Getting Started with the MapleSim Connector for dSPACE DS1104

Getting Started with the MapleSim Connector for dSPACE DS1104

This document was produced using a special version of Maple and DocBook.

Printed in Canada

Contents

Introduction

The MapleSim™ Connector for dSPACE™ software provides all of the tools you need to prepare and export your dynamic systems models to the dSPACE™ DS1104 R&D Controller Board. You can create a model in MapleSim, simplify it in Maple™ by using an extensive range of analytical tools, and then generate from a subsystem an executable that you can run on the dSPACE DS1104 board.

Features include:

- Maple template, which provide an intuitive user interface for defining the mapping between the inputs and outputs of your subsystem and the I/Os of the DS1104 board, and then generate an executable that you can run on this dSPACE board.
- A range of examples illustrating how to prepare and export your models.

Scope of Model Support

MapleSim is a comprehensive modeling tool where it is possible to create models that could go beyond the scope of this MapleSim Connector for dSPACE DS1104 release. In general, the MapleSim Connector for dSPACE™ DS1104 supports systems of any complexity, including systems of DAEs of any index, in any mix of domains, as long as they exhibit continuous behavior. Systems that contain any type of discontinuity, including discrete transforms, switches, logic gates, relational and Boolean operations are not supported by the current release of this product.

Apart from all of the engineering and signal components that are continuous, this product also supports lookup tables, and custom components that do not use discontinuous operations such as piecewise functions.

System Requirements

For installation instructions and a complete list of system requirements, see the **Install.html** file on the product disc.

1 Getting Started

1.1 Getting Help

In Maple, enter **?dSPACEConnector** at a prompt in a worksheet.

1.2 Using the dSPACE Application Generation Template

The MapleSim Connector for dSPACE provides a **dSPACE Application Generation** template in the form of a Maple worksheet for manipulating and exporting MapleSim subsystems. This template contains pre-built embedded components that allow you to generate dSPACE applications from a MapleSim subsystem and save the source code.

Using this template, you can define inputs and outputs for your system and how these inputs and outputs are connected with the dSPACE inputs and outputs. You can also generate the source code and create an executable.

Viewing Examples

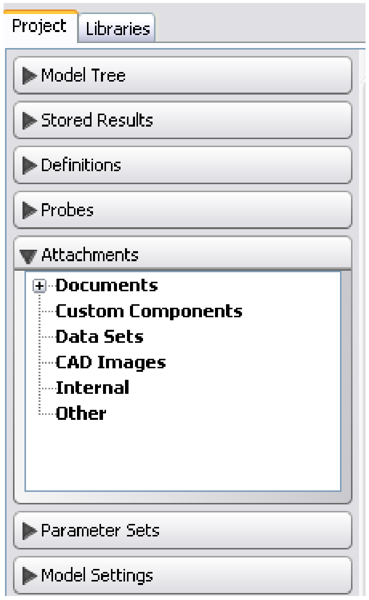
Examples are available in the **dSPACE Connector Examples** palette in MapleSim.

Each example includes a code generation template in its **Attachments** palette.

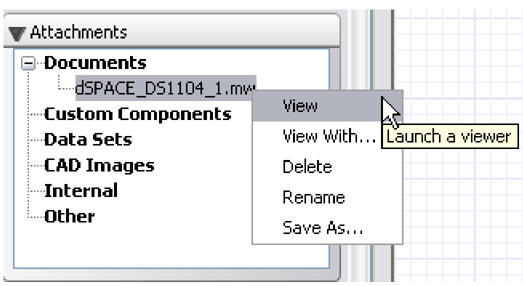
To view an example:

1. In the **Libraries** tab on the left side of the MapleSim window, expand the **dSPACE Connector Examples** palette, and click the entry for the model that you want to view.

2. In the **Project** tab, expand the **Attachments** palette and then expand **Documents**.



3. From the list, right-click the **dSPACE_DS1104** template and select **View**. The code generation template is opened in Maple.



Some models include additional documents, such as templates that display model equations or define custom components. You can open any of these documents by right-clicking its entry in the list and clicking **View**.

1.3 Example: VCFP Model

This example is based on the VCFP (voice-coil-driven flexible positioner) system, a standard demo plant you can purchase from dSPACE. In this example, you will generate a dSPACE executable from the controller subsystem of the closed loop VCFP model that was created in MapleSim. The dSPACE executable that is generated can then be used in ControlDesk.

This example is a ready-to-run application with the following prerequisites

- You have setup the dSPACE software and a DS1104 board on your computer.
- The real dSPACE demo VCFP plant is physically connected to the DS1104 board.

The following steps explain how to configure the existing **dSPACE_DS1104** template to match your configuration.

To generate a dSPACE Application:

1. From the **dSPACE Connector Examples** palette, open the **VCFP_GettingStarted** example.
2. Select the **Project** tab, expand the **Attachments** palette, expand the **Documents** section and double-click on **dSPACE_DS1104_1.** Your MapleSim model is opened in Maple, in the template that you selected.
3. Browse to the **Controller1** subsystem by selecting the subsystem name from the drop-down menu in the toolbar above the model diagram. This menu displays all of the subsystems and components in your MapleSim model.
4. In the **Model Input from MapleSim** section of the template, click **Retrieve System.** All of the template fields are populated with information specific to the subsystem displayed in the model diagram.

You can now check if the mapping between the inputs and outputs of the subsystem and the dSPACE DS11014 inputs and outputs match the real connections on your system

5. Navigate to the **dSPACE DS 1104 Model Generation** section of the template, then the **Input / Output Settings for the dSPACE DS1104 Board** subsection. Select the **Main.Controller1.Reference(t)** signal in the list of model inputs. In this example, this signal is replaced in the application by a square signal generator running on the dSPACE board, as shown in the **DS1104 inputs or Virtual signals** list box at the right.
6. You can modify the configuration of this square signal generator by navigating to, and expanding, the **Virtual Square Signals Generator** section. In the Square 1 part of the table, you can modify the **Amplitude**, the **Frequency** and the **Offset** values. If you want to change the input mapping of this signal, select another virtual signal or a DS1104 channel. For example, if you want to connect it to an ADC channel, such as **ADCH1** (this signal will read the value of the first ADC channel of the DS1104 board, physically connected on the reference of the real plant), select the ADC from the list. There is no configuration possibility for this channel; however, you can review its characteristics by expanding the **Mux ADC Unit** section below the table.
7. In the **Input / Output Settings for the dSPACE DS1104 Board** section, select the **Main.Controller1.Measurement(t)** signal in the list of model inputs. In this example, this signal is connected to **ADCH2**, as shown in the **DS1104 inputs or Virtual signals** list box on the right. It will read the value of the second ADC channel of the DS1104, physically connected to the measurement of the real plant. There is no configuration possibility for this channel but you can review its characteristics by expanding the **Mux ADC Unit** section below the table. You can change the DS1104 channel in order to match the configuration of your system.
8. In the **Input / Output Settings for the dSPACE DS1104 Board** section, select the **Main.Controller1.Command(t)** signal in the list of model outputs. In the **DS1104 outputs** list box on the right, select **DACH1**: this signal will send the value computed by the application to the first DAC channel of the DS1104, physically connected to the command of the real plant.
9. To configure this output, expand the **DAC Unit** section. Define the DAC mode as **transparent**, initialize the DACH1 to **0** and specify a termination value of **0**.

Note: You could also hold the current value when the application terminates. There is no problem in this particular application. However, in general when termination values are different from 0, it is very often unsecure. You can also change the DS1104 channel in order to match the configuration of your system.

10. In the **Solver Setting** section, you can change the base sampling rate of the application.
11. In the **Advanced Code Generation Settings** section, set the **Code Optimization** option to **Full** by moving the slider to the 3 position . This option specifies the degree of simplification applied to the model equations during the code generation process. This option eliminates redundant variables and equations in the system.
12. In the **Advanced Code Generation Settings** section, open the **TRC Variables and Parameters Option** section. These options allow you to optionally add all the possible parameters and/or variables in the TRC file that is generated with the dSPACE application. The TRC file is used by ControlDesk so that you can display variables and modify parameters on the running real-time application.
13. In the **Generate the dSPACE Application** section of the template, specify the dSPACE directory and the directory where the application should be generated..

14. Click **Generate to dSPACE Application** to generate the C code, the .trc file, the .sdf file and the .ppc file of the application.
15. Open **ControlDesk** ® and load the generated application as usual. You will have access to the inputs, outputs, states and states derivatives of the application in the **Model** group. The parameters are in the **Model Parameters** subgroup, the execution time and current time can be accessed from the **Task Info -> Execution** subgroup.

2 Tutorial: Exporting a Subsystem as a dSPACE Application

This tutorial will guide you through the process of creating a dSPACE model using a dSPACE Connector DS1104 template, from a pure closed-loop simulation model.

