Getting Started with the MapleSim Connector for dSPACE DS1104

Copyright

Maplesoft, Maple, and MapleSim are all trademarks of Waterloo Maple Inc.

© Maplesoft, a division of Waterloo Maple Inc. 2011-2014. All rights reserved.
No part of this book may be reproduced, stored in a retrieval system, or transcribed, in any form or by any means — electronic, mechanical, photocopying, recording, or otherwise. Information in this document is subject to change without notice and does not represent a commitment on the part of the vendor. The software described in this document is furnished under a license agreement and may be used or copied only in accordance with the agreement. It is against the law to copy the software on any medium except as specifically allowed in the agreement.

dSPACE is a registered trademark of dSPACE GmbH.

All other trademarks are the property of their respective owners.

This document was produced using Maple and DocBook.
# Contents

Introduction ................................................................. iv  
1 Getting Started ................................................................. 1  
   1.1 Getting Help ................................................................. 1  
   1.2 Using the dSPACE DS1104 Real-Time Application Template ......................................................... 1  
       Viewing Examples ................................................................. 1  
   1.3 Example: VCFP Model ......................................................... 1  
2 Tutorial: Exporting a Subsystem as a dSPACE Application ................................................................. 4  
   2.1 Overview of the Plant to Control ......................................................... 4  
   2.2 Overview of the Initial Closed-loop Simulation Model ......................................................... 4  
   2.3 Preparing the Model for Export ......................................................... 5  
       Adding the Incremental Encoder Model ......................................................... 5  
       Processing the Incremental Encoder Capture Signal in the Real-time Application ......................................................... 6  
       Converting the Controller to a Subsystem ......................................................... 8  
       Adding the current sensor model ......................................................... 9  
       Processing the current sensor ADC signal in the real-time application ......................................................... 11  
   2.4 Map the Controller subsystem inputs and outputs to the dSPACE DS1104 inputs and outputs ......................................................... 13  
   2.5 Exporting the Controller to a dSPACE real-time application running on the DS1104 board ......................................................... 20  
3 Supported Inputs and Outputs on the dSPACE DS1104 board ......................................................... 22  
   3.1 Mux ADC Unit ......................................................... 22  
   3.2 Parallel ADC Converters ......................................................... 22  
   3.3 DAC Unit ......................................................... 22  
   3.4 Bit I/O Unit ......................................................... 23  
   3.5 Incremental Encoder Interface ......................................................... 23  
   3.6 PWM Generation ......................................................... 24  
   3.7 3-Phase PWM Generation ......................................................... 24  
   3.8 PWM Capture ......................................................... 24  
   3.9 Square Wave Signal Generation ......................................................... 24  
   3.10 Square Wave Capture ......................................................... 25  
   3.11 Virtual Signals ......................................................... 25  
       Virtual Square Signal Generator ......................................................... 25  
       Virtual Sine Signal Generator ......................................................... 25  
       Virtual Step Signal Generator ......................................................... 25  
       Virtual Constant Signal Generator ......................................................... 26  
   3.12 Unsupported Features, Inputs, and Outputs ......................................................... 26  
Index ......................................................... 27
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:

- A Maple template, which provides 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 generates an executable that you can run on the dSPACE board.

- A range of examples illustrating how to prepare and export your models.

Scope of Model Support

MapleSim is a comprehensive modeling tool to create models that 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.

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 DS1104 Real-Time Application Template

The MapleSim Connector for dSPACE provides a **dSPACE DS1104 Real-Time Application** 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. Under the **Libraries** tab on the left side of the MapleSim window, expand the **dSPACE Connector Examples** palette, and then click the entry for the model that you want to view.

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

3. From the list, right-click **dSPACE_DS1104_Controller.mw**, and then select **View**. The code generation template opens 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_Controller template to match your configuration.

