port — Checking this box lets you enable the Status output port. If the box is unchecked (disabled), the block does not have an output port. If enabled, a port is shown. The signal leaving the block is a vector of type double where the number of elements depends on the signal dimension of the block input port. There is one element for each CAN message written into the transmit FIFO and the value is identical to the return argument of function

CANPC_send_data(…) described in the Softing user manual. Refer to that manual for more information.

Sample time — Defines the sample time at which the FIFO Write block is executed during a model (target application) run.

You can use as many instances of the FIFO Write block in the model as needed. For example by using two instances of the block, different sample times at which CAN messages are sent out can be defined. Or you can use multiple instances to structure your model more efficiently.

FIFO Read Driver Block

The FIFO Read driver block is used to read CAN messages out of the receive FIFO. The firmware running in FIFO mode puts received events (CAN messages) into the receive FIFO, from where the FIFO Read driver reads it out.

The FIFO Read driver block has at least one output port of type double. The signal of this port is a matrix of size m*6, where m is the FIFO Read depth defined in the block's dialog box (see below). Say the FIFO read depth is 5, then the matrix signal of port 1 has size 5*6, therefore one row for each event read out of the receive FIFO (no new message is considered as an event as well). For information on how to extract data from the matrix signal, see "Examples" on page 4-40.

Each row with its six elements contain all the information defining a CAN message. These are:

```
Port
Identifier
Event type
Data frame size
Timestamp
Data
```

Port — The value will be either 1 (port 1) or 2 (port 2) and reports at which port the CAN message was received.

Identifier — This is the identifier of the CAN message being received. If it is a Standard CAN message, the range is 0 to 2047 If the CAN message is extended, the range is 0 to 2^{29}-1.

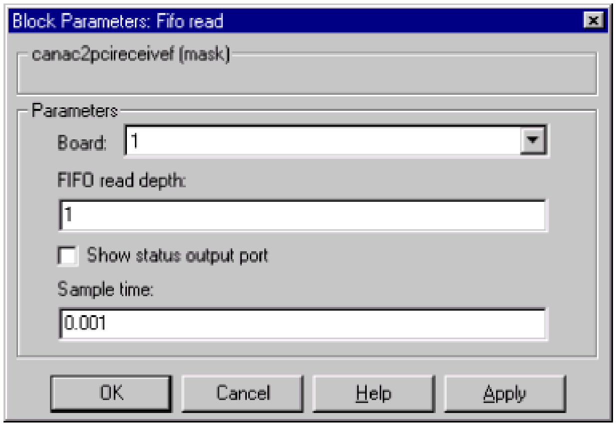
Event type — This value defines the type of event read out of the receive FIFO. The following values are defined from the Softing user manual:

```
16 No new event
17 Standard data frame received
18 Standard remote frame received
19 Transmission of a standard data frame is confirmed
20 -
21 Change of bus state
22 -
23 -
24 Transmission of a standard remote frame is confirmed
25 Extended data frame received
26 Transmission of an extended data frame is confirmed
27 Transmission of an extended remote frame is confirmed
28 Extended remote frame received
29 -
30 -
31 Error frame detected
```

Data frame size — If a data frame has been received the length of the data in bytes is reported by this element. Possible values are 0 to 8.

Timestamp — This element reports the time at which the event was received. The resolution of the timestamp counter is 1us.

Data — This is the data of the data frame itself and is returned as a double value (8 bytes). The CAN unpacking block is used to extract the data out of the double value.



The dialog box of the block lets you define the following settings.

Board — Defines which physically present board is used to send out the CAN messages defined by this block instance. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

FIFO read depth — Defines the number of receive FIFO read attempts. Each time the block gets executed it reads this fixed amount of events (CAN messages) which leads to a deterministic time behavior independent of the number of events currently stored in the receive FIFO. The Read depth (m) defines at the same time the size of the matrix signal (m*6) leaving the first output port. If no event is currently stored in the receive FIFO, the FIFO will be read anyway but the Event type will be reported as 0 (No new event).

Show status output port — Checking this box lets you enable the Status output port. If the box is unchecked (disabled), the block has one output port for the events. If enabled, a second port is shown. The signal leaving that port is a vector of type double with two elements.

```
[Number of lost messages (events), Bus state]
```

The first element returns the current value of the lost messages counter. The receive FIFO can store up to 255 events. If the receive FIFO is not regularly accessed for reading events, the FIFO gets filled and the lost messages counter starts to count up. This is an indicator that events (messages) will be unavoidably lost. The second element returns the current bus state. Possible values are:

```
3 Error active
4 Error passive
5 Bus off
```

Sample time — Defines the sample time at which the FIFO Read block is executed during a model (target application) run.

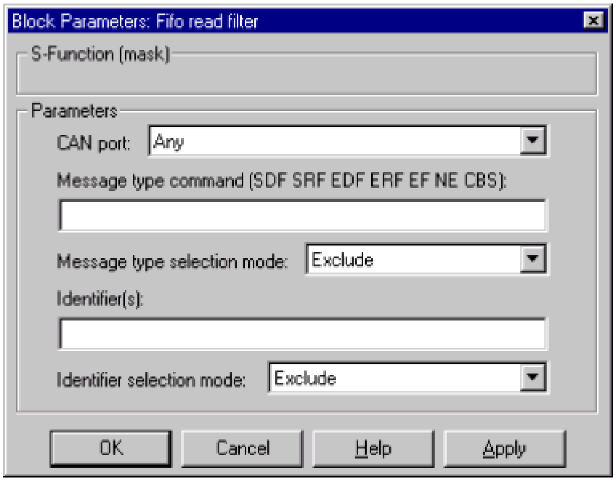
It is strongly recommended that you only use one instance of this block per physical CAN board in your model. Otherwise you may get the unwanted behavior that one instance would read events while they have to be processed by blocks connected to the other, second instance.

FIFO Read Filter Block

This is a utility block for the CAN FIFO driver block set, but does not actually access the CAN board or any other hardware device. This block gets usually connected to the first output port of the FIFO Read driver block and allows filtering events out of the event matrix which is the signal leaving the FIFO Read driver block.

The block code walks through the rows of the incoming event matrix signal and looks out for matching events according to the criteria defined in the block's dialog box. If it matches, the entire event information (row) is written to the block first output port. If more than one row matches the criteria, the later event overwrites the earlier event.

The block has one input port and two output ports. The input port is of type double and accepts a matrix signal of size m*6. The two output ports are of type double as well. The first outputs is a row vector (1*6), the filtered event and the second outputs a scalar value which reports the number of matching events the filter block has processed.



The dialog box of the block lets you define the following settings.

CAN port — Defines the filter criterion for the CAN port. Possible choices are: `Any`, `1` or `2`.

Message type command — Defines the filter criterion for the event types. This entry can consist of a concatenation of space delimited keywords which are:

```
SDF Standard data frame
SRF Standard remote frame
EDF Extended data frame
ERF Extended remote frame
EF  Error frame
NE  No new event
CBS Change of bus state
```

Message type selection mode — Defines how the event type (message type) entered in the control above is treated. If you select Include, the event type criterion is the sum of the concatenated keywords. If you select Exclude, the event type criterion is equal to all event types minus the sum of the concatenated keywords.

Identifier(s) — Defines the filter criterion for the CAN message identifiers. A set of identifiers can be provided as a row vector.

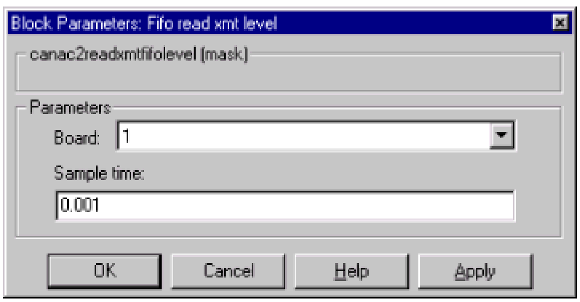
Identifier selection mode — Defines how the identifier criterion entered in the control above is treated. If you select Include, the identifier criterion is the sum of all specified identifiers. If you select Exclude, the identifier criterion is equal to all identifiers minus the specified identifiers.

You can use as many instances of this block in your model as needed. Usually, you connect several instances in parallel to the output of the FIFO Read driver block in order to filter out particular messages or events. For more information on how to do this, see “Examples” on page 4-40.

FIFO Read XMT Level Driver Block

The FIFO Read XMT Level driver block is used to read the current number of CAN messages stored in the transmit FIFO to be processed by the firmware. The transmit FIFO can store up to 255 messages. If it is full and a FIFO write driver block tries to add another messages to the transmit FIFO the passed messages are lost. You can use this driver block to check for this condition and take appropriate action. For example, you could stop the execution or wait for a non-full transmit FIFO.

The block has a single output port of type double returning a scalar value containing the current transmit FIFO level (number of messages to be processed).



The dialog box of the block lets you define the following settings.

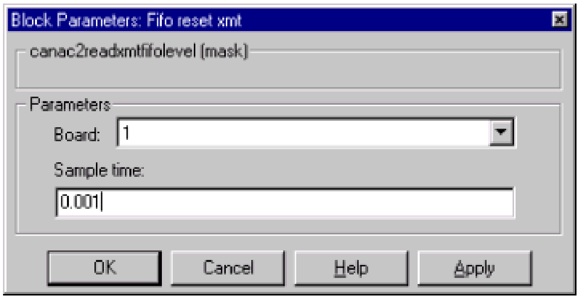
Board — Defines which physically present board is accessed to read the current transmit FIFO level. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Sample time — Defines the sample time at which the FIFO Read XMT Level driver block is executed during a model (target application) run.

FIFO Reset XMT Driver Block

The FIFO Reset XMT driver block is used to reset the transmit FIFO. This will delete all messages currently stored in the transmit FIFO and reset the level counter to 0. As an example, you can use this driver block to reset the transmit FIFO after having detected a fault condition.

The block has a single input port of type double. If a scalar value of 1 is passed, the transmit FIFO gets reset, if 0 is passed no action takes place.



The dialog box of the block lets you define the following settings.

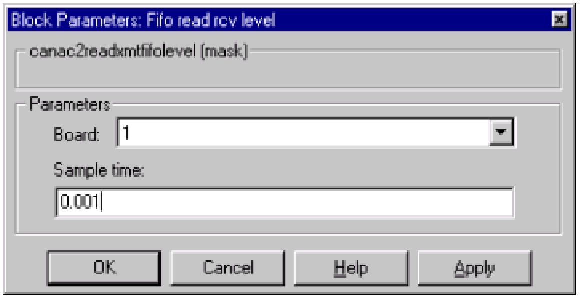
Board — Defines which physically present board is accessed to reset the transmit FIFO. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Sample time — Defines the sample time at which the FIFO Reset XMT driver block is executed during a model (target application) run.

FIFO Read RCV Level Driver Block

The FIFO Read RCV level driver block is used to read the current number of CAN messages stored in the receive FIFO. The receive FIFO can store up to 255 events (messages). If it is full and no FIFO read driver block attempts to read the stored events, new incoming events are lost what is reflected by the lost message counter counting up. You can use this driver block to check for this condition and take appropriate action, like stopping the execution or resetting the receive FIFO.

The block has a single output port of type double returning a scalar value containing the current receive FIFO level (number of messages to be processed).



The dialog box of the block lets you define the following settings.

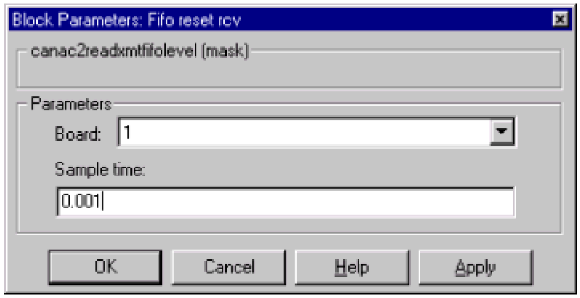
Board — Defines which physically present board is accessed to read the current receive FIFO level. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Sample time — The second control (edit field) defines the sample time at which the FIFO Read RCV Level driver block is executed during a model (target application) run.

FIFO Reset RCV Driver Block

The FIFO Reset RCV driver block is used to reset the receive FIFO. This will delete all messages currently stored in the receive FIFO and reset the level counter to 0. As an example, you can use this driver block to reset the receive FIFO after having detected a fault condition.

The block has a single input port of type double. If a scalar value of 1 is passed, the transmit FIFO gets reset, if 0 is passed no action takes place.



The dialog box of the block lets you define the following settings.

Board — The first control (popup menu) lets you define which physically present board is accessed to reset the receive FIFO. For more information about the meaning of the board number see the Setup driver block described above. If just one board is present in the target system, board number 1 should be selected.

Sample time — Defines the sample time at which the FIFO Reset RCV driver block is executed during a model (target application) run.

Acceptance Filters

As mentioned earlier, the CAN controller's acceptance filters can be used to ensure that certain received messages referenced by their identifiers get written into the receive FIFO. Therefore, fewer read attempts are necessary to get at the messages which are of importance for the target application.

The behavior of the acceptance filter is described for standard and extended identifier ranges individually (one for standard identifiers and one for extended identifiers). Each acceptance filter is defined by a mask parameter and a code parameter.

The mask parameter defines for each bit of the identifier, if the filtering process cares about this bit or not. A 0 means "don't care" and a 1 means "do care."

The code parameter then defines for each bit of the identifier the filtering process cares about (defined by the mask parameter), what the bit value has to be (0 or 1).

For standard identifiers the mask parameter and code parameter have to be both in the range 0 to 2047. For extended identifiers the mask parameter and code parameter have to be both in the range 0 to 2^{29}-1.

The filtering process evaluates the following binary expression

```
and( xor( mask, identifier ), code )
```

If all bits of the resulting value are 0, the message with this identifier will be accepted, if one single bit is 1 the message will be voided.

According to this description, acceptance filters work using binary evaluation while most applications use the mental model of differentiating messages (identifiers) in a decimal or hexadecimal manner. As a consequence, it is possible to filter messages, which identifiers are above a certain decimal number, while the opposite (identifiers below a certain decimal number) can not be achieved in a general way.

Examples

The default values in the FIFO setup driver block are both 0 for the mask parameter and the code parameter. These parameter values assure (the above expression always evaluates to 0) that all incoming messages will reach the receive FIFO (no filtering takes place). All parameter values have to be defined using decimal numbers. You can use the MATLAB function `hex2dec` to also define hexadecimal numbers in the dialog box entry.

Lets assume a CAN application where messages with the following identifiers (standard) are crossing the CAN network:

```
2-30, 48-122 (decimal)
```

Additionally, only incoming messages 4-29 have to be processed by the target application. Ideally, all messages not having identifier 4-29 would be filtered out, but the mask and code parameters don't allow this exact scheme. The closest we can achieve here, is filtering out all messages with identifiers above 31 by using value 2047-31=2016 for the mask parameter and value 0 for the code parameter. The messages with identifier 0,1,2, and 3 cannot be filtered out and will be returned by the FIFO read driver block, even if they have not to be processed by the target application.

Examples

Example 1

Lets start with a simple model using the FIFO Setup block, FIFO Write block, FIFO Read block, and FIFO Read Filter block. The entire CAN network consists of a single physical connection between CAN port 1 and port 2 (loop-back configuration). For this, both CAN ports have to be terminated properly.

The objective of the application is the following:

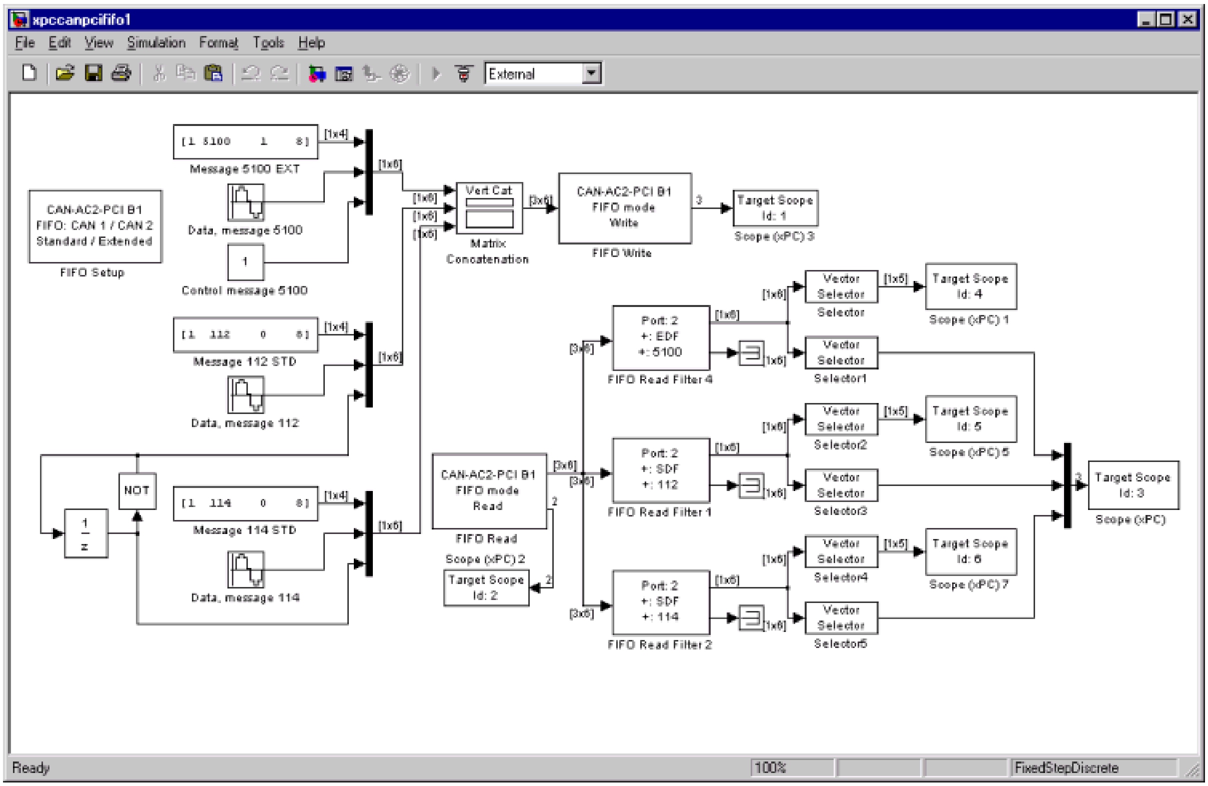
- Send a message with extended identifier 5100 and change data every millisecond on port 1
- Send a message with standard identifier 112 and change data every even millisecond on port 1
- Send a message with standard identifier 114 and change data every odd millisecond on port 1
- Read three events every millisecond from the receive FIFO on port 2
- Display the incoming data of the three messages separately
- Acceptance filtering is not used (all messages are accepted)

The data transmitted with the CAN messages are double values in all the following examples. This has been chosen for simplicity. You should refer to the bit-packing and bit-unpacking chapter of the standard CAN driver documentation, on how to construct from respectively extract into bit fields.

The first implementation uses the following scheme.

The matrix signal entering the FIFO Write block consists of all three messages including the Control element (sixth element), therefore the matrix size will be [3,6]. The sample time of the FIFO Write block is defined as 1 ms. For the standard identifiers which have to be sent out every other millisecond, the Control element is alternated accordingly. This is achieved by using a Unit delay block with corresponding feedback as the Control element value.

The FIFO Read block has a Read depth of 3 and also a base sample of 1 ms. Three FIFO filter blocks are connected to the output of the FIFO read block (in parallel) to extract the information of the incoming CAN messages. You can display the model by typing, in the MATLAB command window, either `xpccanpcififo1.mdl` or `xpccan104fifo1.mdl`.

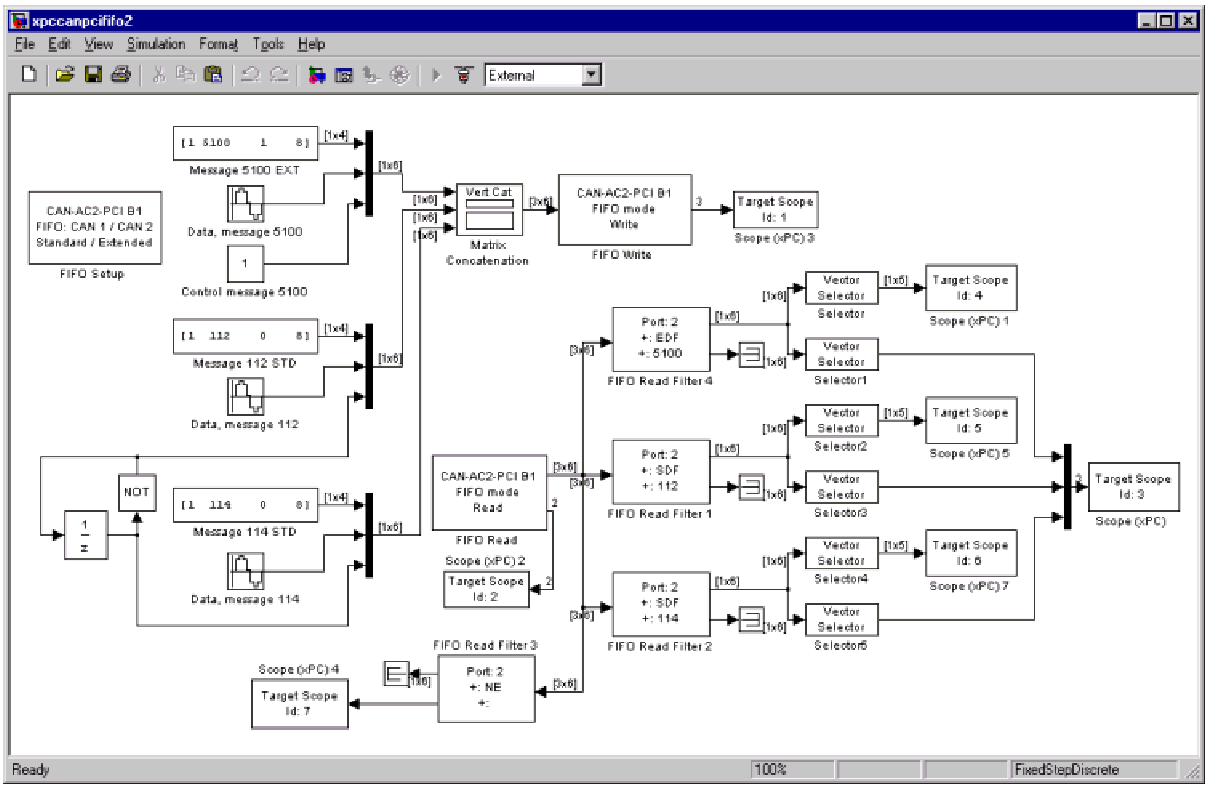


The model uses several xPC Target scope blocks to display different type of data on the target screen:

- Scope 1 (numerical) displays the status vector leaving the FIFO Write block.
- Scope 2 (numerical) displays the status vector [lost-message-counter, bus state] leaving the FIFO Read block.
- Scope 3 (graphical) plots the data of all three CAN messages being received.
- Scopes 4-6 (numerical) displays the other message relevant data of the three incoming CAN messages individually (port, identifier, type, data length, timestamp).

Example 2

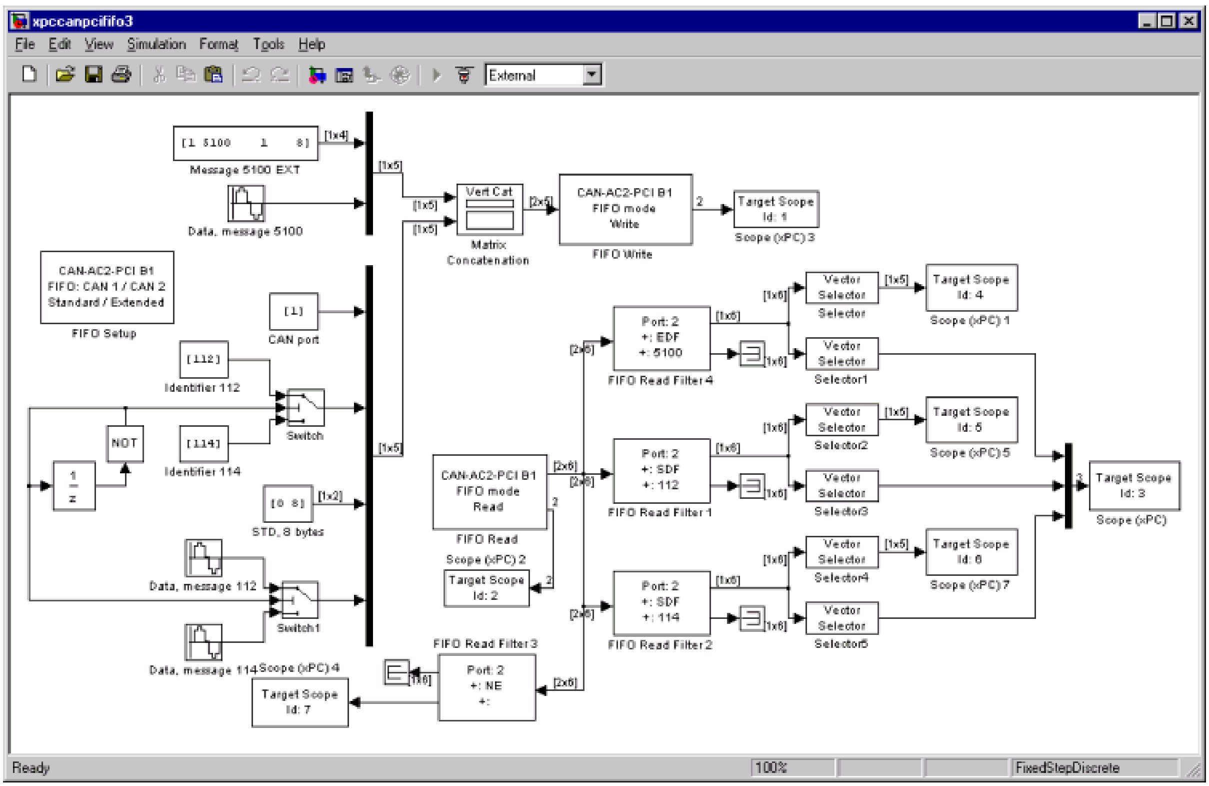
When looking at the time behavior of the model, you can observe that at each millisecond 2 CAN messages are sent out via the FIFO Write block, while the FIFO Read block reads each millisecond three events out of the receive FIFO. This implies that one of the three events leaving the FIFO Read block will be of type "No new event”. This can be visually shown, by attaching another FIFO Filter block in parallel, which filters “No new events”, and then by displaying the second output port, which reports the number of matching events. You can display the model by typing, in the MATLAB command window, either `xpccanpcififo2.mdl` or `xpccan104fifo2.mdl`.



Having observed this, we could then reduce the Read depth of the FIFO Read block from 3 to 2. This would not change anything of the overall behavior of the model. As a positive side effect, the latency time of the FIFO Read block gets smaller and therefore the model's cycle time as well.

Example 3

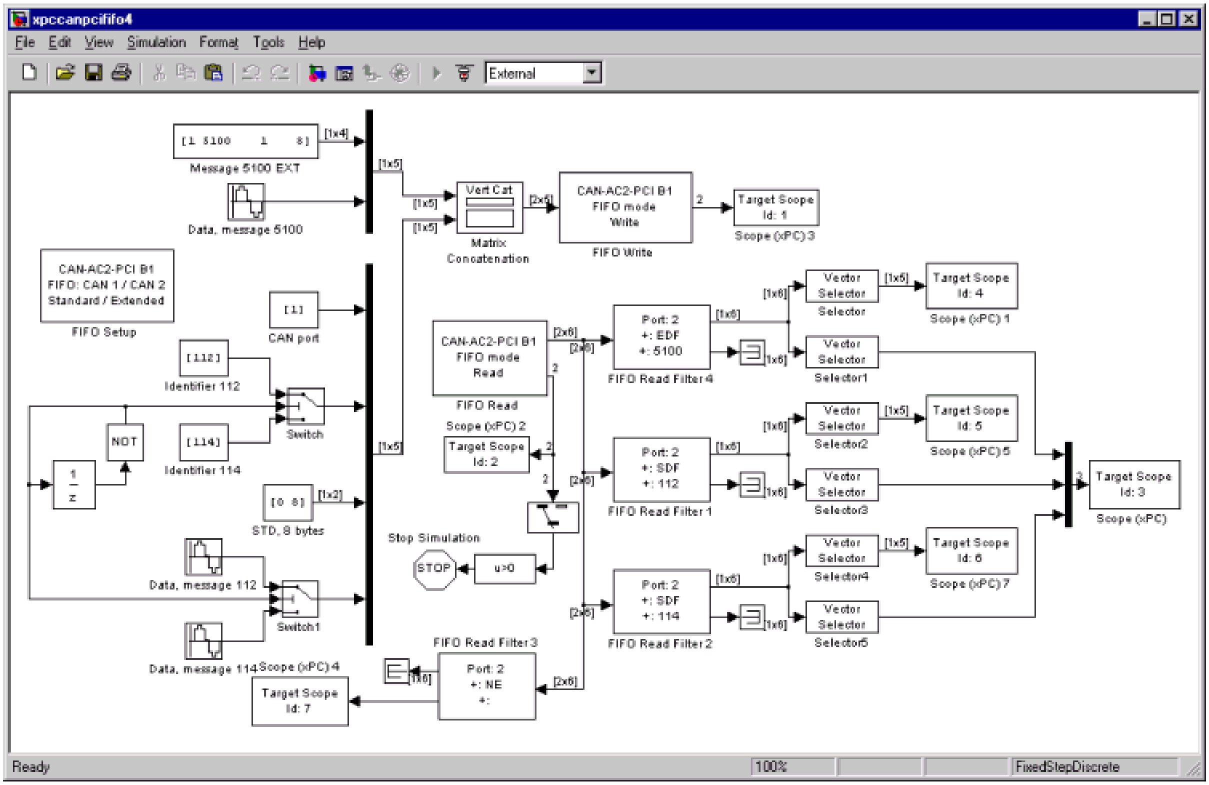
We now look at a second implementation on the FIFO Write side. Instead of providing three messages in parallel, we can just write two messages and then alternate the identifier and data of the second CAN message to be sent. Because the messages are now sent out every millisecond in any case, the Control element has no longer to be provided, therefore reducing the matrix entering the FIFO Write block to a size of [2,5]. You can display the model by typing, in the MATLAB command window, either `xpccanpcififo3.mdl` or `xpccan104fifo3.mdl`.



This implementation behaves exactly like the first implementation, but nicely shows how CAN messages (to be sent out) can be constructed dynamically at run-time.

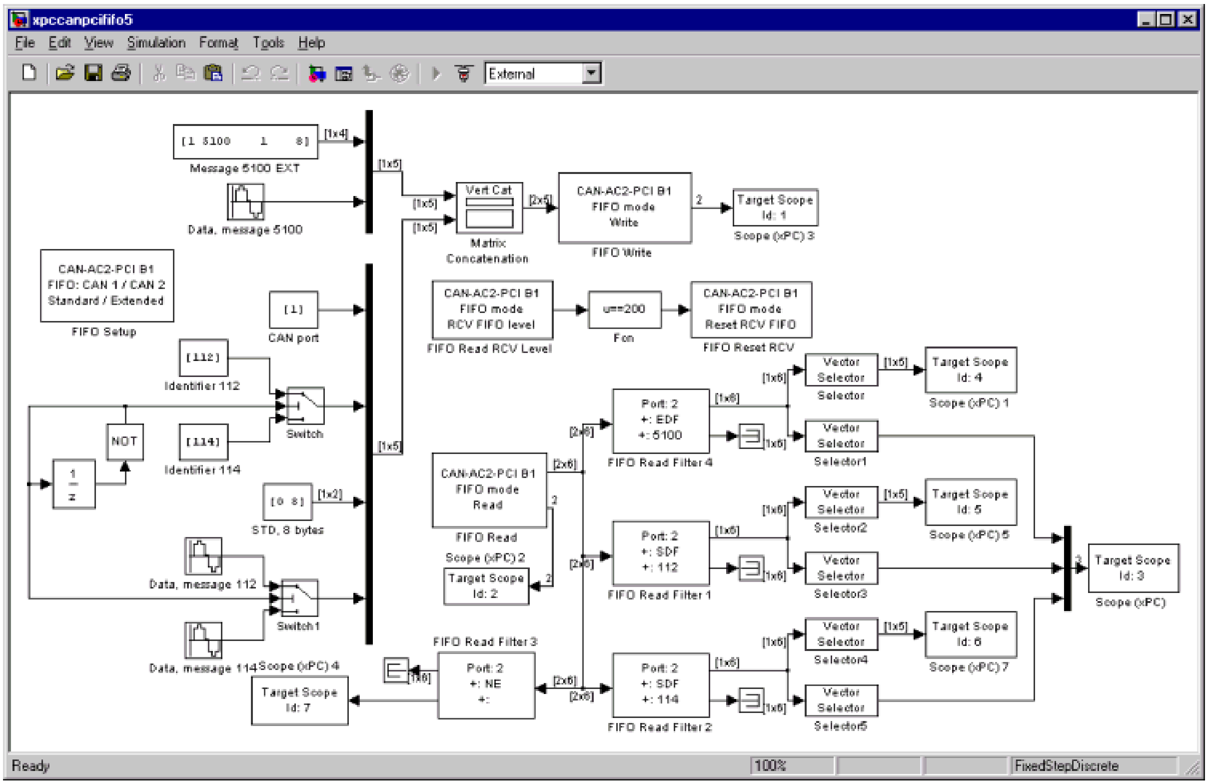
Example 4

Now lets look at the situation where the Read depth parameter of the FIFO Read block in the model above is set to 1 instead of 2 or 3. This leads to a receive FIFO overflow when the execution time reaches 256 ms. Here, as an example, the execution should be stopped, if the overflow occurs. This can be easily achieved by evaluating the lost message counter value leaving the status output port of the FIFO Read block. You can display the model by typing, in the MATLAB command window, either `xpccanpcififo4.mdl` or `xpccan104fifo4.mdl`.



Example 5

Now lets consider a different handling of the receive FIFO overflow: If the receive FIFO level reaches the value of 200, the receive FIFO should be reset in order to delete all currently stored events. The application execution has to continue normally. For this, two new driver blocks have to be added to the model which are used to read the receive FIFO level and then reset it accordingly. You can display the model by typing, in the MATLAB command window, either `xpccanpcififo5.mdl` or `xpccan104fifo5.mdl`.



Example 6

The next example shows the use of the CAN acceptance filters. First the Read depth parameter of the FIFO Read block is set back to a value of 2. Then the identifier of the second standard message is changed from 114 to 188. The goal is to filter any CAN messages with an identifier larger than 127 what would mean that the receive FIFO would never contain the CAN message with identifier 188. Additionally the FIFO Filter block, filtering CAN message with identifier 114 is changed to filter the message with identifier 188.

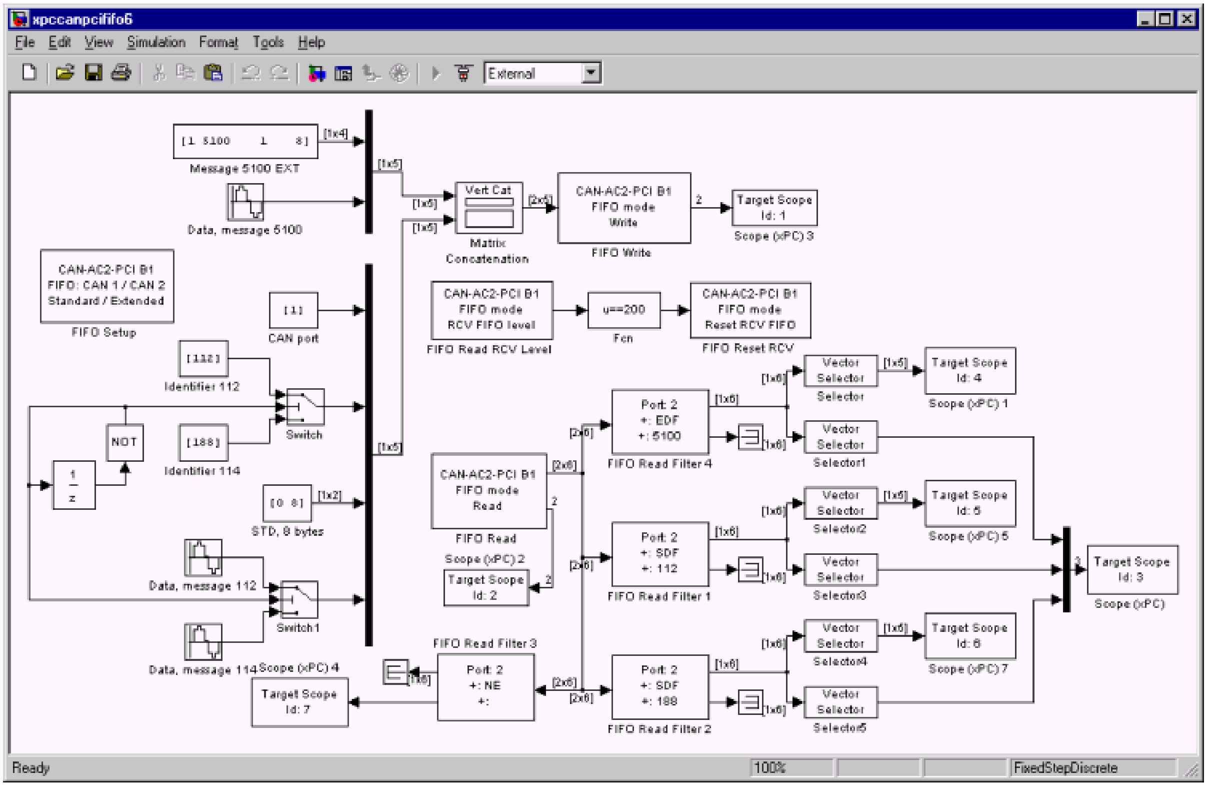
For this the **Acceptance Filters** parameter of CAN port 2 in the FIFO Setup block has to be set accordingly:

```
[2047-127, 0, 0, 0]
```

By rebuilding and re-executing the target application the following can be observed:

- Scope with Id 6 shows 0 for all elements of the vector leaving the corresponding FIFO Filter block. The message with identifier 188 is never received.
- Scope with Id 3 shows one of the data traces always being zero.
- Scope with Id 7 shows a value of 1 which reflects that the Read depth could be reduced to 1, because only one message per millisecond reaches the receive FIFO now.

You can display the model by typing, in the MATLAB command window, either is `xpccanpcififo6.mdl` or `xpccan104fifo6.mdl`.


xpccanpcififo6
File Edit View Simulation Format Tools Help
External
[1 5100 1 8]
Message 5100 EXT
Data, message 5100
CAN-AC2-PCI B1
FIFO: CAN 1 / CAN 2
Standard / Extended
FIFO Setup
[1]
CAN port
[112]
Identifier 112
NOT
[188]
Identifier 114
Switch
1/z
[0 8]
STD, 8 bytes
Data, message 112
Switch1
Data, message 114
Vert Cat
Matrix Concatenation
CAN-AC2-PCI B1
FIFO mode
Write
FIFO Write
Target Scope
Id: 1
Scope (xPC) 3
CAN-AC2-PCI B1
FIFO mode
RCV FIFO level
FIFO Read RCV Level
u==200
Fcn
CAN-AC2-PCI B1
FIFO mode
Reset RCV FIFO
FIFO Reset RCV
CAN-AC2-PCI B1
FIFO mode
Read
FIFO Read
Scope (xPC) 2
Target Scope
Id: 2
Port: 2
+: EDF
+: 5100
FIFO Read Filter 4
Port: 2
+: SDF
+: 112
FIFO Read Filter 1
Port: 2
+: NE
+:
FIFO Read Filter 3
Port: 2
+: SDF
+: 188
FIFO Read Filter 2
Target Scope
Id: 7
Scope (xPC) 4
Vector Selector
Selector
Vector Selector
Selector1
Vector Selector
Selector2
Vector Selector
Selector3
Vector Selector
Selector4
Vector Selector
Selector5
Target Scope
Id: 4
Scope (xPC) 1
Target Scope
Id: 5
Scope (xPC) 5
Target Scope
Id: 6
Scope (xPC) 7
Target Scope
Id: 3
Scope (xPC)
Ready
100%
FixedStepDiscrete

5

UDP I/O Support

xPC Target provides support for the UDP/IP communication protocol. This chapter includes the following sections:

User Datagram Protocol (UDP)

xPC Target supports communication from the target PC to other systems or devices using User Datagram Protocol (UDP) packets. UDP is a transport protocol similar to TCP. However, unlike TCP, UDP provides a direct method to send and receive packets over an IP network. UDP uses this direct method at the expense of reliability by limiting error checking and recovery. This section includes the following topics:

- What Is UDP?
- Why UDP?