It is a *rapid control prototyping* application, where the controller runs in real-time on the DS1104 board in order to control a real system.

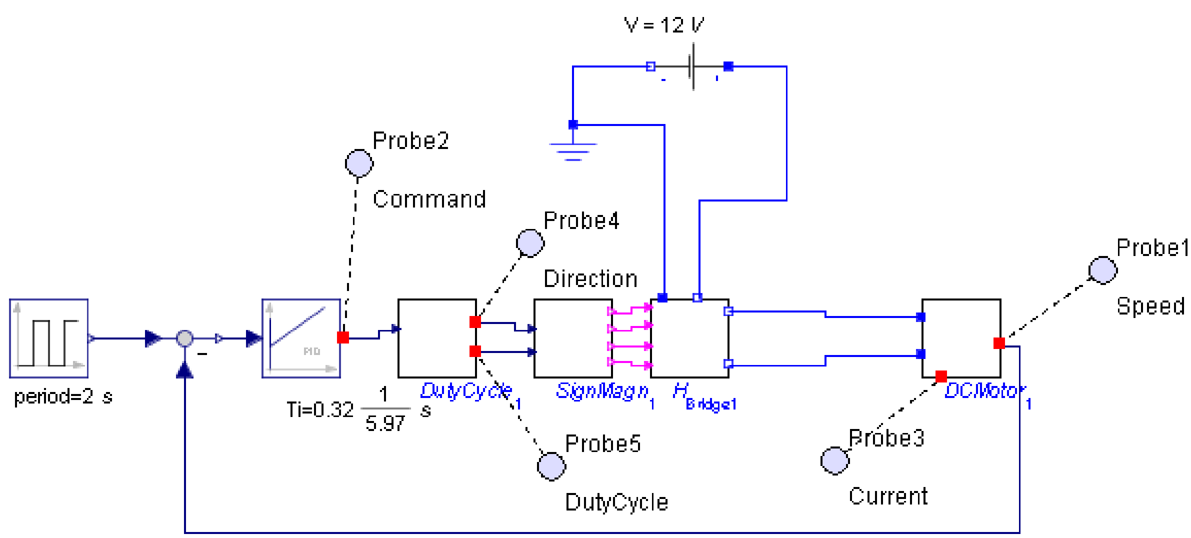
2.1 Overview of the Plant to Control

The goal is to control the speed of a DC Motor. The system is composed of the following real components:

- The DC Motor itself, which includes a gear (ratio=30)
- The H-bridge driver, allowing the bi-directional control of the motor. This component also provides a current measurement output with a 377 µA/A sensitivity, connected in series with a 2 kΩ resistor
- A TTL incremental encoder with 300 lines. This encoder is attached to the motor shaft (the rotor), not to the gear shaft. Thus, there are 9000 lines (300*30) per revolution

2.2 Overview of the Initial Closed-loop Simulation Model

The initial closed-loop simulation model (**DCMotorControl_startTutorial.msim**) is located in the examples directory of the dSPACE Connector installation (**<Maple 15>/toolbox/dSPACEConnector/data/examples**, where <**Maple 15**> is your Maple 15 installation directory).



Open this model in MapleSim 5. The main components are (from right to left):

- The **DC Motor model**. You can explore this model to see the different parameters of the motor.
- A **H-bridge subsystem**, based on ideal switches and diodes components. The H-bridge, supplied with a 12V voltage source, provides the required power to the motor. Each switch is driven by a on/off signal output by the SignMagn subsystem.

- The **SignMagn subsystem** computes the switching logic in order to drive the H-bridge in the "Sign/Magnitude mode".
- The **DutyCycle subsystem** transforms the voltage command (output of the PID controller) into a direction and duty cycle signal, required by the Sign/Magnitude drive mode.
- A **PID controller** which maintains the real motor speed as close as possible to the reference.
- A **pulse signal** which gives the square speed reference.

Run this model and look at the result.

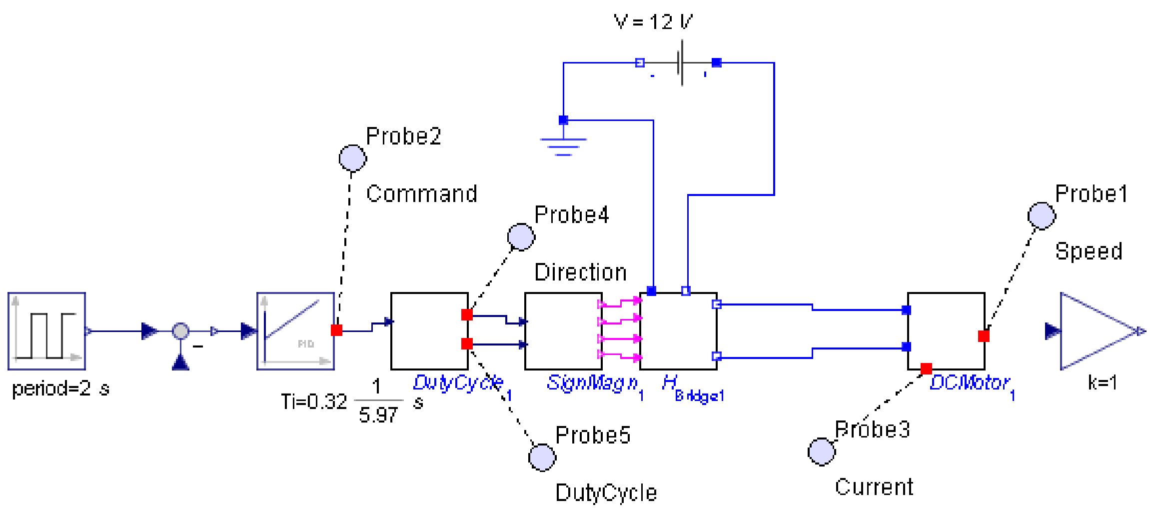
2.3 Preparing the Model for Export

The model you opened in the previous section is not ready to export, mainly because it does not precisely reflect the real sensors and their interaction with the input and output components of the dSPACE DS1104 board. Next, you will add modeling components for the incremental encoder and the current sensor. You will also group into a subsystem all the components needed in the Controller application that will run in real-time on the DS1104 board.

Adding the Incremental Encoder Model

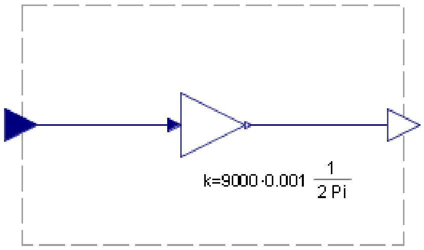
The output of the incremental encoder are pulses corresponding to the angular position of the rotor. In the real-time application, you will connect this sensor to the incremental encoder input of the DS1104 board. The encoder input driver of the DS1104 gives the angular position and the position difference between two time steps. Use this delta position to compute the speed.

1. Disconnect the line between the DCMotor speed output and the minus input of the comparator block.
2. Add a gain block (from the **Signal Blocks -> Common** palette) to the right of the DCMotor subsystem.

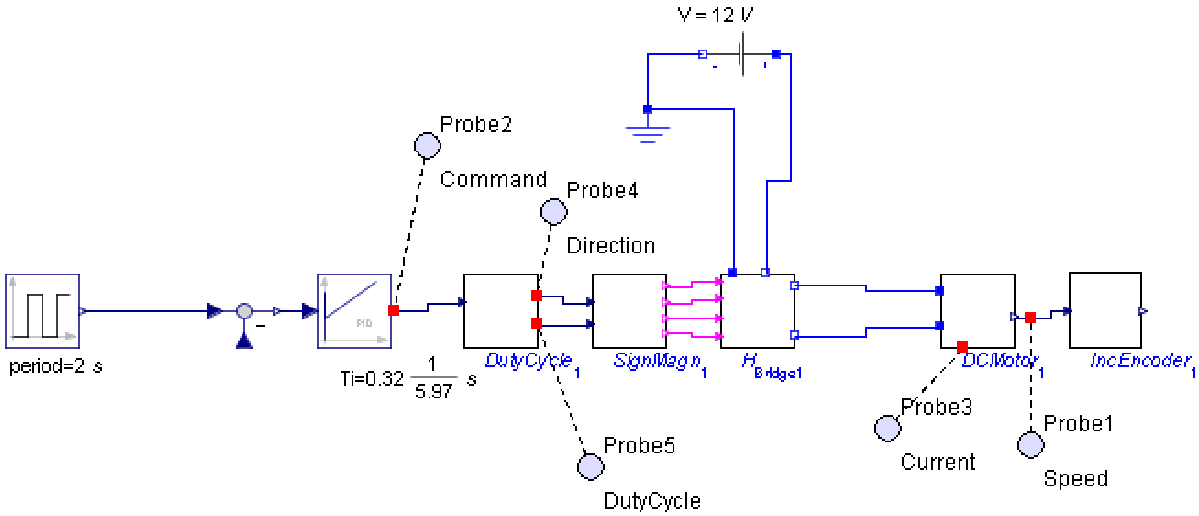


3. Transform this block into a subsystem (select the block, then select **Edit -> Create Subsystem** from the MapleSim menus). Name the subsystem "IncEncoder".
4. Double-click on this subsystem to open it.
5. Connect the input and output of the gain block to the subsystem boundary and name the input,"Speed" and name the output,"DeltaPos".

