dSPACE XIL API

Reference

For dSPACE XIL API .NET 2021-A

Release 2021-A – May 2021

How to Contact dSPACE

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

How to Contact dSPACE Support

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

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

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

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

Software Updates and Patches

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

Important Notice

Contents

About This Reference

Content

This reference gives you detailed information on aspects mentioned in the dSPACE XIL API Implementation Guide 🕮, for example, the error classes available for the supported electrical error simulation hardware.

In the context of dSPACE products that are not based on the ASAM AE XIL API standard, the electrical error simulation is called *failure simulation*. The errors are therefore *failures* that are executed on failure simulation hardware. Usually the hardware provides a failure insertion unit (FIU). You find both terms in this document.

For complete information on the API's capabilities, you should also read the installed documentation of the ASAM AE XIL 2.1.0 standard, and look at the available demo source code.

dSPACE XIL API .NET implementation contains only the implementation of the Framework, Testbench, the MAPort for model access and the EESPort for electrical error simulation. For further information on the dSPACE-specific implementation of the ASAM AE XIL API, also refer to dSPACE XIL API Implementation Guide 🕮.

Related Documents

Below is a list of documents that you are recommended to read when working with dSPACE XIL API .NET implementation.

> **Note**
>
> The documents of the ASAM AE XIL standard are encrypted. The documentation can be viewed only if the dSPACE XIL API .NET installation has been decrypted in the dSPACE Installation Manager. For information on decryption, refer to How to Decrypt Encrypted Archives of dSPACE Software Installations (Managing dSPACE Software Installations 🕮).

The following documents of the ASAM AE XIL standard are available:

- ASAM_AE_XIL_Generic-Simulator-Interface_BS-1-4-Programmers-Guide_V2-1-0.pdf, which contains basic information on the ASAM AE XIL standard.

- ASAM_AE_XIL_Generic-Simulator-Interface_BS-2-4_CSharp-API-Technology-Reference-Mapping-Rules_V2-1-0.pdf, which contains the mapping rules describing the transformation from the generic UML model to C#-specific interfaces.

Use the links in dSPACE Help to access the ASAM AE XIL documents.

Note

Legal Information on ASAM binaries and ASAM documentation
dSPACE software also installs components that are licensed and released by ASAM e.V. (Association for Standardisation of Automation and Measuring Systems).
dSPACE hereby confirms that dSPACE is a member of ASAM and as such entitled to use these licenses and to install the ASAM binaries and the ASAM documentation together with the dSPACE software.
You are not authorized to pass the ASAM binaries and the ASAM documentation to third parties without permission. For more information, visit http://www.asam.net/license.html.

Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
DANGER	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
WARNING	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
CAUTION	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a hazard that, if not avoided, could result in property damage.
Note	Indicates important information that you should take into account to avoid malfunctions.
Tip	Indicates tips that can make your work easier.
	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

%name% Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Special folders

Some software products use the following special folders:

Common Program Data folder A standard folder for application-specific configuration data that is used by all users.

%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>

or

%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>

Documents folder A standard folder for user-specific documents.

%USERPROFILE%\Documents\dSPACE\<ProductName>\
<VersionNumber>

Local Program Data folder A standard folder for application-specific configuration data that is used by the current, non-roaming user.

%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\
<ProductName>

Accessing dSPACE Help and PDF Files

After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com/go/help.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the icon in dSPACE Help. The PDF opens on the first page.

Introduction

Introduction

Gives you general information on the ASAM AE XIL standard and the dSPACE XIL API .NET implementation concerning reference information.

Where to go from here

Information in this section

Basic Information on the ASAM AE XIL API Standard

Introduction

After installing the dSPACE XIL API .NET implementation, some documents of the ASAM are also installed. Here you get an overview on how to work with the ASAM documentation.

ASAM user documentation

For basic information on the ASAM AE XIL standard, refer to ASAM_AE_XIL_Generic-Simulator-Interface_BS-1-4-Programmers-Guide_V2-1-0.pdf.

Note

The dSPACE XIL API .NET implementation only supports Framework, Testbench, MAPort and EESPort.

For details how to implement an application in .NET/C#, refer to the C# Technology Reference provided by ASAM, refer to ASAM_AE_XIL_Generic-Simulator-Interface_BS-2-4_CSharp-API-Technology-Reference-Mapping-Rules_V2-1-0.pdf.

Related topics

Basics

General Concepts of the ASAM AE XIL Standard (dSPACE XIL API Implementation Guide 🕮)

Basic Information on the dSPACE XIL API .NET Implementation

Introduction

Gives you general information on dSPACE-specific implementation details.

Basic configuration for the ASAM XIL API to the dSPACE implementation

When you install the dSPACE XIL API .NET implementation, the relevant files that configure the Framework or Testbench for working with dSPACE hardware are provided.

Related topics

Basics

Features of the dSPACE XIL API Implementation (dSPACE XIL API Implementation Guide 🕮)

Error Codes

Introduction

The ASAM AE XIL API standard also defines the error codes to use.

For an overview of the error codes, refer to *Appendix E. Error Overview* in ASAM_AE_XIL_Generic-Simulator-Interface_BS-1-4-Programmers-Guide_V2-1-0.pdf.

Framework Implementation

Introduction

Gives you detailed information on the dSPACE-specific Framework implementation and the schema definition of a Framework configuration file.

Where to go from here

Information in this section

Information in other sections

Using the Framework (dSPACE XIL API Implementation Guide 📖)

Limitations (dSPACE XIL API Implementation Guide 📖)

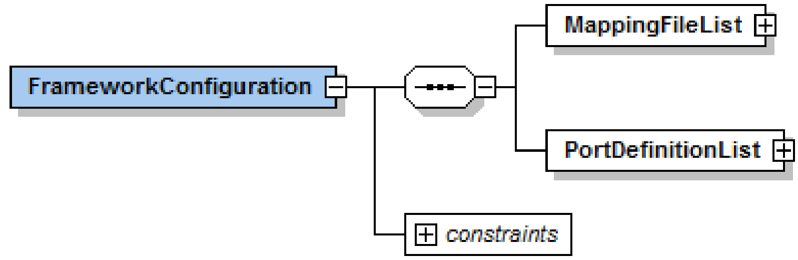
Framework Configuration

Purpose

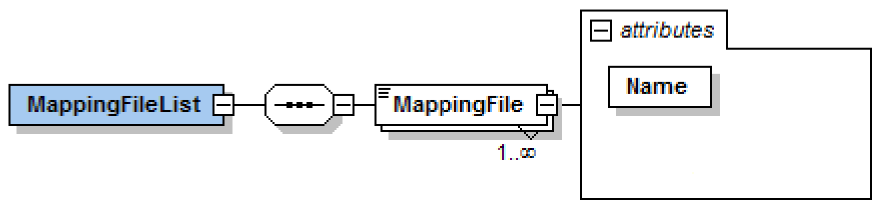
To configure the Framework containing mapping information and all the port configurations that you want to use with your XIL API test application.

FrameworkConfiguration schema definition

The configuration of the Framework is based on the `FrameworkConfiguration.xsd` file that provides the schema definition for the required configuration files in XML format.



MappingFileList Specifies a list of mapping file references, e.g., mapping files for variable mapping or unit conversion.



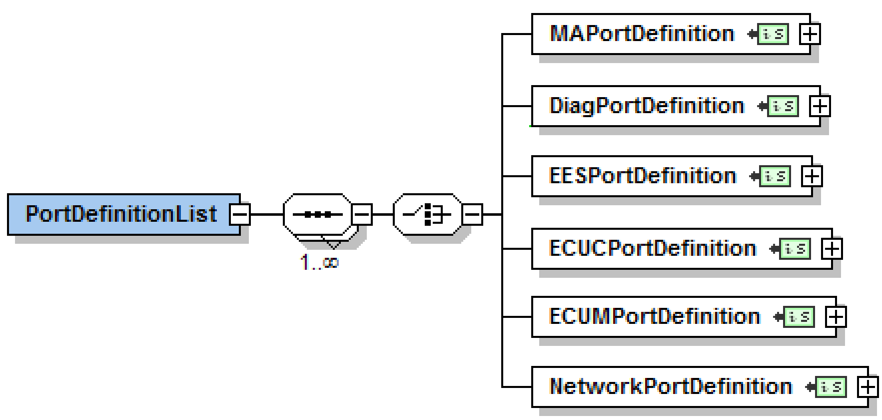
The MappingFile element specifies the path and name of the referenced mapping file.

The following attribute is available:

- Name

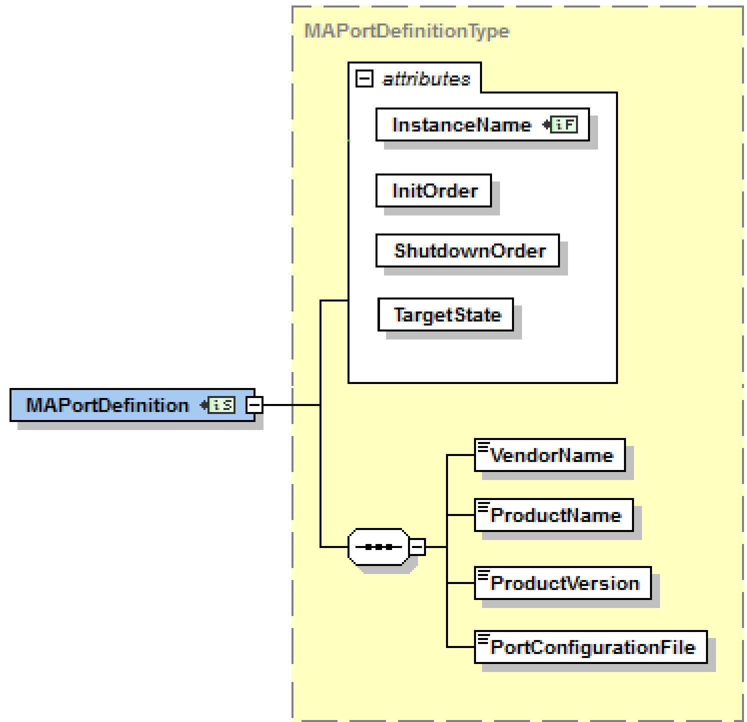
  To specify a descriptive name for the referenced mapping file.

PortDefinitionList Specifies a list of port definitions that consists of a reference to the port configuration file and additional attributes.