What Is UDP?

The User Datagram Protocol (UDP) is a transport protocol layered on top of the Internet Protocol (IP) and is commonly known as UDP/IP. It is analogous to TCP/IP. A convenient way to present the details of UDP/IP is by comparison to TCP/IP as presented below:

- **Connection Versus Connectionless** — TCP is a *connection based* protocol, while UDP is a *connectionless* protocol. In TCP, the two ends of the communication link must be connected at all times during the communication. An application using UDP prepares a packet and sends it to the receiver's address without first checking to see if the receiver is ready to receive a packet. If the receiving end is not ready to receive a packet, the packet is lost
- **Stream Versus Packet** — TCP is a *stream-oriented* protocol, while UDP is a *packet-oriented* protocol. This means that TCP is considered to be a long stream of data that is transmitted from one end to the other with another long stream of data flowing in the other direction. The TCP/IP stack is responsible for breaking the stream of data into packets and sending those packets while the stack at the other end is responsible for reassembling the packets into a data stream using information in the packet headers. UDP, on the other hand, is a packet-oriented protocol where the application itself divides the data into packets and sends them to the other end. The other end does not have to reassemble the data into a stream. Note, some applications may indeed present the data as a stream when the underlying protocol is UDP. However, this is the layering of an additional protocol on top of UDP, and it is not something inherent in the UDP protocol itself.

- **TCP Is a Reliable Protocol, While UDP Is Unreliable** — The packets that are sent by TCP contain a unique sequence number. The starting sequence number is communicated to the other side at the beginning of communication. Also, the receiver acknowledges each packet, and the acknowledgement contains the sequence number so that the sender knows which packet was acknowledged. This implies that any packets lost on the way can be retransmitted (the sender would know that they did not reach their destination since it had not received an acknowledgement). Also, packets that arrive out of sequence can be reassembled in the proper order by the receiver.

  Further, time-outs can be established, since the sender will know (from the first few packets) how long it takes on average for a packet to be sent and its acknowledgment received. UDP, on the other hand, simply sends the packet and does not keep track of them. Thus, if packets arrive out of sequence, or are lost in transmission, the receiving end (or the sending end) has no way of knowing.

TCP communication may be compared to a telephone conversation where a connection is required at all times and two-way streaming data (the words spoken by each party to the conversation) are exchanged. UDP, on the other hand, may be compared to sending letters by mail (without a return address). If the other party is not found, or the letter is lost in transit, it is simply discarded. The analogy fails, however, when considering the speed of communication. Both TCP and UDP communication happen roughly at the same speed since both use the underlying Internet Protocol (IP) layer.

Note *Unreliable* is used in the sense of "not guaranteed to succeed" as opposed to "will fail a lot of the time." In practice, UDP is quite reliable as long as the receiving socket is active, and is processing data as quickly as it arrives.

Why UDP?

UDP was chosen as the transport layer for xPC Target precisely because of its lightweight nature. Since the primary objective of an application running in the xPC Target framework is real-time, the lightweight nature of UDP ensures that the real-time application will have a maximum chance of succeeding in real-time execution. Also, the datagram nature of UDP is ideal for sending samples of data from the Simulink/RTW generated application. Since TCP is stream oriented, separators between sets of data will have to be used for the data to be processed in samples. It is easier to build an application to deal with unreliable data than it is to decode all of this information in real-time. Also, if the application is unable to process the data as quickly as it arrives, the following packets can just be ignored and only the most recent packet can be used.

Communication can involve a packet made up of any Simulink data type (`double`, `int8`, `int32`, `uint8`, etc.), or a combination of these. xPC Target provides blocks for combining various signals into one packet (packing), and then transmitting it. Also, xPC Target provides blocks for splitting a packet (unpacking) into its component signals which can then be used in a Simulink model. The maximum size of a packet is limited to about 500 bytes.

xPC Target UDP Blocks

This section includes the following topics:

- UDP Communication Setup
- UDP Receive Block
- UDP Send Block
- UDP Pack Block
- UDP Unpack Block
- UDP Byte Reversal Block.

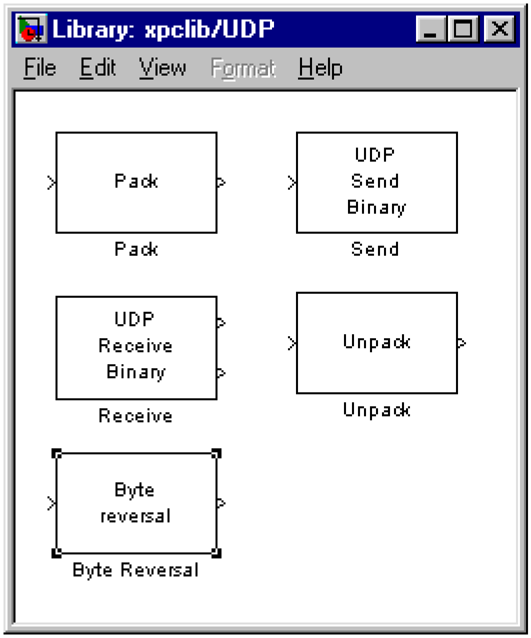
UDP Communication Setup

The infrastructure provided in the xPC Target Library for UDP communication consists mainly of two blocks — a Send block and a Receive block. These blocks may be found in the xPC Target Library available from the Simulink Library under **xPC Target**, or you can access then form the MATLAB command line, by typing

```
xpclib
```

The blocks are located under the UDP heading in the library. The Send block takes as input a vector of type `uint8,` which it sends. This is limited to a length of about 500 bytes (i.e., a 1x500 vector). Similarly, the Receive block outputs a vector of `uint8`. To convert arbitrary Simulink data types into this vector of `uint8`, a Pack block is provided, while an Unpack block is provided to convert a vector of uint8s back into arbitrary Simulink data types. The UDP part of the xPC Target Block Library is shown below.

xPC Target includes a Byte Reversal block for communication with *big-endian* architecture systems. You do not need this block if you are communicating between 80x86-based PC systems running either xPC Target or Microsoft Windows.

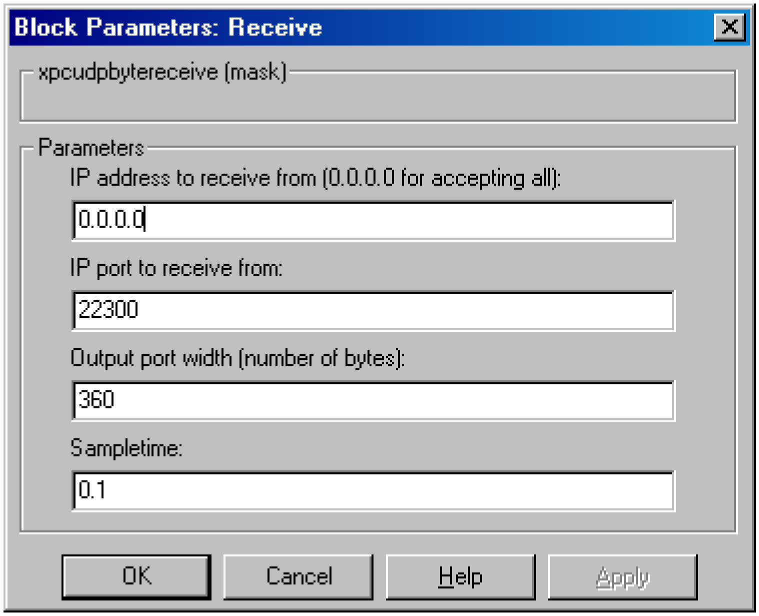


All the blocks are set up to work both from within Simulink and from an application running under xPC Target. However, when using a Simulink simulation and an xPC Target application to communicate, or when using two Simulink models, care must be taken. This is because a Simulink model inherently executes in non-real time and may be several times faster or slower than real time. The sample time of the send and receive blocks and the sample time of the Simulink model must be set so that the communication can proceed properly.

UDP Receive Block

The Receive block has two output ports. The first port is the actual output of the received data as a vector of `uint8` while the second one is a flag indicating whether any new data has been received. This port outputs a value of `1` for the sample when there is new data and a `0`, otherwise. The default behavior of the Receive block is to keep the previous output when there is no new data. This behavior can be modified by the user by using the second port to flag when there is new data.

The Block Parameters for the Review block are shown below.

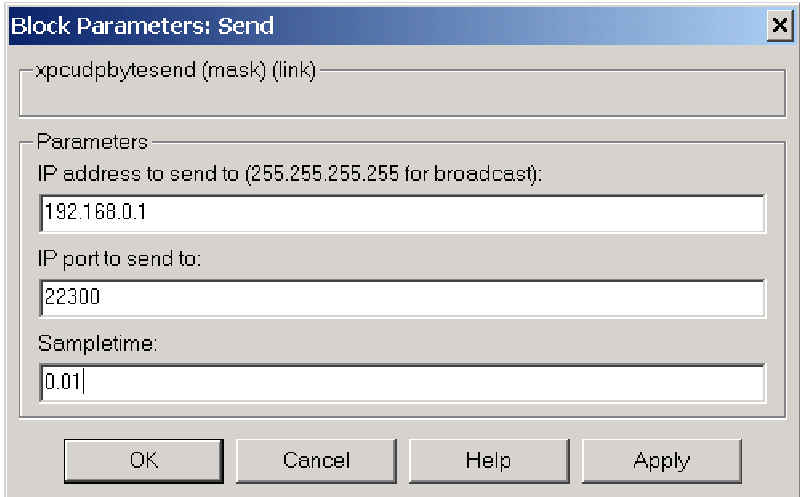


The **IP address to receive from** can be left with the default value of 0.0.0.0. This will accept all UDP packets from any other computer. Otherwise, if set to a specific IP address, only packets arriving from that IP address will be received. The **IP port to receive from** is the port which the block will accept data from. The other end of the communication will have to send data to the port specified here. The output port width is the width of the acceptable packets. This may be obtained when designing the other side (send side) of the communication. The sample time may be set to -1 for inheritable sample time, but it is recommended that this be set to some specific (large) value to eliminate chances of dropped packets. This is especially true when you are using a small base sample time.

UDP Send Block

The Send block has only one input port that receives the `uint8` vector that is sent as a UDP packet.

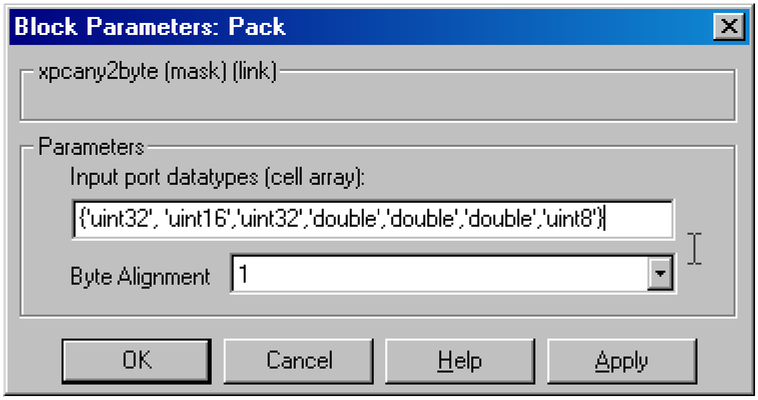
The Block Parameters for the Send block are shown below.



The **IP address to send to** and the **IP port to send to** have to be specified in the appropriate locations. The local IP port that is used for sending will be determined automatically by the networking stack. The sample time should be set to an appropriate value, with similar considerations as in the receive block.

UDP Pack Block

The Pack Block is used to convert one or more Simulink signals of varying data types to a single vector of `uint8` as required by the Send Block. The data types for the different signals must be specified as part of the block parameters while the sizes of each signal are determined automatically.



As seen in the figure above, the data types of each of the signals have to be specified as a cell array of strings in the correct order. Once this is done, the block will automatically convert itself to one with the correct number of input ports. There is always one output port. The supported data types are: `double`, `single`, `int8`, `uint8`, `int16`, `uint16`, `int32`, `uint32`, and `boolean`. The byte alignment field specifies how the data types are aligned. The possible values are: 1, 2, 4 and 8. The byte alignment scheme is simple, and ensures that each element in the list of signals starts on a boundary specified by the alignment relative to the start of the vector. For example, say the Input port data types are specified as

```
{'uint8','uint32','single','int16','double'}
```

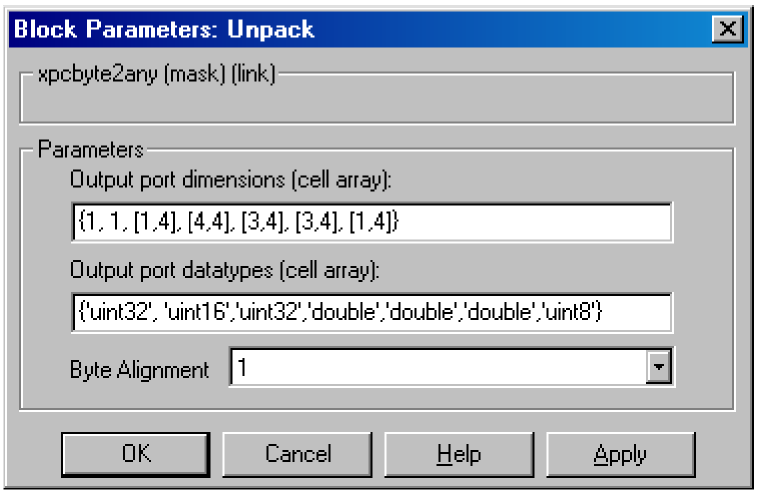
and an alignment of 4 is used. Assume also that all the signals are scalars. The first signal will then start at byte 0 (this is always true), the second at byte 4, the third at byte 8, the fourth at byte 12, and the fifth at byte 16. Note that the sizes of the data types used in this example are 1, 4, 4, 2, and 8 bytes respectively. This implies that there are “holes” of 3 bytes between the first and second signal and 2 bytes between the fourth and fifth signal.

A byte alignment of 1 means the tightest possible packing. That is, there are no holes for any combination of signals and data types.

Note Individual elements of vector/matrix signals are not byte aligned: only the entire vector/matrix is byte aligned. The individual elements are tightly packed with respect to the first element.

UDP Unpack Block

This block is the exact analog of the Pack block. It receives a vector of `uint8` and outputs various Simulink data types in different sizes.



As shown in the figure above, the **Output port datatypes** field is the same as in the **Input port data types** field of the matching Pack block. The Pack block is on the sending side and the Unpack block is on the receiving side in different models. The **Output port dimensions** field contains a cell array, with each element the dimension as returned by the size function in MATLAB of the corresponding signal. This should normally be the same as the dimensions of the signals feeding into the corresponding Pack block.

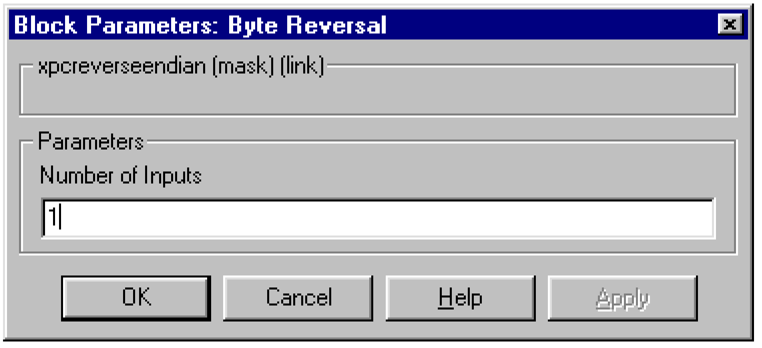
Note on Byte Alignment

The byte alignment feature provided in the Pack and Unpack blocks is primarily intended for interfacing a system running xPC Target to another system that is running neither Simulink nor xPC Target. For example, the data on the other end may be in the form of a C `struct`, which is subject to the byte alignment convention of the compiler used. We recommend using a byte alignment value of 1 (tightly packed) whenever possible. This, of course, is easily accomplished when UDP I/O is used to exchange data between two xPC Target systems or between xPC Target and Simulink.

Even when communication is between xPC Target and a system using a C `struct`, the use of compiler pragmas may help to pack the structure tightly. For example, `#pragma pack(1)` is common to several compilers. The byte alignment blocks are provided for the case when this is not possible.

UDP Byte Reversal Block.

You use the Byte Reversal block for communication between an xPC Target system and a system running with a processor that is *big-endian*. Processors compatible with the Intel 80x86 family are always *little-endian*. For this situation, you should insert a Byte Reversal block before the Pack block and just after the Unpack block to ensure that the values are transmitted properly.



This block takes just one parameter, the number of inputs. The number of input ports adjusts automatically to follow this parameter, and the number of outputs is equal to the number of inputs.

xPC Target UDP Examples

This section includes the following topic:

- UDP Example

UDP Example

In this section, we provide an example of how to set up two way data exchange between two xPC Target systems, between xPC Target and Simulink, or between two Simulink models. When one or both of the systems is running Simulink in non real-time, care must be taken to set the sample time properly.

Our hypothetical models are called Model A and Model B. Two different sets of data are transferred between these two models. One set from Model A to Model B and another set in the opposite direction.

The data to transfer is in the following order:

Model A to Model B

- `uint8` (3x3)
- `int16` (1x1)
- `double` (2x4)

Model B to Model A

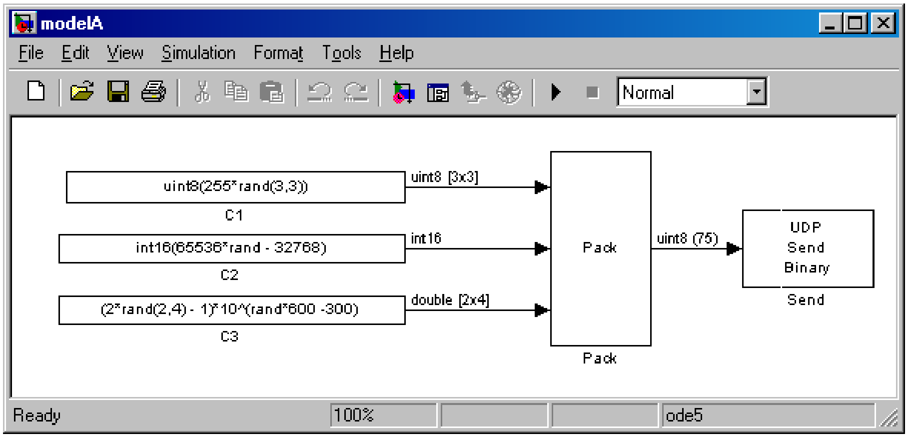
- `single` (4x1)
- `double` (2x2)
- `uint32` (2x2)
- `int8` (5x3)

For the purposes of this example, all the inputs are generated using Simulink Constant blocks that use the MATLAB random number function (rand). The random numbers are generated by Real Time Workshop using this function at the time of code generation. To generate the vector of `uint8` (3x3), use the MATLAB function.

```
uint8(255 * rand(3,3))
```

since 255 is the maximum value for an unsigned 8-bit integer. The other values are generated similarly.

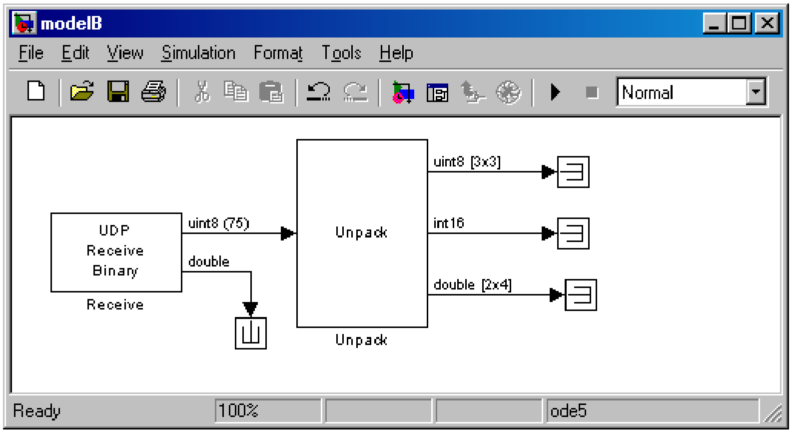
With this setup, construct the send side of modelA.



Note that **Port Data Types** and **Signal dimensions** have been turned on from the **Format** menu, showing us that the width of the UDP packet to be sent is 75 bytes. The parameters used in the Pack block are Input port datatypes `{'uint8','int16','double'}` and Byte Alignment 1.

For the Send block, set the **IP Address to send to** to `192.168.0.2`. This is the hypothetical address of the system that will run Model B. Set the **IP Port to send to** to `25000` (picked arbitrarily). The sample time is set to `0.01`.

Use this information to construct the receive end of Model B.

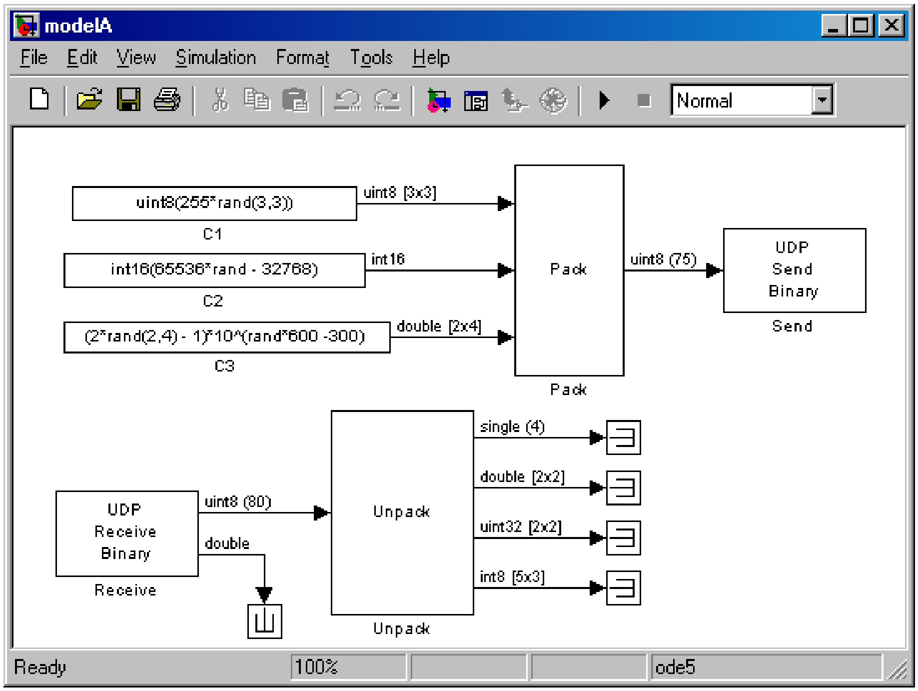


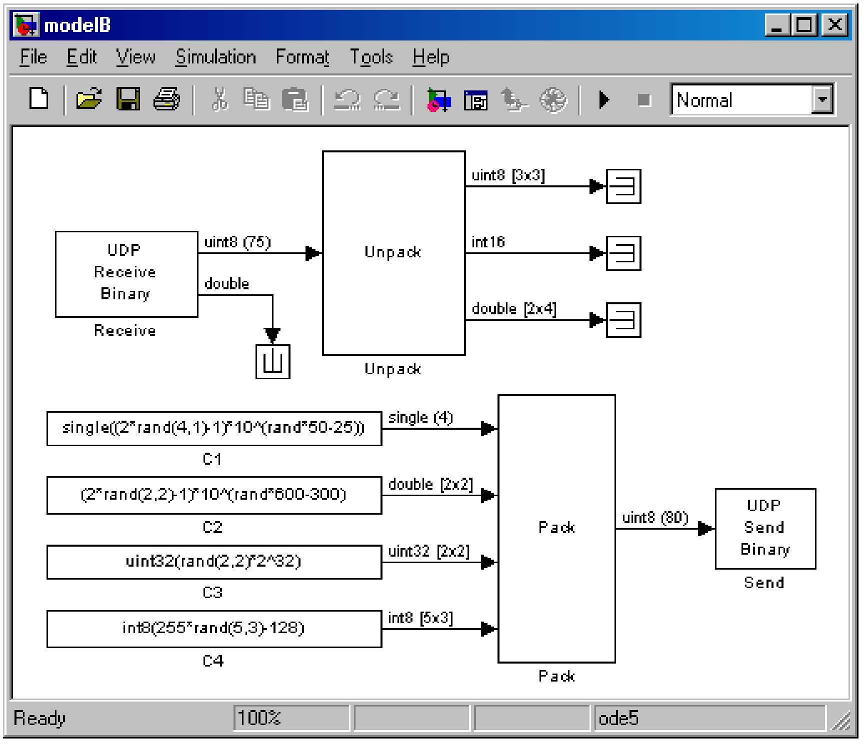
For setting up the Receive block, **IP address to receive from** is set to `192.168.0.1` (the hypothetical address of the system that will run Model B). The **IP port to receive from** is set to `25000` (the same value as set in the Send block in Model A). The **Output port width** is set to `75` which is obtained from the output port width of the Pack block in Model A.

For the Unpack block, **Byte Alignment** is set to `1`, and the **Output port datatypes** is set to `{'uint8','int16','double'}` from the Pack block in Model A. The **Output port dimensions** is set to `{[3 3],1,[2 4]}` from the dimensions of the inputs to the Pack block in modelA.

Note that in Model B, the second output port of the Receive block is sent into a terminator. This may be used to determine when a packet has arrived. The same is true for the outputs of the Unpack block, which in a real model would be used in the model.

For constructing the Model B to Model A side of the communication, follow an analogous procedure. The final models are shown below.





The following table lists the parameters in Model A.

Block	Parameter	Value
Receive	IP address	`192.168.0.2`
	IP port	`25000`
	Output port width	`80`
	Sample time	`0.01`
Unpack	Output port dimensions	`{4,[2 2],[2 2],[5 3]}`

Block	Parameter	Value
	Output port datatypes	{'single','double', 'uint32','int8'}
	Byte Alignment	2

The following table lists the parameters in Model B

Block	Parameter	Value
Pack	Input Port Datatypes	{'single','double', 'uint32','int8'}
	Byte Alignment	2
Send	IP address	192.168.0.1
	IP port	25000
	Sample time	0.01

Note on UDP Communication

The UDP blocks work in the background while the real-time application is not running. Also, the UDP communication has been set up to have a maximum of two UDP packets waiting to be read. All subsequent packets are rejected. This has been done to prevent excessive memory usage, and to minimize the load on the TCP/IP stack. Consequently, when any large background task is being performed, such as uploading a screen shot or communicating large pages through the WWW interface, packet loss may occur. Applications should be designed such that this is not critical. That is, the receipt of further packets after the ones that were lost will ensure graceful continuation.

6

Access IO

I/O boards supported by xPC Target.

WDG-CSM (p. 6-2)	The WDG-CSM is a watchdog timer used to detect computer failure.

WDG-CSM

The WDG-CSM is a watchdog timer used to detect computer failure.

xPC Target supports this board with one driver block:

- "WDG-CSM Watchdog Timer"

Board Characteristics

Board name	WDG-CSM
Manufacturer	Access IO
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

WDG-CSM Watchdog Timer

Driver Block Parameters

Watchdog Time [s] (20us-4800s) - Enter a time value in seconds

Show Reset Port - Select this check box to place an input port on the driver block which resets the board when set to high.

Sampletime - Base sample time or a multiple of the base sample time.

BaseAddress - Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

ADDI-DATA

I/O boards supported by xPC Target.

APCI-1710 (p. 7-2)	The APCI-1710 is a general purpose counting board with four function modules.
PA-1700 (p. 7-5)	The PA1700 is a counter board with three 24-bit counters for connecting three incremental encoders.

APCI-1710

The APCI-1710 is a general purpose counting board with four function modules.

xPC Target supports this board with this driver block:

- "APCI-1710 Incremental Encoder"

Board Characteristics

Board name	APCI-1710
Manufacturer	ADDI-DATA
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

APCI-1710 Incremental Encoder

A function module is individually programmable with different firmware. This is done by using the ADDI-DATA utility SET1710. This driver supports the APCI-1710 if the specified function module is programmed with the incremental encoder firmware.

If the board and its specific module is not programmed with the incremental encoder firmware, SET1710 has to be invoked before the driver can be used within an xPC Target application. In this case, plug the board into a PC running Microsoft Windows and install the board as it is indicated in the ADDI-DATA user manual. Use SET1710 to download the incremental encoder firmware onto the appropriate function module residentially. After this step the board can be removed and be plugged into the target PC.

This driver block has two block outputs. The values output depend on the value of the Type of Evaluation parameter. See below for further information. Refer to the APCI-1710-manual for information on how to electrically connect the encoders to the board.

Driver Block Parameters

Function Module — From the list select 1, 2, 3, or 4. This field specifies which function module (counter) is used for this block. It has to be programmed with the incremental encoder firmware. For the same board two blocks cannot have the same module (channel) specified.

Type of Evaluation — From the list select the type of counter evaluation as either Virtual Absolute or Reset and Index Output Up-Dating.

Choosing Virtual Absolute allows to get the counter value as an absolute value after the reference point of the encoder has been reached for the first time. The first output of the block outputs the actual absolute angle of the connected encoder in radians. As long as the reference point has not been reached for the first time, the second block output is zero. If the reference point is reached for the first time and only for the first time the corresponding counter is reset to zero and the second output goes to 1. From then on the output 1 outputs an absolute angle even in the case the encoder is turned multiple times. The second output can be used for controlling or switching different Simulink submodels.

Choosing Reset and Index Output Up-Dating allows to get the counter value in the range of 0..2*pi or –pi..+pi where the counter is reset every time the reference point is reached. The first output of the block outputs the actual angle of the connected encoder in radian. As long as the reference point has not been reached for the first time, the second block output is zero. Every time the reference point is reached the counter is reset to zero and depending on the direction of the encoder at this event the output value is either incriminated or decremented by the value 1. In other words the second output outputs the actual number of turns n since the reference point has been reached for the first time. If the second output is multiplied by 2*pi and added to the value of the first output you get an absolute multi-turn angle, even if the counter is reset periodically.

Mode — From the list, choose single, double, or quadruple. This parameter specifies the phase detection mode i.e. how many phase-changes are detected of the specified module (see the APC1-1710-manual).

Hystheresis — From the list choose either off or on. The Hystheresis parameter specifies if a counter should skip a tick if the direction changes (see the APC1-1700 manual).

Resolution — This field specifies the resolution of the connected incremental encoder for one revolution.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PA-1700

The PA1700 is a counter board with three 24-bit counters for connecting three incremental encoders.

xPC Target supports this board with this driver block:

- "PA-1700 Incremental Encoder"

Board Characteristics

Board name	PA1700
Manufacturer	ADDI-DATA
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PA-1700 Incremental Encoder

The driver block has two block outputs. The first outputs the actual absolute angle in radians. The second output is zero as long as the index or the reference point was not reached by rotating the encoder. If it is reached for the first time and only for the first time the corresponding counter is reset to zero and this output goes to 1. From then on the output 1 outputs an absolute angle even in the case the encoder is turned multiple times. The second output can be used for controlling or switching different Simulink submodels.

Driver Block Parameters

Counter — From the list select `1`, `2`, or `3`. This parameter specifies which counter is used for this block. For the same board (same base address) two blocks cannot have the same counter (channel) specified.

Mode — From the list select `single`, `double`, or `quadruple`. This parameter specifies the phase detection mode ie. how many phase-changes are detected of the specified counter (see the PA1700-manual).

Hystheresis — From the list choose either `off` or `on`. The Hystheresis parameter specifies if a counter should skip a tick if the direction changes (see the PA1700 manual).

Resolution — This field specifies the resolution of the connected incremental encoder for one revolution.

Sample time — Model base sample time or a multiple of the base sample time.

Base Address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The following jumpers must be set according to the parameters entered above:

- Jumper J16, 17 and 18 must be set to position 1-2
- Jumper J13, 14 and 15 must be set to position 1-2
- Jumper J1, 5 and 9 must be set according to the connected encoders
- Jumper J2, 6 and 10 must be set according to the connected encoders
- Jumper J3, 7 and 11 must be set according to the connected encoders
- Jumper J4, 8 and 12 must be set according to the connected encoders

For information on how to electrically connect the encoders to the board, see the PA1700-manual.

If you want to use the 5V power supply from the board (PIN20), you have to insert Fuse 1 on the board. Refer to the PA1700-manual.

8

Advantech

I/O boards supported by xPC Target (`http://www.advantech.com`)

PCL-1800 (p. 8-3)	16 single or 8 differential analog channels, 2 analog output D/A channels, and 16 digital input lines and 16 digital output lines.
PCL-711B (p. 8-8)	8 single ended analog input channels, 1 analog output channel, and 16 digital input lines and 16 digital output lines.
PCL-726 (p. 8-13)	6 independent analog output D/A channels, 16 digital input lines and 16 digital output lines.
PCL-727 (p. 8-17)	12 independent analog output D/A channels, 16 digital input lines and 16 digital output lines.
PCL-728 (p. 8-21)	2 independent analog output D/A channels.
PCL-812 (p. 8-23)	16 single ended analog input channels, 2 analog output D/A channels, and 16 digital input lines and 16 digital output lines.
PCL-812PG (p. 8-28)	16 single or 8 differential analog channels, 2 analog output D/A channels, and 16 digital input lines and 16 digital output lines.
PCL-818 (p. 8-33)	16 single or 8 differential analog channels, 2 analog output D/A channels, and 16 digital input lines and 16 digital output lines.
PCL-818H (p. 8-38)	16 single or 8 differential analog channels, 1 analog output D/A channel, and 16 digital input lines and 16 digital output lines
PCL-818HD (p. 8-42)	16 single or 8 differential analog channels, 1 analog output D/A channels, and 16 digital input lines and 16 digital output lines.

PCL-818HG (p. 8-46)	16 single or 8 differential analog input (A/D) channels, 1 analog output (D/A) channel, and 16 digital input lines and 16 digital output lines.
PCL-818L (p. 8-51)	16 single or 8 differential analog input (A/D) channels, 1 analog output (D/A) channels, 16 digital input lines, and 16 digital output lines.