6. Modify the gain value: since there are 9000 lines per revolution and that the sample time of the application will be 1 ms, multiply the real speed by $\frac{9000 \cdot 0.001}{2 \cdot \pi}$ in order to get the delta position value



7. Connect the DCMotor speed output to the input of the "IncEncoder" subsystem

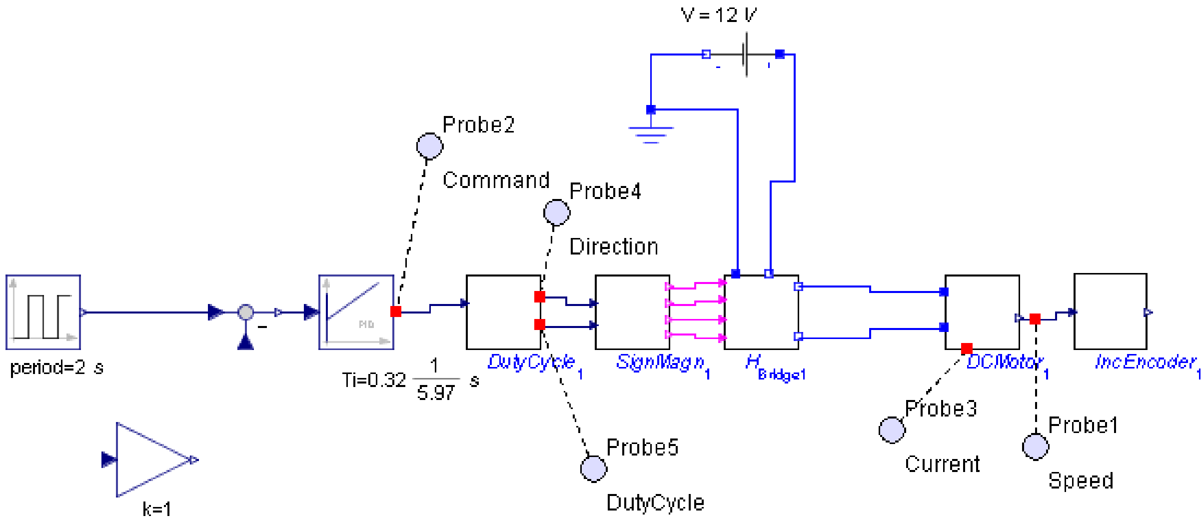


Processing the Incremental Encoder Capture Signal in the Real-time Application

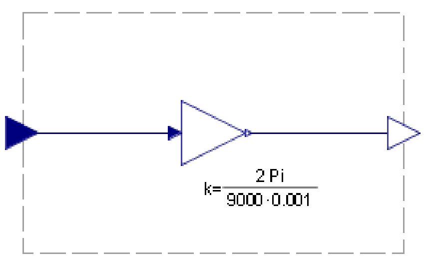
In the real-time application, you will need to transform the incremental encoder capture signal of the DS1104 board (delta position) into a quantity corresponding to the motor speed. For this you just need to add another gain block with a value which is the inverse of the one you set up in the previous section.

Follow steps 2-7 from the *Adding the Incremental Encoder Model (page 6)* section, with the following differences:

- Add the gain block near the minus input of the comparator block (at the left of the PID subsystem)

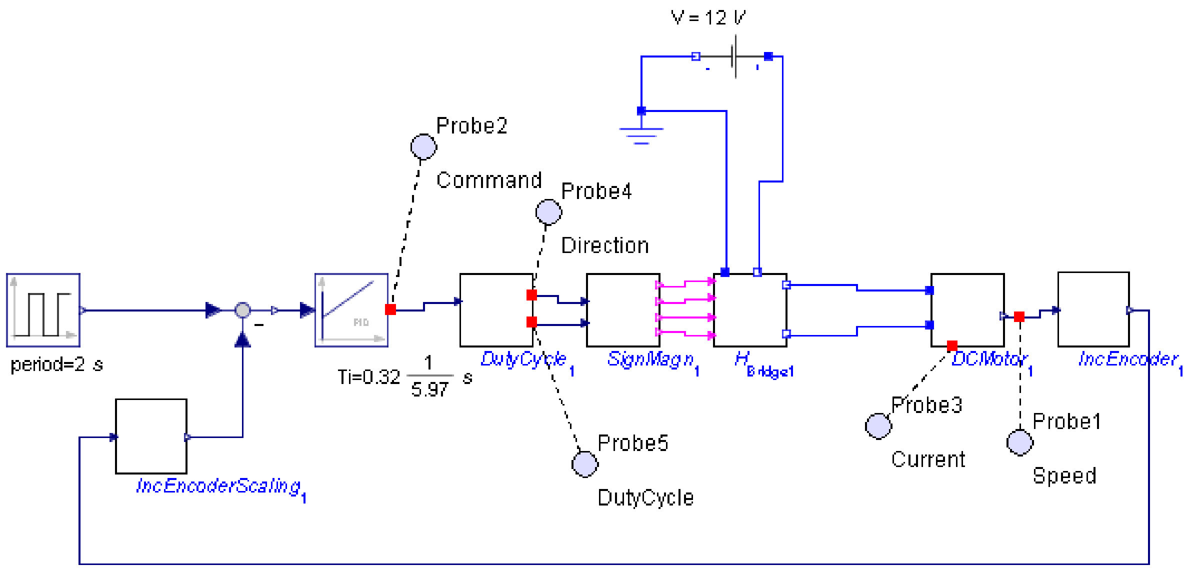


- Create a subsystem with the name "IncEncoderScaling"
- The input and output of this subsystem will be respectively "DeltaPos" and "Speed"
- The gain value is $\dfrac{2 \cdot \pi}{9000 \cdot 0.001}$



Finally, complete the connections:

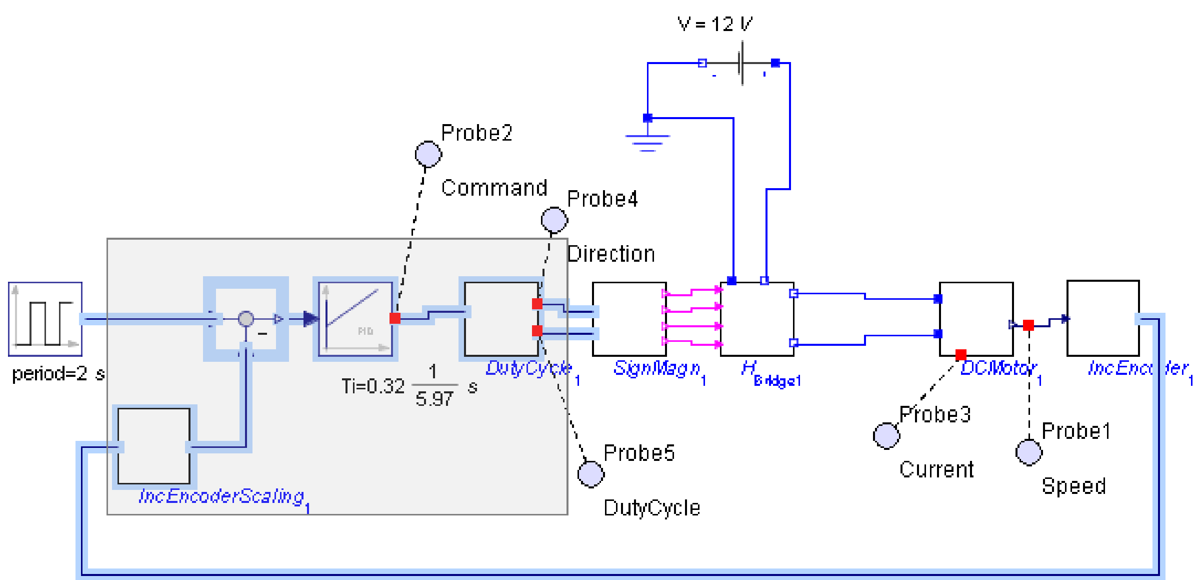
1. Connect the output of the "IncEncoder" subsystem to the input of the "IncEncoderScaling" subsystem
2. Connect the output of the "IncEncoderScaling" subsystem to the minus input of the comparator block