**To generate a dSPACE Application**

1. From the dSPACE Connector Examples palette, open the VCFP Getting Started example.
2. Under the Project tab, expand the Attachments palette, expand the Documents section and double-click on dSPACE_DS1104_Controller. Your MapleSim model opens in Maple, in the template that you selected.
3. Browse to the Controller1 subsystem by selecting the subsystem name from the list box in the toolbar above the model diagram. This menu displays all of the subsystems and components in your MapleSim model.
4. In the Step 1: Subsystem Selection section of the template, click Load Selected Subsystem. 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 DS1104 inputs and outputs match the real connections on your system.
5. Navigate to the Step 2: Input / Output Settings for the dSPACE DS1104 Board section. 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 in the DS1104 inputs or Virtual signals list box.
6. You can modify the configuration of this square signal generator by navigating to, and expanding, the Virtual Square Signals Generator section. In the Square1 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 reads 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 in the DS1104 inputs or Virtual signals list box. 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 store the current value when the application terminates. There is no problem with this particular application. However, in general when termination values are different from 0, it is very often insecure. You can also change the DS1104 channel in order to match the configuration of your system.
10. In the Step 3: Options section, enter the base sampling rate of the application in the Solver Setting area.
11. In the Optimization Options area, set the Level of 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. The optimization options and what they do are as follows:

- **None** (0): performs no optimization; the default equations are used in the generated code.
- **Partial** (1, 2): removes redundant equations from the system.
- **Full** (3): performs index reduction to reduce the system to an ODE system or a differential algebraic equation (DAE) system of index 1, and removes redundant equations.

12. In the **Constraint Handling Options** area, specify whether constraints are satisfied in a DAE system by using constraint projection in the generated C code. Use this option to improve the accuracy of a DAE system that has constraints. If the constraint is not satisfied, the system result may deviate from the actual solution and could lead to an increase in error at an exponential rate. The options in this area are as follows:

- **Maximum number of projection iterations**: maximum number of times that a projection is permitted to iterate to obtain a more accurate solution
- **Error tolerance**: the desirable error tolerance to achieve after the projection
- **Apply projection during even iterations**: interpolate iterations to obtain a more accurate solution

13. In the **Event Handling Options** area, specify whether the events are satisfied in a DAE system by using event projection in the generated C code. Use this option to improve the accuracy of a DAE system with events. If the constraint is not satisfied, the system result may deviate from the actual solution and could lead to an increase in error at an exponential rate. The options in this area are as follows:

- **Maximum number of event iterations**: the maximum number of times that a projection is permitted to iterate to obtain a more accurate solution
- **Width of event hysteresis band**: the desirable error tolerance to achieve after the projection

14. In the **TRC Variables and Parameters Options** area, you can set options to allow you to add all the possible parameters and/or variables in the TRC file that are 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.

15. In the **Step 4: Generate dSPACE Application** section of the template, specify the dSPACE directory and the directory where the application should be generated.

16. Click **Generate dSPACE Application** to generate the C code, the .trc file, the .sdf file and the .ppc file of the application.

17. 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 guides you through the process of creating a dSPACE model using the dSPACE DS1104 Real-Time Application 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 model simulates a DC Motor speed controller. 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 Transistor-Transistor Logic (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 (DC Motor Control Start Tutorial) is located in the dSPACE Connector Examples palette under the Libraries tab. Click DC Motor Control Start Tutorial to open the model in MapleSim.

The main components are (from right to left):
- The DC Motor subsystem. You can explore this model to see the different parameters of the motor.
- A H-bridge subsystem, based on ideal switch and diode components. The H-bridge, supplied with a 12V voltage source, provides the required power to the motor. Each switch is driven by an 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 a duty cycle signal, required by the Sign/Magnitude drive mode.

- **A PID controller** which maintains the real motor speed as closely as possible to the reference.
- **A Pulse component** which outputs a square pulse as a reference speed.

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.

**To add the incremental encoder model**

1. Select, and then delete the line between the **DCMotor** speed output and the minus input of the **Feedback** component.
2. Add a **Gain** component (from the **Signal Blocks** > **Common** palette) to the right of the **DCMotor** subsystem.
3. Transform the **Gain** component into a subsystem (that is, select the **Gain** component, then select the **Edit** menu, and then select **Create Subsystem**). Name the subsystem **IncEncoder**.
4. Double-click on this subsystem to open it.
5. Connect the input and output of the **Gain** component to the subsystem boundary. Name the input **Speed** and name the output **DeltaPos**.
6. Select the **Gain** component, and then enter \( \frac{9000 \cdot 0.001}{2 \cdot \pi} \) for the gain value (k).