PCL-1800

The PCL-1800 is an I/O board with 16 single or 8 differential analog channels (12-bit) with a maximum sample rate of 330 kHz, 2 analog output D/A channels (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-1800 Analog Input (A/D)"
- "PCL-1800 Analog Output (D/A)"
- "PCL-1800 Digital Input"
- "PCL-1800 Digital Output"

Board Characteristics

Board Name	PCL-1800
Manufacturer	Advantech
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-1800 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `single ended` from the MUX list, then enter numbers between 1 and 16. If you choose `differential` from the MUX list, then enter numbers between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.25 to +1.25	-1.25	0 to +1.25	1.25
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.
MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-1800 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
0 to +10	10	0 to +5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-1800 Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-1800 Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-711B

The PCL-711B is an I/O board with 8 single ended analog input channels (12-bit) with a maximum sample rate of 25 kHz, 1 analog output channel (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with four driver blocks:

- "PCL-711B Analog Input (A/D)"
- "PCL-711B Analog Output (D/A)"
- "PCL-711B Digital Input"
- "PCL-711B Digital Output"

Board Characteristics

Board name	PCL-711B
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-711B Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10		
-5 to +5	-5		
-2.5 to +2.5	-2.5		
-1.25 to +1.25	-1.25		
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is `-2.5` to `2.5` volts, enter

```
[-10,-2.5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-711B Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Range — From the list, choose either **0-10V** or **0-5V**.

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-711B Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-711B Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-726

The PCL-726 is an I/O board with, 6 independent analog output D/A channels (12-bit), 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-726 Analog Output (D/A)"
- "PCL-726 Digital Input"
- "PCL-726 Digital Output"

Board Characteristics

Board name	PCL-726
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-726 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 6. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
0 to +10	10	0 to +5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-726 Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-726 Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-727

The PCL-727 is an I/O board with, 12 independent analog output D/A channels (12-bit), 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-727 Analog Output (D/A)"
- "PCL-727 Digital Input"
- "PCL-727 Digital Output"

Board Characteristics

Board name	PCL-727
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-727 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 12. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
		0 to 10	10
-5 to +5	-5	0 to +5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-727 Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-727 Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-728

The PCL-728 is an I/O board with, 2 independent analog output D/A channels (12-bit).

xPC Target supports this board with this driver block:

- "PCL-728 Analog Output (D/A)"

Board Characteristics

Board name	PCL-728
Manufacturer	Advantech
Bus Type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-728 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Channel numbers begin with 1 even if the board manufacturer starts numbering channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812

The PCL-812 is an I/O board with 16 single ended analog input channels (12-bit) with a maximum sample rate of 30 kHz, 2 analog output D/A channels (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-812 Analog Input (A/D)"
- "PCL-812 Analog Output (D/A)"
- "PCL-812 Digital Input"
- "PCL-812 Digital Output"

Board Characteristics

Board name	PCL-812
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-812 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. For example, to use the first and second analog input (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10		
-5 to +5	-5		
-2.0 to +2.0	-2.0		
-1.0 to +1.0	-1.25		

For example, if the first channel is -10 to +10 volts, and the second channel is -5 to 5 volts, enter

```
[-10,-5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range	Range code	Input range	Range code
0 to +5V	5		

For example, if both channels are 0 to +5 volts, enter

```
[5,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812 Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812 Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812PG

The PCL-812PG is an I/O board with 16 single or 8 differential analog channels (12-bit) with a maximum sample rate of 30 kHz, 2 analog output D/A channels (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-812PG Analog Input (A/D)"
- "PCL-812PG Analog Output (D/A)"
- "PCL-812PG Digital Input"
- "PCL-812PG Digital Output"

Board Characteristics

Board name	PCL-812PG
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-812PG Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `single ended` from the MUX list, then enter numbers between 1 and 16. If you choose `differential` from the MUX list,

then enter numbers between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.25 to +1.25	-1.25	0 to +1.25	1.25
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812PG Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range	Range Code	Input Range	Range Code
-10 to +10V	-10	0 to 10 V	10
-5 to +5 V	-5	0 to +5 V	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812PG Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-812PG Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818

The PCL-818 is an I/O board with 16 single or 8 differential analog channels (12-bit) with a maximum sample rate of 100 kHz, 2 analog output D/A channels (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-818 Analog Input (A/D)"
- "PCL-818 Analog Output (D/A)"
- "PCL-818 Digital Input"
- "PCL-818 Digital Output"

Board Characteristics

Board name	PCL-818
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-818 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `single ended` from the MUX list, then enter numbers between 1 and 16. If you choose `differential` from the MUX list,

then enter numbers between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.25 to +1.25	-1.25	0 to +1.25	1.25
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.
MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address - Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range	Range Code	Input Range	Range Code
-10 to +10V	-10	0 to 10 V	10
-5 to +5 V	-5	0 to +5 V	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818 Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818 Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address —Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818H

The PCL-818H is an I/O board with 16 single or 8 differential analog channels (12-bit) with a maximum sample rate of 100 kHz, 1 analog output D/A channel (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-818H Analog Input (A/D)"
- "PCL-818H Analog Output (D/A)"
- "PCL-818H Digital Input"
- "PCL-818H Digital Output"

Board Characteristics

Board name	PCL-818H
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-818H Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `single ended` from the MUX list, then enter numbers between 1 and 16. If you choose `differential` from the MUX list,

then enter numbers between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Channel numbers begins with 1 even if the board manufacturer starts to number channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.25 to +1.25	-1.25	0 to +1.25	1.25
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. **MUX** — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time —Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818H Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Range — From the list, choose either `0-10V` or `0-5V`.

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818H Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818H Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HD

The PCL-818HD is an I/O board with 16 single or 8 differential analog channels (12-bit) with a maximum sample rate of 100 kHz, 1 analog output D/A channels (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-818HD Analog Input (A/D)"
- "PCL-818HD Analog Output (D/A)"
- "PCL-818HD Digital Input"
- "PCL-818HD Digital Output"

Board Characteristics

Board name	PCL-818HD
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-818HD Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `single ended` from the MUX list, then enter numbers between 1 and 16. If you choose `differential` from the MUX list,

then enter numbers between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.25 to +1.25	-1.25	0 to +1.25	1.25
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. **MUX** — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HD Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Range — From the list, choose either `0-10V` or `0-5V`.

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HD Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HD Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low $\geq$0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HG

The PCL-818 is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 100 kHz, 1 analog output (D/A) channel (12-bit), and 16 digital input lines and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- "PCL-818HG Analog Input (A/D)"
- "PCL-818HG Analog Output (D/A)"
- "PCL-818HG Digital Input"
- "PCL-818HG Digital Output"

Board Characteristics

Board name	PCL-818HG
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-818HG Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `single ended` from the MUX list, then enter numbers between 1 and 16. If you choose `differential` from the MUX list, then enter numbers between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +1	1
-2.5 to +2.5	-2.5	0 to +0.1	0.1
-1.25 to +1.25	-1.25	0 to +0.01	0.01
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.
MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HG Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Range — From the list, choose either `0-10V` or `0-5V`.

The range settings have to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HG Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818HG Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818L

The PCL-818L is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate or 40 kHz, 1 analog output (D/A) channels (12-bit), 16 digital input lines, and 16 digital output lines.

xPC Target supports this board with these driver blocks:

- “PCL-818L Analog Input (A/D)”
- “PCL-818L Analog Output (D/A)”
- “PCL-818L Digital Input”
- “PCL-818L Digital Output”

Board Characteristics

Board name	PCL-818
Manufacturer	Advantech
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCL-818L Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `single ended` from the MUX list, then enter channels between 1 and 16. If you choose `differential` from the MUX list, then enter channels between 1 and 8. For example, to use the first and second analog output (A/D) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10		
-5 to +5	-5		
-2.5 to +2.5	-2.5		
-1.25 to +1.25	-1.25		
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is -5 to 5 volts, enter

```
[-10,-5]
```

The range settings have to correspond to the DIP-switch settings on the board.
MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818L Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Range — From the list, choose either `0-10V` or `0-5V`.

The range setting has to correspond to the DIP-switch settings on the board.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818L Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter channels between 1 and 16. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first eight digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCL-818L Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter channels between 1 and 16. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use the first 8 digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Apex

I/O boards supported by xPC Target.

PC-12SD (PC-77SD1) (p. 9-2)	I/O board with up to 12 input channels for positioning sensors of type Synchro or Resolver.

PC-12SD (PC-77SD1)

The PC-12SD is an I/O board with up to 12 input channels for positioning sensors of type Synchro or Resolver. The manufacturer individually programs the board according to the order code or board code. Values for some of the block parameters with this driver are dependent on this board code.

xPC Target supports this board with one driver:

- "PC-12SD (PC-77SD1) Synchro/Resolver"

Board Characteristics

Board name	PC-12SD
Manufacturer	Apex (NAA)
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

PC-12SD (PC-77SD1) Synchro/Resolver

Scaling Input to Output

Hardware Input	Block Input Data Type	Scaling
Synchro or Resolver	double	angle in rad, velocity in rps

Driver Block Parameters

Channel vector — Enter numbers between 1 and 12 to select the Synchro/Resolver (S/D) channels you use with this block. The driver allows the selection of individual S/D channels in any order, but repeating channels is not allowed. For example, to use the first and second S/D channels, enter

```
[1,2]
```

Number channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Sensor type vector — If the board code allows switching between Synchro or Resolver inputs, use this vector to define which type of sensor is attached to the corresponding channel. The vector his to have the same length as the **Channel vector**. Use a value 0 is to specify Resolver input, and a value 1 to specify Synchro input. If the board code stands for a static type of sensor, enter and empty vector ([ ]).

Ratio vector — Enter numbers between 1 and 255 to define the ratio for each channel. This vector has to have the same length as the **Channel vector**. For single speed input, use a ratio value of 1. For double speed input the ratio depends on your sensor and application and can have a value between 2 and 255. In the case of double speed input, two channels are used to provide fine and coarse data.

Output format — From the list, choose **Angle**, **Angle-Status**, Angle-Velocity, or **Angle-Velocity-Status**. This is the output format for each S/D channel and also the format for the output port of this block. The possible selections are:

- **Angle** — The signal width is 1. This scalar is the angular position in radians.
- **Angle - Status** — The signal width is 2. The first element is the angular position, and the second element is the status.
- **Angle - Velocity** — The signal width is 2. The first element is the angular position, and the second element is the angular velocity. The unit for the angular velocity is revolutions/second (rps).
- **Angle - Velocity - Status** —The signal width is 3. The first element is the angular position, the second element the angular velocity, and the third element is the status.

The status signal returns information about test status, signal status and reference status for each S/D channel. All three statuses return binary information (0 is OK, 1 is FAILURE). The test status has weight 2^0, signal status has weight 2^1, and the reference status has weight /2^2.

For example, a status value of 5, means the test status is OK and both signal status and reference status are FAILURE.

Notice, if you do not provide a **Reference Vector** by entering an empty matrix, the reference status is not returned. See the board manual for more information about statuses.

Velocity scaling vector — Enter a scale factor for defining the maximum rotations/second (rps) for each S/D channel. You need to enter a value to read velocity information. This vector has to have the same length as the **Channel vector**. The values entered here define the maximum revolutions/second and affect the accuracy of the velocity readings. Choose values to have the best accuracy.

Show input ports for dynamic velocity scaling — Selecting this check box allows you to update the **Velocity scaling vector** at runtime.

If checked, the block shows the same number of input ports as output ports. That is, one port for each selected S/D channel. The signal width of each input port is 1. You can use the signal entering the corresponding input port to update the **Velocity scaling vector**. Even if you select this check box and you provide values to the input ports, you still have to enter a **Velocity scaling vector**. In this case, the **Velocity scaling vector** defined the initial values.

Reference vector — If the board code includes the Reference Output option, you can use this vector to define the frequency and amplitude of the reference output. If you enter an empty matrix ([ ]), the reference output circuit is not accessed, even if the board is equipped with it.

To activate the reference output, you have to enter a row vector with two elements, where the first element defines the frequency in Hertz and the second element defines the output voltage in Volts.

Sample time - Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address -Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Burr-Brown

I/O boards supported by xPC Target.

PCI-20003M (p. 10-2)	I/O board with 2 analog output (D/A) channels (12-bit).
PCI-20019M (p. 10-4)	I/O board with 8 single analog input (A/D) channels (12-bit).
PCI-20023M (p. 10-7)	I/O board with 8 single analog input (A/D) channels (12-bit).
PCI-20041C (p. 10-10)	Carrier board with 32 digital I/O-lines grouped into four ports that can be configured as digital input or output. Each port has a maximum of 8 digital lines.
PCI-20098C (p. 10-13)	Carrier board with 8 single or 16 differential analog input (A/D) channels (12-bit), and 16 digital I/O-lines grouped into two 8-line ports.

PCI-20003M

The PCI-20003M is an I/O board with two analog output (D/A) channels (12-bit).

xPC Target supports this board when it is installed on a PCI-20041C carrier board with this driver block:

- "PCI-20003M Analog Output (D/A)"

Board Characteristics

Board name	PCI-20003M
Manufacturer	Burr-Brown
Bus type	ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PCI-20003M Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — This parameter is a combined Channel vector and Range vector. The number of elements defines the number of A/D channels used.

Enter a range code for each of the A/ D channels used. This driver allows a different range for each channel with a maximum of two A/D channels.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
		0 - 5	5

For example, if the first channel is -10 to + 10 volts and the second channel is 0 to +5 volts, enter

```
[-10,5]
```

The jumpers W1 to W5, W13, W14, W27, W31, W7 to W11, W30, W32 on the module must correspond to this range setting.

Sample Time — Enter the base sample time or a multiple of the base sample time.

Module Number — Enter a number from 1 to 3 to identify the connector on the carrier board where the I/O module is inserted. This driver verifies if the module is placed on the specified module connector.

Base Address or Carrier Board (ie: 0xd000) — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-20019M

The PCI-20019M is an I/O board with 8 single analog input (A/D) channels (12-bit).

xPC Target supports this board when it is installed on a PCI-20041C carrier board with this driver block:

- "PCI-20019M Analog Input (A/D)"

Board Characteristics

Board name	PCI-20019M
Manufacturer	Burr-Brown
Bus type	ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PCI-20019M Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels.

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Input Range — Enter an input range code for all A/D channels. This driver does not allow the selection of a different range for individual channel. The input range is the same for all A/D channels

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2.5 to +2.5	-2.5		

The jumpers W1 to W5 on the module must correspond to this range setting.

Sample time — Enter a base sample time or a multiple of the base sample time.

Module Number (1-3) — Enter a number from 1 to 3 to identify the connector on the carrier board where the I/O module is inserted. This driver verifies if the module is placed on the specified module connector.

Base Address of Carrier Board (ie. 0xd000) — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Other jumper on this board. The switch and jumper settings, that are not mentioned here, have no influence on the running of xPC Target.

Jumper Number	Jumper	Jumper Number	Jumper
W6	out	W22	out
W8	in	W27	out
W10	out	W30	-

Jumper Number	Jumper	Jumper Number	Jumper
W11	in	W31	-
W12	out		

PCI-20023M

The PCI-20023M is an I/O board with 8 single analog input (A/D) channels (12-bit).

xPC Target supports this board when it is installed on a PCI-20041C carrier board with this driver block:

- "PCI-20023M Analog Input (A/D)"

Board Characteristics

Board name	PCI-20023M
Manufacturer	Burr-Brown
Bus type	ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PCI-20023M Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of A/D channels used. This driver does not the selection of individual channels.

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Input Range — Enter an input range code for all A/D channels. This driver does not allow the selection of a different range for individual channel. The input range is the same for all A/D channels.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10	0 - 10	10
-5 to +5	-5		

The jumpers W1, W2, W4, W5, W33 on the module must correspond to this range setting. The switch and jumper settings, that are not mentioned here, have no influence on running xPC Target.

Sample time — Enter the base sample time or a multiple of the base sample time.

Module Number (1-3) — Enter a number from 1 to 3 to identify the connector on the carrier board where the I/O module is inserted. This driver verifies if the module is placed on the specified module connector.

Base Address of Carrier Board (ie. 0xd000) — Enter the base address of the I/O board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Other jumpers on this board. The switch and jumper settings, that are not mentioned here, have no influence on the running of xPC Target.

Jumper Number	Jumper	Jumper Number	Jumper
W6	out	W12	out
W8	in	W27	out
W9	-	W30	-

Jumper Number	Jumper	Jumper Number	Jumper
W10	out	W31	-
W11	in		

PCI-20041C

The PCI-20041C is a carrier board with 32 digital I/O-lines grouped into four ports that can be configured as digital input or output. Each port has a maximum of 8 digital lines.

xPC Target supports this board with these driver blocks:

- "PCI-20041C Digital Input"
- "PCI-20041C Digital Output"

Board Characteristics

Board name	PCI-20041C
Manufacturer	Burr-Brown
Bus type	ISA
Access method	Memory mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-20041C Digital Input

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of digital input lines used with this port.

Port Number (1-4) — Enter a number from 1 to 4 to identify the port used with this block of digital input lines.

Sample time — Enter a base sample time or a multiple of the base sample time.

Module Number (0-3) — Enter a number from 0 to 3 to identify the connector on the carrier board where the I/O module is inserted. This driver verifies if the module is placed on the specified module connector.

Base Address or Carrier Board (ie: 0xd000) — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-20041C Digital Output

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of digital output lines used with this port.

Port Number (0-3) — Enter a number from 0 to 3 to identify the port used with this block of digital output lines.

Sample Time — Enter a base sample time or a multiple of the base sample time.

Module Number (1-3) — Enter a number from 1 to 3 to identify the connector on the carrier board where the I/O module is inserted. This driver verifies if the module is placed on the specified module connector.

Base Address or Carrier Board (ie: 0xd000) — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-20098C

The PCI-20098C is a carrier board with 8 single or 16 differential analog input (A/D) channels (12-bit), and 16 digital I/O-lines grouped into two 8-line ports.

xPC Target supports this board with these driver blocks:

- "PCI-20098C Analog Input (A/D)"
- "PCI-20098C Digital Input"
- "PCI-20098C Digital Output"

Board Characteristics

Board Name	PCI-20098C
Manufacturer	Burr-Brown
Bus Type	ISA
Access Method	Memory mapped
Multiple block instance support	A/D: No, Digital I/O: Yes
Multiple board support	Yes

PCI-20098C Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of single A/D channels used. If `differential` is chosen from the MUX list, then enter a number between 1 and 8 to select the number of differential A/D channels used. This driver does not allow the selection of individual channels or a different MUX setting for each channel.

Range — From the list, choose either `+-10V` (-10 to +10 volts), `+-5V` (-5 to +5 volts), or `0-10V`.This driver does not allow the selection of a different range for each channel. The input range is the same for all A/D channels

MUX (16/8) — From the list, choose either `16 single-ended` or `8 differential`. This entry must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base Address or Carrier Board (ie: 0xd000) — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-20098C Digital Input

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of digital input lines used with this port.

Port Number — From the list, choose either `A` or `B` to identify the port used with this block of I/O lines.

Sample Time — Enter a base sample time or a multiple of the base sample time.

Base Address or Carrier Board (ie: 0xd000) — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-20098C Digital Output

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of digital output lines used with this port.

Number lines beginning with 1 even if the board manufacturer starts numbering lines with 0.

Port Number — From the list, choose either **A** or **B** to identify the port used with this block of I/O lines.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base Address or Carrier Board (ie: 0xd000) — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

ComputerBoards (Measurement Computing)

I/O boards supported by xPC Target

CIO-DAS1602/16 (p. 11-54)	I/O board with 16 single or 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, 32 digital I/O lines, and 3 counters.
CIO-DDA06 (/12) (p. 11-60)	I/O board with 6 analog output (D/A) channels, and 24 digital I/O lines.
CIO-DDA06/16 (p. 11-66)	I/O board with 6 analog output (D/A) channels, and 24 digital I/O lines.
CIO-DIO24 (p. 11-72)	I/O board with 24 digital I/O lines.
CIO-DIO24H (p. 11-76)	I/O board with 24 digital I/O lines.
CIO-DIO48 (p. 11-79)	I/O board with 48 digital I/O lines.
CIO-DIO48H (p. 11-83)	I/O board with 48 digital I/O lines.
CIO-DIO96 (p. 11-87)	I/O board with 96 digital I/O lines.
CIO-DIO192 (p. 11-91)	I/O board with 192 digital I/O lines.
CIO-DO24DD (p. 11-95)	I/O board with 24 open-collector digital output lines.
CIO-PDISO16 (p. 11-97)	I/O board with 16 isolated digital input lines and 16 relay driven digital output lines.
CIO-QUAD02 (p. 11-100)	24-bit counting board with 2 channels. This board typically connects to incremental encoders.
CIO-QUAD04 (p. 11-103)	24-bit counting board with 4 channels. This board typically connects to incremental encoders.
PC104-DAC06 (/12) (p. 11-106)	I/O board with 6 analog output (D/A) channels.
PC104-DAS16JR/12 (p. 11-109)	I/O board with 16 single or 8 differential analog input channels, 4 digital input lines and 4 digital output lines.
PC104-DAS16JR/16 (p. 11-113)	I/O board with 16 single or 8 differential analog input (A/D) channels, 4 digital input lines and 4 digital output lines.
PC104-DIO48 (p. 11-117)	I/O board with 48 digital I/O lines.
PCI-CTR05 (p. 11-121)	I/O board with 5 counter/timer channels.

PCIM-DAS1602/16 (p. 11-215)	I/O board with 16 single or 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 24 digital input and output lines and 3 counters.
PCIM-DDA06/16 (p. 11-222)	I/O board with 6 analog output (D/A) channels, and 24 digital I/O lines.
PCI-DUAL-AC5 (p. 11-228)	I/O board with 48 digital I/O lines.
PCI-QUAD04 (p. 11-232)	24-bit counting board with 4 channels. This board typically connects to incremental encoders.
PCI-DAS-TC (p. 11-235)	I/O board with 16 differential analog thermocouple input channels. The thermocouple signals are converted by a high frequency synchronous V-F A/D converter.

CIO-CTR05

The CIO-CTR05 is an I/O board with five counter/timer channels (16-bit).

It contains one AM9513A counter/timer chip. For additional information about the various counter/timer modes of that chip see the AM9513A data sheet which is part of the board documentation.

xPC Target supports this board with these driver blocks:

- "CIO-CTR05 Counter PWM"
- "CIO-CTR05 Counter PWM & ARM"
- "CIO-CTR05 Counter FM"
- "CIO-CTR05 Counter FM & ARM"
- "CIO-CTR05 PWM Capture"
- "CIO-CTR05 FM Capture"
- "CIO-CTRxx"

Board Characteristics

Board name	CIO-CTR05
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-CTR05 Counter PWM

The CIOCTR05 has one AM9513A chip with five counters.

The CIO-CTR05 PWM driver programs the AM9513A for PWM (Pulse Width Modulation) signal generation (a square wave with fixed frequency and variable duty cycle). The block has one input which defines the variable duty cycle between 0 and 1. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value between 0 and 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the **Frequency base** and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR05 Counter PWM & ARM

The CIO-CTR05 has one AM9513A chip with five counters.

The CIO-CTR05 PWM & ARM driver programs the AM9513A for PWM or disarmed signal generation (a square wave with fixed frequency and variable duty cycle). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Duty cycle: double Arm: double	0 to 1 <0.5 disarmed $\geq$0.5 armed

Driver Block Parameters

Counter — **F**rom the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is

assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the **Frequency base** and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR05 Counter FM

The CIO-CTR05 has one AM9513A chip with five counters.

The CIO-CTR05 FM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and

variable frequency). For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR05 Counter FM & ARM

The CIO-CTR05 has one AM9513A chip with five counters.

The CIO-CTR05 FM & ARM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Variable frequency: double Arm: double	<0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4` or, `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.

- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the **Output duty cycle** defined in the setting above define the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR05 PWM Capture

This block programs the AMD9513A for capturing PWM signals by using two counters. One counter measures the cycle duration, and the other counter measures the duration the signal is high.

There are two outputs. One output is the relative frequency compared to the base frequency. The other output is the duty cycle. To get the actual frequency, multiply the base frequency by the relative frequency.

The PWM signal has to enter the pins named GATE of both corresponding counter channels (parallel wiring). Both CLK pins have to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1&2`, `2&3`, `3&4`, `4&5`. This selects which two counters the driver block uses to determine the PWM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR05 FM Capture

This block programs the AMD9513A for capturing FM signals.

There is one output for relative frequency compared to the base frequency. To get the actual frequency, multiply the base frequency by the relative frequency.

The FM signal has to enter the pin named GATE of the corresponding counter channel. The CLK pin has to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5`. This selects which counter the driver block uses to determine the FM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTRxx

You can use this block to program the AMD9513A counter. The PWM, PWM & ARM, FM, FM & ARM, PWM Capture, and FM Capture blocks use this block in their underlying subsystems. The API for this block is not currently documented.

CIO-CTR10

The CIO-CTR10 is an I/O board with ten counter/timer channels (16-bit).

It contains one AM9513A counter/timer chip. For additional information about the various counter/timer modes of that chip see the AM9513A data sheet which is part of the board documentation.

xPC Target supports this board with these driver blocks:

- "CIO-CTR10 Counter PWM"
- "CIO-CTR10 Counter PWM & ARM"
- "CIO-CTR10 Counter FM"
- "CIO-CTR10 Counter FM & ARM"
- "CIO-CTR10 PWM Capture"
- "CIO-CTR10 FM Capture"
- "CIO-CTRxx"

Board Characteristics

Board name	CIO-CTR05
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-CTR10 Counter PWM

The CIOCTR10 has one AM9513A chip with ten counters.

The CIO-CTR10 PWM driver programs the AM9513A for PWM (Pulse Width Modulation) signal generation (a square wave with fixed frequency and variable duty cycle). The block has one input which defines the variable duty cycle between 0 and 1. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value between 0 and 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the **Frequency base** and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR10 Counter PWM & ARM

The CIO-CTR10 has two AM9513A chip with ten counters.

The CIO-CTR10 PWM & ARM driver programs the AM9513A for PWM or disarmed signal generation (a square wave with fixed frequency and variable duty cycle). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Duty cycle: double Arm: double	0 to 1 <0.5 disarmed ≥ 0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the **Frequency base** and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave - From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR10 Counter FM

The CIO-CTR10 has two AM9513A chip with ten counters.

The CIO-CTR05 FM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR10 Counter FM & ARM

The CIO-CTR10 has two AM9513A chips with ten counters.

The CIO-CTR10 FM & ARM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Variable frequency: double Arm: double	<0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`,`5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the **Output duty cycle** defined in the setting above define the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR10 PWM Capture

This block programs the AMD9513A for capturing PWM signals by using two counters. One counter measures the cycle duration, and the other counter measures the duration the signal is high.

There are two outputs. One output is the relative frequency compared to the base frequency. The other output is the duty cycle. To get the actual frequency, multiply the base frequency by the relative frequency.

The PWM signal has to enter the pins named GATE of both corresponding counter channels (parallel wiring). Both CLK pins have to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter - From the list, choose `1&2`, `2&3`, `3&4`, `4&5`, `5&6`, `6&7`, `7&8`, `8&9`, `9&10`. This selects which two counters the driver block uses to determine the PWM. In each case, one block is needed for each counter.

Frequency base - From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR10 has to be in position 1MHz not 5MHz.

Sample time - Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address - Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTR10 FM Capture

This block programs the AMD9513A for capturing FM signals.

There is one output for relative frequency compared to the base frequency. To get the actual frequency, multiply the base frequency by the relative frequency.

The FM signal has to enter the pin named GATE of the corresponding counter channel. The CLK pin has to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose 1, 2, 3, 4, 5, 6, 7, 8, 9, or 10. This selects which counter the driver block uses to determine the FM. In each case, one block is needed for each counter.

Frequency base — From the list, choose F1=1MHz, F2=100kHz, F3=10kHz, F4=1kHz, or F5=100Hz to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR10 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-CTRxx

You can use this block to program the AMD9513A counter. The PWM, PWM & ARM, FM, FM & ARM, PWM Capture, and FM Capture blocks use this block in their underlying subsystems. The API for this block is not currently documented.

CIO-DAC08 (/12)

The CIO-DAC08 (/12) is an I/O board with eight analog output (D/A) channels (12-bit).

xPC Target supports this board with this driver block:

- "CIO-DAC08 Analog Output (D/A)"

Board Characteristics

Board name	CIO-DAC08 (CIO-DAC08/12)
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DAC08 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8. This board allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Range code for each of the channels in the channel vector. The range vector must have the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAC08/16

The CIO-DAC08/16 is an I/O board with 8 analog output (D/A) channels (16-bit).

xPC Target supports this board with this driver block:

- "CIO-DAC08/16 Analog Output (D/A)"

Board Characteristics

Board name	CIO-DAC08/16
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DAC08/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8. This board allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAC16 (/12)

The CIO-DAC016 is an I/O board with 16 analog output (D/A) channels (12-bit).

xPC Target supports this board with one driver block:

- "CIO-DAC16 Analog Output (D/A)"

Board Characteristics

Board name	CIO-DAC16 (/12)
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DAC16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 16. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2.5 to +2.5	-2.5	0 to 2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The jumpers by the range DIP switches on the board all have to be in the XFER position. The Wait-State jumper has to be in the off position.

CIO-DAC16/16

The CIO-DAC16/16 is an I/O board with 16 analog output (D/A) channels (16-bit).

xPC Target supports this board with this driver block:

- "CIO-DAC16/16 Analog Output (D/A)"

Board Characteristics

Board name	CIO-DAC08/16
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DAC16/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This board allows the selection of individual A/D channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2.5 to +2.5	-2.5	0 to 2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board.

Reset vector – The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector – The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address - Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The jumpers by the range DIP switches on the board all have to be in the XFER position. The Wait-State jumper has to be in the off position.

CIO-DAS16/330

The CIO-DAS16/330 is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 330 kHz, 4 digital input lines, and 4 digital output lines.

xPC Target supports this board with one driver block:

- "CIO-DAS16/330 Analog Input (A/D)"

Note xPC Target does not support the digital I/O on this board.

Board Characteristics

Board name	CIO-DAS16/330
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

CIO-DAS16/330 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channel used. If differential is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of different range for each channel.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS16/JR (/12)

The CIO-DAS16/JR is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 130 kHz, 4 digital input lines, 4 digital output lines, and 3 counter/timers (16-bit). An external signal conditioning board can be added to the CIO-DAS16/JR board.

xPC Target supports this board with these driver blocks:

- "CIO-DAS16/JR Analog Input (A/D)"
- "CIO-DAS16/JR (/12) Analog Input (A/D) with EXP Signal Conditioning Board"

Note xPC Target does not support the digital I/O or counters on this board.

Board Characteristics

Board name	CIO-DAS16/JR
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

CIO-DAS16/JR Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS16/JR (/12) Analog Input (A/D) with EXP Signal Conditioning Board

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

There are signal conditioning boards (external devices) available from ComputerBoards which can be connected to the CIO-DAS16/JR. Each EXP-board contains its own multiplexer circuit which multiplexes a maximum number of 16 EXP-channels to one A/D-channel of the CIO-DAS16/JR. For this type of operation the CIO-DAS16/JR has to be setup for single-ended input mode and this results in a theoretical number of 256 EXP-channels per CIO-DAS/16JR board.

- EXP16
- EXP32
- EXP-BRIDGE16
- EXP-RTD
- EXP-GP

Driver Block Parameters

EXP Channel vector — This parameter describes the EXP-channels used. Because always a group of 16 EXP-channels are mapped to one A/D-channel of the CIO-DAS16/JR the EXP-channel vector can contain elements between 0 and 15 and no value should occur twice. The number of elements of the vector defines the number of block outputs. The EXP-channel defined as the first element is output at the first block output, the EXP-channel defined as the second element is output at the second block output and so on.

```
Example:EXP Channel Vector:[4,0,12]
the Signal of EXP-channel 4 is output at block outut 1
the Signal of EXP-channel 0 is output at block outut 2
the Signal of EXP-channel 12 is output at block outut 3
```

Note If a EXP32 is used and the EXP-channels 16 to 31 should be acquired, the elements of the EXP Channel Vector have still to be in the range of 0 to 15. Therefore the EXP-channel numbers have to be subsaturated by the constant 16.

A special case is provided by setting the EXP Channel Vector to an empty vector. In this case it is assumed that no EXP-board is connected to the specified A/D-channel (see dialog field A/D Board Channel) and the signal is directly connected to the A/D-input of the CIO-DAS16Jr board. This feature allows to use the A/D-channels of a CIO-DAS16Jr either for EXP-channels or for direct input. Therefore it is not necessary to purchase another A/D-board for direct input.

Note This feature should only be used if at least one EXP-board has to be connected to the CIO-DAS16Jr. If all inputs are directly connected to the A/D board use the CIO-DAS16Jr/12 (2.2.1) driver instead which allows much higher sample rates.

EXP Gain — This parameter describes the gains for each EXP-channel used. This vector corresponds over his indices with the EXP-gain vector and must therefore have the same length. Because this I/O-driver can be used together with all different EXP-boards there is no restriction about the gain value itself. The EXP-board manual should be contacted to know what the gains of the different EXP-boards are. The gains on the EXP-board depend on several DIP-switches on the specific EXP-board.

```
Example:EXP Channel Vector:[4,0,12]
        EXP Gain Vector:[1,1000,200]
EXP-channel 4 has gain 1, channel 0 gain 1000 and channel 12 gain
200
```

If EXP Channel Vector is an empty vector EXP Gain Vector has to be an empty vector as well.

A/D Board Channel — This field specifies to which A/D-channel of the CIO-DAS16Jr the block of 16 EXP-channels are mapped. Because the input coupling of the A/D board has to be single-ended channel 0 to 16 can be used. The channel selection jumpers on the EXP-boards have to be set accordingly to this software setting.

A/D Board Range — This field specifies the input voltage range for the CIO-DAS16/JR which is the same for all 16 single-ended channels.

From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of different range for each channel.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address - Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Note If this driver is used the input coupling switch on the CIO-DAS16Jr has always to be in the 16 (single-ended) position.

CIO-DAS16JR/16

The CIO-DAS16JR/16 is an I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100 kHz, 4 digital input lines, 4 digital output lines and 3 counter/timers.

xPC Target supports this board with this driver block:

- "CIO-DAS16JR/16 Analog Input (A/D)"

Note xPC Target does not support the digital I/O or the counters on this board.

Board Characteristics

Board name	CIO-DAS16JR/16
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

CIO-DAS16JR/16 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1601/12

The CIO-DAS1601/12 is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sampling rate of 160 kHz, 2 analog output (D/A) channels (12-bit), 32 digital input and output lines, and 3 counters (16-bit).

xPC Target supports this board with these driver blocks:

- "CIO-DAS1601/12 Analog Input (A/D)"
- "CIO-DAS1601/12 Analog Output (D/A)"
- "CIO-DAS1601/12 Digital Input"
- "CIO-DAS1601/12 Digital Output"

Note xPC Target supports only 24 digital I/O lines and does not support the counters on this board.

Board and Driver Block Characteristics

Board name	CIO-DAS1601/12
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	A/D: No, D/A: Yes, Digital I/O: Yes
Multiple board support	Yes

CIO-DAS1601/12 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

If a bipolar range is used, the bipolar switch on the board must be in the bipolar position. if a unipolar range is used, the bipolar switch must be in the unipolar position.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1601/12 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1601/12 Digital Input

The DAS1601/12 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1601/12 Digital Output

The DAS1601/12 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1602/12

The CIO-DAS1602/12 is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sampling rate of 100kHz, 2 analog output (D/A) channels (12-bit), 32 digital input and output lines, and 3 counters (16-bit).

xPC Target supports this board with these driver blocks:

- "CIO-DAS1602/12 Analog Input (A/D)"
- "CIO-DAS1602/12 Analog Output (D/A)"
- "CIO-DAS1602/12 Digital Input"
- "CIO-DAS1602/12 Digital Output"

Note xPC Target supports only 24 digital I/O lines and does not support the counters on this board.

Board and Driver Block Characteristics

Board Name	CIO-DAS1602/12
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	A/D: No, D/A: Yes, Digital I/O: Yes
Multiple board support	Yes

CIO-DAS1602/12 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

If a bipolar range is used, the bipolar switch on the board must be in the bipolar position. if a unipolar range is used, the bipolar switch must be in the unipolar position.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1602/12 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be planted according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1602/12 Digital Input

The DAS1601/12 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1602/12 Digital Output

The DAS1601/12 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1602/16

The CIO-DAS1602/16 is an I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sampling rate of 100kHz, 2 analog output (D/A) channels (12-bit), 32 digital I/O lines, and 3 counters 916-bit).

xPC Target supports this board with these driver blocks:

- “CIO-DAS1602/16 Analog Input (A/D)”
- “CIO-DAS1602/16 Analog Output (D/A)”
- “CIO-DAS 1602/16 Digital Input”
- “CIO DAS1602/16 Digital Output”

Note xPC Target supports only 24 digital I/O lines and does not support the counters on this board.

Board and Driver Block Characteristics

Board Name	CIO-DAS1602/16
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	A/D: No, D/A: Yes, Digital I/O: Yes
Multiple board support	Yes

CIO-DAS1602/16 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

If a bipolar range is used, the bipolar switch on the board must be in the bipolar position. if a unipolar range is used, the bipolar switch must be in the unipolar position.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS1602/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be planted according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DAS 1602/16 Digital Input

The DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO DAS1602/16 Digital Output

The DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either **A**, **B**, or **C**. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DDA06 (/12)

The CIO-DDA06 (/12) is an I/O board with 6 analog output (D/A) channels (12-bit), and 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "CIO-DDA06 (/12) Analog Output (D/A)"
- "CIO-DDA06 (/12) Digital Input"
- "CIO-DDA06 (/12) Digital Output"

Board Characteristics

Board name	CIO-DDA06 (/12)
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DDA06 (/12) Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 6. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.67 to +1.67	-1.67	0 to +1.67	1.67
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. The jumpers by the range DIP-switches on the board all have to be in the XFER position. The Wait-State jumper has to be in the off position.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DDA06 (/12) Digital Input

The CIO-DDA06 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DDA06 (/12) Digital Output

The CIO-DDA06 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address - Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DDA06/16

The CIO-DDA06/16) is an I/O board with 6 analog output (D/A) channels (12-bit), and 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- “CIO-DDA06/16 Analog Output (D/A)”
- “CIO-DDA06/16 Digital Input”
- “CIO-DDA06/16 Digital Output”

Board Characteristics

Board name	CIO-DDA06/16
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DDA06/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 6. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.67 to +1.67	-1.67	0 to +1.67	1.67
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. The jumpers by the range DIP-switches on the board all have to be in the XFER position. The Wait-State jumper has to be in the off position.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DDA06/16 Digital Input

The CIO-DDA06/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DDA06/16 Digital Output

The CIO-DDA06/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO24

The CIO-DIO24 is an I/O board with 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "CIO-DIO24 Digital Input"
- "CIO-DIO24 Digital Output"
- "CIO-DIO24 Signal Conditioning"

Board Characteristics

Board name	CIO-DIO24
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DIO24 Digital Input

The CIO-DIO24 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO24 Digital Output

The CIO-DIO24 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO24 Signal Conditioning

Block Parameters

Genix initialization file (path\file) — Provide the filename of the Genix initialization file.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO24H

The CIO-DIO24H is an I/O board with 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- “CIO-DIO24H Digital Input”
- “CIO-DIO24H Digital Output”.

Board Characteristics

Board name	CIO-DIO24H
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DIO24H Digital Input

The CIO-DIO24H has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO24H Digital Output

The CIO-DIO24H has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO48

The CIO-DIO48 is an I/O board with 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "CIO-DIO48 Digital Input"
- "CIO-DIO48 Digital Output"

Board Characteristics

Board name	CIO-DIO48
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DIO48 Digital Input

The CIO-DIO48 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO48 Digital Output

The CIO-DIO48 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO48H

The CIO-DIO48H is an I/O board with 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "CIO-DIO48H Digital Input"
- "CIO-DIO48H Digital Output"

Board Characteristics

Board name	CIO-DIO48H
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DIO48H Digital Input

The CIO-DIO48H has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO48H Digital Output

The CIO-DIO48H has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose 1 or 2. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter 0x300

CIO-DIO96

The CIO-DIO96 is an I/O board with 96 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "CIO-DIO96 Digital Input"
- "CIO-DIO96 Digital Output"

Board Characteristics

Board name	CIO-DIO96
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DIO96 Digital Input

The CIO-DIO96 has four 8255 chips (1,2,3,4). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1`, `2`, `3`, or `4`. The **Chip** parameter defines which of the four 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO96 Digital Output

The CIO-DIO96 has four 8255 chips (1,2,3,4). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1`, `2`, `3`, or `4`. The **Chip** parameter defines which of the four 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO192

The CIO - DIO192 is an I/O board with 192 digital I/O lines.

xPC Target supports this board with these driver blocks:

- “CIO-DIO192 Digital Input”
- “CIO-DIO192 Digital Output”

Board Characteristics

Board name	CIO-DIO192
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DIO192 Digital Input

The CIO-DIO96 has eight 8255 chips (1,2,3,4,5,6,7,8). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, or `8`. The **Chip** parameter defines which of the eight 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DIO192 Digital Output

The CIO-DIO192 has eight 8255 chips (1,2,3,4,5,6,7,8). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, or `8`. The **Chip** parameter defines which of the eight 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-DO24DD

The CIO-DO24DD is an I/O board with 24 open-collector digital output lines.

xPC Target supports this board with this driver block:

- "CIO-DO24DD Digital Output"

Board Characteristics

Board name	CIO-DO24DD
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-DO24DD Digital Output

The CIO-DIO24DD has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that are configured as outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

CIO-PDISO16

The CIO-PDISO16 is an I/O board with 16 isolated digital input lines and 16 relay driven digital output lines.

xPC Target supports this board with these driver blocks:

- "CIO-PDISO16 Digital Input"
- "CIO-PDISO16 Digital Output"

Note xPC Target does not support the 16 relays on this board.

Board Characteristics

Board name	CIO-PDISO16
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-PDISO16 Digital Input

The CIO-PDISO16 has two independent connectors. Each connector has 8 digital input lines.

Use a separate diver block for each connector.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
5 to 24 volts DC/AC	double	~0 volts = 0.0 5 to 24 volts = 1.0

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of digital input lines used with this connector. This driver does not allow the selection of individual digital input lines.

Section(Connector) — From the list, choose either `1 (nearest to backplate)` or `2 (farthest from backplate` to select the connector used.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The Wait-State jumper has to be in the off position.

The switch and jumper settings, that are not mentioned here, have no influence on the running of xPC Target.

CIO-PDISO16 Digital Output

The CIO-PDISO16 has two independent connectors. Each connector has 8 relay driven digital input lines.

Use a separate diver block for each connector.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
relay	double	< 0.5 = Relay open ≥ 0.5 = Relay closed

Driver Block Parameters

Number of Channels — Enter a number between 1 and 8 to select the number of digital output lines used with this connector. This driver does not allow the selection of individual digital output lines.

Section(Connector) — From the list, choose either `1 (nearest to backplate)` or `2 (farthest from backplate` to select the connector used.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The Wait-State jumper has to be in the off position.

The switch and jumper settings, that are not mentioned here, have no influence on the running of xPC Target.

CIO-QUAD02

The CIO-QUAD02 is a 24-bit counting board with 2 channels. This board typically connects to incremental encoders. Incremental encoders convert physical motion into electrical pulses than can be used to determine velocity, direction, and distance.

xPC Target supports this board with this driver block:

- "CIO-QUAD02 Incremental Encoder"

Board Characteristics

Characteristic	
Board name	CIO-QUAD02
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-QUAD02 Incremental Encoder

This driver block has three block outputs: **Angle**, **Turns**, and **Init**.

You can use **Init** to determine when the block output values are valid. **Init** is first set to 0. When the encoder reached the first index, **Init** is set to 1. From then on, you can determine the exact position, direction, and velocity. **Init** remains 1 unless **Counter Reset by Index** is set to **First Only**, and the counter detects a rollover. For more information, see "Driver Block Parameters" on page 11-101.

Turns is the number of complete revolutions made by the encoder. **Angle** is the amount the encoder turns since the last full revolution.

The distance is given by:

```
distance = 2 * pi * Turns + Angle
```

The velocity is given by:

```
velocity = (distance(tS) - distance(tS-1)) / tS
```

The direction is given by:

```
direction = distance(tS) - distance(ts-1)
```

A negative value is reverse, while a positive value is forward.

Driver Block Parameters

Function module — From the list choose `1` or `2`. This parameter specifies which channel you use for this block. For the same board (same base address) two blocks cannot have the same channel number.

Counter Reset by Index — From the list choose either `Only First`, or `Continuous`.

If you choose `Only First`, the first time the encoder reaches the index, the counter is reset to zero, the **Init** output signal is set to 1, and the encoder begins to count. The encoder ignores all other times it reaches the index. **Init** remains `1` until a rollover is detected, and then set to -1. A rollover is when the counter reaches its maximum value and begins to start counting at zero again. A rollover can also occur when the counter reaches its minimum value and the counter resets itself to the maximum value and resumes counting down. The outputs are still accurate after rollover. The -1 flag is used to alert that a rollover has occurred.

If you choose `Continuous`, the first time the encoder reaches the index, the counter is reset to zero, the **Init** output signal is set to 1, and the encoder begins to count. Each time the encoder reaches the index, it resets to zero. **Init** remains always at l because a rollover cannot occur.

Positive Rotation — From the list, choose either `Clockwise` or `Counter Clockwise`. This parameter sets the direction for positive rotation. If you choose `Clockwise`, when the encoder is turned clockwise it counts up, and when turned counter clockwise it counts down. If you choose `Counter Clockwise` the counting direction is reversed.

Mode — From the list, choose `Single`, `Double`, or `Quadruple`. This parameter specifies the phase detection mode. That is, how many phase changes are detected. For more information, see the board manual.

Resolution — This field specifies the divisions of the connected incremental encoder for one revolution.

CIO-QUAD04

The CIO-QUAD04 is a 24-bit counting board with 4 channels. This board typically connects to incremental encoders. Incremental encoders convert physical motion into electrical pulses than can be used to determine velocity, direction, and distance.

xPC Target supports this board with this driver block:

- "CIO-QUAD04 Incremental Encoder"

Board Characteristics

Board name	CIO-QUAD04
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

CIO-QUAD04 Incremental Encoder

This driver block has three block outputs: **Angle**, **Turns**, and **Init**.

You can use **Init** to determine when the block output values are valid. **Init** is first set to `0`. When the encoder reached the first index, **Init** is set to `1`. From then on, you can determine the exact position, direction, and velocity. **Init** remains `1` unless **Counter Reset by Index** is set to **First Only**, and the counter detects a rollover. For more information, see "Driver Block Parameters" on page 11-101.

Turns is the number of complete revolutions made by the encoder. **Angle** is the amount the encoder turns since the last full revolution.

The distance is given by:

```
distance = 2 * pi * Turns + Angle
```

The velocity is given by:

```
velocity = (distance(ts) - distance(ts-1)) / ts
```

The direction is given by:

```
direction = distance(ts) - distance(ts-1)
```

A negative value is reverse, while a positive value is forward.

Driver Block Parameters

Function module — From the list choose, **1**, **2**, **3**, or **4**. This parameter specifies which channel you use for this block. For the same board (same base address) two blocks cannot have the same channel number.

Counter Reset by Index — From the list choose either `Only First`, or `Continuous`.

If you choose `Only First`, the first time the encoder reaches the index, the counter is reset to zero, the **Init** output signal is set to 1, and the encoder begins to count. The encoder ignores all other times it reaches the index. **Init** remains `1` until a rollover is detected, and then set to -1. A rollover is when the counter reaches its maximum value and begins to start counting at zero again. A rollover can also occur when the counter reaches its minimum value and the counter resets itself to the maximum value and resumes counting down. The outputs are still accurate after rollover. The -1 flag is used to alert that a rollover has occurred.

If you choose `Continuous`, The first time the encoder reaches the index, the counter is reset to zero, the **Init** output signal is set to 1, and the encoder begins to count. Each time the encoder reaches the index, it resets to zero. **Init** remains always at l because a rollover cannot occur.

Positive Rotation — From the list, choose either `Clockwise` or `Counter Clockwise`. This parameter sets the direction for positive rotation. If you choose `Clockwise`, when the encoder is turned clockwise it counts up, and when turned counter clockwise it counts down. If you choose `Counter Clockwise` the counting direction is reversed.

Mode — From the list, choose `Single`, `Double`, or `Quadruple`. This parameter specifies the phase detection mode. That is, how many phase changes are detected. For more information, see the board manual.

Resolution — This field specifies the divisions of the connected incremental encoder for one revolution.

PC104-DAC06 (/12)

The PC104-DAC06 (12) is an I/O board with 6 analog output (D/A) channels (12-bit).

xPC Target supports this board with this driver block:

- "PC104-DAC06 (/12) Analog Output (D/A)"

Board Characteristics

Board name	PC104-DAC06 (/12)
Manufacturer	Computer Boards
Bus type	ISA (PC104)
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

PC104-DAC06 (/12) Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 6. This board allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the jumper settings on the board.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The jumpers by the range DIP-switches on the board all have to be in the XFER position. The Wait-State jumper has to be in the off position.

PC104-DAS16JR/12

The PC104-DAS16JR/12 is an I/O board with 16 single or 8 differential analog input channels (12-bit) with a maximum sample rate of 150 kHz, 4 digital input lines and 4 digital output lines.

xPC Target supports this board with these driver blocks:

- "PC104-DAS16JR/12 Analog Input (A/D)"
- "PC104-DAS16JR/12 Digital Input"
- "PC104-DAS16JR/12 Digital Output"

Board and Driver Block Characteristics

Board name	PC104-DAS16JR/12
Manufacturer	ComputerBoards
Bus type	ISA (PC104)
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

PC104-DAS16JR/12 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8

to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC104-DAS16JR/12 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Number of Channels — Enter a number between 1 and 4 to select the number of digital input lines used. This driver does not allow the selection of individual digital input lines.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC104-DAS16JR/12 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Number of Channels — Enter a number between 1 and 4 to select the number of digital output lines used. This driver does not allow the selection of individual digital output lines.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The switch and jumper settings, that are not mentioned here, have no influence on the running of xPC Target.

PC104-DAS16JR/16

The PC104-DAS16JR/16 is an I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100 kHz, 4 digital input lines and 4 digital output lines.

xPC Target supports this board with these driver blocks:

- "PC104-DAS16JR/16 Analog Input (A/D)"
- "PC104-DAS16JR/16 Digital Input"
- "PC104-DAS16JR/16 Digital Output"

Board and Driver Block Characteristics

Board name	PC104-DAS16JR/16
Manufacturer	ComputerBoards
Bus type	ISA (PC104)
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

PC104-DAS16JR/16 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8

to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of different range for each channel.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC104-DAS16JR/16 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Number of Channels — Enter a number between 1 and 4 to select the number of digital input lines used. This driver does not allow the selection of individual digital input lines.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC104-DAS16JR/16 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Number of Channels — Enter a number between 1 and 4 to select the number of digital output lines used. This driver does not allow the selection of individual digital output lines.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC104-DIO48

The PC104-DIO48 is an I/O board with 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PC104-DIO48 Digital Input"
- "PC104-DIO48 Digital Output"

Board and Driver Block Characteristics

Board name	PC104-DIO48
Manufacturer	ComputerBoards
Bus type	ISA (PC104)
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

PC104-DIO48 Digital Input

The CIO-DIO48 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs. Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC104-DIO48 Digital Output

The PC104-DIO48 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs. Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel

vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-CTR05

The PCI-CTR05 is an I/O board with 5 counter/timer channels (16-bit).

It contains one AM9513A counter/timer chip. For additional information about the various counter/timer modes of that chip see the AM9513A data sheet which is part of the board documentation.

xPC Target supports this board with these driver blocks:

- "PCI-CTR05 Counter PWM"
- "PCI-CTR05 Counter PWM & ARM"
- "PCI-CTR05 Counter FM"
- "PCI-CTR05 Counter FM & ARM"
- "PCI-CTR05 PWM Capture"
- "PCI-CTR05 FM Capture"
- "PCI-CTRxx"

Board Characteristics

Board name	PCI-CTR05
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-CTR05 Counter PWM

The PCI-CTR05 PWM driver programs the AM9513A for PWM (Pulse Width Modulation) signal generation (a square wave with fixed frequency and variable duty cycle). The block has one input which defines the variable duty cycle between 0 and 1. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value between 0 and 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the **Frequency base** and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-CTR05 Counter PWM & ARM

The PCI-CTR05 has one AM9513A chip with 5 counters.

The PCI-CTR05 PWM & ARM driver programs the AM9513A for PWM or disarmed signal generation (a square wave with fixed frequency and variable duty cycle). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Duty cycle: double Arm: double	0 to 1 <0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose **F2=100kHz** as the **Frequency base** and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-CTR05 Counter FM

The PCI-CTR05 has one AM9513A chip with 5 counters.

The PCI-CTR05 FM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-CTR05 Counter FM & ARM

The PCI-CTR05 has one AM9513A chip with 5 counters.

The PCI-CTR05 FM & ARM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Variable frequency: double Arm: double	<0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the **Output duty cycle** defined in the setting above define the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-CTR05 PWM Capture

This block programs the AMD9513A for capturing PWM signals by using two counters. One counter measures the cycle duration, and the other counter measures the duration the signal is high.

There are two outputs. One output is the relative frequency compared to the base frequency. The other output is the duty cycle. To get the actual frequency, multiply the base frequency by the relative frequency.

The PWM signal has to enter the pins named GATE of both corresponding counter channels (parallel wiring). Both CLK pins have to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1&2`, `2&3`, `3&4`, `4&5`. This selects which two counters the driver block uses to determine the PWM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is

assumed to be 1MHz, therefore the jumper on the PCI-CTR05 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-CTR05 FM Capture

This block programs the AMD9513A for capturing FM signals.

There is one output for relative frequency compared to the base frequency. To get the actual frequency, multiply the base frequency by the relative frequency.

The FM signal has to enter the pin named GATE of the corresponding counter channel. The CLK pin has to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4` or `5`. This selects which counter the driver block uses to determine the FM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is

assumed to be 1MHz, therefore the jumper on the PCI-CTR05 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-CTRxx

You can use this block to program the AMD9513A counter. The PWM, PWM & ARM, FM, FM & ARM, PWM Capture, and FM Capture blocks use this block in their underlying subsystems. The API for this block is not currently documented.

PCI-DAS1200

The PCI-DAS1200 is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 330 kHz, 2 analog output (D/A) channels (12-bit), and 24 digital input and output lines.

xPC Target supports this board with these driver blocks:

- "PCI-DAS1200 Analog Input (A/D)"
- "PCI-DAS1200 Analog Output (D/A)"
- "PCI-DAS1200 Digital Input"
- "PCI-DAS1200 Digital Output"

Board Characteristics

Board name	PCI-DAS1200
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	A/D: No, D/A: Yes, Digital I/O: Yes
Multiple board support	Yes

PCI-DAS1200 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of different range for each channel.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1200 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1200 Digital Input

The PCI-DAS1200 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1200 Digital Output

The PCI-DAS1200 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1200/JR

The PCI-DAS1200/JR is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 330 kHz, and 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DAS1200/JR Analog Input (A/D)"
- "PCI-DAS1200/JR Digital Input"
- "PCI-DAS1200/JR Digital Output"

Board Characteristics

Board name	PCI-DAS1200/JR
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	A/D: No, Digital I/O: Yes
Multiple board support	Yes

PCI-DAS1200/JR Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of Channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If differential is chosen from the MUX list, then enter a number between 1 and 8

to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of different range for each channel.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. Your choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1200/JR Digital Input

The PCI-DAS1200/JR has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1200/JR Digital Output

The PCI-DAS1200 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1602/12

The PCI-DAS1602/12 is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sampling rate of 200kHz, 2 analog output (D/A) channels (12-bit), and 24 digital input and output lines and 3 counters (16-bit).

xPC Target supports this board with these driver blocks:

- "PCI-DAS1602/12 Analog Input (A/D)"
- "PCI-DAS1602/12 Analog Output (D/A)"
- "PCI-DAS 1602/12 Digital Input"
- "PCI-DAS1602/12 Digital Output"

Note xPC Target supports only 24 digital I/O lines and does not support the counters on this board.

Board and Driver Block Characteristics

Characteristic	
Board Name	PCI-DAS1602/12
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DAS1602/12 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If `differential` is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

If a bipolar range is used, the bipolar switch on the board must be in the bipolar position. if a unipolar range is used, the bipolar switch must be in the unipolar position.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1602/12 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number channels begin with 1 even if the board manufacturer starts numbering channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be planted according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS 1602/12 Digital Input

The DAS1601/12 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1602/12 Digital Output

The DAS1601/12 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1602/16

The PCI-DAS1602/16 is an I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sampling rate of 200kHz, 2 analog output (D/A) channels (16-bit), and 24 digital input and output lines and 3 counters (16-bit).

xPC Target supports this board with these driver blocks:

- "PCI-DAS1602/16 Analog Input (A/D)"
- "PCI-DAS1602/16 Analog Output (D/A)"
- "PCI-DAS 1602/16 Digital Input"
- "PCI-DAS1602/16 Digital Output"

Note xPC Target supports only 24 digital I/O lines and does not support the counters on this board.

Board and Driver Block Characteristics

Characteristic	
Board Name	PCI-DAS1602/16
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DAS1602/16 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If `differential` is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

If a bipolar range is used, the bipolar switch on the board must be in the bipolar position. if a unipolar range is used, the bipolar switch must be in the unipolar position.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1602/16 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number channels begin with 1 even if the board manufacturer starts numbering channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be planted according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS 1602/16 Digital Input

The DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DAS1602/16 Digital Output

The DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA02/12

The PCI-DDA02/12) is an I/O board with 2 analog output (D/A) channels (12-bit), and 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DDA02/12 Analog Output (D/A)"
- "PCI-DDA02/12 Digital Input"
- "PCI-DDA02/12 Digital Output"

Board Characteristics

Board name	PCI-DDA02/12
Manufacturer	Computer Boards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DDA02/12 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - +10	10
-5 to +5	-5	0 - +5	5
-2.5 to +2.5	-2.5	0 - +2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the

initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA02/12 Digital Input

The PCI-DDA02/12 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose 1 or 2. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA02/12 Digital Output

The PCI-DDA02/12 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA02/16

The PCI-DDA02/16) is an I/O board with 2 analog output (D/A) channels (12-bit), and 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DDA02/16 Analog Output (D/A)"
- "PCI-DDA02/16 Digital Input"
- "PCI-DDA02/16 Digital Output"

Board Characteristics

Board name	PCI-DDA06/12
Manufacturer	Computer Boards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DDA02/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - +10	10
-5 to +5	-5	0 - +5	5
-2.5 to +2.5	-2.5	0 - +2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the

initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA02/16 Digital Input

The PCI-DDA02/16 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose 1 or 2. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA02/16 Digital Output

The PCI-DDA02/16 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA04/12

The PCI-DDA04/12) is an I/O board with 4 analog output (D/A) channels (16-bit), and 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DDA04/12 Analog Output (D/A)"
- "PCI-DDA04/12 Digital Input"
- "PCI-DDA04/12 Digital Output"

Board Characteristics

Board name	PCI-DDA04/12
Manufacturer	Computer Boards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DDA04/12 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 4. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use the first and second analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - +10	10
-5 to +5	-5	0 - +5	5
-2.5 to +2.5	-2.5	0 - +2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the

initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA04/12 Digital Input

The PCI-DDA4/12 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose 1 or 2. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA04/12 Digital Output

The PCI-DDA04/12 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA04/16

The PCI-DDA04/16) is an I/O board with 4 analog output (D/A) channels (16-bit), and 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DDA04/16 Analog Output (D/A)"
- "PCI-DDA04/16 Digital Input"
- "PCI-DDA04/16 Digital Output"

Board Characteristics

Board name	PCI-DDA04/16
Manufacturer	Computer Boards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DDA04/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 4. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use the first and second analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - +10	10
-5 to +5	-5	0 - +5	5
-2.5 to +2.5	-2.5	0 - +2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the

initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA04/16 Digital Input

The PCI-DDA4/16 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA04/16 Digital Output

The PCI-DDA04/16 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA08/12

The PCI-DDA08/12) is an I/O board with 8 analog output (A/D) channels (12-bit), and 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DDA08/12 Analog Output (D/A)"
- "PCI-DDA08/12 Digital Input"
- "PCI-DDA08/12 Digital Output"

Board Characteristics

Board name	PCI-DDA08/12
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DDA08/12 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 8. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use the first and second analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - +10	10
-5 to +5	-5	0 - +5	5
-2.5 to +2.5	-2.5	0 - +2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the

initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA08/12 Digital Input

The PCI-DDA08/12 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA08/12 Digital Output

The PCI-DDA08/12 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA08/16

The PCI-DDA08/16) is an I/O board with 8 analog output (A/D) channels (12-bit), and 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DDA08/16 Analog Output (D/A)"
- "PCI-DDA08/16 Digital Input"
- "PCI-DDA08/16 Digital Output"

Board Characteristics

Board name	PCI-DDA06/12
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DDA08/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 8. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use the first and second analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - +10	10
-5 to +5	-5	0 - +5	5
-2.5 to +2.5	-2.5	0 - +2.5	2.5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be placed according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the

initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA08/16 Digital Input

The PCI-DDA08/16 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DDA08/16 Digital Output

The PCI-DDA08/16 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO24

The PCI-DIO24 is an I/O board with 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DIO24 Digital Input"
- "PCI-DIO24 Digital Output"
- "PCI-DIO24 Signal Conditioning"

Board Characteristics

Board name	PCI-DIO24
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DIO24 Digital Input

The PCI-DIO24 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO24 Digital Output

The PCI-DIO24 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO24 Signal Conditioning

Genix initialization file (path\file) — Provide the filename of the Genix initialization file.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO24H

The PCI-DIO24H is an I/O board with 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DIO24H Digital Input"
- "PCI-DIO24H Digital Output".

Board Characteristics

Board name	PCI-DIO24H
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DIO24H Digital Input

The PCI-DIO24H has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO24H Digital Output

The PCI-DIO24H has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO48H

The PCI-DIO48H is an I/O board with 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- “PCI-DIO48H Digital Input”
- “PCI-DIO48H Digital Output”

Board Characteristics

Board name	PCI-DIO48H
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DIO48H Digital Input

The PCI-DIO48H has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO48H Digital Output

The PCI-DIO48H has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO96H

The PCI-DIO96H is an I/O board with 96 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DIO96H Digital Input"
- "PCI-DIO96H Digital Output"

Board Characteristics

Board name	PCI-DIO96
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DIO96H Digital Input

The PCI-DIO96H has four 8255 chips (1,2,3,4). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1`, `2`, `3`, or `4`. The **Chip** parameter defines which of the four 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO96H Digital Output

The PCI-DIO96H has four 8255 chips (1,2,3,4). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose 1, 2, 3, or 4. The **Chip** parameter defines which of the four 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-PDISO8

The PCI-PDISO8 is an I/O board with eight inputs and eight relay outputs.

xPC Target supports this board with these driver blocks:

- "PCI-PDISO8 Digital Input"
- "PCI-PDISO8 Digital Output"

Board Characteristics

Board name	PCI-PDISO8
Manufacturer	ComputerBoards
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-PDISO8 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
OPTO	double	Low (opto off) = 0.0 High (opto on) = 1.0

The input is rectified before being applied to the opto isolator on the input. With no additional input current limiting resistor, the opto turns on when $|V_{in}| > 3$ volts.

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Filter vector — This vector controls the 5 ms noise filter. It is the same length as the channel vector. A value of 1 turns the filter on, and a value of 0 turns the filter off. If you specify a scalar, this value is applied to all channels.

Port — From the list choose `A` because this board only has eight channels.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-PDISO8 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
RELAY	double	< 0.5 = relay off ≥ 0.5 = relay on

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines to drive. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Port — From the list choose `A` because this board only has eight channels.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-PDISO16

The PCI-PDISO16 is an I/O board with 16 inputs and 16 relay outputs.

xPC Target supports this board with these driver blocks:

- "PCI-PDISO16 Digital Input"
- "PCI-PDISO16 Digital Output"

Board Characteristics

Board name	PCI-PDISO16
Manufacturer	ComputerBoards
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-PDISO16 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
OPTO	double	Low (opto off) = 0.0 High (opto on) = 1.0

The input is rectified before being applied to the opto isolator on the input. With no additional input current limiting resistor, the opto turns on when $|V_{in}| > 3$ volts.

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Filter vector — This vector controls the 5 ms noise filter. It is the same length as the channel vector. A value of 1 turns the filter on, and a value of 0 turns the filter off. If you specify a scalar, this value is applied to all channels.

Port — From the list choose either `A` or `B`. The **Port** parameter defines which port is used for this driver block. Each port has a maximum of 8 digital inputs. More than one block can be connected to the same port, but each channel can only be referenced by one block at a time.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-PDISO16 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
RELAY	double	< 0.5 = relay off ≥ 0.5 = relay on

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines to drive. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A` or `B`. The **Port** parameter defines which port is used for this driver block. Each port has a maximum of 8 digital outputs. More than one output block can be used with the same port, but each output channel can only be controlled by one block at a time.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCIM-DAS1602/16

The PCIM-DAS1602/16 is an I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sampling rate of 100kHz, 2 analog output (D/A) channels (16-bit), and 24 digital input and output lines and 3 counters (16-bit).

xPC Target supports this board with these driver blocks:

- "PCIM-DAS1602/16 Analog Input (A/D)"
- "PCIM-DAS1602/16 Analog Output (D/A)"
- "PCIM-DAS 1602/16 Digital Input"
- "PCIM-DAS1602/16 Digital Output"

Note xPC Target supports only 24 digital I/O lines and does not support the counters on this board.

Board and Driver Block Characteristics

Characteristic	
Board Name	PCIM-DAS1602/16
Manufacturer	ComputerBoards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCIM-DAS1602/16 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of channels — If `single-ended` is chosen from the MUX list, then enter a number between 1 and 16 to select the number of A/D channels used. If `differential` is chosen from the MUX list, then enter a number between 1 and 8 to select the number of A/D channels used. This driver does not allow the selection of individual channels or to mix single-ended and differential inputs.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V`, `+-2.5V`, `+-1.25V`, `+-0.625V`, `0-10V`, `0-5V`, `0-2.5V`, or `0-1.25V`.This driver does not allow the selection of a different range for each channel.

If a bipolar range is used, the bipolar switch on the board must be in the bipolar position. If a unipolar range is used, the bipolar switch must be in the unipolar position.

MUX — From the list, choose either `single-ended(16 channels)` or `differential (8 channels)`. This choice must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCIM-DAS1602/16 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number channels begin with 1 even if the board manufacturer starts numbering channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. Also the Bipolar/ Unipolar jumpers have to be inserted according to the ranges used.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCIM-DAS 1602/16 Digital Input

The DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCIM-DAS1602/16 Digital Output

The PCIM-DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The port name defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCIM-DDA06/16

The PCIM-DDA06/16) is an I/O board with 6 analog output (D/A) channels (16-bit), and 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCIM-DDA06/16 Analog Output (D/A)"
- "PCIM-DDA06/16 Digital Input"
- "PCIM-DDA06/16 Digital Output"

Board Characteristics

Board name	PCIM-DDA06/16
Manufacturer	Computer Boards
Bus type	PCIM
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCIM-DDA06/16 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter Numbers between 1 and 6. This driver allows the selection of individual D/A channels in any order. The number of elements

defines the number of D/A channels used. For example, to use the first and second analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to +5	5
-2.5 to +2.5	-2.5	0 to +2.5	2.5
-1.67 to +1.67	-1.67	0 to +1.67	1.67
-.625 to +.625	-0.625		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the DIP-switch settings on the board. The jumpers by the range DIP-switches on the board all have to be in the XFER position. The Wait-State jumper has to be in the off position.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCIM-DDA06/16 Digital Input

The PCIM-DDA6/16 has a 8255 chip with three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCIM-DDA06/16 Digital Output

The PCIM-DDA06/16 has a 8255 chip with three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DUAL-AC5

The PCI-DUAL-AC5 is an I/O board with 48 digital I/O lines.

xPC Target supports this board with these driver blocks:

- "PCI-DUAL-AC5 Digital Input"
- "PCI-DUAL-AC5 Digital Output"

Board Characteristics

Board name	PCI-DUAL-AC5
Manufacturer	ComputerBoards
Bus type	compact PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DUAL-AC5 Digital Input

The PCI-DUAL-AC5 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DUAL-AC5 Digital Output

The PCI-DUAL-AC5 has two 8255 chips (1,2). Each chip has three ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has two 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1` or `2`. The **Chip** parameter defines which of the two 8255 chips is used.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-QUAD04

The PCI-QUAD04 is a 24-bit counting board with 4 channels. This board typically connects to incremental encoders. Incremental encoders convert physical motion into electrical pulses than can be used to determine velocity, direction, and distance.

xPC Target supports this board with this driver block:

- “CIO-QUAD04 Incremental Encoder”

Board Characteristics

Board name	PCI-QUAD04
Manufacturer	ComputerBoards
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-QUAD04 Incremental Encoder

This driver block has three block outputs: **Angle**, **Turns**, and **Init**.

You can use **Init** to determine when the block output values are valid. **Init** is first set to `0`. When the encoder reached the first index, **Init** is set to `1`. From then on, you can determine the exact position, direction, and velocity. **Init** remains `1` unless **Counting reset by index** is set to `Only First`, and the counter detects a rollover. For more information, see “Driver Block Parameters” on page 11-101.

Turns is the number of complete revolutions made by the encoder. **Angle** is the amount the encoder turns since the last full revolution.

The distance is given by:

```
distance = 2 * pi * Turns + Angle
```

The velocity is given by:

```
velocity = (distance(ts) - distance(ts-1)) / ts
```

The direction is given by:

```
direction = distance(ts) - distance(ts-1)
```

A negative value is reverse, while a positive value is forward.

Driver Block Parameters

Function module — From the list choose 1, 2, 3, or 4. This parameter specifies which channel you use for this block. For the same board (same base address) two blocks cannot have the same channel number.

Counting reset by index — From the list choose Only First, Continuous, or Index input disabled.

If you choose Only First, the first time the encoder reaches the index, the counter is reset to zero, the **Init** output signal is set to 1, and the encoder begins to count. The encoder ignores all other times it reaches the index. **Init** remains 1 until a rollover is detected, and then set to -1. A rollover is when the counter reaches its maximum value and begins to start counting at zero again. A rollover can also occur when the counter reaches its minimum value and the counter resets itself to the maximum value and resumes counting down. The outputs are still accurate after rollover. The -1 flag is used to alert that a rollover has occurred.

If you choose Continuous, The first time the encoder reaches the index, the counter is reset to zero, the **Init** output signal is set to 1, and the encoder begins to count. Each time the encoder reaches the index, it resets to zero. **Init** remains always at l because a rollover cannot occur.

Positive Rotation — From the list, choose either Clockwise or Counter Clockwise. This parameter sets the direction for positive rotation. If you choose Clockwise, when the encoder is turned clockwise it counts up, and when turned counter clockwise it counts down. If you choose Counter Clockwise, the counting direction is reversed.

Mode — From the list, choose Single, Double, or Quadruple. This parameter specifies the phase detection mode. That is, how many phase changes are detected. For more information, see the board manual.

Resolution — This field specifies the divisions of the connected incremental encoder for one revolution.

PCI-DAS-TC

The PCI-DAS-TC is an I/O board with 16 differential analog thermocouple input channels. The thermocouple signals are converted by a high frequency synchronous V-F A/D converter. The board is equipped with its own micro-controller responsible for calibration, cold junction compensation, moving average calculations, conversion, and conversion into physical engineering units. The on-board micro-controller significantly off loads the main CPU.

xPC Target supports this board with this driver block:

- "PCI-DAS-TC Thermocouple"

Board Characteristics

Board name	PCI-DAS-TC
Manufacturer	Computer Boards
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

PCI-DAS-TC Thermocouple

Scaling Input to Output

Hardware Input	Block Input Data Type	Scaling
Volts	double	temperature in either degrees C, K, or F.

Driver Block Parameters

Conversion rate (interrogation time) — From the list, choose either `50Hz`, `60Hz`, or `400 Hz`. This is the conversion rate for the V-F A/D converter. The conversion rate is the same for all input channels.

Number of samples for moving average — From the list, choose a value from **1** to **16**. Converted signal values are put into a cyclic buffer of size **n** which is used to calculate the moving average over these **n** samples.

Number of channels to be acquired (1..n) — From the list, choose a value from **1** to **16**. This is the number of input channels activated for conversion. The first channel of the scan is always and input channel with the number 1 and the last channel has the number **n**.

Vector input thermocouple types (cell array of char) — For each acquired channel, enter a valid type of either `'J'`, `'K'`, `'E'`, `'T'`, `'R'`, `'S'`, or `'B'`. This vector defines the type of thermocouple for each channel. The vector must be the same length as the **Number of channels to be acquired**.

Vector of input gains (double array) — For each acquired channel, enter a valid input gain of either `1`, `125`, `166.7`, or `400`. This vector defines the input gain for each channel. The vector must be the same length as the **Number of channels to be acquired**.

Vector or temperature formats (cell array of char) — For each acquired channel, enter a valid format of either `'C'` or `'F'`. `'C'`= Celsius and `'F'`= Fahrenheit. The vector must be the same length as the **Number of channels to be acquired**.

Read and output CJC temperature — If you want the block to read, convert, and output the temperature of the cold junction (CJC) sensor on the board, select this check box. If selected, the block shows an additional output port with the value of the CJC temperature.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

Note Each time a target application containing this driver block is downloaded to the target PC, the board automatically does a full calibration. Thermocouple sensor calibration is an extensive procedure and because it has to take place for each channel independently, the calibration time can easily exceed several seconds, especially when the number of channels to be acquired is 5 or higher.

Because of this long calibration period during the initialization stage of the target application, the download procedure can time-out and return an error message. To avoid this error, increase the default time-out duration. See "Increasing the Time Out Value" on page 3-33.

12

Contec

I/O boards supported by xPC Target (`http://www.contec.com`)

Contec AD12-16(PCI)

The Contec AD12-16(PCI) is an I/O board with 16 single-ended or 8 differential analog input (A/D) channels (12 bit), 4 digital input lines, and 4 digital output lines.

xPC Target supports this board with these driver blocks:

- "AD12-16(PCI) Analog Input (A/D)"
- "AD12-16(PCI) Digital Input"
- "AD12-16(PCI) Digital Output"

xPC Target does not support the Counter/Timer functionality of the board.

Board Characteristics

Board name	AD12-16(PCI)
Manufacturer	Contec
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

AD12-16(PCI) Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the input channels. For example to use the first three analog input (A/D) channels, enter

```
[1, 2, 3]
```

The channel numbers can occur in any order. Number the them beginning with 1 even if the board manufacturer numbers them beginning with 0.

The maximum allowable channel number for this board is 8 (double-ended) or 16 (single-ended). If the highest channel number you specify is n, the hardware converts all the channels between 1 and n, whether or not they occur in your channel vector. It is most efficient to specify a contiguous range of channels. (Permuting the order of such a range has no impact on efficiency however.)

Range vector — This board allows the range of each channel to be selected independently. Enter a scalar, in which case the same range will be used for all channels, or a vector the same length as the channel vector. The range vector entries must be range codes selected from the following table:.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to 5	5
-2.5 to 2.5	-2	0 to 2.5	2
-1.25 to 1.25	-1	0 to 1.25	1

Polarity — Choose `single-ended` or `double-ended`. This setting applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number, use the command

```
getxpcpci all
```

The Device ID of this board is 8153.

AD12-16(PCI) Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the input channels. For example to use the first and third digital input channels enter

```
[1, 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number use the command

```
getxpcpci all
```

The Device ID of this board is 8153.

AD12-16(PCI) Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 TTL high

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels enter

```
[1, 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block To find the slot number, use the command

```
getxpcpci all
```

The Device ID of this board is 8153.

Contec AD12-64(PCI)

The Contec AD12-64(PCI) is an I/O board with 64 single-ended or 32 differential analog input (A/D) channels (12 bit), 4 digital input lines, and 4 digital output lines.

xPC Target supports this board with these driver blocks:

- "AD12-64(PCI) Analog Input (A/D)"
- "AD12-64(PCI) Digital Input"
- "AD12-64(PCI) Digital Output"

xPC Target does not support the Counter/Timer functionality of this board.

Board Characteristics

Board name	AD12-64(PCI)
Manufacturer	Contec
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

AD12-64(PCI) Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first three analog input (A/D) channels, enter

```
[1, 2, 3]
```

The channel numbers can occur in any order. Number the them beginning with 1 even if the board manufacturer numbers them beginning with 0.

The maximum allowable channel number is for this board is 32 (double-ended) or 64 (single-ended). If the highest channel number you specify is n, the hardware will convert all the channels between 1 and n, whether or not they occur in your channel vector. It is most efficient to specify a contiguous range of channels. (Permuting the order of such a range has no impact on efficiency however.)

Range vector — This board allows the range of each channel to be selected independently. If you enter a scalar, the same range is used for all channels. If you enter a vector, it must be the same length as the channel vector. The range vector entries must be range codes selected from the following table:.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to 5	5
-2.5 to 2.5	-2	0 to 2.5	2
-1.25 to 1.25	-1	0 to 1.25	1

Polarity — Choose `single-ended` or `double-ended`. This setting applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block To find the slot number, use the command

```
getxpcpci all
```

The Device ID of this board is 8143.

AD12-64(PCI) Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels enter

```
[1, 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number use the command

```
getxpcpci all
```

The Device ID of this board is 8143.

AD12-64(PCI) Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 TTL high

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels enter

```
[1, 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values (0 or 1) for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number. use the command

```
getxpcpci all
```

The Device ID of this board is 8143.

Contec DA12-4(PCI)

The Contec DA12-4(PCI) is an I/O board with 4 analog output (D/A) channels (12 bit).

xPC Target supports this board with this driver block:

- “DA12-4(PCI) Analog Output (D/A)”

xPC Target does not support the timer, external trigger, or interrupt functionality of this board.

Board Characteristics

Board name	DA12-4(PCI)
Manufacturer	Contec
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

DA12-4(PCI) Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and second analog output (D/A) channels enter

```
[1, 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number, use the command

```
getxpcpci all
```

The Device ID of this board is 8183.

Contec DA12-16(PCI)

The Contec DA12-16(PCI) is an I/O board with 16 analog output (D/A) channels (12 bit).

xPC Target supports this board with this driver block:

- "DA12-16(PCI) Analog Output (D/A)"

xPC Target does not support the timer, external trigger, or interrupt functionality of this board.