Converting the Controller to a Subsystem

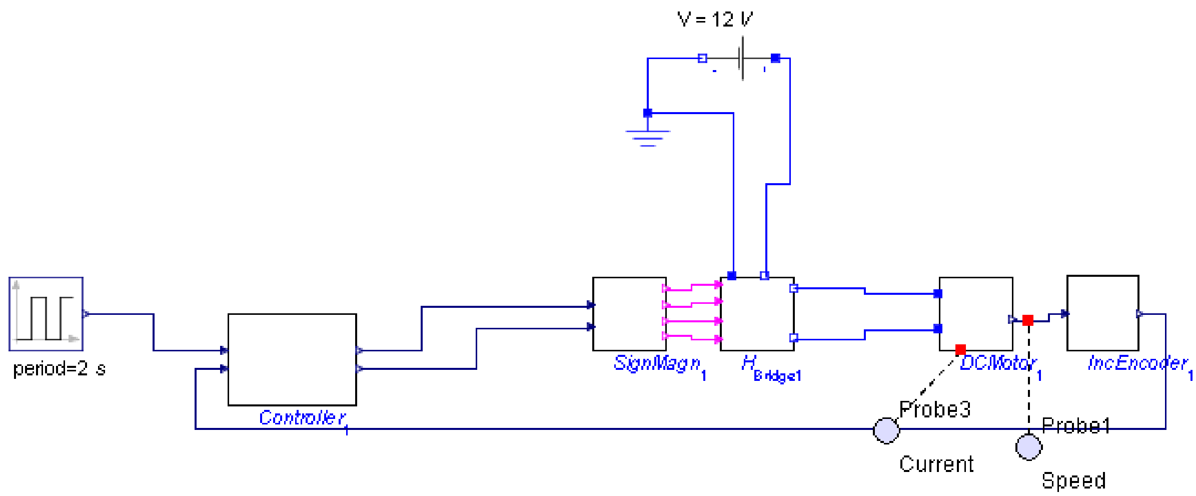
By converting your entire model or part of your model into a subsystem, you identify which parts of the model you want to export to a dSPACE application. In this example, you will group the controller components into a subsystem.

1. Using the selection tool ( ) located above the model workspace, draw a box around the controller components in the model.

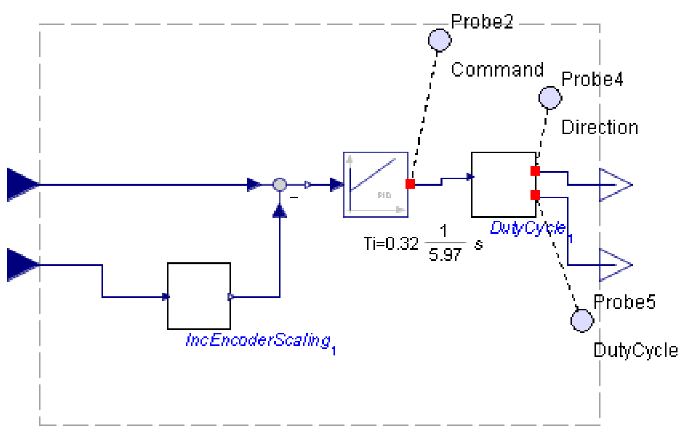


2. From the **Edit** menu, select **Create Subsystem**.
3. In the **Create Subsystem** dialog box, enter **Controller** as the subsystem name.

4. Click **OK**. A **Controller** subsystem block is displayed in the model workspace.



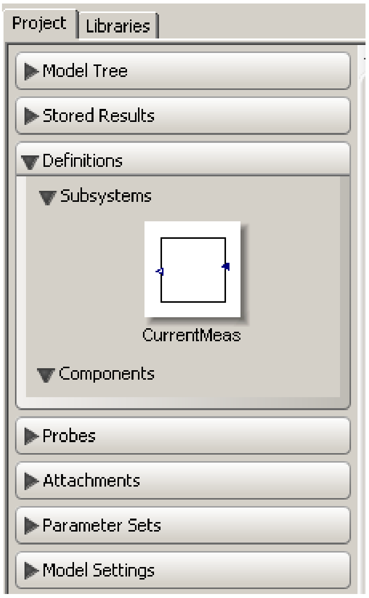
5. Open this subsystem and modify the names of the input and outputs as below:
   First input (connected to the comparator): "SpeedReference"
   Second input (connected to the IncEncoderScaling subsystem): "DeltaPos"
   First output (first output of the DutyCycle subsystem): "Direction"
   Second output (second output of the DutyCycle subsystem): "DutyCycle"



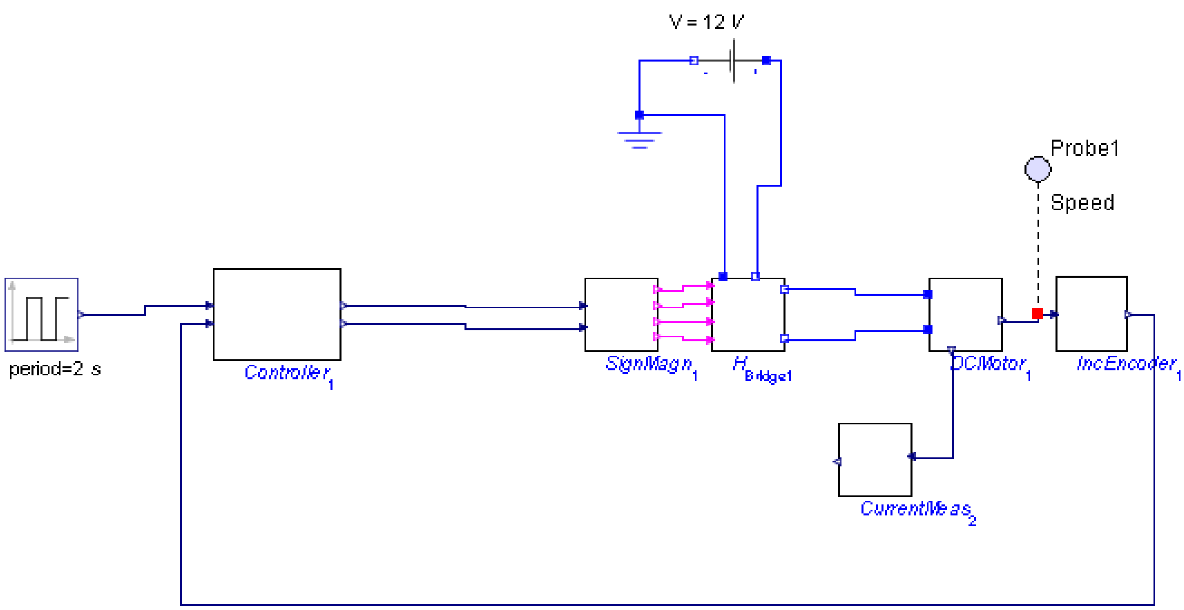
Adding the current sensor model

For the real-time application, connect the current sensor output to an analog input of the DS1104 board in order to monitor the current in the drive.

1. To save time, the current sensor model is already included in this model. You will find it in **Project ->Definitions ->Subsystems**, with the name "CurrentMeas".



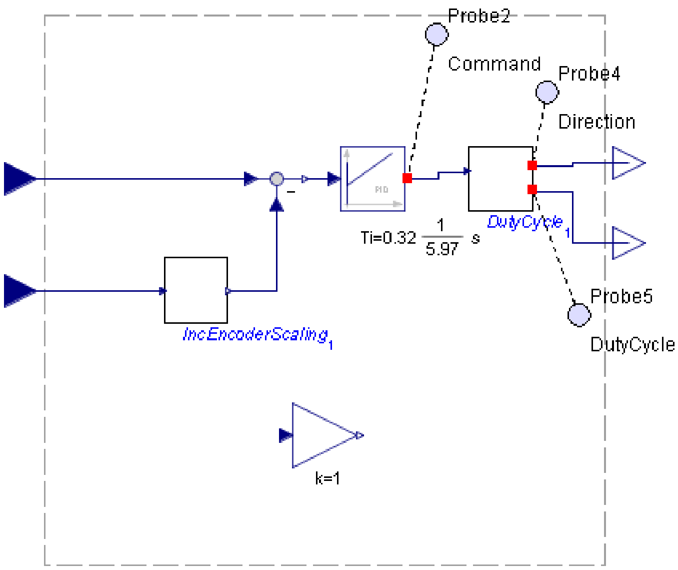
2. Remove the current probe in the model and put it below the DCMotor subsystem
3. Connect the DCMotor current output to the input of the "CurrentMeas" subsystem