When you use dSPACE XIL API .NET, only MAPortDefinition and EESPortDefinition are relevant.

Each port definition provides the same elements and attributes. These elements and attributes are displayed for the MAPortDefinition as an example.



MAPortDefinition A port definition provides the following elements.

Element	Meaning
VendorName	Specifies the name of the vendor.
ProductName	Specifies the name of the product.
ProductVersion	Specifies the version of the product.
PortConfigurationFile	Specifies the path and name of the referenced port configuration.

The following attributes are available:

- InstanceName

  To specify a descriptive name of the port instance.
- InitOrder

  To specify the order in which the ports are to be initialized. A lower value corresponds to a higher precedence.
- ShutdownOrder

  To specify the order in which the ports are to be terminated. A higher value corresponds to a higher precedence.
- TargetState

  To specify the port state to be set up during configuration.

  For restrictions on using the `TargetState` attribute, refer to Limitations (dSPACE XIL API Implementation Guide 🕮).

Note

Before you can execute an XIL API application, you have to register the configured simulation platform. For registering and loading, you can use, for example, ControlDesk.

Examples

The following example shows a framework configuration for a SCALEXIO system. The referenced mapping files are installed in `<InstallationPath>\Demos\Framework\Common\FrameworkMappings`. The referenced MAPort configuration file is installed in `<InstallationPath>\Demos\MAPort\Common\PortConfigurations`. The paths are given as relative paths to the framework configuration file. You find the described framework configuration in `<InstallationPath>\Demos\Framework\Common\FrameworkConfigurations`.

```
<?xml version="1.0" encoding="utf-8"?>
<FrameworkConfiguration xmlns:xsi="http://www.asam.net/XIL/FrameworkConfiguration/2.0">
  <MappingFileList>
    <MappingFile Name="DemoLabels">..\FrameworkMappings\FrameworkLabels.xml</MappingFile>
    <MappingFile Name="DemoMapping_SCALEXIO">..\FrameworkMappings\LabelMapping_SCALEXIO.xml</MappingFile>
    <MappingFile Name="DemoUnits">..\FrameworkMappings\Units.xml</MappingFile>
  </MappingFileList>
  <PortDefinitionList>
    <MAPortDefinition InstanceName="dSPACE SCALEXIO MAPort" InitOrder="0" ShutdownOrder="0"
TargetState="eSIMULATION_RUNNING">
      <VendorName>dSPACE GmbH</VendorName>
      <ProductName>XIL API</ProductName>
      <ProductVersion>2021-A</ProductVersion>
      <PortConfigurationFile>..\..\..\MAPort\Common\PortConfigurations\MAPortConfigSCALEXIO.xml</PortConfigurationFile>
    </MAPortDefinition>
  </PortDefinitionList>
</FrameworkConfiguration>
```

The above mentioned mapping files show you how to configure variable mapping including label mapping and unit conversions.

Model Access Port Implementation

Introduction	Gives you detailed information on the dSPACE-specific MAPort implementation and the schema definition of an MAPort configuration file.
Where to go from here	Information in this section

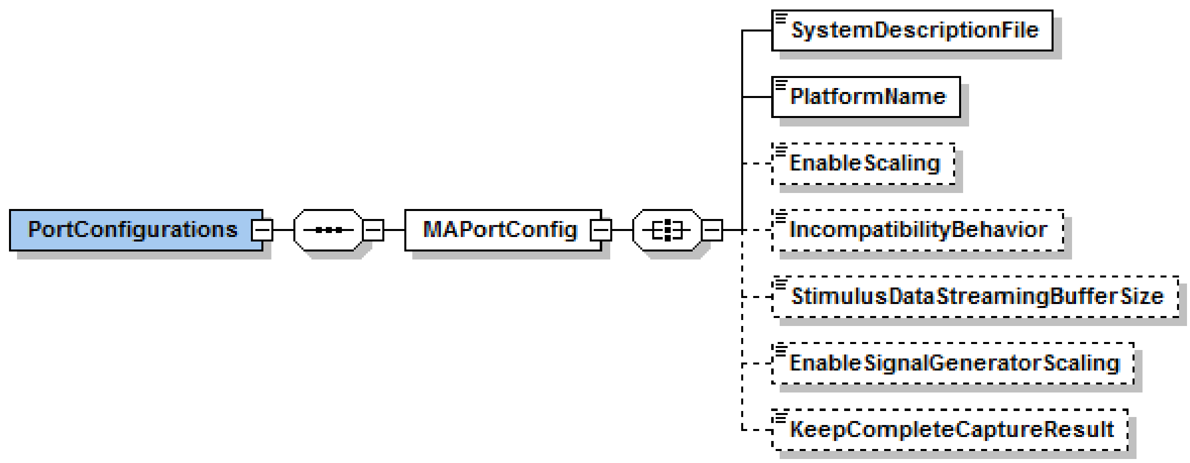
Information in other sections

Implementing an MAPort Application (dSPACE XIL API Implementation Guide 📖)

Limitations (dSPACE XIL API Implementation Guide 📖)

MAPort Configuration

Purpose	To configure the platform you want to use with your XIL API test application.
MAPortConfiguration schema definition	The configuration of the MAPort is based on the `MAPortConfiguration.xsd` that provides the schema definition for the required configuration files in XML format.



PortConfigurations Specifies the base element of the configuration file used for any type of ports.

MAPortConfig Specifies that the configuration is applied to an MAPort.

SystemDescriptionFile Specifies the system description file (SDF file) that is created when you build the simulation application. The file path can be entered as absolute path or relative path relating to the storage location of the MAPort configuration file. The system description file provides information on the simulation application and its contained variables, with which you can work.

> **Note**
>
> If you use the XIL API platform in ControlDesk, you must specify the *absolute* path of the file.

PlatformName Specifies the name of the simulation platform.

You have to specify the same name as used for the registration in ControlDesk, AutomationDesk or ConfigurationDesk (only SCALEXIO systems).

> **Note**
>
> If you want to use a multiprocessor system based on DS1006, you have to enter the name you specified as Platform name in the Register Platforms dialog while registering your multiprocessor system. The default name is *Multiprocessor*.

EnableScaling [Optional] Specifies whether to enable the scaling of values via trace file (TRC file). This function is enabled per default.

For limitations, refer to Limitations (dSPACE XIL API Implementation Guide 📖).

IncompatibilityBehavior [Optional] Specifies how to react if I/O components included in the generated real-time application are not available at the connected platform.

This attribute is only relevant to a SCALEXIO system, DS1007, MicroLabBox, and MicroAutoBox III.

Value	Meaning
0	Abort The download of the real-time application to the platform is stopped. This is the default behavior that is also applied when this attribute is not specified in the MAPort configuration file.
1	Simulate The real-time application is downloaded to the platform and the missing I/O components are simulated.
2	Ignore The real-time application is downloaded to the platform and the missing I/O components are ignored.

StimulusDataStreamingBufferSize [Optional] Specifies the buffer size used by Real-Time Testing for data streaming. The buffer size affects the execution time. This attribute is relevant when you stimulate SignalValueSegment or DataFileSegment signals.

For further information, e.g., how to calculate the required buffer size, refer to MatFile Class Description (Real-Time Testing Library Reference 📖).

EnableSignalGeneratorScaling [Optional] Specifies whether to enable the scaling of values, which are used for stimulating signals, via trace file (TRC file). This function is enabled per default.

KeepCompleteCaptureResult [Optional] Specifies whether to completely store a CaptureResult in the memory or to reduce the size to avoid storage problems, because of a high amount of variables or a long measurement duration.

Value	Meaning
0	Disabled If you set the value to 0, you have to regularly get the capture result with a `Capture.Fetch(false)`. Using the `Capture.CaptureResult` property leads to an exception.
1	Enabled This is the default behavior. The complete CaptureResult is stored in the memory and you can use any kind of capture method.

File writer, such as the `CaptureResultMDFWriter` works independently of this setting.

Note

Before you can execute an XIL API application, you have to register the configured simulation platform. For registering and loading, you can use, for example, ControlDesk.

Examples

Following there are some examples to show the relevant use cases.

Using a real-time application on a modular system based on DS1006

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MAPortConfig>
    <SystemDescriptionFile>D:\Work\TurnSignal_DS1006\turnlamp.sdf</SystemDescriptionFile>
    <PlatformName>DS1006</PlatformName>
  </MAPortConfig>
</PortConfigurations>
```

Using a multi-processor real-time application on modular systems based on DS1006

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MAPortConfig>
    <SystemDescriptionFile>..\SimulationApplications\TurnSignal_DS1006MP\BuildResult\turnlamp.sdf</SystemDescriptionFile>
    <PlatformName>Multiprocessor</PlatformName>
  </MAPortConfig>
</PortConfigurations>
```

Using a multi-core real-time application on a modular system based on DS1007 with changed platform name

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MAPortConfig>
    <SystemDescriptionFile>D:\Work\TurnSignal_DS1007MC\turnlamp.sdf</SystemDescriptionFile>
    <PlatformName>MyDS1007</PlatformName>
    <StimulusDataStreamingBufferSize>512</StimulusDataStreamingBufferSize>
  </MAPortConfig>
</PortConfigurations>
```

Using a real-time application on a SCALEXIO system

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MAPortConfig>
    <SystemDescriptionFile>D:\Work\TurnSignal_SCALEXIO\turnlamp.sdf</SystemDescriptionFile>
    <PlatformName>SCALEXIO</PlatformName>
    <IncompatibilityBehavior>1</IncompatibilityBehavior>
    <StimulusDataStreamingBufferSize>512</StimulusDataStreamingBufferSize>
  </MAPortConfig>
</PortConfigurations>
```

Using an offline simulation application on VEOS

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MAPortConfig>
    <SystemDescriptionFile>D:\Work\TurnSignal_VEOS\turnlamp.sdf</SystemDescriptionFile>
    <PlatformName>VEOS</PlatformName>
  </MAPortConfig>
</PortConfigurations>
```

Using a real-time application with the KeepCompleteCaptureResult option disabled

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MAPortConfig>
    <SystemDescriptionFile>D:\Work\TurnSignal_DS1007\turnlamp.sdf</SystemDescriptionFile>
    <PlatformName>DS1007</PlatformName>
    <KeepCompleteCaptureResult>0</KeepCompleteCaptureResult>
  </MAPortConfig>
</PortConfigurations>
```

TargetScript Configuration

Purpose

To configure the TargetScript you want to use with your XIL API test application.