Board Characteristics

Board name	DA12-16(PCI)
Manufacturer	Contec
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

DA12-16(PCI) Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and second analog output (D/A) channels enter

```
[1, 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number, use the command

```
getxpcpci all
```

The Device ID of this board is 8163.

Contec PIO-32/32T(PCI)

The Contec PIO-32/32T(PCI) is an I/O board with 32 digital input channels and 32 output channels.

xPC Target supports this board with these driver blocks:

- "PIO-32/32T(PCI) Digital Input"
- "PIO-32/32T(PCI) Digital Output"

xPC Target does not support the timer, external trigger, or interrupt functionality of this board.

Board Characteristics

Board name	PIO-32/32T(PCI)
Manufacturer	Contec
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

PIO-32/32T(PCI) Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

I/O format — Select `serial` or `parallel`. If you select `serial`, the block is configured to accept up to 32 one-bit input channels. If you select `parallel`, the block is configured to accept a single 32-bit input channel and the channel vector parameter is unavailable.

Channel vector — If you selected `serial` I/O format, enter a vector of numbers to specify the input channels for serial I/O format. For example, to use the first and third digital input channels enter

```
[1, 3]
```

The channel numbers can occur in any order, but the numbers must lie in the range 1 to 32.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number, use the command

```
getxpcpci all
```

The Device ID of this board is 8152.

PIO-32/32T(PCI) Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 TTL high

Driver Block Parameters

I/O format — Select `serial` or `parallel`. If you select `serial`, the block is configured to accept up to 32 one-bit input channels for output. If you select `parallel`, the block is configured to accept a single 32-bit channel and the channel vector parameter is unavailable.

Channel vector — For `serial` I/O format, enter a vector of numbers to specify the output channels. For example to use the first and third digital output channels enter

```
[1, 3]
```

The channel numbers can occur in any order, but the numbers must lie in the range 1 to 32.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values (0 or 1) for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started. If you selected `parallel` I/O format, the values can be in the form [hex2dec('ffffffff')].

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI slot — If only one board of this type is physically present in your target PC, enter

```
-1
```

Otherwise, enter the PCI slot number of the particular board to be controlled by this block. To find the slot number, use the command

```
getxpcpci all
```

The Device ID of this board is 8152.

Data Translation

I/O boards supported by xPC Target. (www.datatranslation.com)

DT2821 (p. 13-3)	I/O board with 16 single-ended or 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2821-F-8DI (p. 13-8)	I/O board with 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2821-G-8DI (p. 13-13)	I/O board with 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2821-F-16SE (p. 13-18)	I/O board with 16 single-ended analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output
DT2821-G-16SE (p. 13-23)	I/O board with 16 single-ended analog input (A/D) channels, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2823 (p. 13-28)	I/O board 4 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2824-PGH (p. 13-33)	I/O board with 16 single-ended or 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

DT2824-PGL (p. 13-37)	I/O board with 16 single-ended or 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2825 (p. 13-41)	I/O board with 16 single-ended or 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2827 (p. 13-46)	I/O board with 4 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.
DT2828 (p. 13-51)	I/O board with 4 single-ended analog input (A/D) channels, 2 analog output (D/A) channels, and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

DT2821

The DT2821 is an I/O board with 16 single-ended or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 50 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- “DT2821 Analog Input (A/D)”
- “DT2821 Analog Output (D/A)”
- “DT2821 Digital Input”
- “DT2821 Digital Output”

Board Characteristics

Board Name	DT2821
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2821 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `Single-ended (16 channels)` from the Coupling list, enter numbers between 1 and 16. If you choose `Differential (8 channels)` from the Coupling list, then enter numbers between 1 and 8. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of `+-10V`. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), or `0-10V` (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Coupling — From the list, choose either `Single-ended(16 channels)` or `Differential (8 channels)`. Your choice must correspond to the input mode setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821 Digital Input

DT2821 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821 Digital Output

DT2821 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-8DI

The DT2821-F-8DI is an I/O board with 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 150 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "DT2821-F-8DI Analog Input (A/D)"
- "DT2821-F-8DI Analog Output (D/A)"
- "DT2821-F-8DI Digital Input"
- "DT2821-F-8DI Digital Output"

Board Characteristics

Board Name	DT2821-F-8DI
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2821-F-8DI Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of `+-10V`. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), or `0-10V` (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-8DI Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-8DI Digital Input

DT2821-F-8DI series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-8DI Digital Output

DT2821-F-8DI series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-8DI

The DT2821-G-8DI is an I/O board with 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 250 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- “DT2821-G-8DI Analog Input (A/D)”
- “DT2821-G-8DI Analog Output (D/A)”
- “DT2821-G-8DI Digital Input”
- “DT2821-G-8DI Digital Output”

Board Characteristics

Board Name	DT2821-G-8DI
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2821-G-8DI Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of `+-10V`. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), `+-5V` (-5 volts to +5 volts), or `0-10V` (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-8DI Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-8DI Digital Input

DT2821-G-8DI series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-8DI Digital Output

DT2821-G-8DI series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-16SE

The DT2821-F-16SE is an I/O board with 16 single-ended analog input (A/D) channels (12-bit) with a maximum sample rate of 150 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "DT2821-F-16SE Analog Input (A/D)"
- "DT2821-F-16SE Analog Output (D/A)"
- "DT2821-F-16SE Digital Input"
- "DT2821-F-16SE Digital Output"

Board Characteristics

Board Name	DT2821-F-16SE
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2821-F-16SE Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of `+-10V`. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), or `0-10V` (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-16SE Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-16SE Digital Input

DT2821-F-16SE series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-F-16SE Digital Output

DT2821-F-16SE series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-16SE

The DT2821-G-16SE is an I/O board with 16 single-ended analog input (A/D) channels (12-bit) with a maximum sample rate of 250 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "DT2821-G-16SE Analog Input (A/D)"
- "DT2821-G-16SE Analog Output (D/A)"
- "DT2821-G-16SE Digital Input"
- "DT2821-G-16SE Digital Output"

Board Characteristics

Board Name	DT2821-G-16SE
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2821-G-16SE Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter 1, 2, 4, or 8 for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of **+-10V**. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either +-10V (-10 volts to +10 volts), +-5V (-5 volts to +5 volts), or 0-10V (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-16SE Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-16SE Digital Input

DT2821-G-16SE series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2821-G-16SE Digital Output

DT2821-G-16SE series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2823

The DT2823 is an I/O board 4 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100 kHz, 2 analog output (D/A) channels (16-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "DT2823 Analog Input (A/D)"
- "DT2823 Analog Output (D/A)"
- "DT2823 Digital Input"
- "DT2823 Digital Output"

Board Characteristics

Board Name	DT2823
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2823 Analog Input (A/D)

The range for the DT2823 is -10 to + 10 volts.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 4. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2823 Analog Output (D/A)

The range of the DT2823 is -10 to +10 volts.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2823 Digital Input

DT2823 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2823 Digital Output

DT2823 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2824-PGH

The DT2824-PGH is an I/O board with 16 single-ended or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 50 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "DT2824-PGH Analog Input (A/D)"
- "DT2824-PGH Digital Input"
- "DT2824-PGL Digital Output"

Board Characteristics

Board Name	DT2824-PGH
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2824-PGH Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `Single-ended (16 channels)` from the Coupling list, enter numbers between 1 and 16. If you choose `Differential (8 channels)` from the Coupling list, then enter numbers between 1 and 8. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of `+-10V`. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), or `0-10V` (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Coupling — From the list, choose either `Single-ended(16 channels)` or `Differential (8 channels)`. Your choice must correspond to the input mode setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

DT2824-PGH Digital Input

DT2824-PGH series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2824-PGH Digital Output

DT2824-PGH series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output.

Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2824-PGL

The DT2824-PGL is an I/O board with 16 single-ended or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 50 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "DT2824-PGL Analog Input (A/D)"
- "DT2824-PGL Digital Input"
- "DT2824-PGL Digital Output"

Board Characteristics

Board Name	DT2824-PGL
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2824-PGL Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `Single-ended (16 channels)` from the Coupling list, enter numbers between 1 and 16. If you choose `Differential (8 channels)` from the Coupling list, then enter numbers between 1 and 8. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `10`, `100`, or `500` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of `+-10V`. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), or `0-10V` (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Coupling — From the list, choose either `Single-ended(16 channels)` or `Differential (8 channels)`. Your choice must correspond to the input mode setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

DT2824-PGL Digital Input

DT2824-PGL series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2824-PGL Digital Output

DT2824-PGL series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output.

Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2825

The DT2825 is an I/O board with 16 single-ended or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 45 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- “DT2825 Analog Input (A/D)”
- “DT2825 Analog Output (D/A)”
- “DT2825 Digital Input”
- “DT2825 Digital Output”

Board Characteristics

Board Name	DT2825
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2825 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If you choose `Single-ended (16 channels)` from the Coupling list, enter numbers between 1 and 16. If you choose `Differential (8 channels)` from the Coupling list, then enter numbers between 1 and 8. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `10`, `100`, or `500` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of `+-10V`. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either `+-10V` (-10 volts to +10 volts), or `0-10V` (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Coupling — From the list, choose either `Single-ended(16 channels)` or `Differential (8 channels)`. Your choice must correspond to the input mode setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2825 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2825 Digital Input

DT2825 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2825 Digital Output

DT2825 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2827

The DT2827 is an I/O board with 4 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- “DT2827 Analog Input (A/D)”
- “DT2827 Analog Output (D/A)”
- “DT2827 Digital Input”
- “DT2827 Digital Output”

Board Characteristics

Board Name	DT2827
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2827 Analog Input (A/D)

The range for this board is -10 to +10 volts.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector —Enter numbers between 1 and 4. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2827 Analog Output (D/A)

The range for this board is -10 to + 10 volts.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2827 Digital Input

DT2827 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2827 Digital Output

DT2827 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2828

The DT2828 is an I/O board with 4 single-ended analog input (A/D) channels (12-bit) with a maximum sample rate of 100 kHz, 2 analog output (D/A) channels (12-bit), and 16 digital I/O lines that can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "DT2828 Analog Input (A/D)"
- "DT2828 Analog Output (D/A)"
- "DT2828 Digital Input"
- "DT2828 Digital Output"

Board Characteristics

Board Name	DT2828
Manufacturer	Data Translation
Bus Type	ISA
Access Method	I/O mapped
Multiple block instance support	A/D:No, D/A:No, Digital I/O:Yes
Multiple board support	Yes

DT2828 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 4. For example, to use the first and third analog output (A/D) channels, enter

```
[1,3]
```

Number the channels beginning with 1 even if the board manufacturer starts to number the channels with 0.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the **Channel vector** to specify the gain for that channel. The gain vector must be the same length as the **Channel vector**. (If your enter a scalar, it is automatically expanded to channel vector). This driver allows the gain of each channel to be different. The gain is applied prior to sampling the voltage.

For example, if you have an input signal from -1 to +1 volts, and a gain of 8, the signal is amplified -8 to +8 volts. Select a range equal or larger than the amplified voltage. For example, select a range of **+-10V**. After the signal voltage is sampled, this block divides by the gain to output the original signal value.

Range — From the list, choose either **+-10V** (-10 volts to +10 volts), or **0-10V** (0 volts to +10 volts). This specifies the effective range which is the same for all channels and must correspond with the input range setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the bass address setting on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2828 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameter

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1, 2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the **Channel vector**. The range vector must be the same length as the **Channel vector**. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +5	5
-5 to +5	-5	0 to +10	10
-2.5 to +2.5	-2.5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

The range settings have to correspond to the Output Range Selection settings on the board for DAC0, and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2828 Digital Input

DT2828 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Port — From the list, choose `1` or `2`.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines you use with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital inputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DT2828 Digital Output

DT2828 series boards have two I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Port — From the list, choose 1 or 2.

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines you use.

For example, to use all of the digital outputs for this port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

14

Diamond

I/O Boards supported by xPC Target.

Diamond-MM

The Diamond-MM is a DAS16 compatible I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate or 100 kHz, 2 analog output (D/A) channels (12-bit), 8 digital input lines, and 8 digital output lines.

xPC Target supports this board with these driver blocks:

- “Diamond-MM Analog Input (A/D)”
- “Diamond-MM Analog Output (D/A)”
- “Diamond-MM Digital Input”
- “Diamond-MM Digital Output”

Board Characteristics

Board name	Diamond-MM
Manufacturer	Diamond Systems Corporation
Bus type	ISA (PC/104)
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

Diamond-MM Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of channels — If you entered 16 in the MUX box, then enter a number between 1 and 16 to select the number of single A/D channels used. If you entered 8 in the MUX box, then enter a number between 1 and 8 to select the number of differential A/D channels used. This driver does not allow the selection of individual channels or a different MUX setting for each channel.

Input range — Enter an input range code for all A/D channels. This driver does not allow the selection of a different range for each channel. The input range is the same for all A/D channels.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +10	10
-5 to +5	-5	0 to +5	5
-2.5 to + 2.5	-2.5	0 to +2	2
-1 to +1	-1	0 to +1	1
-0.5 to +0.5	-5		

The gain jumpers on the board have to be in the correct positions for the chosen range.The bipolar jumper on the board has to be in the bipolar position, if a bipolar range is used or in the unipolar position, when a unipolar range is used.

MUX (16/8) — Enter 16 for single-ended or 8 for differential A/D channels. This entry must correspond to the MUX-switch setting on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Diamond-MM Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — This parameter is a combined Channel vector and Range vector. The number of elements defines the number of D/A channels used.

Enter a range code for each of the D/A channels used. This driver allows a different range for each channel with a maximum of 2 channels.

The following table is a list of the ranges for this driver and the corresponding range codes. The D/A specific jumpers on the board have to be in the correct positions for the ranges entered.

Input Range (V)	Range Code	Input Range (V)	Range Code
		0 to +10	10
		0 to +5	5

For example, if the first channel is 0 to + 10 volts and the second channel is 0 to +5 volts, enter

```
[10,5]
```

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Diamond-MM Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Number of channels — Enter a number between 1 and 8 to select the number of digital input lines used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Diamond-MM Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low $\geq$ 0.5 = TTL high

Driver Block Parameters

Number of channels — Enter a number between 1 and 8 to select the number of digital output lines used.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Diamond-MM-32

The Diamond MM-32 is a PC104 I/O board with 32 single or 16 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 200 kHz, 4 analog output (D/A) channels (12-bit), 24 digital input and output lines.

xPC Target supports this board with these driver blocks:

- “Diamond-MM-32 Analog Input (A/D)”
- “Diamond-MM-32 Analog Output (D/A)”
- “Diamond-MM-32 Digital Input”
- “Diamond-MM-32 Digital Output”

Board Characteristics

Board name	Diamond MM-32
Manufacturer	Diamond Systems Corporation
Bus type	ISA (PC/104)
Access method	I/O mapped
Multiple block instance support	A/D:No, D/A:Yes, DIO:Yes
Multiple board support	Yes

Diamond-MM-32 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

First channel (1..n) — If you select `single-ended` from the MUX list, then enter a number between 1 and 32 to select the first channel. If you select differential from the MUX list, then enter a number between 1 and 16 to select the first channel.

Number of Channels (1..n) — If you select `single-ended` from the MUX list, then enter a number between 1 and 32 to select the first channel. If you select differential from the MUX list, then enter a number between 1 and 16 to select the first channel. This driver does not allow the selection of individual channels or a different MUX setting for each channel.

Range — From the list, choose a range code. This driver does not allow the selection of a different range for each channel. The input range is the same for all A/D channels.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +10	10
-5 to +5	-5	0 to +5	5
-2.5 to + 2.5	-2.5	0 to +2.5	2.5
-1.25 to +1.25	-1	0 to +1.25	1.25
-0.625 to +0.625	-5		

MUX — From the list choose `single-ended (32 channels)` or `differential (16channels)`. This entry must correspond to the MUX jumpers set on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Diamond-MM-32 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 4. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels you use. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with `1` even if the board manufacturer starts numbering the channels with `0`

Range — From the list, choose a range code. This driver does not allows a different range for each or the 4 channels. This selection has to correspond to the range and bipolar/unipolar jumper settings on the board.

The following table is a list of the ranges for this driver and the corresponding range codes. The D/A specific jumpers on the board have to be in the correct positions for the ranges entered.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +10	10
-5 to +5	-5	0 to +5	5

For example, if the first channel is 0 to + 10 volts and the second channel is 0 to +5 volts, enter

```
[10,5]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Diamond-MM-32 Digital Input

The Diamond-MM-32 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Diamond-MM-32 Digital Output

The Diamond-MM-32 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Onyx-MM

The Onyx-MM is an I/O board with 48 digital I/O lines which can be configured in groups of 8 for either input or output. xPC Target does not support the Counter/Timer functionality of this board.

xPC Target supports this board with these driver blocks:

- "Onyx-MM Digital Input"
- "Onyx-MM Digital Output"

Board Characteristics

Board name	Onyx-MM
Manufacturer	Diamond Systems Corporation
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	DIO: Yes
Multiple board support	Yes

Onyx-MM Digital Input

Onyx-MM boards have two digital I/O chips, each with three 8-bit digital I/O ports, for a total of 48 I/O lines. Each port can be configured independently for either input or output. Use a separate driver block for each port. Select the digital input driver block for a given port to configure the port for input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for the current port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose `A`, `B`, or `C`.

Chip — From the list choose `1` or `2`.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the Base address board setting. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Onyx-MM Digital Output

Onyx-MM boards have two digital I/O chips, each with three 8-bit digital I/O ports, for a total of 48 I/O lines. Each port can be configured independently for either input or output. Use a separate driver block for each port. Select the digital output driver block for a given port to configure a port for output.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for the current port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose `A`, `B`, or `C`.

Chip — From the list choose `1` or `2`.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the Base address board setting. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 5

The Quartz-MM 5 has 8 digital input lines, 8 digital output lines, and 10 counter/timers.

xPC Target supports this board with these driver blocks:

- "Quartz-MM 5 Digital Input"
- "Quartz-MM 5 Digital Output"
- "Quartz-MM5 Counter PWM"
- "Quartz-MM5 Counter PWM & ARM"
- "Quartz-MM5 Counter FM"
- "Quartz-MM5 Counter FM & ARM"
- "Quartz-MM5 PWM Capture"
- "Quartz-MM5 FM Capture"
- "Quartz-MMxx"

Board Characteristics

Board name	Quartz-MM 5
Manufacturer	Diamond Systems Corporation
Bus type	ISA (PC/104)
Access method	I/O mapped
Multiple block instance support	
Multiple board support	

Quartz-MM 5 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 5 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM5 Counter PWM

The Quartz-MM5 has one AM9513A chip with 5 counters.

The Quartz-MM5 PWM driver programs the AM9513A for PWM (Pulse Width Modulation) signal generation (a square wave with fixed frequency and variable duty cycle). The block has one input which defines the variable duty cycle between 0 and 1. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value between 0 and 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the Frequency Base and enter `0.175` as the Relative Output Frequency. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM5 Counter PWM & ARM

The Quartz-MM5 has one AM9513A chip with 5 counters.

The Quartz-MM5 PWM & ARM driver programs the AM9513A for PWM or disarmed signal generation (a square wave with fixed frequency and variable duty cycle). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Duty cycle: double Arm: double	0 to 1 <0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is

assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the Frequency Base and enter `0.175` as the Relative Output Frequency. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM5 Counter FM

The Quartz-MM5 has one AM9513A chip with 5 counters.

The Quartz-MM5 FM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and

variable frequency). For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

Quartz-MM5 Counter FM & ARM

The Quartz-MM5 has one AM9513A chip with 5 counters.

The Quartz-MM5 FM & ARM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Variable frequency: double Arm: double	<0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.

- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the **Output duty cycle** defined in the setting above define the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM5 PWM Capture

This block programs the AMD9513A for capturing PWM signals by using two counters. One counter measures the cycle duration, and the other counter measures the duration the signal is high.

There are two outputs. One output is the relative frequency compared to the base frequency. The other output is the duty cycle. To get the actual frequency, multiply the base frequency by the relative frequency.

The PWM signal has to enter the pins named GATE of both corresponding counter channels (parallel wiring). Both CLK pins have to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1&2`, `2&3`, `3&4`, or `4&5`. This selects which two counters the driver block uses to determine the PWM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the Quartz-MM5 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM5 FM Capture

This block programs the AMD9513A for capturing FM signals.

There is one output for relative frequency compared to the base frequency. To get the actual frequency, multiply the base frequency by the relative frequency.

The FM signal has to enter the pin named GATE of the corresponding counter channel. The CLK pin has to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4` or `5`. This selects which counter the driver block uses to determine the FM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the Quartz-MM5 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MMxx

You can use this block to program the AMD9513A counter. The PWM, PWM & ARM, FM, FM & ARM, PWM Capture, and FM Capture blocks use this block in their underlying subsystems. The API for this block is not currently documented.

Quartz-MM 10

The Quartz-MM 10 has 8 digital input line, 8 digital output lines, and 10 counter/timers.

xPC Target supports this board with these driver blocks:

- "Quartz-MM 10 Digital Input"
- "Quartz-MM 10 Digital Output"
- "Quartz-MM 10 Counter PWM"
- "Quartz-MM 10 Counter PWM & ARM"
- "Quartz-MM 10 Counter FM"
- "Quartz-MM 10 Counter FM & ARM"
- "Quartz-MM 10 PWM Capture"
- "Quartz-MM 10 FM Capture"
- "Quartz-MMxx"

Board Characteristics

Board name	Quartz-MM 10
Manufacturer	Diamond Systems Corporation
Bus type	ISA (PC/104)
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

Quartz-MM 10 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 10 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 10 Counter PWM

The Quartz-MM10 has two AM9513A chips with 5 counters each.

The Quartz-MM10 PWM driver programs the AM9513A for PWM (Pulse Width Modulation) signal generation (a square wave with fixed frequency and variable duty cycle). The block has one input which defines the variable duty cycle between 0 and 1. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value between 0 and 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the Frequency Base and enter `0.175` as the Relative Output Frequency. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or **low-high**.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 10 Counter PWM & ARM

The Quartz-MM10 has one AM9513A chip with 5 counters.

The Quartz-MM10 PWM & ARM driver programs the AM9513A for PWM or disarmed signal generation (a square wave with fixed frequency and variable duty cycle). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Duty cycle: double Arm: double	0 to 1 <0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is

assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the Frequency Base and enter `0.175` as the Relative Output Frequency. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 10 Counter FM

The Quartz-MM10 has one AM9513A chip with 5 counters.

The Quartz-MM10 FM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and

variable frequency). For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

Quartz-MM 10 Counter FM & ARM

The Quartz-MM10 has one AM9513A chip with 5 counters.

The Quartz-MM10 FM & ARM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Variable frequency: double Arm: double	<0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.

- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the **Output duty cycle** defined in the setting above define the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 10 PWM Capture

This block programs the AMD9513A for capturing PWM signals by using two counters. One counter measures the cycle duration, and the other counter measures the duration the signal is high.

There are two outputs. One output is the relative frequency compared to the base frequency. The other output is the duty cycle. To get the actual frequency, multiply the base frequency by the relative frequency.

The PWM signal has to enter the pins named GATE of both corresponding counter channels (parallel wiring). Both CLK pins have to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1&2`, `2&3`, `3&4`, `4&5`, `5&6`, `6&7`, `7&8`, `8&9`, or `9&10`. This selects which two counters the driver block uses to determine the PWM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the Quartz-MM10 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MM 10 FM Capture

This block programs the AMD9513A for capturing FM signals.

There is one output for relative frequency compared to the base frequency. To get the actual frequency, multiply the base frequency by the relative frequency.

The FM signal has to enter the pin named GATE of the corresponding counter channel. The CLK pin has to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose 1, 2, 3, 4, 5, 6, 7, 8, 9, or 10. This selects which counter the driver block uses to determine the FM. In each case, one block is needed for each counter.

Frequency base — From the list, choose F1=1MHz, F2=100kHz, F3=10kHz, F4=1kHz, or F5=100Hz to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the Quartz-MM 10 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Quartz-MMxx

You can use this block to program the AMD9513A counter. The PWM, PWM & ARM, FM, FM & ARM, PWM Capture, and FM Capture blocks use this block in their underlying subsystems. The API for this block is not currently documented.

Ruby-MM

The Diamond Ruby-MM is a PC104 I/O board with 4 or 8 single analog output (D/A) channels (12-bit), unipolar and bipolar operation, +/- 10V, +/- 5V, 0-10V, 0-5V fixed ranges, +/- 2.5V, 0-2.5V user-adjustable ranges, 24 digital input and output lines.

xPC Target supports this board with these driver blocks:

- "Diamond Ruby-MM Analog Output (D/A)"
- "Diamond Ruby-MM Digital Input"
- "Diamond Ruby-MM Digital Output"

Board Characteristics

Board name	Diamond Ruby-MM
Manufacturer	Diamond Systems Corporation
Bus type	ISA (PC/104)
Access method	I/O Mapped
Multiple block instance support	D/A:No, DIO:Yes
Multiple board support	Yes

Diamond Ruby-MM Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
Volts	double	1

Driver Block Parameters

Channel vector — List the output channels as a vector. Up to 8 different channels can be listed. To specify the first three channels, enter

```
[1,2,3]
```

The board comes in two different hardware versions. If the 4-channel version of the hardware is used and the channels 5-8 are listed, the those outputs will not show an error, but the data will be ignored.

Range — The output range of the board is selected with jumpers on the board. Each group of 4 channels can be jumpered for any of the available ranges. The range you select in the Block Parameters must correspond to the range specified by the jumper settings or you will obtain incorrect results.

Reset vector – The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector – The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board in hexadecimal (such as 0x300) as selected with the hardware jumpers on the board.

Note Please consult the appropriate Diamond Ruby-MM hardware manuals for more information on jumper settings and I/O connections.

Diamond Ruby-MM Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines to be read. This driver allows the selection of up to 8 individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering lines with 0.

Port — From the **Port** list, choose either `A`, `B`, or `C`. The **Port** parameter defines which port is used for this driver block. Each port has a maximum of 8 digital inputs. In each case, one block is needed for each port.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board in hexadecimal (such as 0x300) as selected with the hardware jumpers on the board.

Note Please consult the appropriate Diamond Ruby-MM hardware manuals for more information on jumper settings and I/O connections.

Diamond Ruby-MM Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines to drive. This driver allows the selection of up to 8 individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering lines with 0.

Port — From the **Port** list, choose either `A`, `B`, or `C`. The **Port** parameter defines which port is used for this driver block. Each port has a maximum of 8 digital outputs. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board in hexadecimal (such as 0x300) as selected with the hardware jumpers on the board.

Note Please consult the appropriate Diamond Ruby-MM hardware manuals for more information on jumper settings and I/O connections.

Ruby-MM-416

The Ruby-MM-416 is an I/O board with 4 16-bit analog output (D/A) channels, and 24 digital I/O lines which can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "Ruby-MM-416 Analog Output (D/A)"
- "Ruby-MM-416 Digital Input"
- "Ruby-MM-416 Digital Output"

Board Characteristics

Board name	Ruby-MM-416
Manufacturer	Diamond Systems Corporation
Bus type	ISA
Access method	I/O Mapped
Multiple block instance support	D/A:No, DIO:Yes
Multiple board support	Yes

Ruby-MM-416 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter a vector containing channel numbers between 1 and 4. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — The range vector must be a scalar or a vector the same length as the channel vector. The vector entries must range codes from the following table

Input Range (V)	Range Code
-10 to +10	-10
-5 to +5	-5
0 to 10	10

The range codes you enter must be consistent with the jumper settings on the board.

Reset vector — The reset vector must be a scalar or the same length as the channel vector and must contain values of 0 or 1. If it is a scalar it will be automatically expanded to the length of the channel vector. This parameter controls the behavior at model termination: a value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector, whereas a value of 0 causes it to be left at the most recent value attained while the model was running.

Initial value vector – The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started. If you provide a value that is out of the channel's range, the value is reset to the lower or upper range value.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This must correspond to the base address setting on the board (header J6). For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Ruby-MM-416 Digital Input

Ruby-MM-416 boards have three I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for the current port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose A, B, or C.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the Base address board setting. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Ruby-MM-416 Digital Output

Ruby-MM-416 boards have three I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for the current port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose `A`, `B`, or `C`.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the Base address board setting. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Ruby-MM-1612

The Ruby-MM-1612 is an I/O board with 16 12-bit analog output (D/A) channels, and 24 digital I/O lines which can be configured in groups of 8 for either input or output.

xPC Target supports this board with these driver blocks:

- "Ruby-MM-1612 Analog Output (D/A)"
- "Ruby-MM-1612 Digital Input"
- "Ruby-MM-1612 Digital Output"

Board Characteristics

Board name	Ruby-MM-1612
Manufacturer	Diamond Systems Corporation
Bus type	ISA
Access method	I/O Mapped
Multiple block instance support	D/A:No, DIO:Yes
Multiple board support	Yes

Ruby-MM-1612 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
Volts	Double	1

Driver Block Parameters

Channel vector — Enter a vector containing channel numbers between 1 and 16. This driver allows the selection of individual D/A channels in any order.

The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range for bank 1, Range for bank 2 — Bank 1 consists of channels 1-8 and bank 2 consists of channels 9-16. The output range may be specified on a per-bank basis. These ranges must correspond to the jumper settings in header R4 on the board. See the board manual for details.

Note that if you select a range of either `5V to 5V` or `0 to 5V` for one bank, then it is not possible to select a range of either `10V to 10V` or `0 to 10V` for the other bank. This is because jumper 5 in header J4 (On-Board Reference Full-Scale Voltage Selection) affects all channels, not just those of a single bank. See the board manual for details.

This driver supports the Adjustable Reference Voltage. You can use this feature with either output range `2.5V to 2.5V` or `0 to 2.5V`. If for example you adjust potentiometer R4 to 2.3V (instead of the default setting of 2.5), then an input signal of 1.2 will result in an output voltage of 1.2 / 2.5 * 2.3V = 1.1V. See the board manual for details.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started. If you provide a value that is out of the channel's range, the value is reset to the lower or upper range value.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Ruby-MM-1612 Digital Input

Ruby-MM-1612 boards have three I/O ports, each containing 8 digital I/O lines.These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital input driver block for a given port, that port is configured for input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for the current port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose `A`, `B`, or `C`.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the Base address board setting. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

Ruby-MM-1612 Digital Output

Ruby-MM-1612 series boards have three I/O ports, each containing 8 digital I/O lines. These ports can be configured independently for either input or output. Use a separate driver block for each port. By selecting the digital output driver block for a given port, that port is configured for output.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for the current port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose `A`, `B`, or `C`.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the Base address board setting. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

15

Gespac

I/O boards supported by xPC Target.

GESADA-1

The GESADA-1 is an industrial I/O board with 16 single or 8 differential analog input (A/D) channels, and 4 analog output (D/A) channels (10-bit).

xPC Target supports this board with these driver blocks:

- "GESADA-1 Analog Input (A/D)"
- "GESADA-1 Analog Output (D/A)"

Note xPC Target does not support the external trigger and interrupt propagation on this board.

Board Characteristics

Board name	GESADA-1
Manufacturer	Gespac
Bus type	ISA industrial
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

GESADA-1 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Number of channels — If you choose `16 single-ended` from the MUX list, then enter a number between 1 and 16. If you choose 8 differential from the MUX list, then enter a number between 1 and 8. This driver does not allow you to select individual channels or to mix single-ended and differential inputs.

Number the channels beginning with 1 even if the board manufacturer starts numbering channels with 0.

Range — From the list, choose either `+-10V` (-10 to +10 volts), `+-5V` (-5 to +5 volts), or `0-10V`.This driver does not allow you to select a different range for each channel. The input range is the same for all A/D channels.

The input range setting must correspond to the settings of jumper J6 and J9 on the board.

MUX — From the list, choose either `16 single-ended` or `8 differential`. This choice must correspond to the MUX-switch setting on the board.

The differential mode is only supported if the board is equipped with option 1A. The MUX setting must correspond to the settings of jumper J3 and J7 on the board.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

GESADA-1 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — This parameter is a combined Channel vector and Range vector. The number of elements defines the number of D/A channels used.

Enter a range code for each of the channels used. This driver allows a different range for each D/A channel with a maximum of 2 channels.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5		

For example, if the first channel is -10 to + 10 volts and the second, third and fourth channel are -5 to +5 volts, enter

```
[-10,5,5,5]
```

The range settings have to correspond to the jumper setting of J5 on the board.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The base address specifies the base address of the board and has to correspond to the Jumper setting (J12) on the board.

GESPIA-2A

The GESPIA-2A is an industrial I/O board with 32 digital I/O lines. The GESPIA-2A has two 6821 PIAs (0 and 1) from Motorola. Each PIA has two ports (A and B) with 8 digital lints which can be defined as input or output.

xPC Target supports this board with these driver blocks:

- "GESPIA-2A Digital Input"
- "GESPIA-2A Digital Output"

Board Characteristics

Board name	GESPIA-2A
Manufacturer	Gespac
Bus type	ISA industrial
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

GESPIA-2A Digital Input

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Number of channels — Enter a number between 1 and 8 to select the number of digital input lines used with this port.

Port Name. From the list, choose either PIA0A, PIA0B, PIA1A or PIA1B to identify the port used with this block of I/O lines.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

GESPIA-2A Digital Output

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Number of channels — Enter a number between 1 and 8 to select the number of digital output lines used with this port.

Port Name. From the list, choose either PIA0A, PIA0B, PIA1A or PIA1B to identify the port used with this block of I/O lines.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

16

Humusoft

I/O boards supported by xPC Target.

AD 512 (p. 16-2)	I/O board with 8 single analog input (A/D) channels, 2 analog output (D/A) channels, 8 digital inputs, and 8 digital outputs.

AD 512

The AD 512 is an I/O board with 8 single analog input (A/D) channels (12-bit) with a maximum sample rate of 100 kHz, 2 analog output (D/A) channels (12-bit), 8 digital inputs, and 8 digital outputs.

xPC Target supports this board with these driver blocks:

- "AD 512 Analog Input (A/D)"
- "AD 512 Analog Output (D/A)"
- "AD 512 Digital Input"
- "AD 512 Digital Output"

Board Characteristics

Board name	AD 512
Manufacturer	Humusoft
Bus type	ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

AD 512 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver block Parameter

Channel vector — Enter numbers between 1 and 8. This driver allows the selection of individual channels in any order. The number of elements defines the number of A/D channels used.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels entered in the channel vector. The range vector must be the same length as the channel vector. This driver allows a different range for each channel.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Sample time — Model base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the jumper settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

AD 512 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — This parameter is a combined Channel vector and Range vector. The number of elements defines the number of D/A channels used.

Enter a range code for each of the channels used. This driver allows a different range for each channel with a maximum of 2 channels.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to + 5	-5	0 - 5	5

For example, if the first channel is -10 to + 10 volts and the second channel is 0 to +5 volts, enter

```
[-10,5]
```

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

AD 512 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Channel vector — Enter a numbers between 1 and 8. This driver allows the selection of individual digital line numbers in any order. The number of elements defines the number of digital input lines used.

For example, to use the first, second and fifth digital input lines, enter

```
[1,2,5]
```

Number the lines beginning with 1, even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

AD 512 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Channel vector — Enter a numbers between 1 and 8. This driver allows the selection of individual digital line numbers in any order. The number of elements defines the number of digital output lines used.

For example, to use the first, second and fifth digital output lines, enter

```
[1,2,5]
```

Number the lines beginning with 1, even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must correspond to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

17

Keithley

I/O boards supported by xPC Target.

DAS-1800HR (p. 17-2)	I/O board with 16 single or 8 differential analog input (A/D) channels, 4 digital input lines and 4 digital output lines.
KCPI-1801HC (p. 17-6)	I/O board with 64 single or 32 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 4 digital input and output lines.
KPCI-1802HC (p. 17-13)	I/O board with 64 single or 32 differential analog input (A/D) channels, 2 analog output (D/A) channels, and 4 digital input and output lines.

DAS-1800HR

The DAS-1800HR is an I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100 kHz, 4 digital input lines and 4 digital output lines.

xPC Target supports this board with these driver blocks:

- “DAS-1800HR Analog Input (A/D)”
- “DAS-1800HR Digital Input”
- “DAS-1800HR Digital Output”

Board Characteristics

Board name	DAS-1800HR
Manufacturer	Keithley
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

DAS-1800HR Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — If `16 single-ended` or `16 single-ended common mode` is chosen from the MUX list, then enter numbers between 1 and 16 to select the individual channels. If `8 differential` is chosen from the MUX list, then enter numbers between 1 and 8 to select the A/D channels used. This driver allows the selection of individual A/D channels in any order. The number of elements defines the number of A/D channels used/

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Gain vector (1,2,4,8) — Enter `1`, `2`, `4`, or `8` for each of the channels in the channel vector to choose the gain code of that channel. The gain vector must be the same length as the channel vector. This driver allows the gain of each channel to be different.

Notice that by increasing the gain code the voltage range is decreased. The gain divides the input voltage range.

For example, if the first channel has a gain code of 1 (10 volt range) and the second and fifth channels have a gain code of 2 (5 volt range), enter

```
[1,2,2]
```

Range — From the list, choose either `Bipolar` or `Unipolar`.

The range setting defines if the board is working in bipolar or unipolar input mode. This setting is the same for all of the selected channels.

The following table is a list of the ranges for this driver given the gain entered and the range chosen.

Gain	Bipolar Range (V)	Unipolar Range (V)
1	-10 to +10	0 to 10
2	-5 to + 5	0 to +5
4	-2.5 to 2.5	0 to 2.5
8	-1.25 to +1.25	0 to 1.25

MUX — From the list, choose either `8 differential`, `16 single-ended`, or `16 single-ended common mode`. Your choice must correspond to the MUX-switch setting on the board.

Common-mode is similar to single-ended mode but the negative wire of the source to be measured is connected to input AI-SENSE instead of LLGND.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DAS-1800HR Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Number of channels — Enter a number between 1 and 8 to select the number of digital input lines used with this port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DAS-1800HR Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Number of channels — Enter a number between 1 and 4 to select the number of digital output lines used with this port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

KCPI-1801HC

The KCPI-1801 is an I/O board with 64 single or 32 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 333 kHz, 2 analog output (D/A) channels (12-bit), and 4 digital input and output lines.

xPC Target supports this board with these driver blocks:

- "KPCI-1801HC Analog Input (A/D)"
- "KPCI-1801HC Analog Output (D/A)"
- "KPCI-1801HC Digital Input"
- "KPCI-1801HC Digital Output"

xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	KCPI-1801HC
Manufacturer	Keithley Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D:No, D/A:Yes, Digital I/O:Yes
Multiple board support	Yes

KPCI-1801HC Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 64. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-5 to +5	-5	0 to 5	5
-1 to +1	-1	0 to 1	1
-0.1 to +0.1	-0.1	0 to 0.1	0.1
-0.02 to +0.02	-0.02	0 to 0.02	0.02

For example, if the first channel is -5 to + 5 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-5,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
single-ended	0	Analog input line connected to the positive input. Analog input ground (IGND) internally connected to the negative input.
differential	1	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,1]
```

The driver selects a second differential input 32 channels higher than the first channel. In the example above, the driver would select the 37th channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

KPCI-1801HC Analog Output (D/A)

The analog output range of this board is set -10 to +10 volts.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this

driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

KPCI-1801HC Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 4 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

KPCI-1801HC Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 4 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

KPCI-1802HC

The KCPI-1802 is an I/O board with 64 single or 32 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 333 kHz, 2 analog output (D/A) channels (12-bit), and 4 digital input and output lines.

xPC Target supports this board with these driver blocks:

- "KPCI-1802HC Analog Input (A/D)"
- "KPCI-1802HC Analog Output (D/A)"
- "KPCI-1802HC Digital Input"
- "KPCI-1802HC Digital Output"

xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	KPCI-1802HC
Manufacturer	Keithley Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D:No, D/A:Yes, Digital I/O:Yes
Multiple board support	Yes

KPCI-1802HC Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 64. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10
-5 to +5	-5	0 to 5	5
-2.5 to +2.5	-2.5	0 to 2.5	2.5
-1.25to +1.25	-1.25	0 to 1.25	1.25

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
single-ended	0	Analog input line connected to the positive input. Analog input ground (IGND) internally connected to the negative input.
differential	1	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,1]
```

The driver selects a second differential input 32 channels higher than the first channel. In the example above, the driver would select the 37th channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

KPCI-1802HC Analog Output (D/A)

The analog output range of this board is set -10 to +10 volts.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this

driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

KPCI-1802HC Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 4 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

KPCI-1802HC Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 4 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

18

National Instruments

I/O boards supported by xPC Target.

PCI-6527 (p. 18-69)	I/O board with 24 filtered digital input lines and 24 digital output lines. You can individually turn input filtering on or off for each of the 24 input lines.
PCI-6703 (p. 18-74)	I/O board with 16 voltage outputs and 8 digital I/O lines.
PCI-6704 (p. 18-76)	I/O board with 16 voltage outputs, 16 current outputs, and 8 digital I/O lines.
PCI-DIO-96 (p. 18-78)	I/O board with 96 digital input and output lines.
PCI-MIO-16E-1 (p. 18-82)	I/O board with 16 single or 8 differential analog input channels, 2 analog output channels, 8 digital input and output lines, and 2 counter/timers.
PCI-MIO-16E-4 (p. 18-89)	I/O board with 16 single or 8 differential analog input channels, 2 analog output channels, 8 digital input and output lines, and 2 counter/timers.
PCI-MIO-16XE-10 (p. 18-97)	I/O board with 16 single or 8 differential analog input (A/D) channels, 2 analog output (D/A) channels, 8 digital input and output lines, and 2 counter/timers.
PXI-6040E (p. 18-105)	I/O board with 16 single or 8 differential analog input channels, 2 analog output channels, 8 digital input and output lines, and 2 counter/timers.
PXI-6070E (p. 18-113)	I/O board with 16 single or 8 differential analog input channels, 2 analog output channels, 8 digital input and output lines, and 2 counter/timers.
PXI-6508 (p. 18-120)	I/O board with 96 digital input and output lines.
PXI-6527 (p. 18-124)	I/O board with 24 filtered digital input lines and 24 digital output lines. You can individually turn input filtering on or off for each of the 24 input lines.

AT-AO-6

The AT-AO-6 is an I/O board with 6 analog output (D/A) channels (12-bit), and 16 digital I/O lines.

xPC Target supports this board with this driver block:

- “AT-AO-6 Analog Output (D/A)”

Board Characteristics

Board name	AT-AO-6
Manufacturer	National Instruments
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

AT-AO-6 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 6. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 10volts, enter

```
[-10,10]
```

The range settings have to correspond to the jumper settings on the board.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

numbering the lines with 0.

AT-AO-10

The AT-AO-10 is an I/O board with 10 analog output (D/A) channels (12-bit), and 16 digital I/O lines.

xPC Target supports this board with this driver block:

- "AT-AO-10 Analog Output (D/A)"

Board Characteristics

Board name	AT-AO-10
Manufacturer	National Instruments
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

AT-AO-10 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 10. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10	0 to 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 10volts, enter

```
[-10,10]
```

The range settings have to correspond to the jumper settings on the board.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-DIO-24

The PC-DIO-24 is an I/O board with 24 digital input and output lines.

xPC Target supports this board with these driver blocks:

- "PC-DIO-24 Digital Input"
- "PC-DIO-24 Digital Output"

Board Characteristics

Board name	PC-DIO-24
Manufacturer	National Instruments
Bus type	ISA
Access method	I/O-mapped
Multiple block instance support	Yes
Multiple board support	Yes

PC-DIO-24 Digital Input

The PC-DIO24 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-DIO-24 Digital Output

The PC-DIO24 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO-10

The PC-TIO-10 is an I/O board with 16 digital input and output lines, and 10 counter/timer channels (16-bit).

xPC Target supports this board with these driver blocks:

- "PC-TIO-10 Digital Input"
- "PC-TIO-10 Digital Output"
- "PC-TIO-10 Counter PWM"
- "PC-TIO10 Counter PWM & ARM"
- "PC-TIO-10 Counter FM"
- "PC-TIO10 Counter FM & ARM"
- "PC-TIO10 PWM Capture"
- "PC-TIO10 FM Capture"

Board Characteristics

Board Name	PC-TIO10
Manufacturer	National Instruments
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PC-TIO-10 Digital Input

The PC-TIO-10 has one MC6821 chip with 2 ports (PIA A, PIA B). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `PIA A`, or `PCA B`. The I/O board has a MC6821 chip with 2 ports. The **Port** parameter defines which port of the MC6821 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO-10 Digital Output

The PC-TIO-10 has one MC6821 chip with 2 ports (PIA A, PIA B). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `PIA A`, or `PCA B`. The I/O board has a MC6821 chip with 2 ports. The **Port** parameter defines which port of the MC6821 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter the base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO-10 Counter PWM

The PC-TIO-10 has two AM9513A chips each with 5 counters for a total of 10 counters on the board.