Since there are 9000 lines per revolution and the sample time of the application will be 1 ms, we have to multiply the real speed by \( \frac{9000 \cdot 0.001}{2 \cdot \pi} \) in order to get the delta position value.

7. Click **Main** in the **Navigation Toolbar** to return to the top level of your model.

8. 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 component with a value which is the inverse of the one you set up in the previous section.

**To process the incremental encoder capture signal**

1. Add a **Gain** component (from the **Signal Blocks > Common** palette) near the minus input of the **Feedback** component (at the left of the PID subsystem).
2. Transform the newly added Gain component into a subsystem (that is, select the Gain component, then select the Edit menu, and then select Create Subsystem). Name the subsystem IncEncoderScaling.

3. Double-click on this subsystem to open it.

4. Connect the input and output of the Gain component to the subsystem boundary. Name the input DeltaPos and name the output Speed.

5. Select the Gain component, and then enter $\frac{2\pi}{9000\cdot0.001}$ for the gain value (k).

6. Click Main in the Navigation Toolbar to return to the top level of your model.

7. Connect the output of the IncEncoder subsystem to the input of the IncEncoderScaling subsystem.

8. Connect the output of the IncEncoderScaling subsystem to the minus input of the Feedback component.
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.

To convert the controller 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 is displayed in the **Model Workspace**.

![Controller subsystem diagram]

5. Double-click the **Controller** subsystem and modify the names of the inputs and outputs as follows:
   - First input (connected to the **Feedback** component): **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**

![DutyCycle subsystem diagram]

**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. To save time, the current sensor model is already included in this model. It is called **CurrentMeas** and can be found under the **Project** tab, in the **Definitions > Subsystems** palette.
To add the current sensor model

1. Click **Main** in the **Navigation Toolbar** to return to the top level of your model.

2. Under the **Project** tab, navigate to the **Definitions > Subsystems** palette, and then drag the **CurrentMeas** subsystem to the **Model Workspace**. Place the **CurrentMeas** subsystem below the **DCMotor** subsystem.

3. Delete the current probe connected to the **DCMotor** subsystem in the model.

4. Connect the current output of the **DCMotor** subsystem 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.

**To process the current sensor ADC signal**

1. Double-click the **Controller** subsystem to open it.
2. Add a **Gain** component (from the **Libraries > 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 then name the input **Imeas**.
4. Connect the output of **Gain** component to the right border of the subsystem, and then name the output **Current**.

   **Note:** 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** component unconnected.

5. Select the **Gain** component, and then enter \( \frac{1}{2000 \cdot 377 \cdot 10^{-6}} \) for the gain value (k).

   **Note:** The sensitivity is 377 \( \mu \text{A/A} \), and you get your value from the voltage measurement across a 2 \( \text{k}\Omega \) resistor. This results in a scaling value of \( \frac{1}{2000 \cdot 377 \cdot 10^{-6}} \).
6. Connect the output of the **IncEncoderScaling** subsystem to the right border of the subsystem, and then name the output **Speed**. This allows you to monitor the motor speed in the real-time application.

7. Delete the three probes. They will be added to **Controller** subsystem outputs in step 10.

8. Click **Main** in the **Navigation Toolbar** to return to the top level of your model.

9. Connect the third input of the **Controller** subsystem to the output of the **CurrentMeas** subsystem.

10. Add probes to the first and second outputs of the **Controller** subsystem.

    **Optional:** You can add probes to the third and fourth outputs of the **Controller** subsystem to display their values during a MapleSim simulation.
The model should be very similar to the **DC Motor Control** model found in the **dSPACE Connector Examples** palette. The **Controller** subsystem is now ready to be prepared for 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 inputs and outputs, reflecting the real connection between the DS1104 and the DC Motor. This will be done using the Maple template, **dSPACE DS1104 Real-Time Application**, which will then be attached to the model.

**To map the controller**

1. Click **Templates ( )** in the **Main Toolbar**.
2. From the list, select **dSPACE DS1104 Real-Time Application**.
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 **dSPACE DS1104 Real-Time Application** template.
5. Browse to the **Controller** subsystem by selecting **Controller** from the list box next to **Main** above the model diagram.
6. In the **Subsystem Selection** section of the template, click **Load Selected Subsystem**. 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 DS1104 inputs and outputs, according to the following details:

- The speed reference could be connected to an 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, the 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 all of the mapping signals, and then you will configure the input and output parameters.