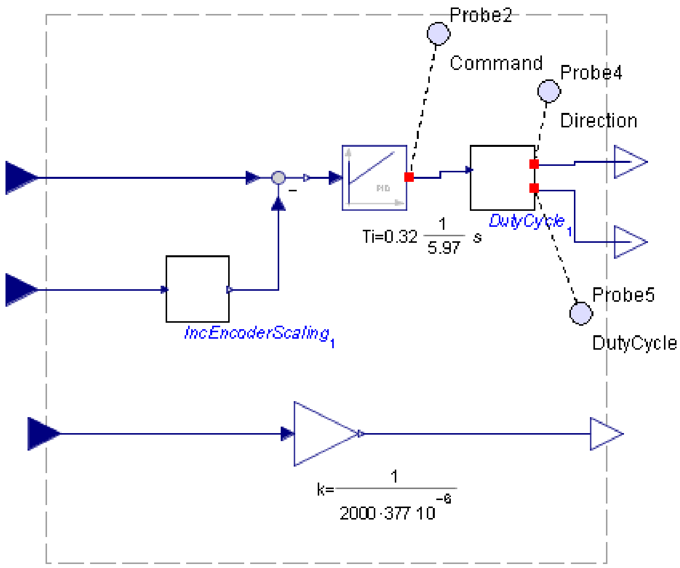
Processing the current sensor ADC signal in the real-time application

In the real-time application, connect the output of the current sensor to an ADC input of the DS1104 board and transform the signal into a quantity corresponding to the motor current. For this you just need to add a scaling gain block in the controller subsystem:

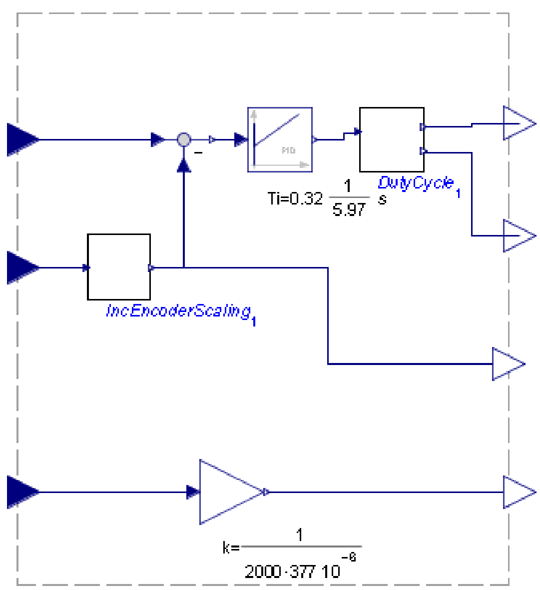
1. Open the Controller subsystem
2. Put a gain block (from the **Signal Blocks->Common** palette) below the other blocks of this subsystem.



3. Connect the input of this gain block to the left border of the subsystem and give the name "Imeas" to the input
4. The current value will not be used in the controller (you just want to monitor it), so you will not connect it to any block of the controller. However, in order to generate code you cannot leave the output of the gain block unconnected. Thus you need to connect the output of this block to the right border of the subsystem and give the name "Current" to the output
5. Modify the gain value: since the sensitivity is 377 µA/A and since you get value through the voltage measurement on a 2 kΩ resistor, the scaling value is $\frac{1}{2000 \cdot 377 \cdot 10^{-6}}$

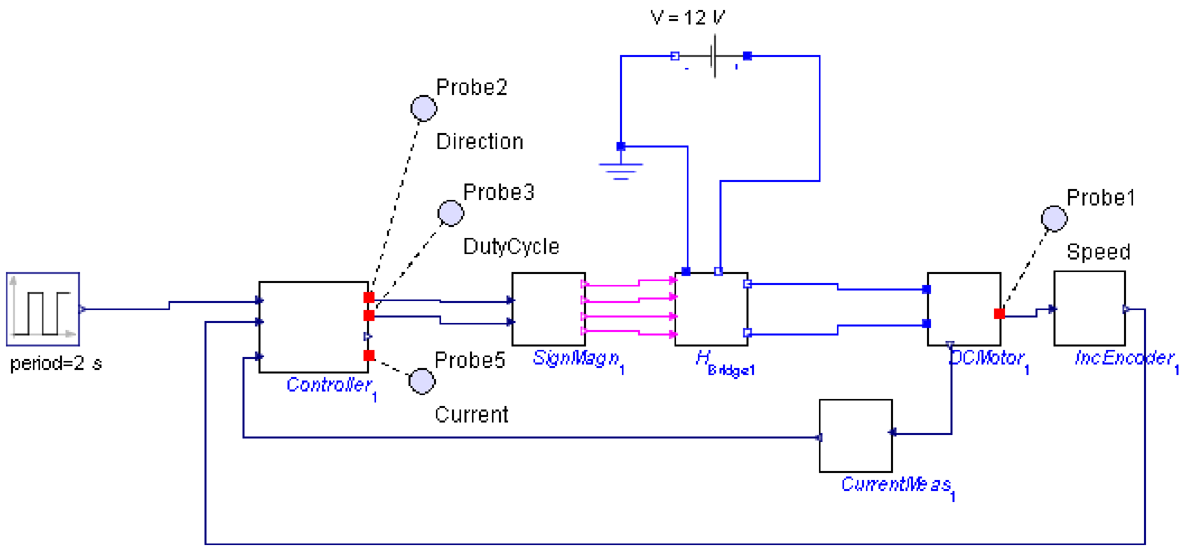


6. You will need also to monitor the motor speed in the real-time application. Thus you need to connect the output of the IncEncoderScaling subsystem to the right border of the subsystem and give the name "Speed" to the output
7. Delete the three probes. They will be added to Controller subsystem outputs in step 9.



8. Go up one level to the Main model and connect the third input of the "Controller" subsystem to the output of the "CurrentMeas" subsystem.
9. Add a probe to the first and second output of the "Controller" subsystem.

10. You can leave the third and fourth output of the "Controller" subsystem unconnected or add a probe to display the values during a MapleSim simulation



This model should now be the same as the one you can find in the **dSPACE Connector Examples** palette.

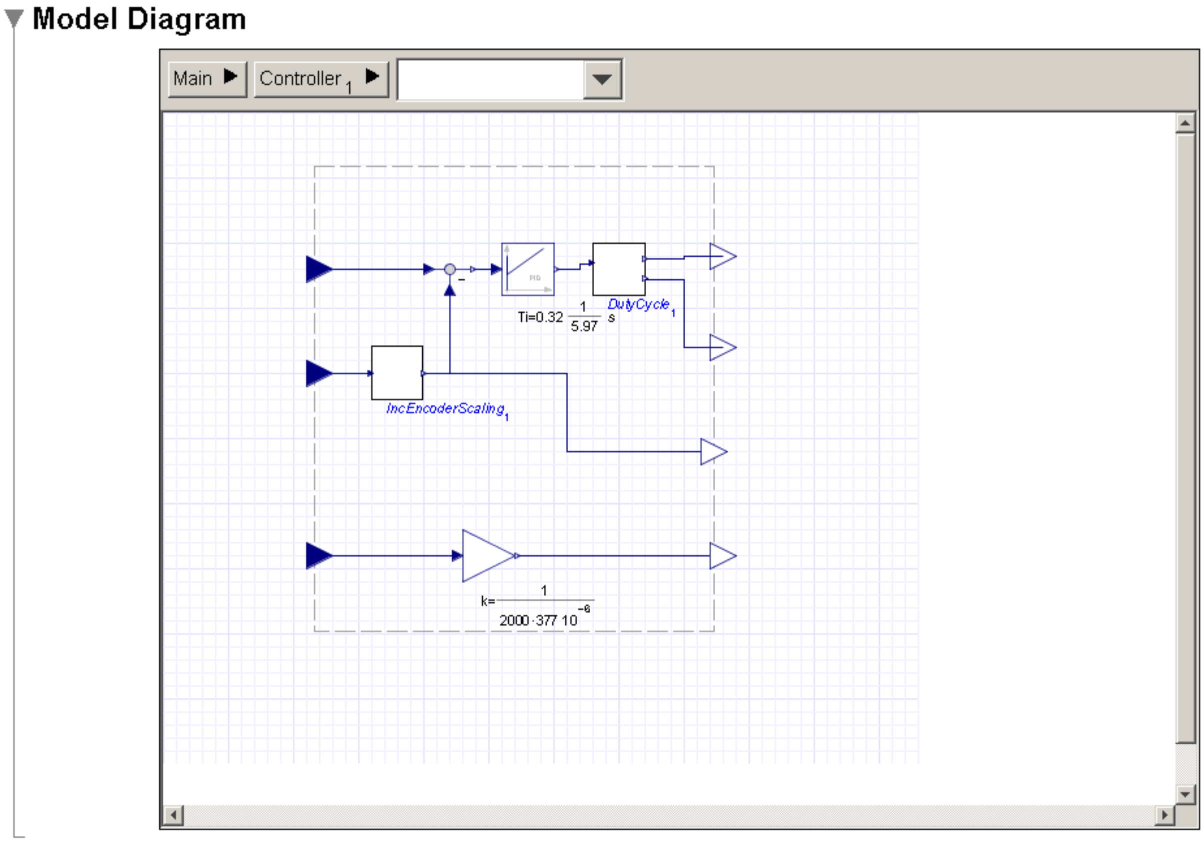
The "Controller" subsystem is now ready to be prepared to export. Note that you can still simulate this model in MapleSim.