The PC-TIO-10 PWM driver programs the AM9513A for PWM (Pulse Width Modulation) signal generation (a square wave with fixed frequency and variable duty cycle). The block has one input which defines the variable duty cycle between 0 and 1. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the Frequency base and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO10 Counter PWM & ARM

The PC-TIO-10 has two AM9513A chips each with 5 counters for a total of 10 counters on the board.

The PC-TIO-10 PWM & ARM driver programs the AM9513A for PWM or disarmed signal generation (a square wave with fixed frequency and variable duty cycle). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Duty cycle: double Arm: double	0 to 1 <0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=100kHz` as the **Frequency base** and enter `0.175` as the **Relative output frequency**. 100kHz x 0.175 = 17.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO-10 Counter FM

The PC-TIO-10 has two AM9513A chips each with 5 counters for a total of 10 counters on the board.

The PC-TIO-10 FM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency.

Output duty Cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO10 Counter FM & ARM

The PC-TIO-10 has two AM9513A chips each with 5 counters for a total of 10 counters on the board.

The PC-TIO-10 FM & ARM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	Variable frequency: double Arm: double	<0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, or `10` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency.

Output duty Cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the **Output duty cycle** defined in the setting above define the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

disarmed. If a value 1 is asserted, the counter gets armed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

The dialogue box of the block allows the following settings:

PC-TIO10 PWM Capture

This block programs the AMD9513A for capturing PWM signals by using two counters. One counter measures the cycle duration, and the other counter measures the duration the signal is high.

There are two outputs. One output is the relative frequency compared to the base frequency. The other output is the duty cycle. To get the actual frequency, multiply the base frequency by the relative frequency.

The PWM signal has to enter the pins named GATE of both corresponding counter channels (parallel wiring). Both CLK pins have to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1&2`, `2&3`, `3&4`, `4&5`. This selects which two counters the driver block uses to determine the PWM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO10 FM Capture

This block programs the AMD9513A for capturing FM signals.

There is one output for relative frequency compared to the base frequency. To get the actual frequency, multiply the base frequency by the relative frequency.

The FM signal has to enter the pin named GATE of the corresponding counter channel. The CLK pin has to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5`. This selects which counter the driver block uses to determine the FM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=1MHz`, `F2=100kHz`, `F3=10kHz`, `F4=1kHz`, or `F5=100Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the CIO-CTR05 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PC-TIO-10xx

You can use this block to program the AMD9513A counter. The PWM, PWM & ARM, FM, FM & ARM, PWM Capture, and FM Capture blocks use this block in their underlying subsystems. The API for this block is not currently documented.

PCI-6023E

The PCI-6023E is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 200 kHz, 8 digital I/O lines, and 2 counter/timers (24-bit)

xPC Target supports this board with these driver blocks:

- "PCI-6023E Analog Input (A/D)"
- "PCI-6023E Digital Input"
- "PCI-6023E Digital Output"

xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-6023E
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, Digital I/O: Yes
Multiple board support	Yes

PCI-6023E Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	-0.5 to +0.5	-0.5
-5 to +5	-5	-0.05 to +0.05	-0.05

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6023E Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6023E Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6024E

The PCI-6024E is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 200 kHz, 2 analog output (D/A) channels (12-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- "PCI-6024E Analog Input (A/D)"
- "PCI-6024E Analog Output (D/A)"
- "PCI-6024E Digital Input"
- "PCI-6024E Digital Output"

xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-6024E
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-6024E Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	-0.5 to +0.5	-0.5
-5 to +5	-5	-0.05 to +0.05	-0.05

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6024E Analog Output (D/A)

The analog output range of this board is set -10 to +10 volts.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this

driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6024E Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6024E Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low $\geq$ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6025E

The PCI-6025E is an I/O board with 16 single or 8 differential analog inputs (A/D) channels (12-bit) with a maximum sample rate of 200 kHz, 2 analog output channels (12-bit), 32 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- “PCI-6025E Analog Input (A/D)”
- “PCI-6025E Analog Output (D/A)”
- “PCI-6025E Digital Input”
- “PCI-6025E Digital Output”

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-6025E
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-6025E Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	-0.5 to +0.5	-0.5
-5 to +5	-5	-0.05 to +0.05	-0.05

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6025E Analog Output (D/A)

The analog output range of this board is set -10 to +10 volts.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this

driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6025E Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6025E Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6031E

The PCI-6031E is an I/O board with 64 single or 32 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100 kHz, 2 analog output (D/A) channels (16-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- "PCI-6031E Analog Input (A/D)"
- "PCI-6031E Analog Output (D/A)"
- "PCI-6031E Digital Input"
- "PCI-6031E Digital Output"

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-6031E
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-6031E Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 64. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1, even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2 to +2	-2	0 - 2	2
-1 to + 1	-1	0 - 1	1
-0.5 to +0.5	-0.5	0 - 0.5	0.5
-0.2 to +0.2	-0.2	0 - 0.2	0.2
-0.1 to +0.1	-0.1	0 - 0.1	0.1

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6031E Analog Output (D/A)

The analog output range of this board is set -10 to +10 volts.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels begin with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 10volts, enter

```
[-10,10]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6031E Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6031E Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6052E

The PCI-6052E is an I/O board with 16 single or 8 differential analog input channels (16-bit) with a maximum sample rate of 333 kHz, 2 analog output channels (16-bit) and 8 digital input and output lines.

xPC Target supports this board with these driver blocks:

- "PCI-6052E Analog Input (A/D)"
- "PCI-6052E Analog Output (D/A)"
- "PCI-6052E Digital Input"
- "PCI-6052E Digital Output"

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-6052E
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-6052E Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2 to +2	-2	0 - 2	2
-1 to + 1	-1	0 - 1	1
-0.5 to +0.5	-0.5	0 - 0.5	0.5
-0.2 to +0.2	-0.2	0 - 0.2	0.2
-0.1 to +0.1	-0.1	0 - 0.1	0.1
-0.05 to +0.05	-0.05		

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6052E Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels begin with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10 V	-10	0 - 10 V	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in

the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6052E Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6052E Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	<0.5 = TTL low ≥0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel

vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6071E

The PCI-6071E is an I/O board with 64 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 1.25 MHz, 2 analog output (D/A) channels (12-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- “PCI-6071E Analog Input (A/D)”
- “PCI-6071E Analog Output (D/A)”
- “PCI-6071E Digital Input”
- “PCI-6071E Digital Output”

xPC Target does not support the counter/timers on this board.

Board Characteristics

Board Name	PCI-6071E
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-6071E Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 64. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +10	10
-5 to +5	-5	0 to +5	5
-2 to +2	-2	0 to +2	2
-1 to + 1	-1	0 to +1	1
-0.5 to +0.5	-0.5	0 to +0.5	0.5
-0.2 to +0.2	-0.2	0 to +0.2	0.2

Input Range (V)	Range Code	Input Range (V)	Range Code
-0.1 to +0.1	-0.1	0 to +0.1	0.1
-0.05 to +0.05	-0.05		

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6071E Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6071E Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6071E Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6503

The PCI-6503 is an I/O board with 24 digital input and output lines.

xPC Target supports this board with these driver blocks:

- “PCI-6503 Digital Input”
- “PCI-6503 Digital Output”

Board Characteristics

Board name	PCI-6503
Manufacturer	National Instruments
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-6503 Digital Input

The PCI-6503 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6503 Digital Output

The PCI-6503 has one 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6527

The PCI-6527 is an I/O board with 24 filtered digital input lines and 24 digital output lines. You can individually turn input filtering on or off for each of the 24 input lines. However, there is only one filter time of all of the input lines that have filtering enabled.

xPC Target supports this board with these driver blocks:

- "PCI-6527 Digital Input"
- "PCI-6527 Digital Output"

Board Characteristics

Board name	PCI-6527
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	Digital I/O: Yes
Multiple board support	Yes

PCI-6527 Digital Input

The PCI-6527 has one chip with 3 ports (A,B,C) for digital input. Each port has a maximum of 8 digital I/O lines. Use a separate driver block for each port.

Note The interrupt on change capability of this board is not used.

Scaling of Input to Output

Hardware Input	Block Input Data Type	Scaling
OPTO	double	$< 0.5 = 0$ $\geq 0.5 = 1$

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The **Port** parameter defines which port is used for this driver block. Each port has a maximum of 8 digital inputs. In each case, one block is needed for each port.

The documentation for this board from National Instruments labels these ports 0, 1, and 2.

Filter vector — This is a Boolean vector that selects which input lines to filter. For example, if you enter a **Channel vector** of `[1,3,5]`, and you want to filter all three lines, enter

[1,1,1]

If you want to filter lines 1 and 5, but not line 3, then enter

[1,0,1]

If the filter vector is a single element, then it is scaler expanded to the same width as the **Channel vector**. In the example above, if the **Filter vector** is `[0]`, it is expanded to `[0,0,0]`. Likewise, if the **Filter vector** is `[1]`it is expanded to `[1,1,1]`.

Filter interval — Enter the time interval the hardware filter uses to determine a stable pulse. If the input pule is shorter then this interval, it is ignored. There is only one filter interval for all filtered inputs, and if you filter on more than one digital input block for this board, you must enter the same **Filter interval** for all blocks.

A reasonable value for the **Filter interval** would be 200 to 300 us.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6527 Digital Output

The PCI-6527 has one chip with 3 ports (A,B,C) for digital output. Each port has a maximum of 8 digital output lines. Use a separate driver block for each port.

Scaling of Input to Output

Hardware Output	Block Output Data Type	Scaling
OPTO	double	<0.5 = OPTO is OFF, no current flowing ≥0.5 = OPTO is ON, current can flow

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The **Port** parameter defines which port is used for this driver block. Each port has a maximum or 8 digital output lines. In each case, one block is needed for each port.

The documentation for this board from National Instruments labels these ports 3, 4, and 5.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this

driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6703

The PCI-6703 is an I/O board with 16 voltage outputs and 8 digital I/O lines.

xPC Target supports this board with this driver block:

- "PCI-6703 Analog Output (D/A)"

Board Characteristics

Board name	PCI-6703
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PCI-6703 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-6704

The PCI-6704 is an I/O board with 16 voltage outputs, 16 current outputs, and 8 digital I/O lines.

xPC Target supports this board with this driver block:

- “PCI-6704 Analog Output (D/A)”

Board Characteristics

Board name	PCI-6704
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PCI-6704 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use the first and second analog output (D/A) channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-DIO-96

The PC-DIO-96 is an I/O board with 96 digital input and output lines.

xPC Target supports this board with these driver blocks:

- "PC-DIO-24 Digital Input"
- "PCI-DIO-96 Digital Output"

Board Characteristics

Board name	PC-DIO-96
Manufacturer	National Instruments
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PCI-DIO96 Digital Input

The PC-DIO96 has four 8255 chips with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has four 8255 chips with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1`, `2`, `3`, or `4`.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-DIO-96 Digital Output

The PC-DIO24 has four 8255 chips with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1`, `2`, `3`, or `4`.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

PCI-MIO-16E-1

The PCI-MIO-16E-1 is an I/O board with 16 single or 8 differential analog input channels (12-bit) with a maximum sample rate of 1.25 MHz, 2 analog output channels (12-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- "PCI-MIO-16E-1 Analog Input (A/D)"
- "PCI-MIO-16E-1 Analog Output (D/A)"
- "PCI-MIO-16E1 Digital Input"
- "PCI-MIO-16E-1 Digital Output"

xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-MIO-16E-1
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-MIO-16E-1 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2 to +2	-2	0 - 2	2
-1 to + 1	-1	0 - 1	1
-0.5 to +0.5	-0.5	0 - 0.5	0.5
-0.2 to +0.2	-0.2	0 - 0.2	0.2
-0.1 to +0.1	-0.1	0 - 0.1	0.1
-0.05 to +0.05	-0.05		

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16E-1 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in

the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16E1 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16E-1 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel

vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16E-4

The PCI-MIO-16E-4 is an I/O board with 16 single or 8 differential analog input channels (12-bit) with a maximum sample rate of 500 kHz, 2 analog output channels (12-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- "PCI-MIO-16E-4 Analog Input (A/D)"
- "PCI-MIO-16E-4 Analog Output (D/A)"
- "PCI-MIO-16E-4 Digital Input"
- "PCI-MIO-16E-4 Digital Input"

xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-MIO-16-4
Manufacturer	National Instruments
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-MIO-16E-4 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2 to +2	-2	0 - 2	2
-1 to + 1	-1	0 - 1	1
-0.5 to +0.5	-0.5	0 - 0.5	0.5
-0.2 to +0.2	-0.2	0 - 0.2	0.2

Input Range (V)	Range Code	Input Range (V)	Range Code
-0.1 to +0.1	-0.1	0 - 0.1	0.1
-0.05 to +0.05	-0.05		

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16E-4 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16E-4 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16E-4 Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16XE-10

The PCI-MIO-16XE-10 is an I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100 kHz, 2 analog output (D/A) channels (16-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- "PCI-MIO-16XE-10 Analog Input (A/D)"
- "PCI-MIO-16XE-10 Analog Output (D/A)"
- "PCI-MIO-16XE-10 Digital Input"
- "PCI-MIO-16XE-10 Digital Output"

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PCI-MIO-16XE-10
Manufacturer	National Instruments
Bus type	PCI
Access method	I/O mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PCI-MIO-16XE-10 Analog Input (A/D)

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1, even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2 to +2	-2	0 - 2	2
-1 to + 1	-1	0 - 1	1
-0.5 to +0.5	-0.5	0 - 0.5	0.5
-0.2 to +0.2	-0.2	0 - 0.2	0.2
-0.1 to +0.1	-0.1	0 - 0.1	0.1

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16XE-10 Analog Output (D/A)

The analog output range of this board is set -10 to +10 volts.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels begin with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 10volts, enter

```
[-10,10]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16XE-10 Digital Input

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PCI-MIO-16XE-10 Digital Output

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6040E

The PXI-6040E is an I/O board with 16 single or 8 differential analog input channels (12-bit) with a maximum sample rate of 500 kHz, 2 analog output channels (12-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- "PXI-6040E Analog Input (A/D)"
- "PXI-6040E Analog Output (D/A)"
- "PXI-6040E Digital Input"
- "PXI-6040E Digital Output"

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PXI-6040E
Manufacturer	National Instruments
Bus type	PXI (Compact PCI)
Access method	Memory mapped
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PXI-6040E Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2 to +2	-2	0 - 2	2
-1 to + 1	-1	0 - 1	1
-0.5 to +0.5	-0.5	0 - 0.5	0.5
-0.2 to +0.2	-0.2	0 - 0.2	0.2

Input Range (V)	Range Code	Input Range (V)	Range Code
-0.1 to +0.1	-0.1	0 - 0.1	0.1
-0.05 to +0.05	-0.05		

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6040E Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6040E Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6040E Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6070E

The PXI-6070E is an I/O board with 16 single or 8 differential analog input channels (12-bit) with a maximum sample rate of 1.25 MHz, 2 analog output channels (12-bit), 8 digital input and output lines, and 2 counter/timers (24-bit).

xPC Target supports this board with these driver blocks:

- "PXI-6070E Analog Input (A/D)"
- "PXI-6070E Analog Output (D/A)"
- "PXI-6070E Digital Input"
- "PXI-6070E Digital Output"

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	PXI-6070E
Manufacturer	National Instruments
Bus type	PXI (Compact PC)
Multiple block instance support	A/D: No, D/A: No, Digital I/O: Yes
Multiple board support	Yes

PXI-6070E Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input range (V)	Range code	Input range (V)	Range code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5
-2 to +2	-2	0 - 2	2
-1 to + 1	-1	0 - 1	1
-0.5 to +0.5	-0.5	0 - 0.5	0.5
-0.2 to +0.2	-0.2	0 - 0.2	0.2
-0.1 to +0.1	-0.1	0 - 0.1	0.1
-0.05 to +0.05	-0.05		

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +1 volts, enter

```
[-10,1,1]
```

Coupling vector — Enter a coupling code for each of the channels in the channel vector. The coupling vector must be the same length as the channel vector. This driver allows a different coupling for each channel.

The following table is a list of the couplings for this driver and the corresponding coupling codes.

Coupling	Coupling Code	Description
RSE	0	Analog input line connected to the positive input of the PGIA. Analog input ground (AIGND) internally connected to the negative input of the PGIA. See the board manual.
NRSE	1	Analog input line connected to the positive input of the PGIA. Analog input sense (AISENSE) connected to the negative input of the PGIA. See the board manual.
DIFF	2	First analog input line connected to the positive input of the PGIA. Second analog input line connected to the negative input of the PGIA. See the board manual.

For example, if the first and second channels are single input and the fifth channel is a differential input, enter

```
[0,0,2]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Model base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6070E Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 2. This driver allows the selection of individual D/A channels in any order. The number of elements defines the number of D/A channels used. For example, to use both of the analog output channels, enter

```
[1,2]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter

```
[-10,5]
```

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in

the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6070E Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6070E Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel

vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6508

The PXI-6508 is an I/O board with 96 digital input and output lines.

xPC Target supports this board with these driver blocks:

- “PXI-6508 Digital Input”
- “PXI-6508 Digital Output”

Board Characteristics

Board name	PXI-6508
Manufacturer	National Instruments
Bus type	PXI (Compact PCI)
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

PXI-6508 Digital Input

The PXI-6508 has four 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling of Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either **A**, **B**, or **C**. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Chip — From the list choose `1`, `2`, `3`, or `4`.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6508 Digital Output

The PXI-6508 has four 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The I/O board has a 8255 chip with 3 ports. The **Port** parameter defines which port of the 8255 chip is used for this driver block. Each port has a maximum or 8 digital lines that can be configured as inputs or outputs depending on which driver block is chosen. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Chip — From the list choose `1`, `2`, `3`, or `4`.

Sample time - Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6527

The PXI-6527 is an I/O board with 24 filtered digital input lines and 24 digital output lines. You can individually turn input filtering on or off for each of the 24 input lines. However, there is only one filter time of all of the input lines that have filtering enabled.

xPC Target supports this board with these driver blocks:

- "PXI-6527 Digital Input"
- "PXI-6527 Digital Output"

Board Characteristics

Board name	PXI-6527
Manufacturer	National Instruments
Bus type	PXI
Access method	Memory mapped
Multiple block instance support	Digital I/O: Yes
Multiple board support	Yes

PXI-6527 Digital Input

The PXI-6527 has one chip with 3 ports (A,B,C) for digital input. Each port has a maximum of 8 digital I/O lines. Use a separate driver block for each port.

Note The interrupt on change capability of this board is not used.

Scaling of Input to Output

Hardware Input	Block Input Data Type	Scaling
OPTO	double	< 0.5 = 0 ≥ 0.5 = 1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The **Port** parameter defines which port is used for this driver block. Each port has a maximum of 8 digital inputs. In each case, one block is needed for each port.

The documentation for this board from National Instruments labels these ports 0, 1, and 2.

Filter vector — This is a Boolean vector that selects which input lines to filter. For example, if you enter a **Channel vector** of `[1,3,5]`, and you want to filter all three lines, enter

[1,1,1]

If you want to filter lines 1 and 5, but not line 3, then enter

[1,0,1]

If the filter vector is a single element, then it is scaler expanded to the same width as the **Channel vector**. In the example above, if the **Filter vector** is `[0]`, it is expanded to `[0,0,0]`. Likewise, if the **Filter vector** is `[1]`it is expanded to `[1,1,1]`.

Filter interval — Enter the time interval the hardware filter uses to determine a stable pulse. If the input pule is shorter then this interval, it is ignored. There is only one filter interval for all filtered inputs, and if you filter on more than one digital input block for this board, you must enter the same **Filter interval** for all blocks.

A reasonable value for the **Filter interval** would be 200 to 300 us.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PXI-6527 Digital Output

The PXI-6527 has one chip with 3 ports (A,B,C) for digital output. Each port has a maximum of 8 digital output lines. Use a separate driver block for each port.

Scaling of Input to Output

Hardware Output	Block Output Data Type	Scaling
OPTO	double	<0.5 = OPTO is OFF, no current flowing ≥0.5 = OPTO is ON, current can flow

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even though the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The **Port** parameter defines which port is used for this driver block. Each port has a maximum or 8 digital output lines. In each case, one block is needed for each port.

The documentation for this board from National Instruments labels these ports 3, 4, and 5.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial voltage values for the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that value is the initial value for all channels. The channels are set to the initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this

driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

Real Time Devices

I/O boards supported by xPC Target.

DM6420

The DM6420 is an I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 500 kHz, 2 analog output (D/A) channels (12-bit), 8 independent digital I/O lines, 8 dependent digital I/O lines, and 2 counter/timers (16-bit).

xPC Target supports this board with these driver blocks:

- "DM6420 Analog Input (A/D)"
- "DM6420 Analog Output (D/A)"
- "DM6420 Digital Input"
- "DM6420 Digital Output"

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	DM6420
Manufacturer	Real Time Devices
Bus type	ISA (PC104)
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

DM6420 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual A/D channels in any order. The number of elements defines the number of A/D channels used.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Coupling vector — Enter either 0 (single-ended) or 1 (differential) for each of the channels in the channel vector to choose the coupling code. The coupling vector must be the same length as the channel vector. This driver allows the coupling of each channel to be different.

For example, if the first and second channels are single-ended and the fifth channel is a differential input, enter

```
[0,0,1]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the channel vector to choose the gain code of that channel. The gain vector must be the same length as the channel vector. This driver allows the gain of each channel to be different.

Note While this board has programmable input ranges of ±5, ±10 and 0 to 10, this driver sets the input range to +10, and then lets you select different input ranges by choosing different gains.

The following table is a list of the ranges for this driver given the gain entered in the gain vector.

Gain	Range (V)
1	-10 to 10
2	-5 to +5
4	-2.5 to 2.5
8	-1.25 to 1.25

Notice that by increasing the gain code the voltage range is decreased. The gain divides the input voltage range.

For example, if the first channel has a gain code of 1 (10 volt range) and the second and fifth channels have a gain code of 2 (5 volt range), enter

```
[1,2,2]
```

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6420 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the analog output (D/A) channels used. This driver allows the selection of individual channels in any order. The number of elements defines the number of D/A channels used.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels entered in the channel vector. The range vector must be the same length as the channel vector. This driver allows a different range for each channel.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +5 volts, enter

```
[-10,5,5]
```

Sample time — Enter the model base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6420 Digital Input

The DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6420 Digital Output

The DAS1601/16 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as outputs. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6430

The DM6430 is an ISA PC/104 I/O board with 16 single or 8 differential analog input (A/D) channels (16-bit) with a maximum sample rate of 100 kHz, 1 analog output (D/A) channel (16-bit), 16 digital I/O lines, and 2 counter/timers (16-bit).

xPC Target supports this board with these driver blocks:

- “DM6430 Analog Input (A/D)”
- “DM6430 Analog Output (D/A)”
- “DM6430 Digital Input”
- “DM6430 Digital Output”

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	DM6430
Manufacturer	Real Time Devices
Bus type	ISA (PC104)
Access method	I/O mapped
Multiple block instance support	No
Multiple board support	Yes

DM6430 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual A/D channels in any order. The number of elements defines the number of A/D channels used.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Gain vector — Enter `1`, `2`, `4`, or `8` for each of the channels in the channel vector to choose the gain code of that channel. The gain vector must be the same length as the channel vector. This driver allows the gain of each channel to be different.

The following table is a list of the ranges for this driver given the gain entered in the gain vector.

Gain	Range (V)
1	-10 to 10
2	-5 to +5
4	-2.5 to 2.5
8	-1.25 to 1.25

Notice that by increasing the gain code the voltage range is decreased. The gain divides the input voltage range.

For example, if the first channel has a gain code of 1 (10 volt range) and the second and fifth channels have a gain code of 2 (5 volt range), enter

```
[1,2,2]
```

Coupling vector — Enter either 0 (single-ended) or 1 (differential) for each of the channels in the channel vector to choose the coupling code. The coupling vector must be the same length as the channel vector. This driver allows the coupling of each channel to be different.

For example, if the first and second channels are single-ended and the fifth channel is a differential input, enter

```
[0,0,1]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6430 Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

This board has only 1 analog output (D/A) with a fixed range of -10 to +10 volts.

Sample time — Base sample time of a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6430 Digital Input

The DM6430 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital input driver block, the port is configured as input.

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6430 Digital Output

The DM6430 has a 8255 chip with 3 ports (A,B,C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate diver block for each port. By selecting the digital output driver block, the port is configured as output.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as outputs. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

DM6604

The DM6604 is an ISA PC/104 I/O board with 8 analog output (D/A) channels (12-bit), and 24 digital I/O lines.

xPC Target supports this board with these driver blocks:

- “DM6604 Analog Output (D/A)”
- “DM6604 Digital Input”
- “DM6604 Digital Output”

Board Characteristics

Board name	DM6604
Manufacturer	Real Time Devices
Bus type	ISA (PC104)
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

DM6604 Analog Output (D/A)

Scaling of Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 8 to select the analog output (D/A) channels used. This driver allows the selection of individual channels in any order. The number of elements defines the number of D/A channels used.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector —Enter a range code for each of the channels entered in the channel vector. The range vector must be the same length as the channel vector. This driver allows a different range for each channel.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5	0 - 5	5

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +5 volts, enter

```
[-10,5,5]
```

Sample time — Enter the model base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. It is important that this entry corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6604 Digital Input

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6604 Digital Output

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either A, B, or C. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as outputs. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804

The DM6804 is an ISA PC/104 I/O board with 24 digital I/O lines and 5 counter/timer channels (16-bit).

It contains one 8255 chip with 3 digital I/O ports and one AM9513A counter/timer chip. For additional information about the various counter/timer modes of that chip see the AM9513A data sheet which is part of the board documentation.

xPC Target supports this board with these driver blocks:

- “DM6804 Digital Input”
- “DM6804 Digital Output”
- “DM6804 Counter PWM”
- “DM6804 Counter PWM & ARM”
- “DM6804 Counter FM”
- “DM6804 Counter FM & ARM”
- “DM6804 PWM Capture”
- “DM6804 FM Capture”
- “DM6804xx”

Board Characteristics

Board name	DM6804
Manufacturer	Real Time Devices
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

DM6804 Digital Input

The DM6804 has a 8255 chip with 3 ports (A, B, C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate driver block for each port. By selecting the digital input driver block, the port is configured as input.

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as inputs. In each case, one block is needed for each port.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804 Digital Output

The DM6804 has a 8255 chip with 3 ports (A, B, C). Each port has a maximum of 8 digital I/O lines that can be configured as inputs or outputs.

Use a separate driver block for each port. By selecting the digital output driver block, the port is configured as output.

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `A`, `B`, or `C`. The port name defines which port is used for this driver block. Each port has a maximum of 8 digital lines that can be configured as outputs. In each case, one block is needed for each port.

Reset vector — The reset vector controls the behavior of the channel at model termination. Enter a scalar or a vector that is the same length as the channel vector. If you specify a scalar value, that setting is used for all channels. If you specify a value of 1, the corresponding channel is reset to the value specified in the initial value vector. If you specify a value of 0, the channel remains at the last value attained while the model was running.

Initial value vector — The initial value vector contains the initial values (0 or 1) of the output channels. Enter a scalar or a vector that is the same length as the channel vector. If you enter a scalar, that value is used for all channels. The channels are set to these initial values between the time the model is downloaded and the time it is started.

Sample time — Enter a base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804 Counter PWM

The DM6804 has one AM9513A chip with 5 counters.

The DM6804 PWM driver programs the AM9513A for PWM (Pulse Width Modulation) signal generation (a square wave with fixed frequency and variable duty cycle). The block has one input which defines the variable duty cycle between 0 and 1. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose 1, 2, 3, 4, or 5 to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose F1=5MHz, F2=500kHz, F3=50kHz, F4=5kHz, or F5=500Hz to set the base frequency.

Relative output frequency — Enter a value between 0 and 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose F2=500kHz as the Frequency Base and enter 0.175 as the Relative Output Frequency. 500kHz x 0.175 = 87.5 kHz

Level sequence of square wave — From the list, choose either high-low or low-high.

- If you choose high-low, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose low-high, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either high or low. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address —Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804 Counter PWM & ARM

The DM6804 has one AM9513A chip with 5 counters.

The DM6804 PWM & ARM driver programs the AM9513A for PWM or disarmed signal generation (a square wave with fixed frequency and variable duty cycle). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Duty cycle: double Arm: double	0 to 1 <0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=5MHz`, `F2=500kHz`, `F3=50kHz`, `F4=5kHz`, or `F5=500Hz` to set the base frequency.

Relative output frequency — Enter a value less than 1. The **Relative output frequency** is multiplied by the **Frequency base** to set the fixed output frequency of the PWM-signal.

For example, if the output frequency of a square wave has to be 17.5 kHz, then choose `F2=500kHz` as the Frequency Base and enter `0.175` as the Relative Output Frequency. 500kHz x 0.175 = 87.5 kHz

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804 Counter FM

The DM6804 has one AM9513A chip with 5 counters.

The DM6804 FM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=5MHz`, `F2=500kHz`, `F3=50kHz`, `F4=5kHz`, or `F5=500Hz` to set the base frequency.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the duty cycle entering the block defines the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running and gets armed when the application begins running. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804 Counter FM & ARM

The DM6804 has one AM9513A chip with 5 counters.

The DM6804 FM & ARM driver programs the AM9513A for FM (Frequency Modulation) signal generation (a square wave with fixed duty cycle and variable frequency). Additionally the driver allows to arm and disarm the counter by the second block input. For the corresponding counter channel, the PWM signal is output at the pin named OUT.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	Variable frequency: double Arm: double	<0.5 disarmed ≥0.5 armed

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5` to select which counter is used with this driver block. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=5MHz`, `F2=500kHz`, `F3=50kHz`, `F4=5kHz`, or `F5=500Hz` to set the base frequency. The XTAL frequency is assumed to be 1MHz, therefore the jumper on the DM6804 has to be in position 1MHz not 5MHz.

Output duty cycle — Enter a value between `0` and `1` to set the duty cycle of the square wave. The Duty Cycle is held fixed during execution of the target application.

Level sequence of square wave — From the list, choose either `high-low` or `low-high`.

- If you choose `high-low`, the square wave period starts with the TTL high part followed by the TTL low part.
- If you choose `low-high`, the square wave period starts with the TTL low part followed by the TTL high part.

In either case, the **Output duty cycle** defined in the setting above define the duration of the TTL high part.

Level when disarmed — From the list, choose either `high` or `low`. The counter is automatically disarmed when the target application is not running. If the application is running, the second input port controls whether the counter is armed or disarmed. This parameter sets the TTL level when the counter is disarmed.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804 PWM Capture

This block programs the AMD9513A for capturing PWM signals by using two counters. One counter measures the cycle duration, and the other counter measures the duration the signal is high.

There are two outputs. One output is the relative frequency compared to the base frequency. The other output is the duty cycle. To get the actual frequency, multiply the base frequency by the relative frequency.

The PWM signal has to enter the pins named GATE of both corresponding counter channels (parallel wiring). Both CLK pins have to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1&2`, `2&3`, `3&4`, `4&5`. This selects which two counters the driver block uses to determine the PWM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=5MHz`, `F2=500kHz`, `F3=50kHz`, `F4=5kHz`, or `F5=500Hz` to set the base frequency. The XTAL frequency is

assumed to be 1MHz, therefore the jumper on the DM6804 has to be in position 1MHz not 5MHz.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804 FM Capture

This block programs the AMD9513A for capturing FM signals.

There is one output for relative frequency compared to the base frequency. To get the actual frequency, multiply the base frequency by the relative frequency.

The FM signal has to enter the pin named GATE of the corresponding counter channel. The CLK pin has to be left unconnected.

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	0 to 1

Driver Block Parameters

Counter — From the list, choose `1`, `2`, `3`, `4`, or `5`. This selects which counter the driver block uses to determine the FM. In each case, one block is needed for each counter.

Frequency base — From the list, choose `F1=5MHz`, `F2=500kHz`, `F3=50kHz`, `F4=5kHz`, or `F5=500Hz` to set the base frequency.

Sample time — Enter the base sample time or a multiple of the base sample time. The sample time indicates the update rate of registration on the input (Duty Cycle)

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM6804xx

You can use this block to program the AMD9513A counter. The PWM, PWM & ARM, FM, FM & ARM, PWM Capture, and FM Capture blocks use this block in their underlying subsystems. The API for this block is not currently documented.

DM6814

The DM6814 is a 16-bit counting board with 3 channels. This board typically connects to incremental encoders. Incremental encoders convert physical motion into electrical pulses than can be used to determine velocity, direction, and distance.

xPC Target supports this board with this driver block:

- "DM6814 Incremental Encoder"

Note xPC Target does not support the 12 digital input lines on this board.

Board Characteristics

Board name	DM6814
Manufacturer	Real-Time Devices
Bus type	ISA
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

DM6814 Incremental Encoder

Driver Block Parameters

Encoder channel — From the list choose, 1, 2, or 3. This parameter specifies which channel you use for this block. For the same board (same base address) two blocks cannot have the same channel number.

Counter initial value — Enter the initial value of the counter. The value must be between 1 and $2^{16} - 1$.

Enable counter reset on px.2 (index input) — If this check box is selected and the incremental encoder is moved over its index mark, the counter is reset to its initial value (default zero).

Sample time — Base sample time or a multiple of the base sample time.

Base address — Enter the base address of the board. This entry must corresponds to the DIP-switch settings on the board. For example, if the base address is 300 (hexadecimal), enter

```
0x300
```

DM7420

The DM7420 is a PCI PC/104 I/O board with 16 single or 8 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 600 kHz, 8 independent digital I/O lines, 8 dependent digital I/O lines, and 9 counter/timers.

xPC Target supports this board with these driver blocks:

- "DM7420 Analog Input (A/D)"
- "DM7420 Digital Input"
- "DM7420 Digital Output"

Note xPC Target does not support the counter/timers on this board.

Board Characteristics

Board name	DM6604
Manufacturer	Real Time Devices
Bus type	PCI (PC104)
Access method	I/O mapped
Multiple block instance support	Yes
Multiple board support	Yes

DM7420 Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Channel vector — Enter numbers between 1 and 16. This driver allows the selection of individual A/D channels in any order. The number of elements defines the number of A/D channels used.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 - 10	10
-5 to +5	-5		

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 10 volts, enter

```
[-10,10]
```

Gain vector — Enter 1, 2, 4, 8, 16, or 32 for each of the channels in the channel vector to choose the gain code of that channel. The gain vector must be the same length as the channel vector. This driver allows the gain of each channel to be different.

The following table is a list of the ranges for this driver given the gain entered in the gain vector.

Gain	Range (V)
1	0 to 10
2	0 to +5
4	0 to 2.5
8	0 to 1.25
16	0 to 0.625
32	0 to 0.312

Notice that by increasing the gain code the voltage range is decreased. The gain divides the input voltage range.

For example, if the first channel has a gain code of 1 (10 volt range) and the second and fifth channels have a gain code of 2 (5 volt range), enter

```
[1,2,2]
```

Coupling vector — Enter either 0 (single-ended) or 1 (differential) for each of the channels in the channel vector to choose the coupling code. The coupling vector must be the same length as the channel vector. This driver allows the coupling of each channel to be different.

For example, if the first and second channels are single-ended and the fifth channel is a differential input, enter

```
[0,0,1]
```

The driver selects a second differential input 8 channels higher than the first channel. In the example above, the driver would select the thirteenth channel as a differential input with the fifth channel.

Sample time — Base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

DM7420 Digital Input

Channel vector — Enter numbers between 1 and 8 to select the digital input lines used with this port. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital inputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either `0` or `1`.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

DM7420 Digital Output

Channel vector — Enter numbers between 1 and 8 to select the digital output lines used with this port. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of digital lines used.

For example, to use all of the digital outputs for one port, enter

```
[1,2,3,4,5,6,7,8]
```

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Port — From the list choose either 0 or 1.

Sample time — Base sample time of a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

SBS Technologies

I/O boards supported by xPC Target.

Flex/104A PC/104 IP Carrier Board (p. 20-2)	PC/104 format IP carrier board with a capacity of 2 IP modules.
IP-16ADC (p. 20-4)	16 single or 8 differential analog input (A/D) channels (16-bit) with a combined throughput of 100kHz.
IP-16DAC (p. 20-6)	3 independent precalibrated analog output (D/A) channels (16-bit).
IP-DAC (p. 20-8)	6 independent precalibrated analog output (D/A) channels (12-bit).
IP-Digital 24 (p. 20-10)	24 digital I/O lines which can be independently configured for input or output.
IP-HiADC (p. 20-13)	16 analog input (A/D) channels (12-bit).
IP-Synchro (p. 20-15)	Two channels of position measurement using synchro, resolver, LVDT, or Inductosyn® transducers.
IP-Unidig-E-48 (p. 20-17)	48 digital I/O lines which can be independently configured for input or output
PCI-40A Carrier Board (p. 20-20)	ISA format IP carrier board with a capacity of 4 IP modules.

Flex/104A PC/104 IP Carrier Board

Each IP module must be physically plugged into an IP carrier board such as the Flex/104A. Models containing IP module blocks must also contain corresponding IP carrier blocks. The relationship 'IP module A is plugged into carrier B' is expressed by selecting the same Carrier ID for both A and B.

The Flex/104A holds up to two IP modules.

xPC Target supports this board with this driver blocks:

- "Flex-104A"

Board Characteristics

Board name	Flex-104A PC/104 carrier board
Manufacturer	SBS Technologies
Bus type	PC/104
Access method	I/O Mapped
Multiple block instance support	No
Multiple board support	Yes

Flex-104A

Driver Block Parameters

Carrier ID — Enter a number to uniquely identify this IP carrier board within the model.

Base address — Enter the base address of the board. For example, if the base address is 300 (hexadecimal), enter

0x300

This board permutes the expected pinout of the 50-pin I/O connector for each IP module as follows

Expected Pin	Actual Pin	Expected Pin	Actual Pin	Expected Pin	Actual Pin
1	1	18	35	35	20
2	3	19	37	36	22
3	5	20	39	37	24
4	7	21	41	38	26
5	9	22	43	39	28
6	11	23	45	40	30
7	13	24	47	41	32
8	15	25	49	42	34
9	17	26	2	43	36
10	19	27	4	44	38
11	21	28	6	45	40
12	23	29	8	46	42
13	25	30	10	47	44
14	27	31	12	48	46
15	29	32	14	49	48
16	31	33	16	50	50
17	33	34	18		

IP-16ADC

The IP-16ADC I/O board has 16 single or 8 differential analog input (A/D) channels (16-bit) with a combined throughput of 100kHz.

xPC Target supports this board with this driver block:

- "IP-16ADC Analog Input (A/D)"

Board Characteristics

Board name	IP-16ADC
Manufacturer	SBS Technologies
Bus type	N/A
Access method	I/O Mapped
Multiple block instance support	Yes
Multiple board support	Yes

IP-16ADC Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Carrier slot — Select the slot on the carrier board into which the IP-16ADC module is plugged. Note that different carrier boards can have different slot capacities.

Channel vector — Enter a vector of numbers between 1 and 16. The channel numbers can occur in any order. For example, to use the first and third analog output (A/D) channels, enter

[1, 3]

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Range Code	Input Range (V)
-10	-10 to +10
-5	-5 to +5
5	0 to 5
10	0 to 10

Range codes –10 and –5 specify differential channels, which each use two pins on the I/O cable. Range codes 10 and 5 specify single-ended channels, which each use only one pin. Certain combinations of channel numbers and range codes can refer to conflicting physical I/O pins and will cause an error of the form “Bipolar channel x and unipolar channel y use the same I/O pins”. Consult the section “I/O Pin Wiring” in the IP-16ADC User’s Manual as you select the channel and range vectors to avoid this.