**To define all of the mapping signals:**

1. 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.

2. 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.

3. 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 reads the value of the first ADC channel of the DS1104.

4. 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 sends 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 to 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.

5. 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.

The `Main.Controller1.Current(t)` signal is left unconnected. You can monitor it using ControlDesk.

**To configure the parameters**

1. To configure the first digital I/O channel, scroll down to the **Bit I/O Unit** section, expand this section, and then do the following:
   - In the **Initialization and Termination** section, select 1 for the **Initial Value** (this corresponds to a forward turn of the motor).
   - In the **Termination** section, select **Hold the current value**. (You want the termination direction of the motor to be the same as it was just before the termination of the application.)

   **Note:** The first channel, `Bit_ch0`, should be marked as **Used as Output**. All of the other channels should be marked as **Not used**.
2. To configure the Incremental Encoder Interface, scroll down to the Incremental Encoder Interface section, expand this section, and then do the following:

- In the General parameters section, select TTL for the Encoder 1 mode.
- For the Encoder 1 reset-on-index mode, select false, since there is no index signal on this encoder.
- In the Initialization section, leave the default value of 0 for the initial position of the encoder, since you will not be using the position capture.
3. To configure the first single phase PWM channel, scroll down to the **PWM Generation** section, expand this section, and then do the following:

- In the **General Parameters** section do the following:
  - Select **Asymmetric** as the **PWM mode**.
  - Enter **2** in the **PWM Period (ms)** text area.
  - Select **Active high** for **PWM ch1 polarity** (the polarity for the first channel).

- In the **Initialization** section, enter **0** in the **PWM ch1** text area. This is the initial duty cycle for the first PWM channel; the initial speed of the motor should be 0.

- In the **Termination** section, select **Termination value**, and then enter **0** as the termination duty cycle for **PWM ch1**, the first PWM channel. (For safety, the termination speed of the motor should be 0.)
**General parameters**

<table>
<thead>
<tr>
<th>PWM mode</th>
<th>Asymmetric</th>
<th>Symmetric</th>
</tr>
</thead>
</table>

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

**PWM Period (ms):**

|        | 2          |

Specifies the duration of the PWM period in milliseconds. 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.
4. To configure the **Virtual Square Signal Generator**, scroll down to the **Virtual Square Signals Generator** section, expand it, and then do the following in the **Square 1** part of the table:

- Enter **5** for the **Amplitude**.
- Enter **0.5** for the **Frequency (Hz)**.

### Initialization
Specify the duty cycle value of the corresponding channel during the initialization phase (range = 0..1). There is no scale factor. Thus, the value assigned here should directly correspond to the value assigned in the MapleSim subsystem.

<table>
<thead>
<tr>
<th>PWM ch1:</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>PWM ch2:</td>
<td>0</td>
</tr>
<tr>
<td>PWM ch3:</td>
<td>0</td>
</tr>
<tr>
<td>PWM ch4:</td>
<td>0</td>
</tr>
</tbody>
</table>

### Termination
Hold the current duty cycle value or specify a termination value of the corresponding channel in the stop state (range = 0..1). There is no scale factor. Thus, the value assigned here should directly correspond to the value assigned in the MapleSim subsystem.

<table>
<thead>
<tr>
<th>PWM ch1:</th>
<th>◦ Hold the current value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>◦ Termination value: 0</td>
</tr>
<tr>
<td>PWM ch2:</td>
<td>◦ Hold the current value</td>
</tr>
<tr>
<td></td>
<td>◦ Termination value: 0</td>
</tr>
<tr>
<td>PWM ch3:</td>
<td>◦ Hold the current value</td>
</tr>
<tr>
<td></td>
<td>◦ Termination value: 0</td>
</tr>
<tr>
<td>PWM ch4:</td>
<td>◦ Hold the current value</td>
</tr>
<tr>
<td></td>
<td>◦ Termination value: 0</td>
</tr>
</tbody>
</table>
• Enter 10 for the Offset.