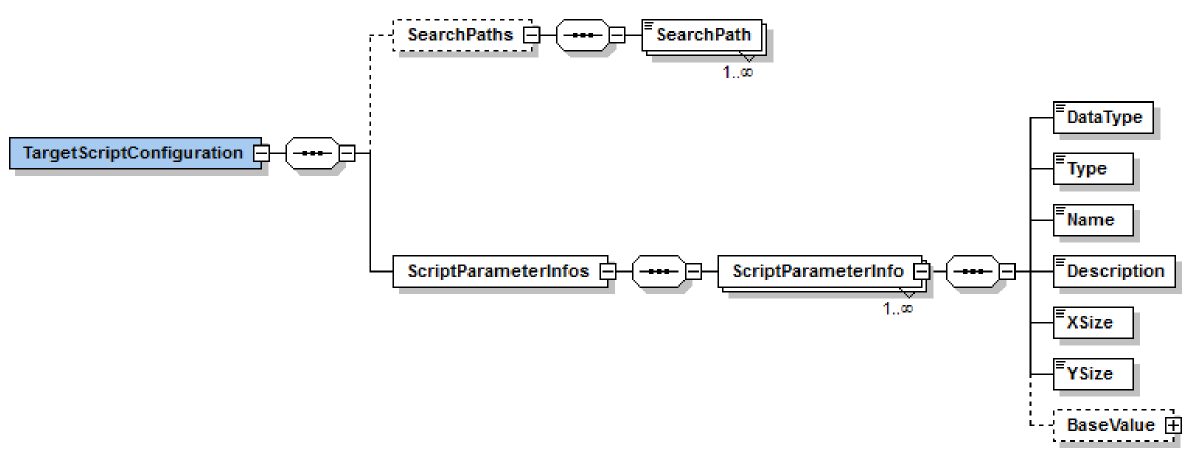
This file is optional. It is only required, if the TargetScript provides parameters that you want to modify, or if you have to add further target scripts to the search path.

File name

The name of the TargetScript configuration file must be the same name of the TargetScript. For example, if your Real-Time Testing sequence, which you want to use as a TargetScript, is called `MyTargetScript.py`, the configuration file must be named `MyTargetScript.xml`. The TargetScript configuration file must be stored in the same folder as the TargetScript.

TargetScriptConfiguration schema definition

The configuration of the TargetScript is based on the following schema definition for the required configuration file in XML format.



TargetScriptConfiguration Specifies the base element of the configuration file used for a TargetScript.

SearchPaths Groups the specified search paths.

SearchPath For each custom Python script that the TargetScript requires as an import, you have to specify its path in a separate SearchPath element.

ScriptParameterInfos Groups the ScriptParameterInfo elements.

ScriptParameterInfo Specifies a variable in the TargetScript that you want to access in your XIL API test application. A script parameter is specified by the below mentionend elements.

DataType Specifies the data type of the parameter.

Possible values are:

- eBOOLEAN,eBOOLEAN_VECTOR, eBOOLEAN_MATRIX
- eINT, eINT_VECTOR, eINT_MATRIX
- eUINT, eUINT_VECTOR, eUINT_MATRIX
- eFLOAT, eFLOAT_VECTOR, eFLOAT_MATRIX
- eSTRING, eSTRING_VECTOR, eSTRING_MATRIX
- eCURVE
- eXYVALUE
- eMAP

Type Specifies the type of the parameter.

Possible values are:

- eLOAD_PARAMETER

  The input parameter can be written in `eIN_CONFIGURATION` state of the script only, i.e., before you call `TargetScript.LoadToTarget()`.
- eSTART_PARAMETER

  The input parameter can be modified before the script is running, i.e., before you call `TargetScript.Start()`. The script is then in the `eFINISHED`, `eREADY`, or `eSTOPPED` state.
- eOUTPUT_PARAMETER

  The output parameter is read-only and read access is restricted to the `eFINISHED` state of the script.

  The output parameters must be specified as `DynamicValue`. For more information, refer to DynamicVariable Class (Real-Time Testing Library Reference 📖).

Name Specifies the name of the parameter.

Description Specifies a description of the parameter.

XSize Specifies the size of a vector or the x-dimension of a matrix. For a scalar value XSize is 0.

YSize Specifies the y-dimension of a matrix. For a scalar value or a vector YSize is 0.

BaseValue Specifies the parameter with its schema-defined data type.

Possible values are:

- BooleanValue, BooleanVectorValue, BooleanMatrixValue
- IntValue, IntVectorValue, IntMatrixValue
- UintValue, UintVectorValue, UintMatrixValue
- FloatValue, FloatVectorValue, FloatMatrixValue
- StringValue, StringVectorValue, StringMatrixValue
- CurveValue
- XYValue
- MapValue

The BaseValue provides the following elements depending on the specified data type.

Element	Description
Type	Specifies the type of the BaseValue, i.e., the same value as specified in the **DataType** element.
Values	Groups the **Value** entries for parameters of matrix type.
Value	Specifies the value of a parameter. For parameters of vector and matrix type, you have to specify as many **Value** elements as specified by the **XSize** and **YSize** elements.
XVector	Specifies the x-axis values for parameters of **Curve** and **XYValue** types.
YVector	Specifies the y-axis values for parameters of **MAP** type.
FcnValues	Specifies the functional values for parameters of **Curve**, **XYValue**, and **MAP** types.

Examples

Using parameters of scalar, vector and matrix types

```
<?xml version="1.0" encoding="utf-8"?>
<TargetScriptConfiguration
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi:"http://www.w3.org/2001/XMLSchema-instance">
  <SearchPaths>
    <SearchPath>C:\Work\MyPythonModules</SearchPath>
  </SearchPaths>
```

```
<ScriptParameterInfos>
  <ScriptParameterInfo>
    <DataType>eBOOLEAN</DataType>
    <Type>eOUTPUT_PARAMETER</Type>
    <Name>MyBoolScalarValue</Name>
    <Description>false: Off, true: On</Description>
    <XSize>0</XSize>
    <YSize>0</YSize>
    <BaseValue xsi:type="BooleanValue">
      <Type>eBOOLEAN</Type>
      <Value>false</Value>
    </BaseValue>
  </ScriptParameterInfo>
  <ScriptParameterInfo>
    <DataType>eINT_VECTOR</DataType>
    <Type>eOUTPUT_PARAMETER</Type>
    <Name>MyIntVectorValue</Name>
    <Description>1: Start, 2: Stop, 3: Pause</Description>
    <XSize>3</XSize>
    <YSize>0</YSize>
    <BaseValue xsi:type="IntVectorValue">
      <Type>eINT_VECTOR</Type>
      <Value>1</Value>
      <Value>2</Value>
      <Value>3</Value>
    </BaseValue>
  </ScriptParameterInfo>
```

```
    <ScriptParameterInfo>
      <DataType>eFLOAT_MATRIX</DataType>
      <Type>eOUTPUT_PARAMETER</Type>
      <Name>MyFloatMatrixValue</Name>
      <Description />
      <XSize>2</XSize>
      <YSize>2</YSize>
      <BaseValue xsi:type="FloatMatrixValue">
        <Type>eFLOAT_MATRIX</Type>
        <Values>
          <Value>1</Value>
          <Value>2</Value>
        </Values>
        <Values>
          <Value>3</Value>
          <Value>4</Value>
        </Values>
      </BaseValue>
    </ScriptParameterInfo>
  </ScriptParameterInfos>
</TargetScriptConfiguration>
```

Using parameters of curve, xyvalue and map types

```
<?xml version="1.0" encoding="utf-8"?>
<TargetScriptConfiguration
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi:"http://www.w3.org/2001/XMLSchema-instance">
  <ScriptParameterInfos>
    <ScriptParameterInfo>
      <DataType>eCURVE</DataType>
      <Type>eOUTPUT_PARAMETER</Type>
      <Name>MyCurveValue</Name>
      <Description />
      <XSize>1</XSize>
      <YSize>0</YSize>
      <BaseValue xsi:type="CurveValue">
        <Type>eCURVE</Type>
        <XVector xsi:type="FloatVectorValue">
          <Type>eFLOAT_VECTOR</Type>
          <Value>1</Value>
          <Value>2</Value>
          <Value>3</Value>
        </XVector>
        <FcnValues xsi:type="FloatVectorValue">
          <Type>eFLOAT_VECTOR</Type>
          <Value>1</Value>
          <Value>2</Value>
          <Value>3</Value>
        </FcnValues>
      </BaseValue>
    </ScriptParameterInfo>
```

```
<ScriptParameterInfo>
  <DataType>eXYValue</DataType>
  <Type>eOUTPUT_PARAMETER</Type>
  <Name>MyXYValue</Name>
  <Description />
  <XSize>1</XSize>
  <YSize>0</YSize>
  <BaseValue xsi:type="XYValue">
    <Type>eXYValue</Type>
    <XVector xsi:type="FloatVectorValue">
      <Type>eFLOAT_VECTOR</Type>
      <Value>1</Value>
      <Value>2</Value>
      <Value>3</Value>
    </XVector>
    <FcnValues xsi:type="FloatVectorValue">
      <Type>eFLOAT_VECTOR</Type>
      <Value>1</Value>
      <Value>2</Value>
      <Value>3</Value>
    </FcnValues>
  </BaseValue>
</ScriptParameterInfo>
```

```
    <ScriptParameterInfo>
      <DataType>eMAP</DataType>
      <Type>eOUTPUT_PARAMETER</Type>
      <Name>MyMapValue</Name>
      <Description />
      <XSize>2</XSize>
      <YSize>2</YSize>
      <BaseValue xsi:type="MapValue">
        <Type>eMAP</Type>
        <XVector xsi:type="FloatVectorValue">
          <Type>eFLOAT_VECTOR</Type>
          <Value>1</Value>
          <Value>2</Value>
        </XVector>
        <YVector xsi:type="FloatVectorValue">
          <Type>eFLOAT_VECTOR</Type>
          <Value>1</Value>
          <Value>2</Value>
        </YVector>
        <FcnValues xsi:type="FloatMatrixValue">
          <Type>eFLOAT_MATRIX</Type>
          <Values>
            <Value>1</Value>
            <Value>2</Value>
          </Values>
          <Values>
            <Value>3</Value>
            <Value>4</Value>
          </Values>
        </FcnValues>
      </BaseValue>
    </ScriptParameterInfo>
  </ScriptParameterInfos>
</TargetScriptConfiguration>
```

MAPort Limitations

Introduction

There are some limitations to be noticed when using the dSPACE XIL API .NET Implementation software for the MAPort. For further information, refer to Limitations (dSPACE XIL API Implementation Guide 🕮).