Sample time — Enter the base sample time or a multiple of the base sample time.

IP-16DAC

The IP-16DAC I/O board has 3 independent precalibrated analog output (D/A) channels (16-bit).

xPC Target supports this board with this driver block:

- "IP-16DAC Analog Output (D/A)"

Board Characteristics

Board name	IP-16DAC
Manufacturer	SBS Technologies
Bus type	N/A
Access method	I/O Mapped
Multiple block instance support	Yes
Multiple board support	Yes

IP-16DAC Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Carrier slot — Select the slot on the carrier board into which the IP-16DAC module is plugged. Note that different carrier boards can have different slot capacities.

Channel vector — Enter a vector of numbers between 1 and 3. The channel numbers can occur in any order. For example, to use the first and second analog output (D/A) channels, enter

[1,2]

Channel numbers 1, 2, and 3 correspond to DAC A, DAC B, and DAC C respectively.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Range Code	Input Range (V)
-10	-10 to +10
-5	-5 to +5
5	0 to 5
10	0 to 10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter [-10,5]

The range settings must correspond to the jumper settings on the board for DAC A and DAC B and DAC C.

Sample time — Base sample time of a multiple of the base sample time.

IP-DAC

The IP-DAC I/O board has 6 independent precalibrated analog output (D/A) channels (12-bit).

xPC Target supports this board with this driver blocks:

- "IP-DAC Analog Output (D/A)"

Board Characteristics

Board name	IP-DAC
Manufacturer	SBS Technologies
Bus type	N/A
Access method	I/O Mapped
Multiple block instance support	Yes
Multiple board support	Yes

IP-DAC Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Carrier slot — Select the slot on the carrier board into which the IP-16ADC module is plugged. Note that different carrier boards can have different slot capacities.

Channel vector — Enter a vector of numbers between 1 and 6. The channel numbers can occur in any order. For example, to use the first and second analog output (D/A) channels, enter

[1,2]

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This board allows the range of each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Range Code	Input Range (V)
-10	-10 to +10
-5	-5 to +5
-2.5	-2.5 to 2.5
5	0 to 5
10	0 to 10

For example, if the first channel is -10 to +10 volts, and the second channel is 0 to 5 volts, enter [-10,5]

The range settings have to correspond to the OUTPUT RANGE SELECTION settings on the board for DAC0 and DAC1 (channel 1 and 2 respectively).

Sample time — Base sample time of a multiple of the base sample time.

IP-Digital 24

IP-Digital 24 boards have 24 digital I/O lines which can be independently configured for input or output.

xPC Target supports this board with these driver blocks:

- “IP-Digital 24 Digital Input”
- “IP-Digital 24 Digital Output”

Board Characteristics

Board name	IP-Digital-24
Manufacturer	SBS Technologies
Bus type	N/A
Access method	I/O Mapped
Multiple block instance support	Yes
Multiple board support	Yes

IP-Digital 24 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Carrier slot — Select the slot on the carrier board into which the IP-Digital 24 module is plugged. Note that different carrier boards can have different slot capacities.

Channel vector — Enter numbers between 1 and 24 to select the digital input lines to be used. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of lines used.

For example, to use the first three digital inputs, enter

[1,2,3]

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

IP-Digital 24 Digital Output

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Channel vector — Enter numbers between 1 and 24 to select the digital output lines to be used. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of lines used.

For example, to use the first three digital outputs, enter

[1,2,3]

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

IP-HiADC

The IP-HiADC I/O board has 16 analog input (A/D) channels (12-bit).

xPC Target supports this board with this driver block:

- “IP-HiADC Analog Input (A/D)”

Board Characteristics

Board name	IP-HiADC
Manufacturer	SBS Technologies
Bus type	N/A
Access method	I/O Mapped
Multiple block instance support	Yes
Multiple board support	Yes

IP-HiADC Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a ‘Carrier ID’ parameter, which must be set to a number not shared by any other carrier board block in the model.

Carrier slot — Select the slot on the carrier board into which the IP-HiADC module is plugged. Note that different carrier boards can have different slot capacities.

Channel vector — Enter a vector of numbers between 1 and 16. The channel numbers can occur in any order. For example, to use the first and third analog output (A/D) channels, enter

[1, 3]

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range — Select either `-5V to 5V` or `-10V to 10V`. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

IP-Synchro

IP-Synchro provides two channels of position measurement using synchro, resolver, LVDT, or Inductosyn® transducers.

xPC Target supports this board with this driver block:

- "IP-Synchro"

Board Characteristics

Board name	IP-Synchro
Manufacturer	SBS Technologies
Bus type	N/A
Access method	I/O Mapped
Multiple block instance support	No
Multiple board support	Yes

IP-Synchro

Scaling Input to Output

Hardware Output	Block Output Data Type	Scaling
Synchro or Resolver	double	angle in radians

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP-Synchro module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Carrier slot — Select the slot on the carrier board into which the IP-Synchro module is plugged. Note that different carrier boards can have different slot capacities.

Channel vector — Enter a vector with one or two elements to select the Synchro/Resolver input channels you use with this block. 1 represents channel A and 2 represents channel B. This driver allows the selection of inputs in either order.

For example to use channels A and B in that order, enter

[1,2]

Position compare vector — This must be a scalar or a vector the same length as the channel vector. For each input channel it specifies a position in radians with which the current position is compared. When they match to within 12 bits of precision, a status bit for the channel is set.

Precision vector — This must be a scalar or a vector the same length as the channel vector. For each input channel it specifies a precision of either 10, 12, 14, or 16 bits. If the automatic precision option is in effect, then the selected precision for each channel will be used for the entire run. Otherwise it merely specifies the initial precision.

Automatic precision — Select this check box to have the IP-Synchro automatically change its precision (resolution) in order to track the input velocity.

Output status 1 — Select this option to make the IP-Synchro STATUS 1 register available in an 8-bit output port.

Output status 2 — Select this option to make the IP-Synchro STATUS 2 register available in an 8-bit output port.

Sample time — Enter a base sample time or a multiple of the base sample time.

IP-Unidig-E-48

IP-Unidig-E-48 boards have 48 digital I/O lines which can be independently configured for input or output.

xPC Target supports this board with these driver blocks:

- "IP-Unidig-E-48 Digital Input"
- "IP-Unidig-E-48 Digital Output"

Board Characteristics

Board name	IP-Unidig-E-48
Manufacturer	SBS Technologies
Bus type	N/A
Access method	I/O Mapped
Multiple block instance support	Yes
Multiple board support	Yes

IP-Unidig-E-48 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Carrier slot — Select the slot on the carrier board into which the IP-Unidig-E-48 module is plugged. Note that different carrier boards can have different slot capacities.

Channel vector — Enter numbers between 1 and 48 to select the digital input lines to be used. This driver allows the selection of individual digital input lines in any order. The number of elements defines the number of lines used.

For example, to use the first three digital inputs, enter

[1,2,3]

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

IP-Unidig-E-48 Digital Output

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Carrier ID — Enter the Carrier ID of the IP carrier board into which the IP module is plugged.

In order to use an IP module, it must be physically plugged into an IP carrier board and the model must contain a block representing this carrier board. Each carrier block has a 'Carrier ID' parameter, which must be set to a number not shared by any other carrier board block in the model.

Channel vector — Enter numbers between 1 and 48 to select the digital output lines to be used. This driver allows the selection of individual digital output lines in any order. The number of elements defines the number of lines used.

For example, to use the first three digital outputs, enter

[1,2,3]

Number the lines beginning with 1 even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter a base sample time or a multiple of the base sample time.

PCI-40A Carrier Board

Each IP module must be physically plugged into an IP carrier board such as the PCI-40A. Models containing IP module blocks must also contain corresponding IP carrier blocks. The relationship 'IP module A is plugged into carrier B' is expressed by selecting the same Carrier ID for both A and B.

The PCI-40A holds up to four IP modules.

xPC Target supports this board with this driver block:

- "PCI-40A"

Board Characteristics

Board name	PCI-40A carrier board
Manufacturer	SBS Technologies
Bus type	PCI
Access method	I/O Mapped
Multiple block instance support	No
Multiple board support	Yes

PCI-40A

Driver Block Parameters

Carrier ID — Enter a number to uniquely identify this IP carrier board within the model.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter -1

If two or more boards of this type are physically present in the target PC, enter the PCI slot number of the board associated with this driver block.

21

Softing

I/O boards supported by xPC Target.

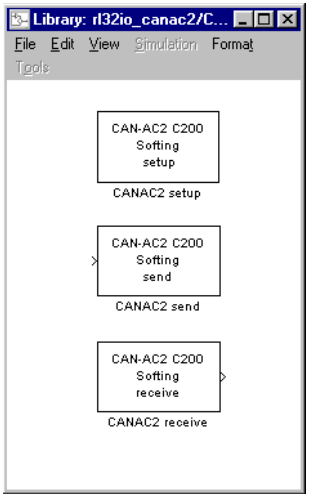
CAN-AC2-ISA (p. 21-2)	For I/O-drivers to connect xPC Target-applications to the CAN-fieldbus xPC Target CAN-AC2 is provided as an extension to the xPC Target basic package.
CAN-AC2-PCI (p. 21-12)	For I/O-drivers to connect xPC Target-applications to the CAN-fieldbus xPC Target CAN-AC2 is provided as an extension to the xPC Target basic package.
CAN-AC2 and CANopen Devices (p. 21-17)	xPC Target CAN-AC2 supports CAN specification 2.0a and 2.0b but this does not generally include the CANopen protocol on driver level. Nevertheless it is possible to access CANopen devices by the CAN-AC2 drivers in a general way.

CAN-AC2-ISA

For I/O drivers to connect xPC Target applications to the CAN fieldbus, xPC Target provides CAN-AC2 as an extension to the xPC Target basic package. See the xPC Target documentation for additional information.

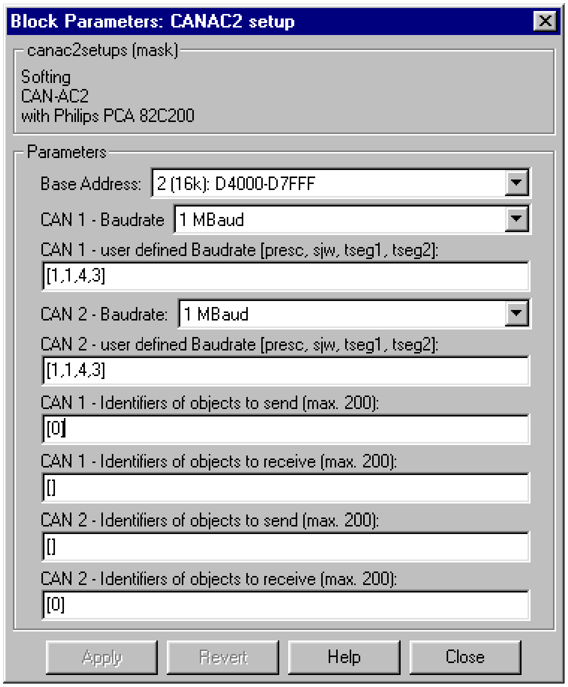
CAN-AC2-ISA with Philips PCA82C200

The second level of the library contains three driver blocks; one for setting up the board, one for sending CAN-messages, and one for receiving CAN-messages.



Setup Block

Every Simulink model which sends and receives CAN-messages over the CAN-AC2 board has to contain exactly one setup-block. The setup-block does not have any inputs or outputs.



The dialog box allows to define general settings for the CAN-AC2 board. The corresponding code (initializing the board) is executed once during the “initializing blocks” phase after the xPC Target application has been downloaded.

The first dialog field (pop-up) allows to specify the memory-address range used to access the board. The CAN-AC2 can be mapped into memory between D0000-EFFFF. See the CAN-AC2 user's guide for further information. If used with xPC Target memory mapped I/O-devices can only be mapped into a subarea of the choosable memory range of the CAN-AC2.

We recommend to use the following configurations if using xPC Target Version 1.1

```
2:  D4000-D7FFF
3:  D8000-D8FFF
```

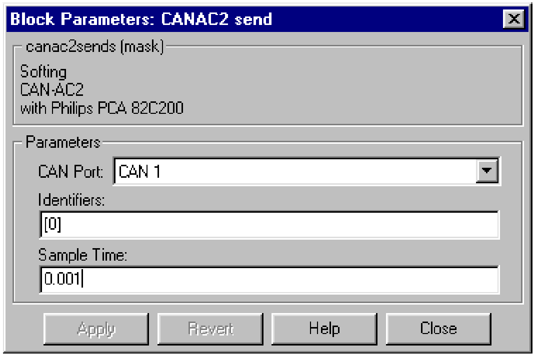
The second and third dialog field allows to choose the baudrate of CAN-port 1. If the baudrate within the pop-up menu is not set to `User defined`, the entries in the user defined dialog field have no meaning. If it is set to `User defined`, a wide range of baudrates can be set by setting Prescaler, Width, Tseg1, Tseg2 to appropriate values. See the CAN-AC2 manual for detailed information how to set the user defined baudrates.

With the fourth and fifth dialog field the baudrate for CAN-port 2 can be set.

The last 4 dialog entries are used to define the identifiers of all CAN-messages sent or received within the current Simulink model. There is one dialog field for send- and receive-identifiers for CAN-port 1 and 2. Each entry can contain a row vector with a maximal number of 200 identifiers. Each identifier can be in the range of 0..2031. In a vector each identifier can only be set once.

Send Block

To send CAN-messages specified in the Setup block, a Simulink model can contain as many as needed Send blocks.



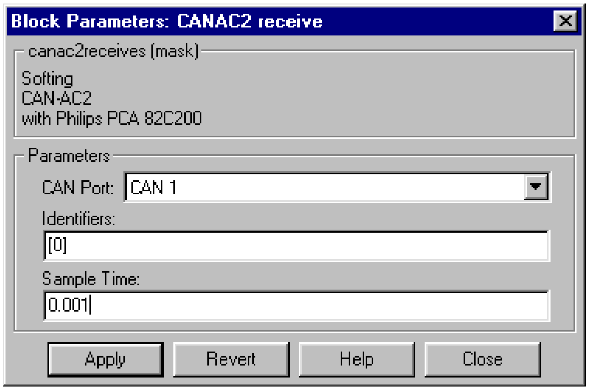
The first dialog entry specifies via which CAN-port the CAN-message should be sent.

The second dialog entry specifies the identifiers for CAN-messages to be sent. The identifiers, entered as a row vector, have to be a subset of the send identifiers defined in the Setup block of either CAN-port 1 or 2. If an identifier is specified that is not defined in the Setup block, an error message appears after the xPC Target application is downloaded. The block has as many inputs as the row vector has elements. The data (double / 8byte) of the first input is sent as the CAN-message with the identifier of the first element of the vector. The second input is sent with the identifier of the second element of the vector and so on.

The third dialog field specifies at which sample time intervals the CAN-messages are sent. By using more than one Send block, it is possible to send CAN-messages at different sample time intervals even with the same identifiers by entering appropriate sample times for each Send block.

Receive Block

To receive CAN-messages specified in the Setup block, a Simulink model can contain as many as needed Receive blocks.



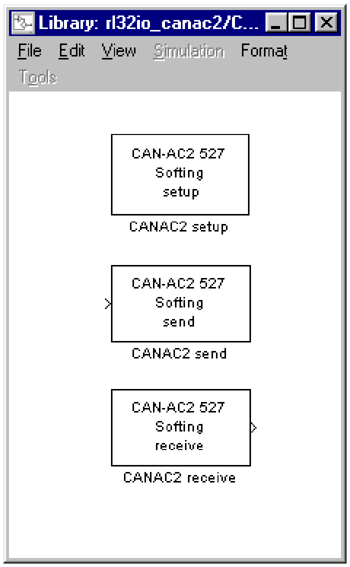
The first dialog entry specifies from which CAN-port the CAN-message should be received.

The second dialog entry specifies the identifiers for CAN-messages to be received. The identifiers, entered as a row vector, have to be a subset of the receive identifiers defined in the Setup block of either CAN-port 1 or 2. If the an identifier is specified that is not defined in the Setup block, an error message appears after the xPC Target application is downloaded. The block has as many outputs as the row vector has elements. The data (double / 8byte) received with the identifier as the first element of the output is the first block output and so on.

The third dialog field specifies at which sample time intervals the CAN-messages have to be read out of the object buffer. By using more than one Receive block, it is possible to get CAN-messages at different sample time intervals even with the same identifiers by entering appropriate sample times for each Receive block.

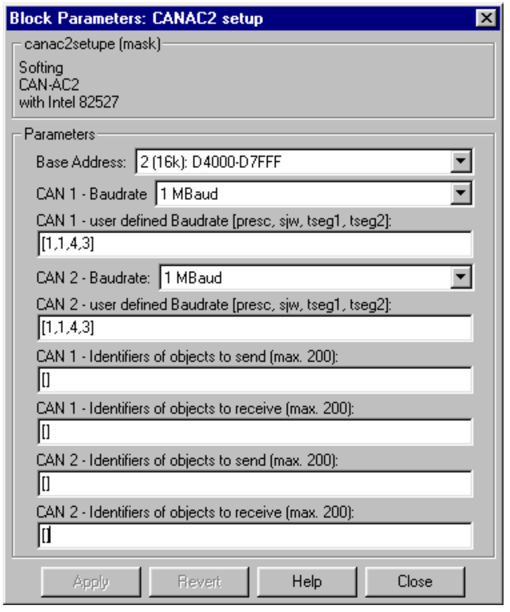
CAN-AC2-ISA with Intel 82527

The second level of the library contains three driver blocks; one for setting up the board, one for sending CAN-messages, and one for receiving CAN-messages.



Setup Block

Every Simulink model that sends and receives CAN-messages over the CAN-AC2 board must contain exactly one Setup block. The Setup block does not have any inputs or outputs.



The dialog box allows you to define general settings for the CAN-AC2 board. The corresponding code (initializing the board) is executed once during the "initializing blocks" phase after the xPC Target application has been downloaded.

The first dialog field (pop-up) allows to specify the memory-address range used to access the board. The CAN-AC2 can be mapped into memory between D0000-EFFFF. See the CAN-AC2 user's guide for further information. If used with xPC Target, memory mapped I/O-devices can only be mapped into a subarea of the choosable memory range of the CAN-AC2.

Use the following configurations if you are using xPC Target Version 1.1

```
2:   D4000-D7FFF
3:   D8000-D8FFF
```

The second and third dialog field allows you to choose the baudrate of CAN-port 1. If the baudrate within the pop-up menu is not set to `User defined` the entries in the user-defined dialog field have no meaning. If it is set to `User defined,` a wide range of baudrates can be set by setting Prescaler, Width, Tseg1, Tseg2 to appropriate values. See the CAN-AC2 manual for detailed information how to set the user defined baudrates.

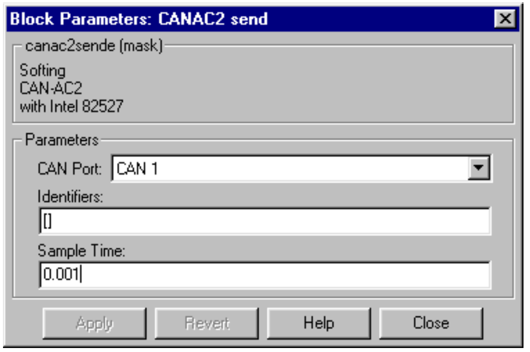
With the fourth and fifth dialog field the baudrate for CAN-port 2 can be set.

The last four dialog entries are used to define the identifiers of all CAN-messages sent or received within the current Simulink model. There is one dialog field for send- and receive-identifiers for CAN-port 1 and 2. Each entry can contain a row vector with a maximal number of 200 identifiers. Each identifier can be in the range of -2032..(2^29-1). Because CAN-specification 2.0B allows you to send and receive messages with standard (11bit) and extended identifiers (29bit) concurrently, the following identifier numbering method has been implemented:

- Positive numbers specify extended identifiers and can be in the range from 0..2^29-1
- Negative numbers specify standard identifiers. Because the number zero is reserved for the extended identifier 0, the standard identifier 0 has the number -1. The standard identifier 1 has the number -2 and so on. Therefore, the standard identifier range 0 to 2031 is mapped to the range -1 to -2032.

Send Block

To send CAN-messages specified in the Setup block, a Simulink model can contain as many as needed Send blocks.



The first dialog entry specifies via which CAN-port the CAN-message should be sent.

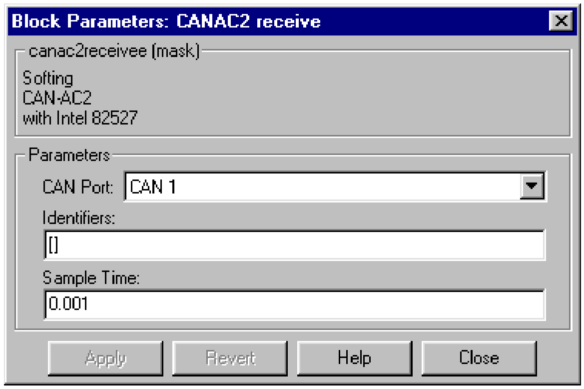
The second dialog entry specifies the identifiers for CAN-messages to be sent. The identifiers, entered as a row-vector, have to be a subset of the send identifiers defined in the Setup block of either CAN-port 1 or 2. If an identifier is specified that is not defined in the Setup block, an error message appears after the xPC Target application is downloaded. The block has as many inputs as the row vector has elements.

The data (double / 8byte) of the first input is sent as the CAN-message with the identifier of the first element of the vector. The second input is sent with the identifier of the second element of the vector and so on.

The third dialog field specifies at which sample time intervals the CAN-messages are sent. By using more than one Send block, it is possible to send CAN-messages at different sample time intervals even with the same identifiers by entering appropriate sample times for each Send block.

Receive Block

To receive CAN-messages specified in the Setup block, a Simulink model can contain as many as needed Receive blocks.



The first dialog entry specifies from which CAN-port the CAN-message should be received.

The second dialog entry specifies the identifiers for CAN-messages to be received. The identifiers, entered as a row vector, have to be a subset of the receive identifiers defined in the Setup block of either CAN-port 1 or 2. If an identifier is specified that is not defined in the Setup block, an error message appears after the xPC Target application is downloaded. The block has as many outputs as the row vector has elements. The data (double / 8byte) received with the identifier as the first element of the output is the first block output and so on.

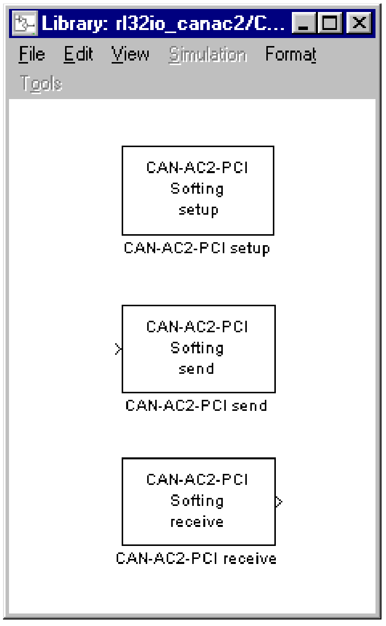
The third dialog field specifies at which sample time intervals the CAN-messages have to be read out of the object buffer. By using more than one Receive block, it is possible to get CAN-messages at different sample time intervals even with the same identifiers by entering appropriate sample times for each Receive block.

CAN-AC2-PCI

For I/O-drivers to connect xPC Target applications to the CAN-fieldbus, xPC Target provides CAN-AC2 as an extension to the xPC Target basic package. See the xPC Target User's Guide for additional information.

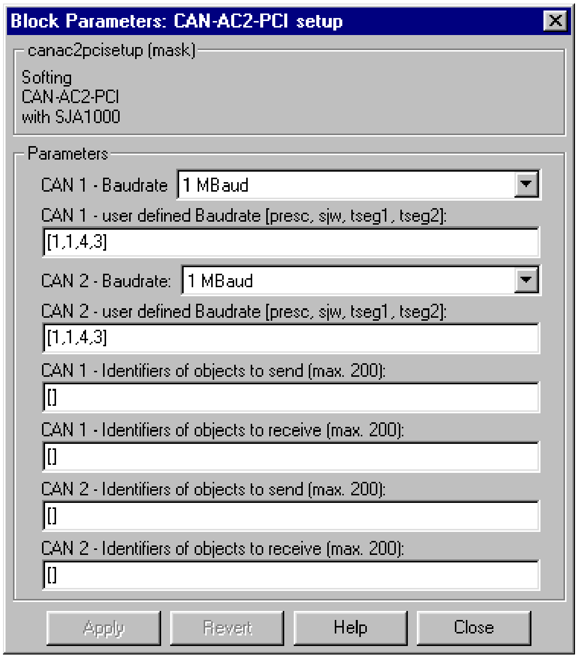
CAN-AC2-PCI with SJA 1000

The second level of the library contains three driver blocks; one for setting up the board, one for sending CAN-messages, and one for receiving CAN-messages.



Setup Block

Every Simulink model which sends and receives CAN-messages over the CAN-AC2-PCI board has to contain exactly one Setup block. The Setup block does not have any inputs or outputs.



The dialog box allows to define general settings for the CAN-AC2-PCI board. The corresponding code (initializing the board) is executed once during the "initializing blocks" phase after the xPC Target application has been downloaded.

The first and second dialog field allows to choose the baudrate of CAN-port 1. If the baudrate within the pop-up menu is not set to `User defined`, the entries in the user defined dialog field have no meaning. If it is set to `User defined`, a wide range of baudrates can be set by setting Prescaler, Width, Tseg1, Tseg2 to appropriate values. See the CAN-AC2-PCI manual for detailed information how to set the user defined baudrates.

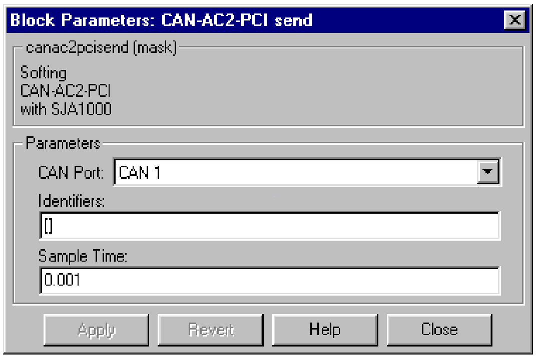
With the third and fourth dialog field the baudrate for CAN-port 2 can be set.

The last four dialog entries are used to define the identifiers of all CAN-messages sent or received within the current Simulink model. There is one dialog field for send- and receive-identifiers for CAN-port 1 and 2. Each entry can contain a row vector with a maximal number of 200 identifiers. Each identifier can be in the range of -(2^29)..2031). Because CAN-specification 2.0B allows to send and receive messages with standard (11bit) and extended identifiers (29bit) concurrently, the following identifier numbering method has been implemented:

- Positive numbers specify standard identifiers and can therefore be in the range from 0..2031
- Negative numbers specify extended identifiers. Because the number zero is reserved for the standard identifier 0 the extended identifier 0 has the number -1 and the extended identifier 1 the number -2 and so on. Therefore the standard identifier range 0 to 2^29-1 is mapped to the range -1 to -(2^29).

Send-block

To send CAN-messages specified in the setup-block, a Simulink model can contain as many as needed send-blocks.



The first dialog entry specifies via which CAN-port the CAN-message should be sent.

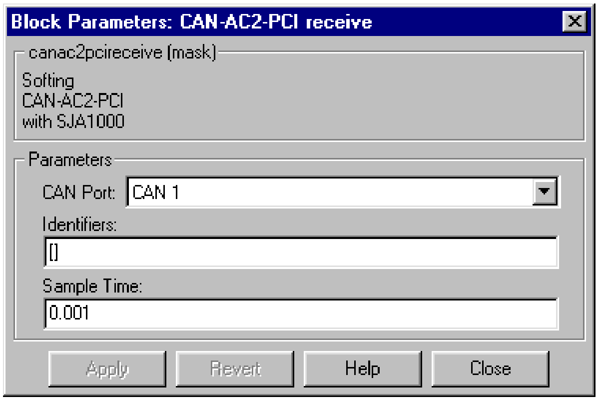
The second dialog entry specifies the identifiers for CAN-messages to be sent. The identifiers, entered as a row vector, have to be a subset of the send identifiers defined in the setup-block of either CAN-port 1 or 2. If an identifier is specified that is not defined in the Setup block, an error message appears after the xPC Target application is downloaded. The block has as many inputs as the row vector has elements.

The data (double / 8byte) of the first input is sent as the CAN-message with the identifier of the first element of the vector. The second input is sent with the identifier of the second element of the vector and so on.

The third dialog field specifies at which sample time intervals the CAN-messages are sent. By using more than one Send block, it is possible to send CAN-messages at different sample time intervals even with the same identifiers by entering appropriate sample times for each Send block.

Receive Block

To receive CAN-messages specified in the Setup block, a Simulink model can contain as many as needed Receive blocks.



The first dialog entry specifies from which CAN-port the CAN-message should be received.

The second dialog entry specifies the identifiers for CAN-messages to be received. The identifiers, entered as a row vector, have to be a subset of the receive identifiers defined in the Setup block of either CAN-port 1 or 2. If an identifier is specified that is not defined in the Setup block, an error message appears after the xPC Target application is downloaded. The block has as many outputs as the row vector has elements. The data (double / 8byte) received with the identifier as the first element of the output is the first block output and so on.

The third dialog field specifies at which sample time intervals the CAN-messages have to be read out of the object buffer. By using more than one Receive block, it is possible to get CAN-messages at different sample time intervals even with the same identifiers by entering appropriate sample times for each Receive block.

CAN-AC2 and CANopen Devices

xPC Target CAN-AC2 supports CAN specification 2.0a and 2.0b but this does not generally include the CANopen protocol on driver level. Nevertheless it is possible to access CANopen devices by the CAN-AC2 drivers in a general way.

CANopen knows two types of messages i.e. SDO and PDO. SDOs are used to setup or initialize a CANopen device for a certain behavior. PDOs are messages that contain real-time data (i.e. converted A/D values from a analog input device) and are CAN-type messages with no CANopen object, index, and subindex information.

xPC Target applications that have to access CANopen devices over the CAN-AC2 drivers transmit SDOs during the initialization phase and the termination phase of the driver. PDOs are sent or received during the simulation phase of the driver.

Because SDOs and PDOs are regular CAN-messages the CAN-AC2 drivers have to provide a way to transmit SDOs during the initialization and termination phase of the CAN-AC2 set up driver block to initialize the different CANopen devices in the network. This is done by providing a c-file within your project directory that describes the SDO messages to send to setup and terminate the CANopen device. During the compilation stage of the xPC Target application (build-process) this c-file, which has to have the filename `CANAC2_setup.c`, is then included into the setup driver.

This implementation has the advantage of accessing a specific CANopen device without the need to have special driver blocks for this device. It is, therefore, a general implementation but has the disadvantage that the user must be able to provide the information (messages) to properly set up and terminate the communication with a specific CANopen device. This information is provided either by the CANopen device manufacturer or by the CAN-CIA association (`www.can-cia.de`).

For an explanation of how to write the `CANAC2_setup.c` file for a specific CANopen device, see the example below. In this example an analog input device from Selectron (`www.selectron.ch`) with the name AIC711 is used to get the A/D-converted values over the CAN-network into the xPC Target application.

Note CANopen initialization and termination is only supported if the CAN-AC2 board is equipped with the Philips C200 controller for standard identifiers.

Example: Accessing the AIC711 CANopen Device from Selectron

The AIC711 contains four analog input channels with a resolution of 12bits and a minimal update-time (sample time) of 10ms.

As explained in earlier chapters, the CAN-AC2 drivers use the dynamic object model to reach low latency times. Therefore the A/D values from the AIC711 have to be received in such a way that they are compatible to the object model of the driver.

The AIC711 has to be seen as a CANopen server and the xPC Target CAN-AC2 drivers (the xPC Target application) as a CANopen client. The AIC711 offers two ways of getting the converted A/D-values over the network:

- Synchronous
- Asynchronous

In the synchronous mode, the client transmits a remote frame to the server to invoke an A/D-conversion of a specified channel. It then waits (poll) until the converted value is received by an ordinary CAN data-message which will contain the values. Synchronous mode leads to large latency times up to 20ms (Tsmin=10ms). During this time period, the xPC Target gets stuck and this is inacceptable.

In addition, the synchronous mode does not fit into the dynamic object model implementation of the xPC Target CAN-drivers because remote frames have to be transmitted.

In the asynchronous mode, the AIC711 sends PDOs automatically in a regular manner to the client. A change in an analog input value invokes automatically an A/D-conversion. After conversion, a PDO-message is constructed and sent automatically to the client. This mode fits well into the object model of the drivers. Therefore the CANopen devices should always be used in asynchronous mode if used with xPC Target.

Regarding the information in the AIC711 CANopen manual (provided by Selectron), the following initialization messages (SDOs) and termination messages (SDOs) must be invoked.

- **Initialization phase** — Enable global interrupts to enable asynchronous mode (object 6423). Put device from preoperational mode into operational mode (transmission of PDOs starts).
- **Simulation phase** — CAN-AC2 Receive block outputs the latest received A/D values.
- **Termination phase —** Put device from operational mode into preoperational mode (transmission of PDOs stops)

The node id of the AIC711 device is set over DIP-switches and, in this example, it is assumed that the node id is set to 11 (decimal). The device is connected to CAN-port 1 of the CAN-AC2-board.

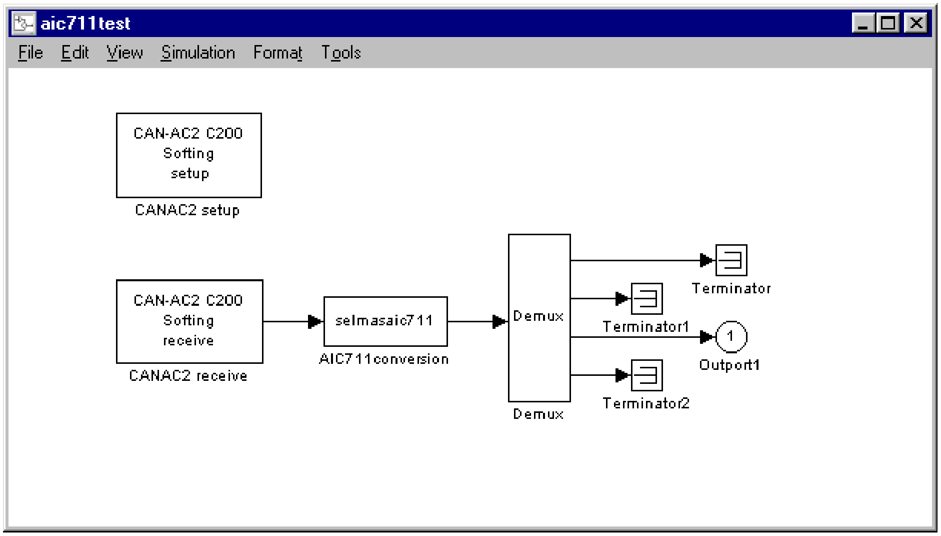
Then the `CANAC2_setup.c` file could look as follows

```
//////////////////////////////////////////////////////////////////////
// Number of initialization and termination messages
//////////////////////////////////////////////////////////////////////
#define CANAC2_init_number2
#define CANAC2_term_number1
//#define DEBUG_CANAC2

// do not change the following four lines
#define CANAC2_setup_present
CANAC2_type CANAC2_init[CANAC2_init_number+1];
CANAC2_type CANAC2_term[CANAC2_term_number+1];
int CANAC2_counter;
//


//////////////////////////////////////////////////////////////////////
// Identifier and constant section
//////////////////////////////////////////////////////////////////////


#define AIC711_node_111
#define AIC711_sdo_base1536
#define MAS_boot   0
```

```
////////////////////////////////////////////////////////////////
// Initialization section
////////////////////////////////////////////////////////////////

// AIC711 SDO object 6423: enable global interupts
CANAC2_init[0].port=1;
CANAC2_init[0].identifier=AIC711_sdo_base+AIC711_node_1;
CANAC2_init[0].data[0]=0x22;
CANAC2_init[0].data[1]=0x23;
CANAC2_init[0].data[2]=0x64;
CANAC2_init[0].data[3]=0x00;
CANAC2_init[0].data[4]=0x01;
CANAC2_init[0].no_bytes=5;
CANAC2_init[0].wait_ms=20;

// put AIC711_node_1 from pre-operational into operational state
CANAC2_init[1].port=1;
CANAC2_init[1].identifier=MAS_boot;
CANAC2_init[1].data[0]=0x01;
CANAC2_init[1].data[1]=AIC711_node_1;
CANAC2_init[1].no_bytes=2;
CANAC2_init[1].wait_ms=20;


////////////////////////////////////////////////////////////////
// Termination section
////////////////////////////////////////////////////////////////


// put AIC711_node_1 from operational into pre-operational state
CANAC2_term[0].port=1;
CANAC2_term[0].identifier=MAS_boot;
CANAC2_term[0].data[0]=0x80;
CANAC2_term[0].data[1]=AIC711_node_1;
CANAC2_term[0].no_bytes=2;
CANAC2_term[0].wait_ms=20;
```

As soon as this file is placed into your project directory and the xPC Target application is rebuilt, the messages defined above will be sent during initialization and termination phase of the Setup block.

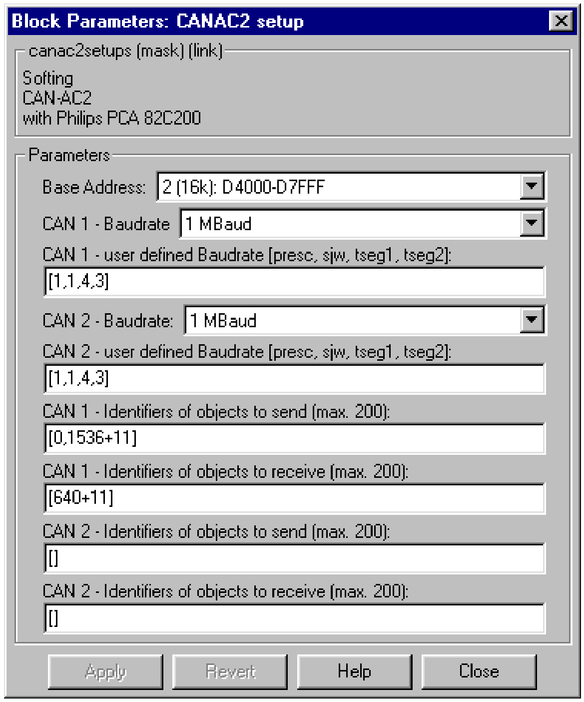
The Simulink model could look as follows



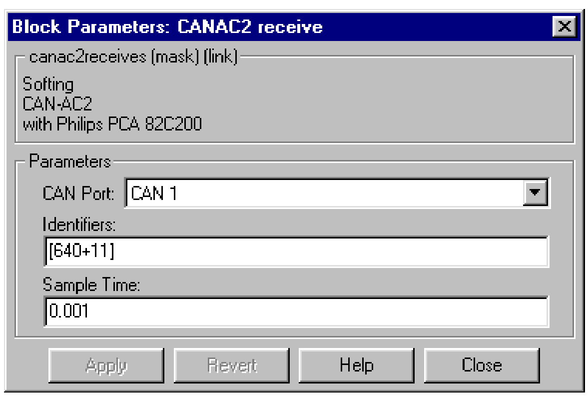
The Receive block will read continuously the object to which the AIC711 sends the PDOs (i.e. the converted A/D-values).