<table>
<thead>
<tr>
<th>Square 1</th>
<th>Amplitude:</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Frequency (Hz):</td>
<td>.5</td>
</tr>
<tr>
<td></td>
<td>Offset:</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Square 2</th>
<th>Amplitude:</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Frequency (Hz):</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Offset:</td>
<td>0</td>
</tr>
</tbody>
</table>

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, for example, you need to define different I/O mappings 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 a dSPACE executable that can be run on a DS1104 from the Controller subsystem.

**To Export the Controller**

1. Scroll down to the **Step 3: Options** section.
2. In the **Solver Settings** area, enter 0.001 s for the **Base Rate**.

   **Solver Settings:**

   Base Rate: 0.001 seconds

3. In the **Optimization Options** area, select 3 (that is **Full**) for the **Level of code optimization**.

   This option specifies the degree of simplification applied to the model equations during the code generation process. The full optimization option eliminates redundant variables and equations in the system.

4. In the **TRC Variables and Parameters Options** section do the following:
   
   • Select **Include the maximum number of variables**.
   
   • Select **Include the maximum number of parameters**.

   These options add all the possible variables and parameters 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 in the real-time application as it is running.
5. Scroll down to the Step 4: Generate dSPACE Application section of the template. Enter the following information for your real-time application:
   - In the dSPACE directory text area, enter the directory where dSPACE is installed.
   - In the Target directory text area, enter the directory where you want to store your real-time application.
   - In the Application name text area, enter the name for your real-time application.

   ![Image of TRC Variables and Parameters Options]

   - Parameters | Value | Export | Updated Row
   - 1

6. Click Generate dSPACE Application to generate the C code, the .trc file, the .sdf file and the .ppc file for your application.

7. Open ControlDesk and load the generated application as usual. You will have access to the inputs, outputs, states and state 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, outputs, and features of the dSPACE DS1104 board.

More details on each of the inputs, outputs, and features can be found in the dSPACE DS1104 R&D Controller Board Features Guide. You will also find more details on the units and range values of the different configurations parameters in the dSPACE DS1104 Real-Time Application template. Attach this template to a model when you want to export a MapleSim subsystem to a dSPACE application.

3.1 Mux ADC Unit

One A/D converter (ADC1) is multiplexed to four channels (signals ADCH1, ADCH2, ADCH3, and 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

Four parallel A/D converters (ADC2, ADC3, ADC4, and ADC5) with one channel each (signals ADCH5, ADCH6, ADCH7, and 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, DACH2, DACH3, DACH4, DACH5, DACH6, DACH7, and 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 modes

The following parameters can be defined independently 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 independently for each channel:
• Input or 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 independently 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.
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 in 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 independently 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 independently 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 independently 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 an 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 typically connect to external signal generators.

#### Virtual Square Signal Generator

Two virtual square signals are provided.

The following parameters can be defined independently for each signal:

- Amplitude
- Frequency
- Offset

#### Virtual Sine Signal Generator

Two virtual sine signals are provided.

The following parameters can be defined independently for each signal:

- Amplitude
- Frequency
- Offset
- Phase

#### Virtual Step Signal Generator

Two virtual step signals are provided.

The following parameters can be defined independently for each signal:

- Amplitude
- Offset
- Time of step
Virtual Constant Signal Generator

Ten virtual constant signals are provided.

The constant value can be defined independently for each signal.

3.12 Unsupported Features, Inputs, and Outputs

The following features, 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

C
converting Controller to Subsystem, 8

D
dSPACE Connector Examples Palette, 1
dSPACE DS1104 board
 supported inputs and outputs, 22
  3-Phase PWM Generation, 24
  Bit IO Unit, 23
  DAC Unit, 22
  Incremental Encoder Interface, 23
  Mux ADC Unit, 22
  Parallel ADC Converters, 22
  PWM Capture, 24
  PWM Generation, 24
  Square Wave Capture, 25
  Square Wave Signal Generation, 24
  Unsupported inputs and outputs, 26
  Virtual Signals, 25
dSPACE DS1104 inputs and outputs, 13

E
Examples
  VCFP Model, 1
exporting Controller to dSPACE real-time application, 20

P
Processing Internal Encoder Capture Signal, 6

T
Templates
  dSPACE Application Generation, 1
Tutorial
  Exporting subsystem as dSPACE Application, 4