2.4 Map the Controller subsystem inputs and outputs to the dSPACE DS1104 inputs and outputs

You will now define the mapping of the Controller subsystem inputs and outputs to the DS1104 hardware input and outputs, reflecting the real connection between the DS1104 and the DC Motor. This will be done using the Maple template, dSPACE_DS1104, which will then be attached to the model.

1. Click the templates button () in the main toolbar
2. From the list, select **dSPACE_DS1104.**
3. In the **Attachment** field, enter **dSPACE_DS1104_Controller** as the worksheet name.
4. Click **Create Attachment**. Your MapleSim model is opened in Maple, in the **DS 1104 Code Generation** template.

5. Browse to the **Controller1** subsystem by selecting the subsystem name from the drop-down menu in the toolbar above the model diagram.



6. In the **Model Input from MapleSim** section of the template, click **Retrieve System.** All of the template fields are populated with information specific to the subsystem displayed in the model diagram

You can now define the mapping between the inputs and outputs of the subsystem and the dSPACE DS11014 inputs and outputs, according to the following details:

- The speed reference could be connected to a external signal generator signal. However, it can be interesting to run a virtual signal generator in real-time on the dSPACE board in order to tune the amplitude, frequency and/or offset directly from the dSPACE ControlDesk software. Thus, you will connect the speed reference input of the controller to a virtual square wave signal. The other choices for the virtual signals are Sine, Step or Constant
- The incremental encoder of the DC Motor will be connected to the first incremental encoder input of the DS1104 board.
- Next, current sensor of the H-Bridge will be connected to the first ADC input of the DS1104 board.
- The direction input of the H-Bridge will be connected to the first digital output of the DS1104 board.
- The duty cycle input of the H-Bridge will be connected to the first PWM output of the DS1104 board.
- The current sensor signal monitored by the controller (third output of the controller) will be left unconnected.

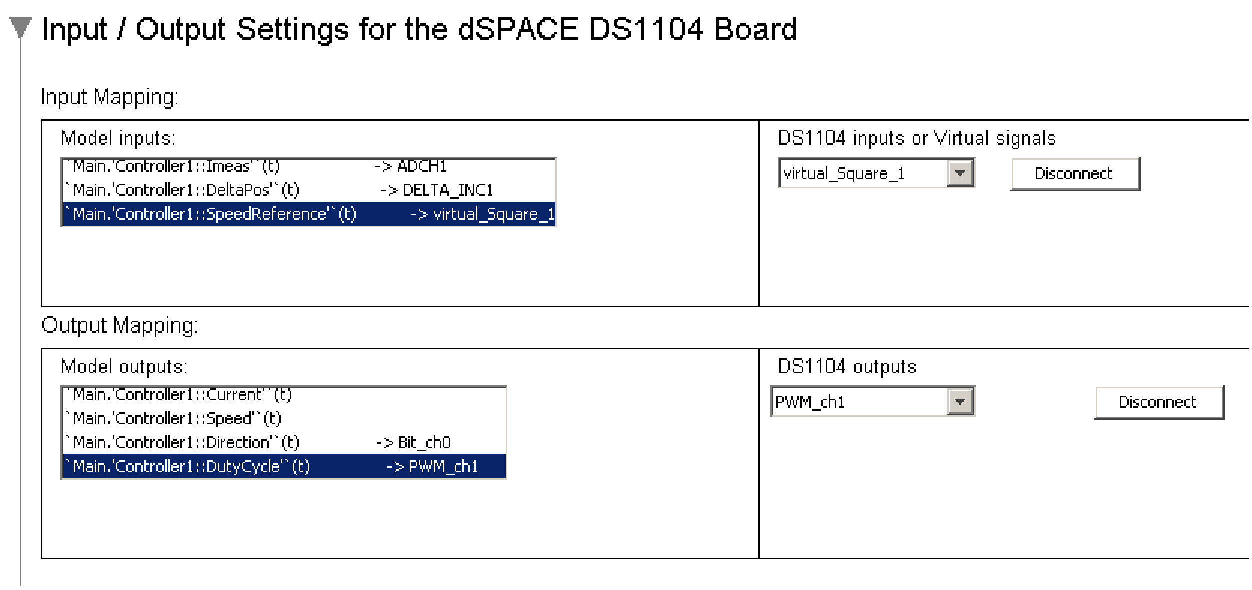
You will proceed in two steps. First you will define of all the mapping signals, and then you will configure the input and output parameters.

To define all of the mapping signals, follow steps 7-12:

7. In the **Input / Output Settings for the dSPACE DS1104 Board** section, **Input Mapping** table, select the **Main.Controller1.SpeedReference(t)** signal in the list of model inputs. In the **DS1104 inputs or Virtual signals**

list box on the right, select **virtual_Square_1.** This signal will be replaced in the application by a square signal generator running on the dSPACE board.

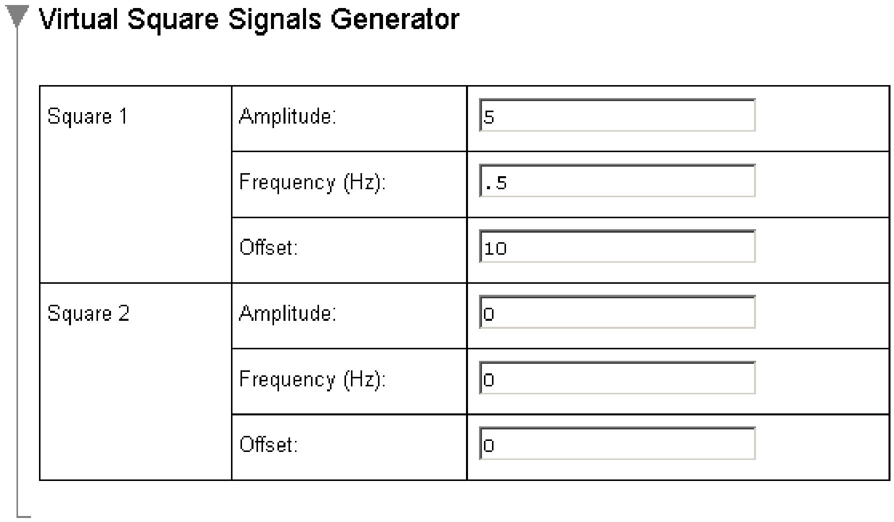
8. In the **Input Mapping** table, select the **Main.Controller1.DeltaPos(t)** signal in the list of model inputs. In the **DS1104 inputs or Virtual signals** list box on the right, select **DELTA_INC1**, which is the delta position information given by the first incremental encoder input of the DS1104 board.
9. In the **Input Mapping** table, select the **Main.Controller1.Imeas(t)** signal in the list of model inputs. In the **DS1104 inputs or Virtual signals** list box on the right, select **ADCH1.** This signal will read the value of the first ADC channel of the DS1104.
10. In the **Output Mapping** table, select the **Main.Controller1.Direction(t)** signal in the list of model outputs. In the **DS1104 outputs** list box at the right, select **Bit_ch0.** This signal will send the value computed by the application to the first digital channel of the DS1104. Note that the digital inputs and outputs of the DS1104 board can be configured either as inputs or outputs. In the present case, since the Bit_ch0 channel is defined as an output, it can no longer be used as an input. To verify this, browse the **DS1104 inputs or Virtual signals** list box and look for the Bit_ch0 signal. You will see that it is marked as unavailable. If you try to connect an input signal to this channel, you will get an error message, until you disconnect the channel from its output signal.
11. In the **Output Mapping** table, select the **Main.Controller1.DutyCycle(t)** signal in the list of model outputs. In the **DS1104 outputs** list box on the right, select **PWM_ch1.** This signal will send the value computed by the application to the first single phase PWM channel of the DS1104.
12. The **Main.Controller1.Current(t)** signal is left unconnected. You will just monitor it with ControlDesk.



To configure the parameters of the different inputs and outputs, follow steps 13-17:

13. To configure the virtual square signal generator, scroll down in the template and expand the **Virtual Square Signals Generator** section. In the Square 1 part of the table, enter **5** for the Amplitude value, **0.5 Hz** for the Frequency and

10 for the offset.