Electrical Error Simulation Port Implementation

Introduction

Gives you detailed information on the dSPACE-specific implementation of the Electrical Error Simulation port (EESPort) and the supported dSPACE FIU hardware with its failure classes.

Where to go from here

Information in this section

Information in other sections

Basic Information on the EESPort Implementation (dSPACE XIL API Implementation Guide 🕮)

Limitations (dSPACE XIL API Implementation Guide 🕮)

Configuration Files for Electrical Error Simulation

Introduction

For using an EESPort, you have to specify an EESPort configuration file and an error configuration file. For both files you find information on their schema definitions.

Where to go from here

Information in this section

dSPACE EESPort Configuration File

Purpose

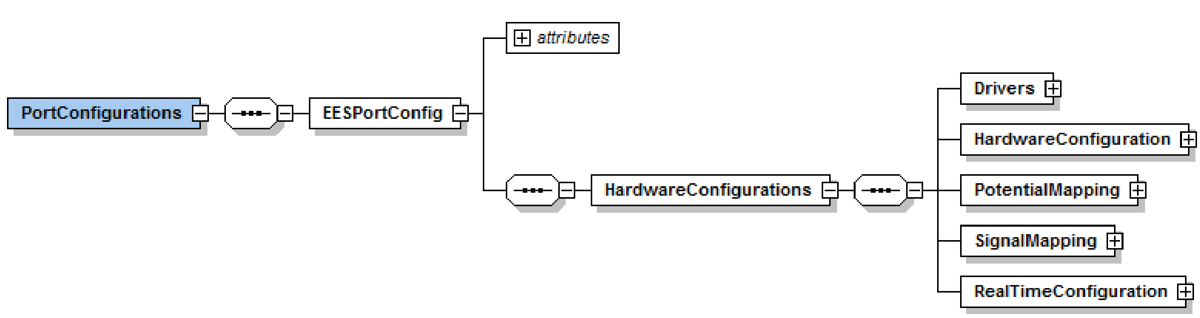
The port configuration files encapsulate the port-specific configuration settings required to setup the Testbench. These files are not standardized, since the port settings are specific to the respective tool or port provider.

EESPortConfiguration schema definition

The dSPACE EESPort configuration file is based on the following schema definition.

Tip

- An EESPort configuration file can be initially created by using the EESPortConfiguration API. For more information, refer to Creating dSPACE EESPort Configuration Files (dSPACE XIL API Implementation Guide 📖).
- For the standard dSPACE hardware, you find EESPort configuration files as examples in `<InstallationPath>\Demos\EESPort\CommonData\PortConfigurations`.



PortConfigurations Specifies the base element of the configuration file used for any type of ports.

EESPortConfig Specifies that the configuration is applied to an EESPort.

Attributes	Description
Logging	For support only, lets you enable logging. Logging essentially increases the execution time. Do not enable logging in normal operating situations. Enable logging only if you want to contact dSPACE Support. ▪ `Logging="false"`: Logging disabled ▪ `Logging="true"`: Logging enabled
OfflineMode	For tests on the host PC without a physical connection to a failure simulation hardware, lets you simulate the activation and triggering of errors without connecting the EESPort to the failure simulation hardware. If you select this property, the use of the tracing variables to monitor the switching behavior and transition states of the failure simulation hardware is disabled automatically. If you are working with software triggers, trigger conditions cannot be checked in offline mode. Error sets that use trigger conditions are therefore activated immediately without checking the specified condition. ▪ `OfflineMode="false"`: Online mode enabled. Failure simulation hardware is required to be connected. ▪ `OfflineMode="true"`: Offline mode enabled
Version	To specify the product version of the EESPort used to create the configuration file, e.g., `"2021-A"`.

HardwareConfigurations Groups the specified hardware configurations.

HardwareConfiguration Specifies the used FIU hardware.

Attributes	Description
ID	To specify a unique identifier as positive integer. This identifier is used for the potential mapping.
SignalListPath	To specify the file containing the list of available signals with an absolute path or relative to the EESPort configuration file. The file format differs for the various platforms: ▪ CSV file for systems with discrete FIU ▪ RTA file for systems with integrated SCALEXIO FIU
DriverId	To specify the driver by its identifier to be used for the current hardware configuration.
EcuVersion	To specify the ECU, if multiple ECUs are specified in the signal list, e.g., 1.

Drivers Groups the specified hardware driver.

Driver Specifies the parameters required to connect to the FIU hardware.

Attributes	Description
ID	To specify a unique identifier as positive integer. This identifier is used for the hardware configuration.
DriverType	To specify the physical connection between the simulator platform and the FIU hardware. ▪ RS232 when using a serial interface, e.g., when you use a system with discrete FIU. ▪ DSCanApi2.0 when using a CAN interface. ▪ SCLX when using a system with integrated SCALEXIO FIU.

Note

Reduced Performance by Using External RS232 Converters

You are strongly recommended to use a physical RS232 port of the host PC to control the failure simulation hardware. If external RS232 ports are missing, try to use an internal RS232 port of the host PC's motherboard. Software triggers and dynamic errors are not supported if you use an external RS232 converter. Communication via an external RS232 converter is also time-critical and can cause communication errors.

If there is no alternative to using an external RS232 converter:

- Use the IOLAN DS1 from Perle as an Ethernet-to-RS232 converter. For configuring this tool, refer to http://www.dspace.com/go/eth2rs232.
- Otherwise, use an USB-to-RS232 converter with an FTDI chipset and the newest FTDI driver, refer to http://www.ftdichip.com/FTDrivers.htm.

The following elements depend on the specified driver type.

Driver Type	Subelements	Description
RS232		
	COMPort	To specify the COM port when using an RS232 serial interface to control the failure simulation hardware, e.g., COM1. The COM port must be physically connected to the failure simulation hardware. The COM port must not be used by another client.
DSCanApi2.0		
	VendorName	To specify the vendor name of the host PC's CAN interface that you are using to control the connected failure simulation hardware. The following vendor names are supported: ▪ dSPACE ▪ Eberspaecher ▪ Kvaser ▪ Vector Informatik ▪ Unknown vendor
	InterfaceName	To specify the name of the host PC's CAN interface. The following CAN interface names are supported: ▪ Calibration Hub ▪ DCI-CAN/LIN1 ▪ DCI-CAN2 ▪ Leaf ▪ Memorator Professional ▪ USBcan II ▪ USBcan Professional ▪ PCAN-miniPCIe FD ▪ CANcaseXL ▪ VN1610 ▪ VN1611

Driver Type	Subelements	Description
		▪ VN1630 ▪ VN1640 ▪ VN5610 ▪ VN5610A ▪ VN7600 ▪ VN8900 ▪ Virtual ▪ Unknown interface
	ChannelIdentifier	To specify the identifier of the channel used for the CAN interface.
	SerialNumber	To specify the serial number of the CAN interface.
SCLX		
	PlatformName	To specify the name of the platform used by the Platform Manager.
	OverrideAccess	To specify whether to automatically disconnect another client that might be connected to the integrated SCALEXIO FIU when you configure the XIL API EESPort. ▪ `OverrideAccess="false"`: The previously connected EESPort client keeps connected to the SCALEXIO FIU hardware. ▪ `OverrideAccess="true"`: The newly configured EESPort client is connected to the SCALEXIO FIU hardware. The disconnected client is not able to simulate errors.

PotentialMapping To map a potential name and a potential type to a unique identifier (e.g., a natural number, starting with 0). The identification of potentials with unique identifiers is required by the ASAM AE XIL API standard. The list order of the potentials in the mapping corresponds to the unique identifiers assigned for the single potentials. For systems with discrete FIU, you find the potential names and potential types in the simulator's signal file. For SCALEXIO systems with the integrated SCALEXIO FIU, you define potentials with ConfigurationDesk as power switches.

Potential mapping is not required for all the hard-wired potentials of your failure simulation hardware, because the dSPACE XIL API implementation knows these potentials. For more information, refer to Creating dSPACE EESPort Configuration Files (dSPACE XIL API Implementation Guide 📖).

The **Potential** element provides the following attributes.

Attributes	Description
ID	To specify a unique identifier as positive integer for the corresponding potential.
Name	To specify the name of the potential provided by your dSPACE simulator.
Type	To specify the type of the potential: ▪ Potential ▪ Ubat

Attributes	Description
	▪ Gnd The type is used to specify which potential is to be connected to the signal.
HardwareConfigurationId	To specify the hardware configuration the mapping is to be used for.

SignalMapping Specifies an optional mapping between the signal names and abstract names used in the XIL API application. The **Signal** element provides the following attributes.

Attributes	Description
MappingName	To specify an abstract name for the signal according to the ASAM AE XIL standard.
ECUName	To specify the name of the ECU.
PinName	To specify the name of the pin.

The ECUs and ECU pins of your HIL system are specified in a signal file (for systems with discrete FIU) or via ConfigurationDesk (for SCALEXIO systems with the integrated SCALEXIO FIU).

RealTimeConfiguration Specifies the monitoring of the switching behavior of the electrical error simulation hardware.

The **RealTimeConfiguration** element provides the following attributes.

Attributes	Description
PlatformName	To specify the name of the platform used by the Platform Manager.
SystemDescriptionFilePath	To specify the path and name of the system description file that is generated when you build the real-time application.
Enabled	To specify whether to activate the monitoring of the switching behavior and triggering by software. ▪ False: The monitoring and software trigger are deactivated. The specified monitoring variables do not represent the current switching states. The conditions for the specified software triggers are ignored. ▪ True: The monitoring and software trigger are activated. The specified monitoring variables represent the current switching states. The conditions for the specified software triggers are evaluated.

With the **Tracing** element of a **RealTimeConfiguration** element, you can specify the variables to be used for monitoring.

With a **Variable** element in the **Tracing** element, you can specify the monitoring variables. The **Variable** element provides the following attributes.