Because the output of this block contains the 8 bytes of the received CAN-data as a double value, a conversion block (AIC711conversion) is necessary to split the 8 bytes (double) into 4 doubles (output signals) that represent the A/D value in volts for each of the four analog input channels. The conversion is made according to the data representation of object 6401. Use the `aselmasaic711.c` file as a template to implement conversion blocks for other CANopen devices. The third channel is then stored with an outport block, which can be visualized by the xPC Target scope functionality.

Because CAN-messages with id 0 (boot) and 1536+node_id (SDO) have to be sent and CAN-messages with id 640+node_id (PDO) have to be received over CAN-port 1, the dialog box of the Setup block must look as follows:



The Receive block receives the data (PDO) over CAN-message 640+node-i and must look as follows:



If more than one CANopen device is connected to the network, the dialog boxes of the Setup and Receive blocks and the `CANAC2_setup.c` file have to be extended accordingly. If you need for-loops in the `CANAC2_setup.c`, use the variable CANAC2_counter.

If an analog output device (or digital output device) is connected to the network, you must drag an additional Send block into the model to send the PDOs to the newly connected CANopen server.

United Electronic Industries (UEI)

Groups of boards supported by xPC target.

PDXI-MF 12-Bit Series (p. 22-43)	16 or 64 single or 8 or 32 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 1.25MHz. The gain varies from 1 to 8 or 1 to 1000. The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines.
PDXI-MF 14-Bit Series (p. 22-49)	16 or 64 single or 8 or 32 differential analog input (A/D) channels (14-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines.
PDXI-MF 16-Bit Series (p. 22-55)	16 or 64 single or 8 or 32 differential analog input (A/D) channels (16-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines.
PDXI-MFS 12-Bit Series (p. 22-62)	4 or 8 single analog input (A/D) channels (12-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines.
PDXI-MFS 14-Bit Series (p. 22-68)	4 or 8 single analog input (A/D) channels (14-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines.
PDXI-MFS 16-Bit Series (p. 22-75)	4 or 8 single analog input (A/D) channels (16-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines.
PD2-AO Series (p. 22-81)	8, 16, or 32 analog output (D/A) channels (16-bit) with a maximum sample rate of 100 kHz per channel. They also have 8 digital input and 8 digital output lines.
PDXI-AO Series (p. 22-86)	8, 16, or 32 analog output (D/A) channels (16 bit) with a maximum sampling rate of 100 kHz per channel. They also have 8 digital input lines, and 8 digital output lines.

Grouping the UEI Boards

The United Electronic Industries (UEI) PowerDAQ board series contains a large number of boards. The board names follow a standard pattern:

[Form Factor]-[Board Type]-[Channel]-[Speed]/[Resolution][Gain]

For example, one UEI board is named PD2-MF-16-1M/12L. The possibilities for the parts of the name are

- **Form Factor** — The form factor can be one of two things. PD2 denotes a 32-bit, 33-MHz PCI bus board. PDXI indicates a 32-bit, 33-MHz PXI/cPCI bus board.
- **Board Type** — The board type can be one of three things. MF indicates a multifunction board. MFS indicates a multifunction board with sample and hold. AO indicates an analog output board.
- **Channel** — For MF and MFS board types, the channel is the number of analog input channels. For AO board types, the channel is the number of analog output channels.
- **Speed** — The speed is the number of samples per second supported by the board.
- **Resolution** — For MF and MFS boards, the Resolution is the analog input resolution. For AO board types, the Resolution is the analog output resolution.
- **Gain** — Gain is denoted by letters that represent ranges. DG indicates a gain of 1, 2, 5, or 10. L indicates a range of 1, 10, 100, or 1000. H indicates a range of 1, 2, 4, or 8.

For example, the PD2-MF-16-1M/12L is a PCI multifunction board with sixteen 12-bit analog input channels supporting one million (1M) samples per second with available gains of 1, 10, 100, or 1000.

Changing the Board Associated with the Block

Note that the board displayed on the block is the current board type. To specify a different board, double-click on the block. In the **Board type** list, choose a different board.

Getting Information on a Specific Board

The boards in this manual are grouped by Form Factor, Board Type, and Resolution. For more information on a specific board, open the MATLAB Help browser. Click on the **Search** tab. Verify that the **Product Filter** is set to **All** or that xPC Target is one of your selected products. In the **Search type** list, choose **Full Text**. In the **Search for** text box, type the name of your board. Click **Go**. Browse through the search results for more information on your board.

PD2-MF 12-Bit Series

The PD2-MF 12-bit series contains I/O boards with 16 or 64 single or 8 or 32 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 1.25MHz. The gain varies from 1 to 8 or 1 to 1000. The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PD2-MF 12-Bit Series Analog Input (A/D)"
- "PD2-MF 12-Bit Series Analog Output (D/A)"
- "PD2-MF 12-Bit Series Digital Input"
- "PD2-MF 12-Bit Series Digital Output"

Board Characteristics

Board type	PD2-MF-16-1M/12L PD2-MF-16-1M/12H PD2-MF-64-1M/12L PD2-MF-64-1M/12H
Manufacturer	United Electronic Industries (UEI)
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PD2-MF 12-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 12-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 12-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 12-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 14-Bit Series

The PD2-MF 14-bit series contains I/O boards with 16 or 64 single or 8 or 32 differential analog input (A/D) channels (14-bit). The gain varies from 1 to 8 or 1 to 1000. The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input lines, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PD2-MF 14-Bit Series Analog Input (A/D)"
- "PD2-MF 14-Bit Series Analog Output (D/A)"
- "PD2-MF 14-Bit Series Digital Input"
- "PD2-MF 14-Bit Series Digital Output"

Board Characteristics

Board type	PD2-MF-16-400/14L PD2-MF-16-400/14H PD2-MF-64-400/14L PD2-MF-64-400/14H PD2-MF-16-800/14L PD2-MF-16-800/14H PD2-MF-64-800/14L PD2-MF-64-800/14H PD2-MF-16-2M/14L PD2-MF-16-2M/14H PD2-MF-64-2M/14L PD2-MF-64-2M/14H
Manufacturer	United Electronic Industries (UEI)
Bus type	PCI, ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PD2-MF 14-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 14-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 14-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 14-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 16-Bit Series

The PD2-MF 16-bit series contains I/O boards with 16 or 64 single or 8 or 32 differential analog input (A/D) channels (16-bit). The gain varies from 1 to 8 or 1 to 1000. The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PD2-MF 16-Bit Series Analog Input (A/D)"
- "PD2-MF 16-Bit Series Analog Output (D/A)"
- "PD2-MF 16-Bit Series Digital Input"
- "PD2-MF 16-Bit Series Digital Output"

Board Characteristics

Board type	PD2-MF-16-50/16L PD2-MF-16-50/16H PD2-MF-64-50/16L PD2-MF-64-50/16H PD2-MF16-100/16L PD2-MF-16-100/16H PD2-MF-64-100/16L PD2-MF-64-100/16H PD2-MF-16-333/16L PD2-MF-16-333/16H PD2-MF-64-333/16L PD2-MF-64-333/16H PD2-MF-16-500/16L PD2-MF-16-500/16H PD2-MF-64-500/16L PD2-MF-64-500/16H
Manufacturer	United Electronic Industries (UEI)
Bus type	PCI, ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PD2-MF 16-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 16-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 16-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MF 16-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 12-Bit Series

The PD2-MFS 12-bit series contains I/O boards with 4 or 8 single analog input (A/D) channels (12-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- “PD2-MFS 12-Bit Series Analog Input (A/D)”
- “PD2-MFS 12-Bit Series Analog Output (D/A)”
- “PD2-MFS 12-Bit Series Digital Input”
- “PD2-MFS 12-Bit Series Digital Output”

Board Characteristics

Board type	PD2-MFS-4-1M/12 PD2-MFS-4-1M/12DG PD2-MFS-4-1M/12H PD2-MFS-8-1M/12 PD2-MFS-8-1M/12DG PD2-MFS-8-1M/12H
Manufacturer	United Electronic Industries (UEI)
Bus type	PCI, ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PD2-MFS 12-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 12-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 12-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 12-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 14-Bit Series

The PD2-MFS 14-bit series contains I/O boards with 4 or 8 single analog input (A/D) channels (14-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PD2-MFS 14-Bit Series Analog Input (A/D)"
- "PD2-MFS 14-Bit Series Analog Output (D/A)"
- "PD2-MFS 14-Bit Series Digital Input"
- "PD2-MFS 14-Bit Series Digital Output"

Board Characteristics

Board type	PD2-MFS-4-500/14 PD2-MFS-4-500/14DG PD2-MFS-4-500/14H PD2-MFS-8-500/14 PD2-MFS-8-500/14DG PD2-MFS-8-500/14H PD2-MFS-4-800/14 PD2-MFS-4-800/14DG PD2-MFS-4-800/14H PD2-MFS-8-800/14 PD2-MFS-8-800/14DG PD2-MFS-8-800/14H PD2-MFS-4-2M/14 PD2-MFS-4-2M/14DG PD2-MFS-4-2M/14H PD2-MFS-8-2M/14 PD2-MFS-8-2M/14DG PD2-MFS8-2M/14H
Manufacturer	United Electronic Industries (UEI)
Bus type	PCI, ISA

Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PD2-MFS 14-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 14-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 14-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 14-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 16-Bit Series

The PD2-MFS 16-bit series contains I/O boards with 4 or 8 single analog input (A/D) channels (16-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PD2-MFS 16-Bit Series Analog Input (A/D)"
- "PD2-MFS 16-Bit Series Analog Output (D/A)"
- "PD2-MFS 16-Bit Series Digital Input"
- "PD2-MFS 16-Bit Series Digital Output"

Board Characteristics

Board type	PD2-MFS-4-300/16 PD2-MFS-4-300/16DG PD2-MFS-4-300/16H PD2-MFS-8-300/16 PD2-MFS-8-300/16DG PD2-MFS-8-300/16H PD2-MFS-4-500/16 PD2-MFS-4-500/16DG PD2-MFS-4-500/16H PD2-MFS-8-500/16 PD2-MFS-8-500/16DG PD2-MFS-8-500/16H
Manufacturer	United Electronic Industries (UEI)
Bus type	PCI, ISA
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PD2-MFS 16-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 16-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 16-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-MFS 16-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 12-Bit Series

The PDXI-MF 12-bit series contains I/O boards with 16 or 64 single or 8 or 32 differential analog input (A/D) channels (12-bit) with a maximum sample rate of 1.25MHz. The gain varies from 1 to 8 or 1 to 1000. The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PDXI-MF 12-Bit Series Analog Input (A/D)"
- "PDXI-MF 12-Bit Series Analog Output (D/A)"
- "PDXI-MF 12-Bit Series Digital Input"
- "PDXI-MF 12-Bit Series Digital Output"

Board Characteristics

Board type	PDXI-MF-16-1M/12L PDXI-MF-16-1M/12H PDXI-MF-64-1M/12L PDXI-MF-64-1M/12H
Manufacturer	United Electronic Industries (UEI)
Bus type	CompactPCI, PXI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PDXI-MF 12-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 12-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 12-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 12-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 14-Bit Series

The PDXI-MF 14-bit series contains I/O boards with 16 or 64 single or 8 or 32 differential analog input (A/D) channels (14-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- “PDXI-MF 14-Bit Series Analog Input (A/D)”
- “PDXI-MF 14-Bit Series Analog Output (D/A)”
- “PDXI-MF 14-Bit Series Digital Input”
- “PDXI-MF 14-Bit Series Digital Output”

Board Characteristics

Board type	PDXI-MF-16-400/14L PDXI-MF-16-400/14H PDXI-MF-64-400/14L PDXI-MF-64-400/14H PDXI-MF-16-800/14L PDXI-MF-16-800/14H PDXI-MF-64-800/14L PDXI-MF-64-800/14H PDXI-MF-16-2M/14L PDXI-MF-16-2M/14H PDXI-MF-64-2M/14L PDXI-MF-64-2M/14H
Manufacturer	United Electronic Industries (UEI)
Bus type	CompactPCI, PXI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PDXI-MF 14-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 14-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 14-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 14-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 16-Bit Series

The PDXI-MF 16-bit series contains I/O boards with 16 or 64 single or 8 or 32 differential analog input (A/D) channels (16-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PDXI-MF 16-Bit Series Analog Input (A/D)"
- "PDXI-MF 16-Bit Series Analog Output (D/A)"
- "PDXI-MF 16-Bit Series Digital Input"
- "PDXI-MF 16-Bit Series Digital Output"

Board Characteristics

Board type	PDXI-MF-16-50/16L PDXI-MF-16-50/16H PDXI-MF-64-50/16L PDXI-MF-64-50/16H PDXI-MF-16-200/16L PDXI-MF-16-200/16H PDXI-MF-64-200/16H PDXI-MF-64-200/16L PDXI-MF-64-200/16H PDXI-MF-16-333/16L PDXI-MF-16-333/16H PDXI-MF-64-333/16L PDXI-MF-64-333/16H PDXI-MF-16-500/16L PDXI-MF-16-500/16H PDXI-MF-64-500/16L PDXI-MF-64-500/16H
Manufacturer	United Electronic Industries (UEI)
Bus type	CompactPCI, PXI
Access method	Memory mapped

Multiple block instance support	No
Multiple board support	Yes

PDXI-MF 16-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 16-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 16-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MF 16-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 12-Bit Series

The PDXI-MFS 12-bit series contains I/O boards with 4 or 8 single analog input (A/D) channels (12-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PDXI-MFS 12-Bit Series Analog Input (A/D)"
- "PDXI-MFS 12-Bit Series Analog Output (D/A)"
- "PDXI-MFS 12-Bit Series Digital Input"
- "PDXI-MFS 12-Bit Series Digital Output"

Board Characteristics

Board type	PDXI-MFS-4-1M/12 PDXI-MFS-4-1M/12DG PDXI-MFS-4-1M/12H PDXI-MFS-8-1M/12 PDXI-MFS-8-1M/12DG PDXI-MFS-8-1M/12H
Manufacturer	United Electronic Industries (UEI)
Bus type	CompactPCI, PXI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PDXI-MFS 12-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 12-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 12-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 12-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 14-Bit Series

The PDXI-MFS 14-bit series contains I/O boards with 4 or 8 single analog input (A/D) channels (14-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- “PDXI-MFS 14-Bit Series Analog Input (A/D)”
- “PDXI-MFS 14-Bit Series Analog Output (D/A)”
- “PDXI-MFS 14-Bit Series Digital Input”
- “PDXI-MFS 14-Bit Series Digital Output”

Board Characteristics

Board type	PDXI-MFS-4-500/14 PDXI-MFS-4-500/14DG PDXI-MFS-4-500/14H PDXI-MFS-8-500/14 PDXI-MFS-8-500/14DG PDXI-MFS-8-500/14H PDXI-MFS-4-800/14 PDXI-MFS-4-800/14DG PDXI-MFS-4-800/14H PDXI-MFS-8-800/14 PDXI-MFS-8-800/14DG PDXI-MFS-8-800/14H PDXI-MFS-4-2M/14 PDXI-MFS-4-2M/14DG PDXI-MFS-4-2M/14H PDXI-MFS-8-2M/14 PDXI-MFS-8-2M/14DG PDXI-MFS8-2M/14H
Manufacturer	United Electronic Industries (UEI)
Bus type	CompactPCI, PXI

Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PDXI-MFS 14-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 14-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 14-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 14-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 16-Bit Series

The PDXI-MFS 16-bit series contains I/O boards with 4 or 8 single analog input (A/D) channels (16-bit). The boards also have 2 analog output (D/A) channels (12-bit), 16 digital input, and 16 digital output lines. xPC Target does not support the counter/timers on these boards.

xPC Target supports this series of boards with these driver blocks:

- "PDXI-MFS 16-Bit Series Analog Input (A/D)"
- "PDXI-MFS 16-Bit Series Analog Output (D/A)"
- "PDXI-MFS 16-Bit Series Digital Input"
- "PDXI-MFS 16-Bit Series Digital Output"

Board Characteristics

Board type	PDXI-MFS-4-300/16 PDXI-MFS-4-300/16DG PDXI-MFS-4-300/16H PDXI-MFS-8-300/16 PDXI-MFS-8-300/16DG PDXI-MFS-8-300/16H PDXI-MFS-4-500/16 PDXI-MFS-4-500/16DG PDXI-MFS-4-500/16H PDXI-MFS-8-500/16 PDXI-MFS-8-500/16DG PDXI-MFS-8-500/16H
Manufacturer	United Electronic Industries (UEI)
Bus type	CompactPCI, PXI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PDXI-MFS 16-Bit Series Analog Input (A/D)

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, use the first and third analog input (A/D) channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 4, 8, 16, or 64 depending on the board type.

Gain vector — Enter a vector of numbers to select the gains for each of the channels. Available gains vary depending on board type. MF PGL gains may be set to 1, 10, 100, or 1000. MF PGH gains may be set to 1, 2, 4, or 8. MF DG-option gains may be set to 1, 2, 5, or 10.

Mux settling time factor — This vector must be the same length as the channel vector. It contains values of 0 or 1. If 1, the corresponding channel has a longer settling time. This is useful when using a high gain such as 100 or 1000.

Range — Select the voltage range from the list provided. This applies to all channels.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 16-Bit Series Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 16-Bit Series Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-MFS 16-Bit Series Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-AO Series

The PD2-AO series boards have 8, 16, or 32 analog output (D/A) channels (16-bit) with a maximum sample rate of 100 kHz per channel. They also have 8 digital input and 8 digital output lines.

xPC Target supports this series of boards with these driver blocks:

- "PD2-AO Analog Output (D/A)"
- "PD2-AO Digital Input"
- "PD2-AO Digital Output"

Board Characteristics

Board type	PD2-AO-8/16 PD2-AO-16/16 PD2-AO-32/16
Manufacturer	United Electronic Industries (UEI)
Bus type	PCI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PD2-AO Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-AO Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PD2-AO Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-AO Series

The PDXI-AO series boards have 8, 16, or 32 analog output (D/A) channels (16 bit) with a maximum sampling rate of 100 kHz per channel. They also have 8 digital input lines, and 8 digital output lines.

xPC Target supports this series of boards with these driver blocks:

- "PDXI-AO Analog Output (D/A)"
- "PDXI-AO Digital Input"
- "PDXI-AO Digital Output"

Board Characteristics

Board type	PDXI-AO-8/16 PDXI-AO-16/16 PDXI-AO-32/16
Manufacturer	United Electronic Industries (UEI)
Bus type	CompactPCI, PXI
Access method	Memory mapped
Multiple block instance support	No
Multiple board support	Yes

PDXI-AO Analog Output (D/A)

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
volts	double	1

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels between 1 and 16. For example, to use the first and second analog output (D/A) channels, enter

```
[1 2]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number for PD2-AO and PDXI-AO series boards is 8, 16, or 32 depending on the specific board type. For all other boards series the maximum channel number is 2.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector and should contain the initial voltage values for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-AO Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the input channels. For example, to use the first and third digital input channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all boards except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

PDXI-AO Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low > 0.5 = TTL high

Driver Block Parameters

Board type — Select the specific board type from the list provided.

Channel vector — Enter a vector of numbers to specify the output channels. For example, to use the first and third digital output channels, enter

```
[1 3]
```

The channel numbers can occur in any order. Number the channels beginning with 1 even if the board manufacturer numbers them beginning with 0. The maximum allowable channel number is 16 for all board types except for the PD2-AO and PDXI-AO series. For those boards, the maximum channel number is 8.

Reset vector — The reset vector must be the same length as the channel vector. It contains values of 0 or 1. This parameter controls the behavior at model termination. A value of 1 causes the corresponding channel to be reset to the value specified in the initial value vector. A value of 0 causes the channel to remain at the last value attained while the model was running.

Initial value vector — The initial value vector must be the same length as the channel vector. It contains the initial values (0 or 1) for the output channels. The channels are set to these values between the time the model is downloaded and the time it is started.

Sample time — Enter the base sample time or a multiple of the base sample time.

PCI Slot (-1:autosearch) — Enter a number between -1 and n. If only one board of this type is physically present in the target PC, enter

```
-1
```

If two or more boards of this type are physically present in the target PC, enter the bus number and the PCI slot number of the board associated with this driver block. Use the format [BusNumber, SlotNumber]. To determine the bus number and the PCI slot number, type

```
getxpcpci
```

23

Versalogic

I/O boards supported by xPC Target.

VSBC-6 (p. 23-2)	A single board computer with 8 signal ended analog input (A/D) channels, 16 digital I/O lines, and a watchdog timer.

VSBC-6

The VSBC-6 is a single board computer with 8 signal ended analog input (A/D) channels, 16 digital I/O lines, and a watchdog timer.

xPC Target supports this board with these driver blocks:

- "VSBC-6 Analog Input (A/D)"
- "VSBC-6 Digital Input"
- "VSBC-6 Digital Output"
- "VSBC-6 Watch Dog"

Board Characteristics

Board name	VSBC-6
Manufacturer	Versalogic
Bus type	N/A
Access method	N/A
Multiple block instance support	Yes
Multiple board support	No

VSBC-6 Analog Input (A/D)

Channel vector — Enter numbers between 1 and 8. This driver allows you to enter channel numbers in any order.

For example, to use the first, second and fifth channels, enter

```
[1,2,5]
```

Number the channels beginning with 1 even if the board manufacturer starts numbering the channels with 0.

Range vector — Enter a range code for each of the channels in the channel vector. The range vector must be the same length as the channel vector. This driver allows each channel to be different.

The following table is a list of the ranges for this driver and the corresponding range codes.

Input Range (V)	Range Code	Input Range (V)	Range Code
-10 to +10	-10	0 to +10	10
-5 to +5	-5	0 to +5	5

For example, if the first channel is -10 to + 10 volts and the second and fifth channels are 0 to +5 volts, enter

```
[-10,5,5]
```

Sample time — Model base sample time or a multiple of the base sample time.

VSBC-6 Digital Input

Scaling Input to Output

Hardware Input	Block Output Data Type	Scaling
TTL	double	TTL low = 0.0 TTL high = 1.0

Driver Block Parameters

Channel vector — Enter a numbers between 1 and 16 to select the number of digital input lines used. This driver allows the selection of individual digital input lines in any order.

For example, to use the first, second and fifth digital input lines, enter

```
[1,2,5]
```

Number the lines beginning with 1, even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter the base sample time or a multiple of the base sample time.

VSBC-6 Digital Output

Scaling Input to Output

Hardware Output	Block Input Data Type	Scaling
TTL	double	< 0.5 = TTL low ≥ 0.5 = TTL high

Driver Block Parameters

Channel vector — Enter a numbers between 1 and 16 to select the number of digital output lines used. This driver allows the selection of individual digital output lines in any order.

For example, to use the first, second and fifth digital output lines, enter

```
[1,2,5]
```

Number the lines beginning with 1, even if the board manufacturer starts numbering the lines with 0.

Sample time — Enter the base sample time or a multiple of the base sample time.

VSBC-6 Watch Dog

Block Parameters

Show enable port — Select this check box to show, on the driver block, the digital input that allows enabling and disabling.

Show reset port — Select this check box to show, on the driver block, the digital input that resets the computer if set to 1.

Sample time — Enter the base sample time or a multiple of the base sample time.

24

Miscellaneous I/O Blocks

Miscellaneous I/O supported by xPC Target.

xPC Target Scope Block

For information about this block, see “Adding an xPC Target Scope Block” on page 3-11 and “Entering Parameters for an xPC Target Scope Block” on page 3-15.

From xPC Target

This block behaves like a source and its output is usually connected to the input of a gauge. This block creates and controls an xPC Target scope object running on the target PC. Because only one numerical value per signal has to uploaded at a time step, the number of samples of the scope object is set to 1. The block uses the signal tracing capability of the xPC Target command-line interface and is implemented as an M-file S-function.

To xPC Target

This block behaves as a sink and usually receives its input data from a dial. The main purpose of this block is to write a new value to a specific parameter on the target application while it is running. This block is implemented as an M-file S-function. The block is optimized so that it only changes a parameter on the target application when the input value differs from the value that existed at the last time step. This block uses the parameter downloading feature of the xPC Target command-line interface.

xPC Target Software Reboot

You can use the Software Reboot block to reboot the target PC when your simulation reaches a certain state. For example, if your control system becomes unstable you may want to reboot the target PC.

This block has one input. The input is the reboot signal and accepts the following values:

- 1 — Reboots the target PC.

- 0 — If you use a Software Reboot block and you don't want the target PC to reboot, the input value to this block must be `0`.

Block Parameters	Description
Sample time	Enter a base sample time or a multiple of the base sample time.

Note Not all systems support rebooting from software. You can test whether your system supports a software reboot. From the MATLAB command window, enter `xpctest`. For more information, see "Testing the Installation" on page 2-28.

I/O Port Read

Intel 80x86 and compatible processors have a special address space reserved for I/O devices. To access this address space and communicate directly to a device, xPC Target provides the I/O Port Read and I/O Port Write blocks. These blocks enable the transfer of data from and to the I/O ports. See also, "I/O Port Write" on page 24-5.

1 Double click the I/O Port Read block.

   The **Block Parameters: I/O Port Read** dialog box opens.

2 In the **I/O-Port address** box, enter the beginning address for each value this block reads.

   For example, if you want to read a word (16 bits) starting at I/O port `0x300`, followed by a byte (8 bits) at `0x302`, enter

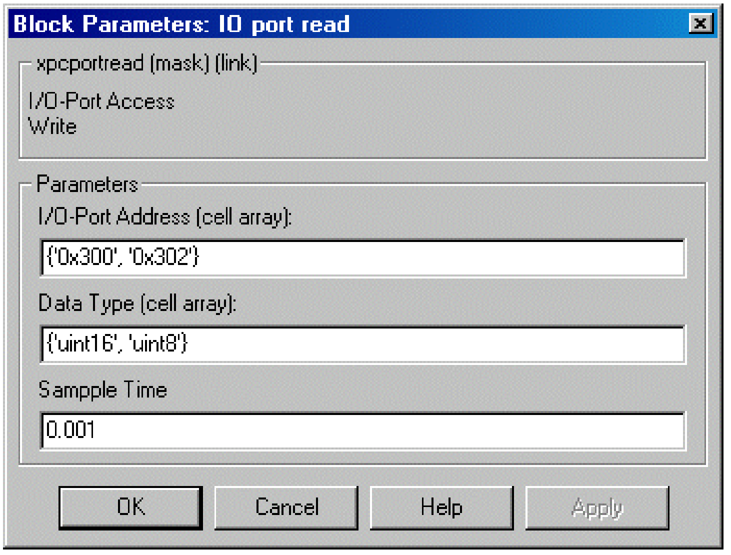
   ```
   {'0x300','0x302'}
   ```

3 In the **Data type** box, enter the type for each value this block reads. There is one type for each address you entered in the **I/O-Port address** box.

   For example if you want to read a word and then a byte, enter

```
{'uint16','uint8'}
```

Your dialog box should look similar to the figure shown below.



4 Click **OK**.

The number of outputs from the block changes to reflect the length of the I/O-Port address cell array.

Block Parameters	**Description**
I/O-Port address (cell array)	This is the cell array containing the beginning I/O port addresses for the data you want to read. These addresses are specified in terms of hexadecimal strings.
Data type (cell array)	This is the cell array containing the types of data you want to read from I/O port. The **Data type** cell array has one value for each value in the **I/O-Port address** cell array. The type `uint32` reads a double word (32 bits), `uint16` reads a word, and a `uint8` reads a byte.
Sample time	Enter a base sample time or a multiple of the base sample time.

I/O Port Write

Intel 80x86 and compatible processors have a special address space reserved for I/O devices. To access this address space and communicate directly to a device, xPC Target provides the I/O Port Read and I/O Port Write blocks. These blocks enable the transfer of data from and to the I/O ports. See also, "I/O Port Read" on page 24-3.

1 Double click the I/O Port Write block.

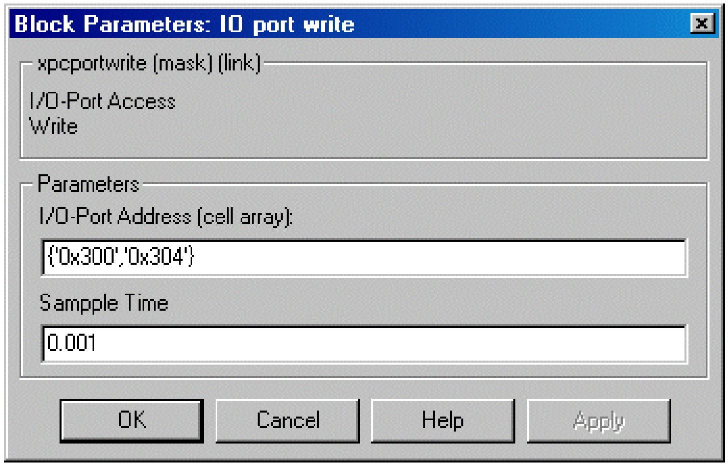
The **Block Parameters: I/O Port Write** dialog box opens.

2 Enter the parameters.

For example, if you want to write a double word (32 bits) starting at I/O port `0x300`, followed by a word (16 bits) at `0x304`, enter

```
{'0x300','0x304'}
```

Your dialog box should look similar to the figure shown below.



3 Click **OK**.

The number of inputs to the block changes to reflect the length of the I/O-Port address cell array. The data type of the input signal reflects the type of value written to the I/O port.For example, an input signal of type `uint32` writes a double word, an `uint16` input signal writes a word, and an uint8 input signal writes a byte.

Block Parameters	**Description**
I/O-Port address (cell array)	This is the cell array containing the I/O port addresses for the data that you want read. These addresses are specified in terms of hexadecimal strings.
Sample time	Enter a base sample time or a multiple of the base sample time.

xPC Target TET

This block outputs the Task Execution Time (TET) in seconds.

Use the output of this block as an input to an xPC Target Scope block. This allows you to visualize the TET while your target application is running.

xPC Target Time

This block outputs the time in clock ticks that the kernel has been executing. This is not the same as the execution time of the target application, but it is the time since you booted up the target PC.

A use for this block is to determine the execution time of subsystems in your Simulink model. Add an xPC Target Time block before and after the subsystem, and then calculate the difference in time.

The time is in clock ticks, and it uses the target PC interval timer to generate the ticks. Since the interval timer runs at a frequency of `1.193` Mhz, you can calculate the time in seconds by dividing 1.193×10^6

Asynchronous Event Support

xPC Target incudes support for asynchronous events. These events are triggered by a hardware interrupt asynchronously to normal execution. Some I/O boards raise interrupts that the CPU can use to interrupt the normal execution of code and jump to another sections of code called an Interrupt Service routine (ISR).

This section includes the following topics:

- **Adding an Asynchronous Event** — Add an Async IRQ Source block to your Simulink model
- **Async IRQ Source Block** — Reference for block parameters
- **Async Rate Transition Block** — Reference for block parameters
- **Async Buffer Write and Read Blocks** — Reference for block parameters
- **Asynchronous Interrupt Examples** — Examples for data transfer with a Transition block, buffered data transfer with Real/Write blocks, and interrupt CAN communication with PCI and PC/104 boards

Adding an Asynchronous Event

When developing a model in Simulink that runs in the xPC Target environment, an Interrupt Server Routine (ISR) is modeled by using a Function-Call Subsystem. Additionally, you need to add an IRQ Source block connected to the Function-Call Subsystem block. This subsystem is then executed when an interrupt occurs and the CPU is ready to accept it.

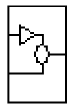
After you install an I/O board with interrupt support into your target PC, you can add xPC Target asynchronous blocks to your Simulink model.

1 In the MATLAB command window, type

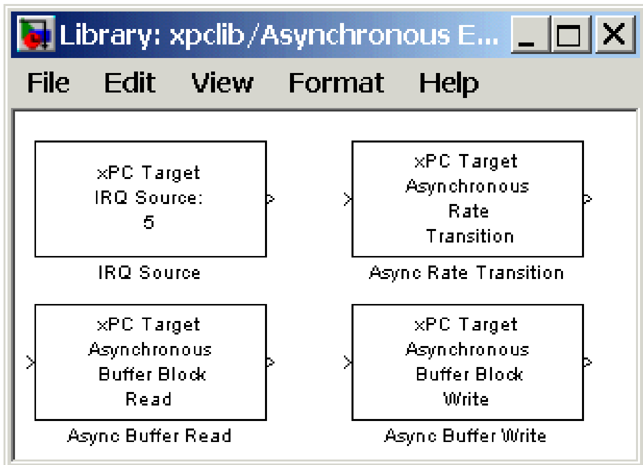
```
xpclib
```

The xPC Target Library opens.

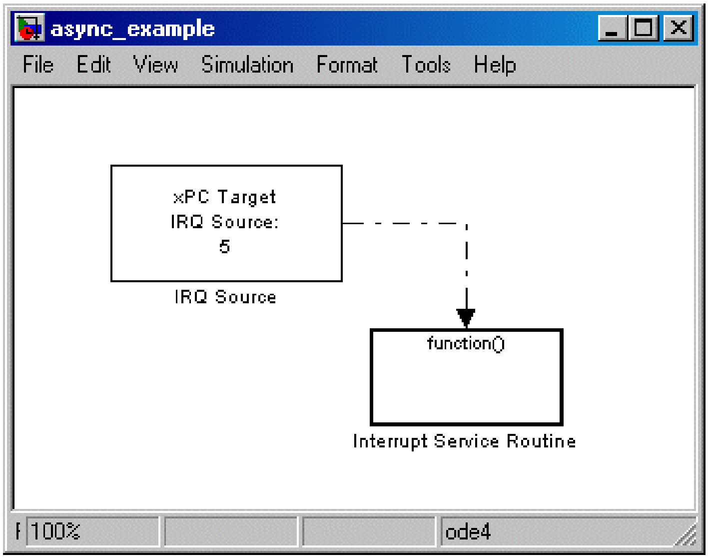
2 Double-click the Asynchronous Event group block.



The Library: xpclib/Asynchronous Event window opens.



3 Drag-and-drop the xPC Target IRQ block in your Simulink model and connect the output to this block to the input of a Function-Call Subsystem. For more information on Function-Call subsystems, see *Using Simulink* and the *Real-Time Workshop User's Guide*.



In the setup shown above, the CPU executes the contents of the Function Call-Subsystem whenever IRQ 5 occurs.

4 Double-click the IRQ Source block.

   The **Block Parameters: IRQ Source** dialog box opens.

5 From the IRQ line number list, choose **1**, **2**, . . . , **15**. To determine the available IRQ line numbers on the target PC, use the function `getxpcpci`.

6 Select or deselect the Allow preemption of function call subsystem check box.

7 From the I/O board generating the interrupt list, select **Computer Boards CIO-CTR05**, **Softing CAN-AC2-104**, or **CAN-AC2-PCI**. Currently, these are the only interrupt boards supported by xPC Target.

8 In the PCI slot box, enter the PCI slot number or enter `-1` to let xPC Target determine the number.

9 Click **OK**.

For more information about the IRQ Source block, see “Async IRQ Source Block” on page 24-11.

If you need to transfer data from your ISR, add an Async Transition Block or Async Read/Write blocks to your Simulink model. See “Async Rate Transition Block” on page 24-13, “Async Buffer Write and Read Blocks” on page 24-13, and “Asynchronous Interrupt Examples” on page 24-14.

If you are using a CAN fieldbus with interrupts, see “Asynchronous Interrupt Examples” on page 24-14.

Async IRQ Source Block

The main block that notifies Simulink and xPC Target that a particular Function-Call Subsystem should be treated as an ISR is the IRQ Source block. This block is actually a virtual block and does not exist at model execution time. However, the model initialization code sets things up with the CPU to execute the ISR when the proper interrupt occurs.

Block Parameters	**Description**
IRQ line number	Select the IRQ line number you are using for this block. This depends on the characteristics of your hardware. You may need to query the PCI bus on the target PC to find what IRQ the PCI bus assigned to your hardware. Use the function `getxpcpci`. Valid IRQ numbers are between 5 and 15.
Allow preemption of function call subsystem	Normally, while the ISR is executing, another interrupt will not cause re-execution of the ISR (That is the ISR will not *preempt* itself). If this check box is checked, the ISR will interrupt itself.

Block Parameters	Description
I/O board generating the interrupt	For many I/O boards, it is necessary to set up the board properly to generate the interrupt. It may also be necessary to set up board specific features at the beginning and/or end of an ISR. you should select the correct board that you intend to use from the drop-down list.
	Currently xPC Target supports the following boards from: • Measurement Computing Corporation (formerly, Computer Boards) — Counter boards CIO-CTR05 (ISA bus) and PCI-CTR05(PCI bus) • Softing AG — CAN boards CAN-AC2-PCI (PCI bus) and CAN-AC2-104 (PC/104 bus) Check the xPC Target Product News Page for library updates at `http://www.mathworks.com/support/product/XP/productnews/productnews.shtml`.
PCI slot (-1: autodetect)	Enter a number between `-1` and *n*. If only one board of this type is physically present in the target PC, enter `-1`. If two or more boards of this type are physically present in the target PC, enter the PCI slot number of the board associated with this driver block. If you do not specify the board you want to use for this interrupt, xPC Target uses the first card it finds of this type.

Async Rate Transition Block

Use the Asynchronous Rate Transition block to double buffer data between the function call subsystem and the rest of the model which executes rate-monotonically in real-time.

Normally, the interrupt service routine writes to the first buffer. When the next model step executes, the first buffer is copied to the second buffer and its value is used for model calculations.

If a second interrupt occurs while the buffer is being copied, data is corrupted. This corruption happens when part of the data is copied from the first buffer, the interrupt occurs and writes over the entire first buffer, and then the transition block continues the copy operation from the first buffer that now has data from the second interrupt.

To prevent possible data corruption, use Async Buffer Write and Read blocks. See "Async Buffer Write and Read Blocks" on page 24-13d.

Block Parameters	Description
Sample time	Enter a base sample time or a multiple of the base sample time.

Async Buffer Write and Read Blocks

These blocks provide double buffering of data between the ISR and the model which executes rate-monotonically in real-time. Always use these blocks in pairs with an Async Buffer Write Block leading into an Async Buffer Read block. The Async Buffer Write Block has to be part of the ISR, and the Async Buffer Read block is outside the ISR.

Use Async Buffer Write and Read blocks for a more secure method of transferring data. Unlike the rate transition block, the data from one buffer is not copied to the second buffer. Instead, interrupts are disabled, and pointers to the buffers are swapped. This method produces the smallest time the interrupts are disabled and protects against data corruption caused by overwriting partially copied buffers.

Block Parameters	Description
Sample time	Enter a base sample time or a multiple of the base sample time.

Asynchronous Interrupt Examples

xPC Target provides several example models. If you installed MATLAB in the default location, these models are located in

```
C:\MATLAB6p1\toolbox\rtw\targets\xpc\xpcdemos.
```

To access any of these models, in the MATLAB command window, type the name of the model. Each model contain annotations documenting its purpose, and should serve as an example of how to use these blocks.

- xpcasyncbuffer — Model using an external TTL signal to trigger and interrupt on the PCI-CTR05 board. Data exchange between an asynchronous task and a monotonic task using **Async Buffer Read/Write** blocks.
- xpcasynctrans — Model using an external TTL signal to trigger and interrupt. on the PCI-CTR05 board. Data exchange between and asynchronous task and a rate monotonic task using a **Async Rate Transition** block.
- xpccanintpc104 — Model using interrupt driven CAN I/O communication with the CAN-AC2-104 board.
- xpccanintpci — Model using interrupt driven CAN I/O communication with the CAN-AC2-PCI board.