14. To configure the incremental encoder interface, scroll up in the template and expand the **Incremental Encoder Interface** section. In the **General parameters** section, select the **Encoder 1** mode as **TTL**. Since there is no index signal on this encoder, for reset on **Encoder 1** index mode, click **false**. Leave the default value of 0 in the initial

position of the encoder in the Initialization section as zero, since you will not be using the position capture.

Incremental Encoder Interface

The master PPC on the DS1104 controls an incremental encoder interface. It has the following characteristics:
- Input channels for two digital incremental encoders
- Support of single-ended TTL and differential RS422 signals
- 24-bit position counter
- 1.65 MHz maximum encoder line count frequency.
- Line termination for differential inputs
- Power supply for incremental encoders (5V and 0.1A)

General parameters

Channel characteristics:

Encoder 1 mode:	(•) TTL ( ) RS422
Encoder 2 mode:	(•) TTL ( ) RS422

If the reset-on-index mode is set, the counter of the specified channel is reset to 0 when an index signal occurs:

Encoder 1 reset on index mode:	( ) true (•) false
Encoder 2 reset on index mode:	( ) true (•) false

Initialization

Specifies an initial value for the position, given in lines including the line subdivision as decimal place. Since the encoder channels use a 4-fold line subdivision, you can specify this value in multiples of 0.25..

Encoder 1:	0.
Encoder 2:	0.

15. The ADC1 to which the current sensor is connected, has no configuration possibility. However, you can check the characteristics of this channels by scrolling up in the template and expanding the **Mux ADC Unit** section.
16. To configure the first single phase PWM channel, scroll down in the template and expand the **PWM Generation** section. In the **General parameters** section, select the PWM mode as asymmetric, the PWM period as 2 ms and the polarity as active high for the first channel. In the **Initialization** section, select 0 as the initial duty cycle for the first PWM channel (the initial speed of the motor should be 0). In the **Termination** section, select the **Termination value** radio button and select 0 as the termination duty cycle for the first PWM channel (for safety the termination speed of the motor should be 0)

PWM Generation

The slave DSP provides four output channels for 1-phase PWM signal generation.

Conflicting I/O features
When using D2F channel 4, you cannot generate standard PWM signals.

General parameters

PWM mode	◉ Asymmetric ○ Symmetric

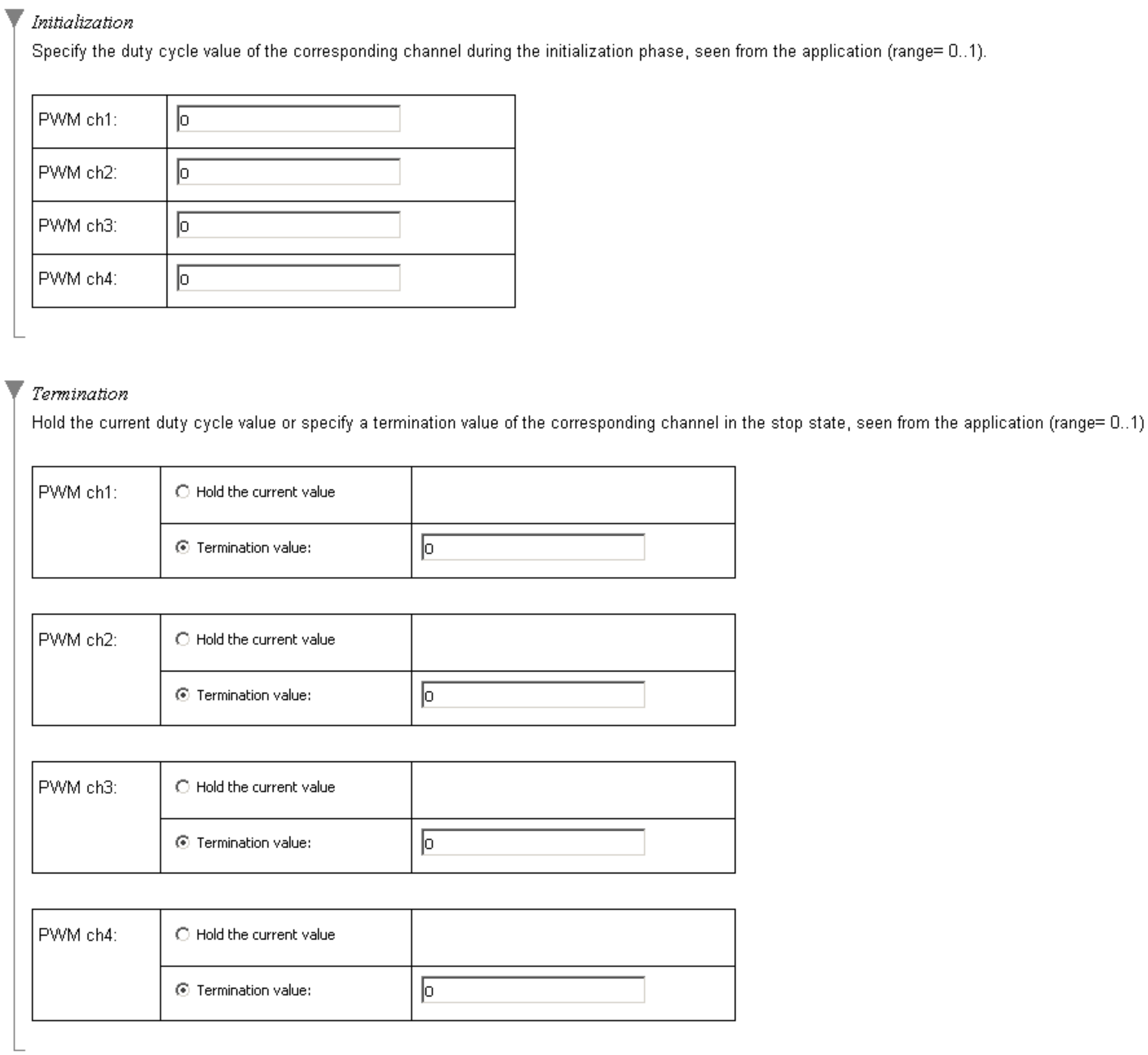
Determines whether a mid-symmetrical or begin synchronized PWM should be used.

PWM Period (ms):	2

Specifies the duration of the PWM period in seconds. The minimum and maximum periods depend on the mode:
- Symmetric mode: period between 100 ns and 400 ms
- Asymmetric mode: period between 200 ns and 800 ms

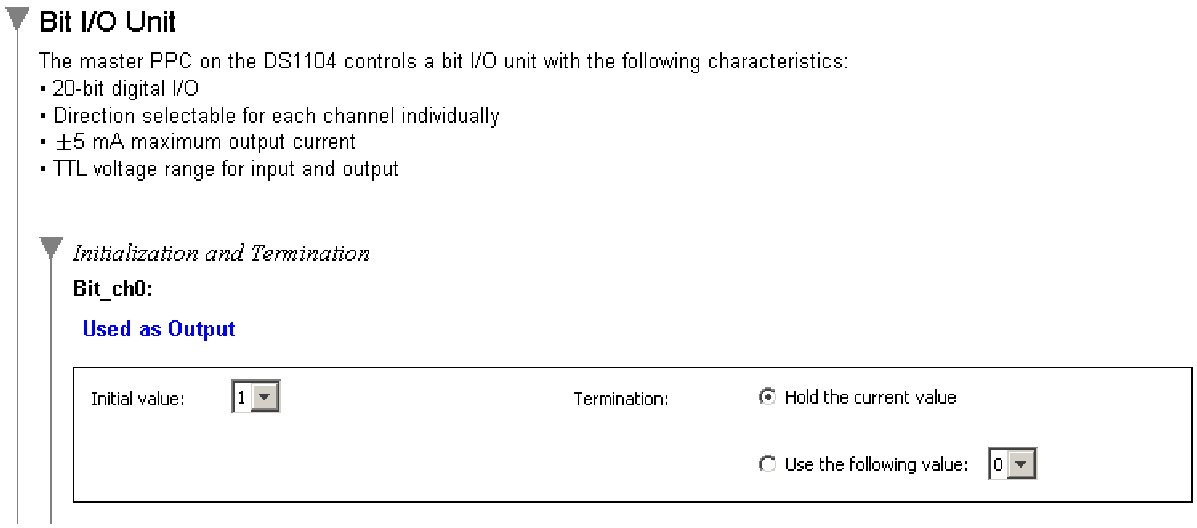
PWM ch1 polarity:	◉ Active high ○ Active low
PWM ch2 polarity:	◉ Active high ○ Active low
PWM ch3 polarity:	◉ Active high ○ Active low
PWM ch4 polarity:	◉ Active high ○ Active low

Specifies the output polarity for each channel.