Attributes	Description
Value	To specify the name of the monitoring variable, including the path within the model. The following names are used by default: ▪ XIL API/EESPort/Active ErrorSet ▪ XIL API/EESPort/Error Activated ▪ XIL API/EESPort/Error Switching ▪ XIL API/EESPort/Flags ▪ XIL API/EESPort/Trigger

Attributes	Description
Type	To specify the information type that the variable represents. ▪ ErrorActivated ▪ ActiveErrorSet ▪ ErrorSwitching ▪ Flags ▪ Trigger

For more information, refer to Monitoring the Switching Behavior of Electrical Error Simulation Hardware (dSPACE XIL API Implementation Guide 📖).

With the **SoftwareTrigger** element of a **RealTimeConfiguration** element, you can specify the behavior of a software trigger if one is configured in the error configuration used. The **SoftwareTrigger** element provides the following attribute.

Attributes	Description
PollingInterval	To specify the sample rate for polling the trigger variable of a trigger condition in seconds.

Examples

Following there are some examples to show the relevant use cases.

Using electrical error simulation hardware with a CAN interface As an example, the DS293 FIU module of a dSPACE Simulator Full-Size is to be connected via CAN interface to the host PC.

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations>
  <EESPortConfig>
    <HardwareConfigurations>
      <Drivers>
        <Driver ID="1" DriverType="DSCanAPI2.0">
          <VendorName>dSPACE</VendorName>
          <InterfaceName>DCI-CAN2</InterfaceName>
          <ChannelIdentifier>1</ChannelIdentifier>
          <SerialNumber>0</SerialNumber>
        </Driver>
      </Drivers>
      <HardwareConfiguration ID="1" SignalListPath="signallist full-size ds293.csv" DriverId="1" />
      <PotentialMapping>
        <Potential ID="0" Name="Pot0" Type="Potential" HardwareConfigurationId="1" />
        <Potential ID="1" Name="Pot1" Type="Ubat" HardwareConfigurationId="1" />
        <Potential ID="2" Name="Pot2" Type="Gnd" HardwareConfigurationId="1" />
        <Potential ID="3" Name="Pot3" Type="Potential" HardwareConfigurationId="1" />
        <Potential ID="4" Name="Pot4" Type="Potential" HardwareConfigurationId="1" />
      </PotentialMapping>
      <SignalMapping>
        <Signal MappingName="Signal 1" ECUName="ECU1" PinName="Pin 1" />
        <Signal MappingName="Signal 2" ECUName="ECU1" PinName="Pin 2" />
        <Signal MappingName="Signal 3" ECUName="ECU2" PinName="Pin 3" />
      </SignalMapping>
      <RealTimeConfiguration PlatformName="SCALEXIO Real-Time PC" SystemDescriptionFilePath="Scalexio_Application.sdf"
Enabled="true">
        <Tracing>
          <Variable Value="XIL API/EESPort/Error Activated" Type="ErrorActivated" />
          <Variable Value="XIL API/EESPort/Active ErrorSet" Type="ActiveErrorSet" />
          <Variable Value="XIL API/EESPort/Error Switching" Type="ErrorSwitching" />
          <Variable Value="XIL API/EESPort/Flags" Type="Flags" />
          <Variable Value="XIL API/EESPort/Trigger" Type="Trigger" />
        </Tracing>
       <SoftwareTrigger PollingInterval="0.001" />
      </RealTimeConfiguration>
    </HardwareConfigurations>
  </EESPortConfig>
</PortConfigurations>
```

Using electrical error simulation hardware with a serial interface As an example, the DS291 FIU module of a dSPACE Simulator Full-Size is to be connected via serial interface to the host PC.

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <EESPortConfig>
    <HardwareConfigurations>
      <Drivers>
        <Driver ID="1" DriverType="RS232">
          <COMPort>COM1</COMPort>
        </Driver>
      </Drivers>
      <HardwareConfiguration ID="1" SignalListPath="signallist full-size ds291.csv" DriverId="1" />
      <PotentialMapping>
      </PotentialMapping>
      <SignalMapping>
        <Signal MappingName="Signal 1" ECUName="ECU1" PinName="Pin 1" />
        <Signal MappingName="Signal 2" ECUName="ECU1" PinName="Pin 2" />
        <Signal MappingName="Signal 3" ECUName="ECU2" PinName="Pin 3" />
      </SignalMapping>
      <RealTimeConfiguration PlatformName="SCALEXIO Real-Time PC" SystemDescriptionFilePath="Scalexio_Application.sdf"
Enabled="true">
        <Tracing>
          <Variable Value="XIL API/EESPort/Error Activated" Type="ErrorActivated" />
          <Variable Value="XIL API/EESPort/Active ErrorSet" Type="ActiveErrorSet" />
          <Variable Value="XIL API/EESPort/Error Switching" Type="ErrorSwitching" />
          <Variable Value="XIL API/EESPort/Flags" Type="Flags" />
          <Variable Value="XIL API/EESPort/Trigger" Type="Trigger" />
        </Tracing>
        <SoftwareTrigger PollingInterval="0.001" />
      </RealTimeConfiguration>
    </HardwareConfigurations>
  </EESPortConfig>
</PortConfigurations>
```

Using a SCALEXIO system with integrated SCALEXIO FIU As an example, the following configuration for a SCALEXIO system provides three different potentials and three signal mappings. The configuration also provides the variables used for monitoring the switching behavior.

```
<?xml version="1.0" encoding="utf-8"?>
<PortConfigurations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <EESPortConfig>
    <HardwareConfigurations>
      <Drivers>
        <Driver ID="1" DriverType="SCLX">
          <PlatformName>SCALEXIO Real-Time PC</PlatformName>
        </Driver>
      </Drivers>
      <HardwareConfiguration ID="1" SignalListPath="SCALEXIO_Application.rta" DriverId="1" />
      <PotentialMapping>
        <Potential ID="0" Name="Power Switch 1\VBat" Type="Potential" HardwareConfigurationId="1" />
        <Potential ID="1" Name="Power Switch 2\VBat" Type="Ubat" HardwareConfigurationId="1" />
        <Potential ID="2" Name="Power Switch 3\VBat" Type="Potential" HardwareConfigurationId="1" />
      </PotentialMapping>
      <SignalMapping>
        <Signal MappingName="Signal 1" ECUName="ECU1" PinName="Pin 2" />
        <Signal MappingName="Signal 2" ECUName="ECU2" PinName="Pin 1" />
        <Signal MappingName="Signal 3" ECUName="ECU2" PinName="Pin 2" />
      </SignalMapping>
      <RealTimeConfiguration PlatformName="SCALEXIO Real-Time PC" SystemDescriptionFilePath="Scalexio_Application.sdf"
Enabled="true">
        <Tracing>
          <Variable Value="XIL API/EESPort/Error Activated" Type="ErrorActivated" />
          <Variable Value="XIL API/EESPort/Active ErrorSet" Type="ActiveErrorSet" />
          <Variable Value="XIL API/EESPort/Error Switching" Type="ErrorSwitching" />
          <Variable Value="XIL API/EESPort/Flags" Type="Flags" />
          <Variable Value="XIL API/EESPort/Trigger" Type="Trigger" />
        </Tracing>
        <SoftwareTrigger PollingInterval="0.001" />
      </RealTimeConfiguration>
    </HardwareConfigurations>
  </EESPortConfig>
</PortConfigurations>
```

Related topics

Basics

Creating dSPACE EESPort Configuration Files (dSPACE XIL API Implementation Guide 📖)
Using the EESPortConfiguration API (dSPACE XIL API Implementation Guide 📖)

ASAM XIL API Error Configuration File

Purpose

An error configuration that you store to a file is saved in XML format.

ErrorConfiguration schema definition

The schema definition of an error configuration is included in the ASAM distribution. It will not be described in details here. For details and further information, refer to the ASAM documentation.

The usual way to create an error configuration file is to use the methods of the error factory to implement errors and error sets and then store this implemented errors to an error configuration file. Error configuration files can be used in other XIL API applications by loading them. You can find an example in `<InstallationPath>\Demos\EESPort\Python\ErrorConfiguration.py`.

You therefore do not need to know the schema definition. The following description only shows you the elements required to specify a simple interrupt error.

ErrorConfiguration Specifies the base element of the error configuration file.

Attributes	Description
name	To specify a name for the error configuration.

ErrorSet Specifies an error set contained in the error configuration. The error sets defined within an error configuration are executed in the specified order.

Attributes	Description
name	To specify a name for the error set.
triggerType	To specify the trigger type to be used for starting an error set. ▪ MANUAL ▪ SOFTWARE ▪ HARDWARE **Note** dSPACE XIL API .NET EESPort implmentation supports only manual and software triggers.

SimpleError Specifies the error to be executed within an error set. With its subelement **Signal** you specify the name of the signal to be interfered.

Attributes	Description
errorCategory	To specify the error category of the error. ▪ INTERRUPT

Custom properties

You can optionally add a `CustomProperties` element to an `ErrorConfiguration`, `ErrorSet`, and any kind of errors. A `CustomProperties` element contains `Entry` elements to specify key-value pairs. This lets you add additional data to the EESPort objects that you can use during run time or for result evaluation, for example.

Entry Specifies the key-value pair to be added as a custom property.

Attributes	Description
key	To specify the key of a custom property.
value	To specify the value of the related key.

Examples

Following there is an example to show a relevant use case.

Simulating interrupt errors

```
<?xml version="1.0" encoding="utf-8"?>
<ErrorConfiguration name="Basic error configuration" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <CustomProperties>
    <Entry key="MyProperty1" value="MyValue1" />
  </CustomProperties>
  <ErrorSet name="First error set" triggerType="MANUAL">
    <CustomProperties>
      <Entry key="MyProperty2" value="MyValue2" />
    </CustomProperties>
    <SimpleError errorCategory="INTERRUPT">
      <CustomProperties>
        <Entry key="MyProperty2" value="MyValue2" />
        <Entry key="MyProperty3" value="MyValue3" />
      </CustomProperties>
      <Signal name="Signal 1" />
    </SimpleError>
    <SimpleError errorCategory="INTERRUPT">
      <Signal name="Signal 2" />
    </SimpleError>
    <SimpleError errorCategory="INTERRUPT">
      <Signal name="Signal 3" />
    </SimpleError>
  </ErrorSet>
  <ErrorSet name="Second error set" triggerType="MANUAL">
    <SimpleError errorCategory="INTERRUPT">
      <Signal name="Signal 2" />
    </SimpleError>
    <SimpleError errorCategory="INTERRUPT">
      <Signal name="Signal 3" />
    </SimpleError>
  </ErrorSet>
  <ErrorSet name="Empty error set (deactivate all errors) triggerType="MANUAL" />
</ErrorConfiguration>
```

Failure Classes

Introduction

A failure class describes an error that can be simulated on a pin.

The failure classes that you find in your signal list, or for a SCALEXIO system in ConfigurationDesk, must be mapped to the error categories and error types of the EESPort implementation.

Where to go from here

Information in this section

Overview of EESPort Errors Supported by dSPACE FIU Hardware

Introduction

The following table shows you which EESPort errors are supported by the dSPACE FIU hardware. This information is required if you create an error configuration file for your simulator. You find the legend below the table.

Simulator Type	FIU Hardware	Error Type	Error Category										
			InterruptError	InterruptAtPosition	ErrorToGround	Multi ErrorToGround	ErrorToUbatt	Multi ErrorToUbatt	ErrorToPotential	Multi ErrorToPotential	ErrorPin2Pin	Multi ErrorPin2Pin	InterchangedPins
FullSize Variant 1	DS291	Simple	(w/wo)	-	w/wo	w/wo	w/wo	w/wo	-	-	w/wo	w/wo	-
		Resistor	-	-	-	-	-	-	-	-	-	x	x
		Dynamic	(w/wo)	-	w/wo	w/wo	w/wo	w/wo	-	-	w/wo	w/wo	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	-	x	x
		Loose Contact	-	-	-	-	-	-	-	-	-	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x
	DS5355 / DS5390	Simple	(w/wo)	-	w/wo	w/wo	w/wo	w/wo	-	-	w/wo	w/wo	-
		Resistor	-	-	-	-	-	-	-	-	-	x	x
		Dynamic	(w/wo)	-	w/wo	w/wo	w/wo	w/wo	-	-	w/wo	w/wo	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	-	x	x
		Loose Contact	-	-	-	-	-	-	-	-	-	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x
FullSize Variant 2	DS293	Simple	(w/wo)	-	w	w	w	w	w	w	w	w	-
		Resistor	(w/wo)	-	w	w	w	w	w	w	w	x	x
		Dynamic	(w/wo)	-	w	w	w	w	w	w	w	w	x
		Dynamic Resistor	(w/wo)	-	w	w	w	w	w	w	w	x	x
		Loose Contact	-	-	-	-	-	-	-	-	-	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x

Simulator Type	FIU Hardware	Error Type	Error Category										
			InterruptError	InterruptAtPosition	ErrorToGround	Multi ErrorToGround	ErrorToUbatt	Multi ErrorToUbatt	ErrorToPotential	Multi ErrorToPotential	ErrorPin2Pin	Multi ErrorPin2Pin	InterchangedPins
MidSize 2210	DS749	Simple	(w/wo)	-	wo	wo	wo	wo	-	-	-	-	-
		Resistor	-	-	-	-	-	-	-	-	-	x	x
		Dynamic	(w/wo)	-	wo	wo	wo	wo	-	-	-	-	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	-	x	x
		Loose Contact	-	-	-	-	-	-	-	-	-	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x
	D789	Simple	(w/wo)	-	wo	wo	wo	wo	wo	wo	wo	wo	-
		Resistor	-	-	-	-	-	-	-	-	-	x	x
		Dynamic	(w/wo)	-	wo	wo	wo	wo	wo	wo	wo	wo	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	-	x	x
		Loose Contact	-	-	-	-	-	-	-	-	-	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x

Simulator Type	FIU Hardware	Error Type	Error Category										
			InterruptError	InterruptAtPosition	ErrorToGround	Multi ErrorToGround	ErrorToUbatt	Multi ErrorToUbatt	ErrorToPotential	Multi ErrorToPotential	ErrorPin2Pin	Multi ErrorPin2Pin	InterchangedPins
MidSize 2211	DS791	Simple	(w/wo)	-	wo	wo	wo	wo	wo	wo	wo	wo	-
		Resistor	-	-	-	-	-	-	-	-	-	x	x
		Dynamic	(w/wo)	-	wo	wo	wo	wo	wo	wo	wo	wo	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	-	x	x
		Loose Contact	-	-	-	-	-	-	-	-	-	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x
	DS793	Simple	(w/wo)	-	wo	wo	wo	wo	wo	wo	wo	wo	-
		Resistor	-	-	-	-	-	-	-	-	-	x	x
		Dynamic	(w/wo)	-	wo	wo	wo	wo	wo	wo	wo	wo	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	-	x	x
		Loose Contact	(w/wo)	-	wo	wo	wo	wo	wo	wo	wo	wo	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x
Bus FIU	DS1450	Simple	(w/wo)	w	w	w	w	w	-	-	w	-	-
		Resistor	-	-	-	-	-	-	-	-	w	x	x
		Dynamic	(w/wo)	w	w	w	w	w	-	-	w	-	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	w	x	x
		Loose Contact	-	-	-	-	-	-	-	-	-	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x

Simulator Type	FIU Hardware	Error Type	Error Category										
			InterruptError	InterruptAtPosition	ErrorToGround	Multi ErrorToGround	ErrorToUbatt	Multi ErrorToUbatt	ErrorToPotential	Multi ErrorToPotential	ErrorPin2Pin	Multi ErrorPin2Pin	InterchangedPins
SCALEXIO	SCALEXIO FIU	Simple	(w/wo)	-	w/wo	w/wo	w/wo	w/wo	w/wo	w/wo	w/wo	w/wo	-
		Resistor	-	-	-	-	-	-	-	-	-	x	x
		Dynamic	(w/wo)	-	w/wo	w/wo	w/wo	w/wo	w/wo	w/wo	w/wo	w/wo	x
		Dynamic Resistor	-	-	-	-	-	-	-	-	-	x	x
		Loose Contact	(w/wo)	-	w/wo	-	w/wo	-	w/wo	-	w/wo	-	x
		Loose Contact Resistor	-	-	-	-	-	-	-	-	-	x	x

Legend

Symbol	Meaning
x	Error is not supported by the ASAM standard.
-	Error is not supported by dSPACE FIU hardware.
w	Error is supported by dSPACE FIU hardware. The load type must be specified as `eWITH_LOAD`. Load rejection is not allowed.
wo	Error is supported by dSPACE FIU hardware. The load type must be specified as `eWITHOUT_LOAD`. Load rejection is enforced.
w/wo	Error is supported by dSPACE FIU hardware. Both load types are supported. The load rejection depends on hardware restrictions and user settings.
(w/wo)	Error is supported by dSPACE FIU hardware. It is not allowed to specify a load type.

Failure Classes for the DS291 FIU Module

Supported failure classes

The following table lists the failure classes that can be set if your simulator contains a DS291 FIU Module. Every failure class is assigned to an identifier (failure class ID).

Failure Class ID	Description
1	Cable break
2	Short circuit to GND (KL31) with load
3	Short circuit to GND (KL31) without load
4	Short circuit to +Ubat (KL30) with load
5	Short circuit to +Ubat (KL30) without load
6	Short circuit to COM (other ECU pin) with load
7	Short circuit to COM (other ECU pin) without load

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide)
DS291 FIU Module (dSPACE XIL API Implementation Guide)
Overview of EESPort Errors Supported by dSPACE FIU Hardware... 39

Failure Classes for the DS293 FIU Module

Supported failure classes

The following table lists the failure classes that can be set if your simulator contains a DS293 FIU Module. Every failure class is assigned to an identifier (failure class ID).

Failure Class ID	Description
100	Direct short circuit to POTENTIAL 0
101	Short circuit to POTENTIAL 0 via RSIM (transition impedance)
110	Direct short circuit to POTENTIAL 1
111	Short circuit to POTENTIAL 1 via RSIM (transition impedance)
120	Direct short circuit to POTENTIAL 2
121	Short circuit to POTENTIAL 2 via RSIM (transition impedance)
130	Direct short circuit to POTENTIAL 3
131	Short circuit to POTENTIAL 3 via RSIM (transition impedance)
140	Direct short circuit to POTENTIAL 4
141	Short circuit to POTENTIAL 4 via RSIM (transition impedance)
200	Direct short circuit of ECU signals
201	Short circuit of ECU signals via RSIM (transition impedance)
300	Cable break of ECU signal
301	Additional RSIM in current path of ECU signal

Rules for Pin Failure IDs

The pin failure ID consists of the following three digits

```
<failure group><failure subgroup><additional hardware component>
```

These have the following meanings:

Failure group Identifying the failure group:

Failure Group	Description
1	Connection of an ECU signal to POTENTIAL 0 … 4 from the FIU module
2	Connection of two or more ECU signals
3	Failure in the current path of an ECU signal

Failure subgroup Identifying the failure subgroup:

Failure Subgroup	Description
0	Failure with POTENTIAL 0
1	Failure with POTENTIAL 1
2	Failure with POTENTIAL 2
3	Failure with POTENTIAL 3
4	Failure with POTENTIAL 4

Additional hardware component Identifying the additional hardware component:

Additional Hardware Component	Description
0	None
1	RSIM module DS289MK

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide)
DS293 FIU Module, DS282 Load Module and DS289MK RSim Module (dSPACE XIL API Implementation Guide)
Overview of EESPort Errors Supported by dSPACE FIU Hardware..39

Failure Classes for the DS749 FIU Module

Supported failure classes

The following table lists the failure classes that can be set if your simulator contains a DS749 FIU Module. Every failure class is assigned to an identifier (failure class ID).

Failure Class ID	Description
1	Cable break
2	Short circuit to +Ubat (KL30)
3	Short circuit to GND (KL31)

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide 📖)
DS749 FIU Module (dSPACE XIL API Implementation Guide 📖)
Overview of EESPort Errors Supported by dSPACE FIU Hardware.. 39

Failure Classes for the DS789 Sensor FIU Module

Supported failure classes

The following table lists the failure classes that can be set if your simulator contains a DS789 Sensor FIU Module. Every failure class is assigned to an identifier (failure class ID).