17. To configure the first digital I/O channel, scroll up in the template and expand the **Bit I/O Unit** section. In the Initialization and Termination section, you will notice that the first channel is marked as output, whereas the other channels are marked as unused. Select 1 as the initial value (corresponds to forward turn of the motor). In the **Termination** section, select the **Hold the current value** radio button. You want the termination direction of the motor

be the same as it was just before the termination of the application.



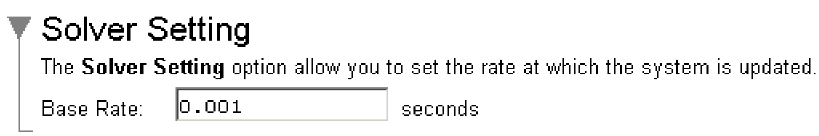
The I/O mapping and configuration is now complete. You can now proceed to the next step; exporting the controller to a real-time dSPACE application.

Note that you can attach more than one template of this kind if you need, for example, to define different I/O mapping for the same controller.

2.5 Exporting the Controller to a dSPACE real-time application running on the DS1104 board

In this section, you will create an dSPACE executable than can run on a DS1104, from the Controller subsystem. Please follow these steps:

1. In the **Solver Setting** section, you must specify a base sampling rate of 1 ms



2. In the **Advanced Code Generation Settings** section, set the **Code Optimization** option to **Full**. This option specifies the degree of simplification applied to the model equations during the code generation process. This option eliminates redundant variables and equations in the system.
3. In the **Advanced Code Generation Settings** section, open the **TRC Variables and Parameters Option** section. These options allows you to add or not all the possible parameters and/or variables in the TRC file that is generated with the dSPACE application. The TRC file is used by ControlDesk so that you can display variables and modify parameters on the running real-time application.
4. In the **Generate the dSPACE Application** section of the template, specify the dSPACE directory and the directory where the application should be generated.
5. Click **Generate to dSPACE Application** to generate the C code, the .trc file, the .sdf file and the .ppc file of the application.
6. Open **ControlDesk** and load the generated application as usual. You will have access to the inputs, outputs, states and states derivatives of the application in the **Model** group. The parameters are in the **Model Parameters** subgroup, the execution time and current time can be accessed from the **Task Info -> Execution** subgroup. Of course, this application should be connected to the real system so that you get interesting values.

3 Supported Inputs and Outputs on the dSPACE DS1104 board

This chapter defines the list of supported inputs and outputs and features on the dSPACE DS1104 board.

You will find more details on each feature and input and output in the dSPACE DS1104 R&D Controller Board Features Guide.

You will find more details on the units and range values of the different configurations parameters in the dSPACE_DS1104 Maple template you need to attach to a model when you want to export a MapleSim subsystem to a dSPACE application.

3.1 Mux ADC Unit

1 A/D converter (ADC1) multiplexed to four channels (signals ADCH1 … ADCH4).The input signals of the converter are selected by a 4:1 input multiplexer. The A/D converters have the following characteristics:

- 16-bit resolution
- ±10 V input voltage range
- ± 5 mV offset error
- ± 0.25% gain error
- > 80 dB (at 10 kHz) signal-to-noise ratio (SNR)

These inputs are used in polling mode.

The end of A/D conversion interrupt and the synchronized start of A/D conversion are not supported.

The application range is the same as the voltage range: -10..10.

3.2 Parallel ADC Converters

4 parallel A/D converters (ADC2 … ADC5) with one channel each (signals ADCH5 … ADCH8). The A/D converters have the following characteristics:

- 12-bit resolution
- ±10 V input voltage range
- ± 5 mV offset error
- ± 0.5% gain error
- > 70 dB signal-to-noise ratio (SNR)

These inputs are used in polling mode.

In case more than one channel is used, all the channels are read at the same time.

The end of A/D conversion interrupt and the synchronized start of A/D conversion are not supported.

The application range is the same as the voltage range: -10..10.

3.3 DAC Unit

The master PPC on the DS1104 controls a D/A converter. It has the following characteristics:

- 8 parallel DAC channels (signals DACH1 … DACH8)
- 16-bit resolution

- ±10 V output voltage range
- ± 1 mV offset error, 10 µV/K offset drift
- ± 0.1% gain error, 25 ppm/K gain drift
- > 80 dB (at 10 kHz) signal-to-noise ratio (SNR)
- Transparent and latched mode

The following parameters can be defined independantly for each channel

- Initial value
- Termination value: hold the last value of the application or termination value specified by the user

The synchronized update of the DAC is not supported.

The application range is the same as the voltage range: -10..10.

3.4 Bit I/O Unit

The master PPC on the DS1104 controls a bit I/O unit with the following characteristics:

- 20-bit digital I/O
- Direction selectable for each channel individually
- ±5 mA maximum output current
- TTL voltage range for input and output

The following parameters can be defined independantly for each channel

- Input our output mode
- Initial value
- Termination value: hold the last value of the application or termination value specified by the user

3.5 Incremental Encoder Interface

The master PPC on the DS1104 controls an incremental encoder interface. It has the following characteristics:

- Input channels for two digital incremental encoders
- Support of single-ended TTL and differential RS422 signals
- 24-bit position counter
- 1.65 MHz maximum encoder line count frequency.
- Line termination for differential inputs
- Power supply for incremental encoders (5V and 0.1A)

The following parameters can be defined independantly for each channel

- TTL or RS422 mode
- Reset on index mode: true or false
- Initial position

A line subdivision of 4 is used. This configuration cannot be changed.

The synchronized incremental encoder position strobe is not supported.A

3.6 PWM Generation

The slave DSP provides four output channels for 1-phase PWM signal generation.

The PWM mode (asymmetric or symmetric) and the PWM period can be specified globally for the four channels.

The minimum and maximum periods depend on the mode:

- Symmetric mode: period between 100 ns and 400 ms
- Asymmetric mode: period between 200 ns and 800 ms

The following parameters can be defined independently for each channel

- Polarity (active high or active low)
- Initial duty cycle
- Termination duty cycle: hold the last value of the application or termination value specified by the user

Conflicting I/O features

When using D2F channel 4, you cannot generate standard PWM signals.

3.7 3-Phase PWM Generation

The slave DSP provides 3 output channels (phases) for 3-phase PWM signal generation (PWM3) in the frequency range 1.25 Hz … 5 MHz.

PWM3 signals are centered on the middle of the PWM period (symmetric mode). The polarity of the non-inverted PWM3 signals is active high.

The period and dead band can be specified globally.

The following parameters can be defined independantly for each channel

- Initial duty cycle
- Termination duty cycle: hold the last value of the application or termination value specified by the user

The PWM interrupt is not supported.

Conflicting I/O features

When using 3-phase PWM (PWM3), you cannot generate the D2F square wave signals.

3.8 PWM Capture

The slave DSP provides input channels for the measurement of the duty cycles and PWM periods of up to four PWM signals.

Conflicting I/O features

When using the PWM measurement, you cannot perform F2D frequency measurement.

3.9 Square Wave Signal Generation

The slave DSP provides four output channels for square-wave signal generation.

The frequency range can be specified globally.

The following parameters can be defined independantly for each channel

- Initial frequency
- Termination duty frequency: hold the last value of the application or termination value specified by the user

Conflicting I/O features

- When using D2F square wave signal generation, you cannot generate 3-phase PWM.
- When using D2F channel 4, you cannot generate standard PWM signals.

3.10 Square Wave Capture

The slave DSP provides input channels for the measurement of the frequencies of up to four square-wave signals.

The minimum frequency to be measured can be specified independantly for each channel.

Conflicting I/O features

When using the F2D frequency measurement, you cannot perform PWM2D measurement.

3.11 Virtual Signals

In order to avoid the use of external signal generator and to allow the tuning or reference signals directly from ControlDesk, a library of virtual signals, that can be generated in real-time, is provided. These virtual signals replace inputs and outputs that would be typically connected to external signal generators.

Virtual Square Signals Generator

2 virtual square signals are provided.

The following parameters can be defined independantly for each signal

- Amplitude
- Frequency
- Offset

Virtual Sine Signals Generator

2 virtual sine signals are provided.

The following parameters can be defined independantly for each signal

- Amplitude
- Frequency
- Offset
- Phase

Virtual Step Signals Generator

2 virtual step signals are provided.

The following parameters can be defined independantly for each signal

- Amplitude
- Offset
- Time of step

Virtual Constant Signals Generator

10 virtual constant signals are provided.

The constant value can be defined independantly for each signal

3.12 Unsupported features or Inputs and Outputs

The following features or inputs and outputs of the dSPACE DS1104 board are not supported:

- All the synchronizing I/O features are not supported
- Serial Interface
- Slave DSP Bit I/O Unit
- Space Vector PWM Signal Generation
- Slave DSP Serial Peripheral Interface

Index