Failure Group	Failure Class ID	Description	Possibilities
1	100	Short circuit to potential 0 (POT0)	1
1	110	Short circuit to potential 1 (POT1)	1
1	120	Short circuit to potential 2 (POT2)	1
1	130	Short circuit to potential 3 (POT0)	1
1	140	Short circuit to potential 4 (POT1)	1
1	150	Short circuit to potential 5 (POT2)	1
1	160	Short circuit to ground (GND)	1
3	300	Cable break	1

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide 📖)
DS789 Sensor FIU Module (dSPACE XIL API Implementation Guide 📖)
Overview of EESPort Errors Supported by dSPACE FIU Hardware.. 39

Failure Classes for the DS791 Actuator FIU Module and the DS793 Sensor FIU Module

Supported failure classes

The following table lists the failure classes that can be set if your simulator contains a DS791 Actuator FIU Module or DS793 Sensor FIU Module. Every failure class is assigned to an identifier (failure class ID).

Failure Group	Failure Class ID	Description	Possibilities
1	100	Short circuit to potential 0 (POT0)	2
1	110	Short circuit to potential 1 (POT1)	2
1	120	Short circuit to potential 2 (POT2)	2
2	200	Short circuit to another ECU pin	2
3	300	Cable break	1

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide 📖)
DS791 Actuator FIU Module (dSPACE XIL API Implementation Guide 📖)
DS793 Sensor FIU (dSPACE XIL API Implementation Guide 📖)
Overview of EESPort Errors Supported by dSPACE FIU Hardware.. 39

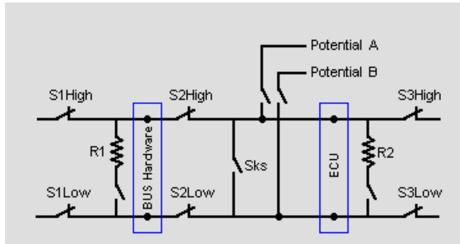
Failure Classes for the DS1450 Bus FIU Board

Supported failure classes

The following table lists the failure classes that can be set if your dSPACE system contains a DS1450 Bus FIU Board. Every failure class is assigned to an identifier (failure class ID).

Pin Failure ID	Description	Possibilities
100	Short circuit of a bus line to potentials (battery voltage or ground)	1
200	Short circuit between both bus lines of a bus channel (Sks)	1
300	Open circuit S1	1
301	Open circuit S2	1
302	Open circuit S3	1
400	Termination resistance	1

For a better understanding, refer to the following illustration showing the circuit diagram of one bus channel of the DS1450 error configuration.



With XIL API.NET 2015-A, the configuration has changed. For details, refer to Signallist Area (dSPACE XIL API Implementation Guide 📖).

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide 📖)
Overview of EESPort Errors Supported by dSPACE FIU Hardware.. 39

References

DS1450 Bus FIU Board (PHS Bus System Hardware Reference 📖)

Failure Classes for the DS5355/DS5390 High Current FIU System

Supported failure classes

The following table lists the failure classes that can be set if your simulator contains FIU system based on a DS5355 High Current FIU Controller Card and a DS5390 High Current FIU board. Every failure class is assigned to an identifier (failure class ID).

Failure Class ID	Description
1	Cable break
2	Short circuit to GND (KL31) with load
3	Short circuit to GND (KL31) without load
4	Short circuit to +Ubat (KL30) with load
5	Short circuit to +Ubat (KL30) without load
6	Short circuit to COM (other ECU pin) with load
7	Short circuit to COM (other ECU pin) without load

Limitations

You have to observe the following limitations when you set a short circuit to GND or +Ubat to the DS5355/DS5390 high current FIU system (failure class IDs 2 … 5):

- All DS5390 of a FIU system support either a short circuit to GND or a short circuit to +Ubat. The boards do not support a short circuit to GND and a short circuit to +Ubat for different pins at the same time.
- If a short circuit to GND or +Ubat is set for a DS5355/DS5390 high current FIU system, no other electrical errors can be simulated by these boards.
- If a short circuit to GND or +Ubat is set for a DS5355/DS5390 high current FIU system, make sure that no DS291 FIU Module simulates a short circuit to other ECU pins at the same time (failure class IDs 6 … 7).

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide 🕮)
DS5355/DS5390 High Current FIU System (dSPACE XIL API Implementation Guide 🕮)
Overview of EESPort Errors Supported by dSPACE FIU Hardware... 39

Failure Classes for SCALEXIO Systems

Supported failure classes

These failure classes are relevant to SCALEXIO systems with integrated SCALEXIO FIU.

You can specify the allowed failure classes for each channel type:

Allowed Failure	Description
Open circuit	The connection between ECU and simulator can be opened. This simulates a cable break.
Short to GND	The ECU signal can be connected to ground.
Short to VBAT	The ECU signal can be connected to the battery voltage. This can be one channel or several channels (if parallel channels are used) on the power switch board.
Short to pin	The ECU signal can be connected to the signal/bus line of another signal generation channel, signal measurement channel, or bus channel.

Note

There are some limitations for pulsed switching and load/signal disconnection. Refer to Basics on Electrical Error Simulation with SCALEXIO Systems (SCALEXIO – Hardware and Software Overview 🕮).

For SCALEXIO systems with discrete FIU, the following failure classes are relevant:

- Failure Classes for the DS291 FIU Module on page 43
- Failure Classes for the DS5355/DS5390 High Current FIU System on page 48

Electrical error simulation and parallel channels

If you use several channels for one signal (parallel channels), for example, for current enhancement, the configuration of allowed failure classes applies to all the parallel channels. When the electrical error is switched, all the channels are switched simultaneously.

Related topics

Basics

Basic Information on Configuring Errors (dSPACE XIL API Implementation Guide 📖)
Electrical Error Simulation with the Integrated SCALEXIO FIU (dSPACE XIL API Implementation Guide 📖)
Hardware for Electrical Error Simulation on SCALEXIO Systems (dSPACE XIL API Implementation Guide 📖)
Overview of EESPort Errors Supported by dSPACE FIU Hardware.. 39
Specifying Allowed Failure Classes for ECU Pins (SCALEXIO – Hardware and Software Overview 📖)

Latencies when Performing Electrical Error Simulation

Introduction

Simulating electrical errors is limited by the latencies of the used simulation system.

Where to go from here

Information in this section

Latencies of the Host PC

Introduction

The latency of the host PC and the communication is independent from the switched error set.

For basic information on latencies, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).

Test equipment

The following host PC was used for the measurements:

Parameter		Specification
PC	Type	Dell Optiplex 7010
Main processor	Type	Intel® Core™ i7-3770
	Frequency	3.4 GHz
	CPU cores	4
	L2-cache	4x 256 kByte
	L3-cache	8 MByte
Chip set		Intel Q77 Express
Memory		32 GB DDR RAM 1600 MHz

Parameter		Specification
BIOS settings	Hyper threading	on
	Turbo boost	on
	Speed step	on
	C states	on
Operating system		Windows 7 (64-bit version)

Measured latencies

The time values in the table result from the statistical evaluation of 1000 measurements.

Time Factor	Measured Latency			
	Mean [ms]	Median [ms]	Min [ms]	Max[1)] [ms]
t_{eval}	1.2	1	0.5	12

1) The maximum value indicates (sporadic) latencies due to a high CPU usage that is caused by other processes or services on the host PC.

For information on t_{eval}, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).

Related topics

Basics

Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖)

Latencies when Using a DS291 FIU Module

Introduction

The following table shows the latencies when performing electrical error simulation with the XIL API .NET implementation and a DS291 FIU Module. The time values in the tables result from the statistical evaluation of 1000 measurements.

- For basic information on latencies and the meanings of t_{FIU} and $t_{error\ activated}$, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).
- For the additional latencies caused by the host PC, refer to Latencies of the Host PC on page 51.

Measured latencies

Error Set Used for the Measurements	Time Factor	Measured Latency			
		Mean [ms]	Median [ms]	Min [ms]	Max[1] [ms]
Interrupt of 1 signal	t_{FIU}	14.7	14.5	14.5	15.5
	$t_{error\ activated}$	48.3	48.5	47.0	55.5
Interrupt of 5 signals	t_{FIU}	14.7	14.5	14.5	15.5
	$t_{error\ activated}$	78.4	78.5	78.0	89.5
Short to ground of 1 signal	t_{FIU}	14.8	14.5	14.5	16.0
	$t_{error\ activated}$	48.6	48.5	48.0	59.0
Short to ground of 5 signals	t_{FIU}	14.7	14.5	14.5	15.0
	$t_{error\ activated}$	78.3	78.5	78.0	84.5
Short to U_{Batt} of 1 signal	t_{FIU}	14.7	14.5	14.5	16.5
	$t_{error\ activated}$	48.3	48.5	48.0	58.5
Short to U_{Batt} of 5 signals	t_{FIU}	14.7	14.5	14.5	16.5
	$t_{error\ activated}$	78.4	78.5	78.0	85.5
Pin to pin of 2 signals	t_{FIU}	14.7	14.5	14.5	15.0
	$t_{error\ activated}$	55.6	55.5	55.0	66.5
Pin to pin of 5 signals	t_{FIU}	14.7	14.5	14.5	15.0
	$t_{error\ activated}$	78.4	78.5	78.0	90.5

[1] The maximum values indicate (sporadic) latencies due to a high CPU usage that is caused by other processes or services on the host PC.

Related topics

Basics

Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖)

Latencies when Using a DS293 FIU Module

Introduction

The following table shows the latencies when performing electrical error simulation with the XIL API .NET implementation and a DS293 FIU Module. The time values in the tables result from the statistical evaluation of 1000 measurements.

- For basic information on latencies and the meanings of t_{FIU} and $t_{error\ activated}$, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).

- For the additional latencies caused by the host PC, refer to Latencies of the Host PC on page 51.

Measured latencies

Error Set Used for the Measurements	Time Factor	Measured Latency			
		Mean [ms]	Median [ms]	Min [ms]	Max[1] [ms]
Interrupt of 1 signal	t_{FIU}	29.7	29.5	28.5	31.0
	$t_{error\ activated}$	75.8	75.5	74.5	84.5
Interrupt of 5 signals	t_{FIU}	29.7	29.5	29.5	30.5
	$t_{error\ activated}$	75.6	75.5	75.0	84.5
Short to ground of 1 signal	t_{FIU}	29.6	29.5	29.5	31.0
	$t_{error\ activated}$	75.6	75.5	75.0	84.5
Short to ground of 5 signals	t_{FIU}	29.7	29.5	29.0	31.0
	$t_{error\ activated}$	75.5	75.5	75.0	80.5
Short to U_{Batt} of 1 signal	t_{FIU}	29.7	29.5	29.0	31.0
	$t_{error\ activated}$	75.7	75.5	75.0	85.0
Short to U_{Batt} of 5 signals	t_{FIU}	29.7	29.5	29.5	30.5
	$t_{error\ activated}$	75.7	75.5	75.0	80.5
Pin to pin of 2 signals	t_{FIU}	29.7	29.5	29.5	31.0
	$t_{error\ activated}$	75.6	75.5	75.0	80.5
Pin to pin of 5 signals	t_{FIU}	29.6	29.5	29.5	30.5
	$t_{error\ activated}$	75.7	75.5	75.0	81.0

[1] The maximum values indicate (sporadic) latencies due to a high CPU usage that is caused by other processes or services on the host PC.

Related topics

Basics

Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖)

Latencies when Using a DS791 Actuator FIU Module

Introduction

The following table shows the latencies when performing electrical error simulation with the XIL API .NET implementation and a DS791 Actuator FIU Module. The time values in the tables result from the statistical evaluation of 1000 measurements.

- For basic information on latencies and the meanings of t_{FIU} and $t_{error\ activated}$, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).
- For the additional latencies caused by the host PC, refer to Latencies of the Host PC on page 51.

Measured latencies

Error Set Used for the Measurements	Time Factor	Measured Latency			
		Mean [ms]	Median [ms]	Min [ms]	Max[1] [ms]
Interrupt of 1 signal	t_{FIU}	71.7	71.5	71.5	75.0
	$t_{error\ activated}$	145.5	145.5	145.0	180.5
Interrupt of 5 signals	t_{FIU}	76.8	76.5	76.5	81.0
	$t_{error\ activated}$	155.5	155.5	155.0	165.5
Short to ground of 1 signal	t_{FIU}	71.8	71.5	71.5	75.0
	$t_{error\ activated}$	145.5	145.5	145.0	155.5
Short to ground of 5 signals	t_{FIU}	76.8	76.5	76.5	81.0
	$t_{error\ activated}$	155.5	155.5	155.0	164.5
Short to U_{Batt} of 1 signal	t_{FIU}	71.7	71.5	71.5	75.5
	$t_{error\ activated}$	145.5	145.5	145.0	156.5
Short to U_{Batt} of 5 signals	t_{FIU}	78.6	76.5	76.5	82.0
	$t_{error\ activated}$	155.6	155.5	155.0	165.0
Pin to pin of 2 signals	t_{FIU}	72.8	72.5	72.5	76.0
	$t_{error\ activated}$	147.5	147.5	147.0	158.5
Pin to pin of 5 signals	t_{FIU}	76.8	76.5	76.5	81.0
	$t_{error\ activated}$	155.6	155.5	155.0	166.5

1) The maximum values indicate (sporadic) latencies due to a high CPU usage that is caused by other processes or services on the host PC.

Related topics

Basics

Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖)

Latencies when Using a DS5355/DS5390 High Current FIU System

Introduction

The following table shows the latencies when performing electrical error simulation with the XIL API .NET implementation and a DS5355 High Current FIU

Controller Card with a DS5390 High Current FIU. The time values in the tables result from the statistical evaluation of 1000 measurements.

- For basic information on latencies and the meanings of t_{FIU} and $t_{error\ activated}$, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).
- For the additional latencies caused by the host PC, refer to Latencies of the Host PC on page 51.

Measured latencies

Error Set Used for the Measurements	Time Factor	Measured Latency			
		Mean [ms]	Median [ms]	Min [ms]	Max[1] [ms]
Interrupt of 1 signal	t_{FIU}	119.7	119.5	119.5	120.0
	$t_{error\ activated}$	155.4	155.5	155.0	161.5
Interrupt of 5 signals	t_{FIU}	119.7	119.5	119.5	120.0
	$t_{error\ activated}$	211.4	211.5	211.0	213.5
Short to ground of 1 signal	t_{FIU}	119.7	119.5	119.5	120.5
	$t_{error\ activated}$	165.3	165.5	165.0	166.5
Short to U_{Batt} of 1 signal	t_{FIU}	119.7	119.5	119.5	120.0
	$t_{error\ activated}$	211.4	211.5	211.0	216.5
Pin to pin of 2 signals	t_{FIU}	119.7	119.5	119.5	120.0
	$t_{error\ activated}$	155.4	155.5	155.0	160.5
Pin to pin of 5 signals	t_{FIU}	119.7	119.5	119.5	120.0
	$t_{error\ activated}$	155.5	155.5	155.0	164.5

[1] The maximum values indicate (sporadic) latencies due to a high CPU usage that is caused by other processes or services on the host PC.

Related topics

Basics

Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖)

Latencies when Using SCALEXIO Signal Measurement Channels

Introduction

The following table shows the latencies when performing electrical error simulation with the XIL API .NET implementation and SCALEXIO signal measurement channels, for example, of a DS2601 Signal Measurement Board. The time values in the tables result from the statistical evaluation of 1000 measurements.

- For basic information on latencies and the meanings of t_{FIU} and $t_{error\ activated}$, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).
- For the additional latencies caused by the host PC, refer to Latencies of the Host PC on page 51.

Measured latencies

Error Set Used for the Measurements	Time Factor	Measured Latency			
		Mean [ms]	Median [ms]	Min [ms]	Max[1] [ms]
Interrupt of 1 signal	t_{FIU}	31.5[2]	31.5[2]	31.0[2]	32.0[2]
	$t_{error\ activated}$	64.8	64.5	63.0	75.5
Interrupt of 5 signals[3]	t_{FIU}	31.5	31.5	31.0	32.0
	$t_{error\ activated}$	97.5	97.5	96.0	108.5
Short to ground of 1 signal	t_{FIU}	< 0.01	< 0.01	< 0.01	< 0.01
	$t_{error\ activated}$	1.2	1.0	0.5	12.5
Short to ground of 5 signals[3]	t_{FIU}	31.5	31.5	31.0	32.0
	$t_{error\ activated}$	65.6	65.5	64.0	75.5
Short to U_{Batt} of 1 signal	t_{FIU}	< 0.01	< 0.01	< 0.01	< 0.01
	$t_{error\ activated}$	1.1	1.0	0.5	2.5
Short to U_{Batt} of 5 signals[3]	t_{FIU}	31.5	31.5	31.0	32.0
	$t_{error\ activated}$	65.5	65.5	64.0	75.5
Pin to pin of 2 signals (with load rejection)	t_{FIU}	31.5	31.5	31.0	32.0
	$t_{error\ activated}$	64.7	64.5	63.0	76.0
Pin to pin of 2 signals (without load rejection)	t_{FIU}	< 0.01	< 0.01	< 0.01	< 0.01
	$t_{error\ activated}$	1.2	1.0	0.5	8.0
Pin to pin of 5 signals[3]	t_{FIU}	31.5	31.5	31.0	32.0
	$t_{error\ activated}$	65.6	65.5	64.0	77.0

1) The maximum values indicate (sporadic) latencies due to a high CPU usage that is caused by other processes or services on the host PC.
2) The signal is interrupted by semiconductor switches in times below 0.01 ms. To avoid parasitic capacitance on the signal channel during electrical error simulation, the SCALEXIO failrails are *additionally* disconnected from the channel. This additional disconnection is responsible for the higher measured latency.
3) Error sets involving more than one signal cannot be completely activated by semiconductor switches. They must be partly activated by relays that affect the switching time for activation. The error activation on one signal can be therefore up to 30 ms prior to another signal.

Related topics

Basics

Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖)

Latencies when Using SCALEXIO Signal Generation Channels

Introduction

The following table shows the latencies when performing electrical error simulation with the XIL API .NET implementation and SCALEXIO signal generation channels, for example, of a DS2621 Signal Generation Board. The time values in the tables result from the statistical evaluation of 1000 measurements.

- For basic information on latencies and the meanings of t_{FIU} and $t_{error\ activated}$, refer to Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 📖).
- For the additional latencies caused by the host PC, refer to Latencies of the Host PC on page 51.

Measured latencies

Error Set Used for the Measurements	Time Factor	Measured Latency			
		Mean [ms]	Median ms]	Min [ms]	Max[1] [ms]
Interrupt of 1 signal	t_{FIU}	11.5[2]	11.5[2]	11.0[2]	12.0[2]
	$t_{error\ activated}$	24.7	24.5	23.0	52.0
Interrupt of 5 signals[3]	t_{FIU}	11.5	11.5	11.0	12.0
	$t_{error\ activated}$	37.5	37.5	36.0	48.5
Short to ground of 1 signal	t_{FIU}	< 0.01	< 0.01	< 0.01	< 0.01
	$t_{error\ activated}$	1.1	1.0	0.5	2.5
Short to ground of 5 signals[3]	t_{FIU}	11.5	11.5	11.0	12.0
	$t_{error\ activated}$	25.4	25.5	24.0	36.5
Short to U_{Batt} of 1 signal	t_{FIU}	< 0.01	< 0.01	< 0.01	< 0.01
	$t_{error\ activated}$	1.1	1.0	0.5	7.0
Short to U_{Batt} of 5 signals[3]	t_{FIU}	11.5	11.5	11.0	12.0
	$t_{error\ activated}$	25.7	25.5	25.0	68.5
Pin to pin of 2 signals (with load rejection)	t_{FIU}	11.5	11.5	11.0	12.0
	$t_{error\ activated}$	24.6	24.5	23.0	35.5
Pin to pin of 2 signals (without load rejection)	t_{FIU}	< 0.01	< 0.01	< 0.01	< 0.01
	$t_{error\ activated}$	1.2	1.0	0.5	3.0
Pin to pin of 5 signals[3]	t_{FIU}	11.5	11.5	11.0	12.0
	$t_{error\ activated}$	25.5	25.5	24.0	35.0

1) The maximum values indicate (sporadic) latencies due to a high CPU usage that is caused by other processes or services on the host PC.

2) The signal is interrupted by semiconductor switches in times below 0.01 ms. To avoid parasitic capacitance on the signal channel during electrical error simulation, the SCALEXIO failrails are *additionally* disconnected from the channel. This additional disconnection is responsible for the higher measured latency.

3) Error sets involving more than one signal cannot be completely activated by semiconductor switches. They must be partly activated by relays that affect the switching time for activation. The error activation on one signal can be therefore up to 10 ms prior to another signal.

Related topics

Basics

Latencies when Performing Electrical Error Simulation (dSPACE XIL API Implementation Guide 🕮)

A

B

C

D

E

F

H

I

L

M

N

P