

# Namespace NationalInstruments.SemiconductorTestLibrary.Common Classes

## [NI Semiconductor Test Exception](#)

Defines a specific exception for NI Semiconductor Test Library.

## [Parallel Execution](#)

Defines utility methods for handling parallel execution.

## [Publish](#)

Defines measurement results publish methods.

## [Site Pin Info](#)

Defines an object that contains the information for an individual site-pin pair. Such as, the site number, pin name, instrument channel string used by the driver, instrument model, etc.

## [System Settings](#)

Defines system settings.

## [Utilities](#)

Provides general helper methods.

# Enums

## [Measurement Type](#)

Defines whether to measure voltage or current.

# Enum MeasurementType

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines whether to measure voltage or current.

```
public enum MeasurementType
```

## Fields

**Current = 1**

Measure current

**Voltage = 0**

Measure voltage

# Class NI Semiconductor TestException

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines a specific exception for NI Semiconductor Test Library.

```
public sealed class NI SemiconductorTestException : Exception, ISerializable, _Exception
```

## Inheritance

[object](#) ← [Exception](#) ← NI SemiconductorTestException

## Implements

[ISerializable](#), [Exception](#)

## Inherited Members

[Exception.GetBaseException\(\)](#), [Exception.ToString\(\)](#),  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#), [Exception.GetType\(\)](#),  
[Exception.Message](#), [Exception.Data](#), [Exception.InnerException](#), [Exception.TargetSite](#),  
[Exception.StackTrace](#), [Exception.HelpLink](#), [Exception.Source](#), [Exception.HResult](#),  
[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.GetHashCode\(\)](#)

## Constructors

[NI SemiconductorTestException\(\)](#)

Initializes a new instance of the [NI SemiconductorTestException](#) class.

[NI SemiconductorTestException\(int, Exception\)](#)

Initializes an instance of [NI SemiconductorTestException](#) with an error code and an inner exception.

[NI SemiconductorTestException\(string\)](#)

Initializes a new instance of the [NI SemiconductorTestException](#) class with an error message.

[NI SemiconductorTestException\(string, Exception\)](#)

Initializes a new instance of the [NI SemiconductorTestException](#) class with an error message and a reference to the inner exception resulting in this exception.

# Methods

## [Throw\(Exception\)](#)

Wraps an exception into a [NI Semiconductor Test Exception](#) and throws.

# Constructor NI Semiconductor TestException

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## NI Semiconductor TestException()

Initializes a new instance of the [NI Semiconductor TestException](#) class.

```
public NI Semiconductor TestException()
```

## NI Semiconductor TestException(string)

Initializes a new instance of the [NI Semiconductor TestException](#) class with an error message.

```
public NI Semiconductor TestException(string message)
```

### Parameters

**message** [string](#) ↗

The error message that explains the reason for the exception.

## NI Semiconductor TestException(string, Exception)

Initializes a new instance of the [NI Semiconductor TestException](#) class with an error message and a reference to the inner exception resulting in this exception.

```
public NI Semiconductor TestException(string message, Exception innerException)
```

### Parameters

**message** [string](#) ↗

The error message that explains the reason for the exception.

`innerException` [Exception](#)

The exception resulting in the current exception.

## NISemiconductorTestException(int, Exception)

Initializes an instance of [NISemiconductorTestException](#) with an error code and an inner exception.

```
public NISemiconductorTestException(int errorCode, Exception innerException)
```

### Parameters

`errorCode` [int](#)

The error code to be included.

`innerException` [Exception](#)

The exception to be embedded.

# Method Throw

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Throw(Exception)

Wraps an exception into a [NISemiconductorTestException](#) and throws.

```
public static void Throw(Exception e)
```

### Parameters

e [Exception](#) ↗

The exception to be wrapped.

# Class ParallelExecution

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines utility methods for handling parallel execution.

```
public static class ParallelExecution
```

## Inheritance

[object](#) ← ParallelExecution

## Inherited Members

[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

## Methods

[DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\], string>\)](#)

Does an operation and publishes the results.

[DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#)

Does an operation and publishes the results.

[DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\], string>\)](#)

Does an operation and publishes the results.

[DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#)

Does an operation and publishes the results.

[DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>, string, string\)](#)

Does an operation and publishes the results.

[DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#)

Does an operation on all sessions and channels in parallel and returns per-instrument per-channel results.

[DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#)

Does an operation on all sessions in parallel with the same inputs and returns per-instrument per-channel results.

[DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#)

Does an operation on all sessions in parallel with the same inputs and returns per-instrument per-channel results.

[DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#)

Does an operation on all sessions and channels in parallel and returns per-site per-pin results.

[DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, int, TResult\[\]>\)](#)

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

[DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[,\]>\)](#)

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

[DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[\]>\)](#)

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

[DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>](#)  
[\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>\)](#)

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

[Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, SitePinInfo>\)](#)

Does an operation on all sessions and channels in parallel.

[Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int, SitePinInfo>\)](#)

Does an operation on all sessions in parallel.

[Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int>\)](#)

Does an operation on all sessions in parallel with session specific inputs.

[Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation>\)](#)

Does an operation on all sessions in parallel with the same inputs.

# Method Do

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**Do<TSessionInformation>**  
**(ISessionsBundle<TSessionInformation>,**  
**Action<TSessionInformation>)**

Does an operation on all sessions in parallel with the same inputs.

```
public static void Do<TSessionInformation>(this ISessionsBundle<TSessionInformation>
sessionsBundle, Action<TSessionInformation> action)
```

Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**action** [Action](#)<TSessionInformation>

The operation to happen on each session info.

Type Parameters

**TSessionInformation**

The type of the session info.

**Do<TSessionInformation>**  
**(ISessionsBundle<TSessionInformation>,**  
**Action<TSessionInformation, int>)**

Does an operation on all sessions in parallel with session specific inputs.

```
public static void Do<TSessionInformation>(this ISessionsBundle<TSessionInformation>
sessionsBundle, Action<TSessionInformation, int> action)
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**action** [Action](#)<TSessionInformation, [int](#)>

The operation to happen on each session info. The second parameter is the session index.

## Type Parameters

**TSessionInformation**

The type of the session info.

**Do<TSessionInformation>**  
**(ISessionsBundle<TSessionInformation>,**  
**Action<TSessionInformation, SitePinInfo>)**

Does an operation on all sessions and channels in parallel.

```
public static void Do<TSessionInformation>(this ISessionsBundle<TSessionInformation>
sessionsBundle, Action<TSessionInformation, SitePinInfo> action)
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**action** [Action](#)<TSessionInformation, [SitePinInfo](#)>

The operation to happen on each session info. The second parameter is the [SitePinInfo](#) associated with a single channel.

## Type Parameters

## TSessionInformation

The type of the session info.

## Do<TSessionInformation> (ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int, SitePinInfo>)

Does an operation on all sessions in parallel.

```
public static void Do<TSessionInformation>(this ISessionsBundle<TSessionInformation>
sessionsBundle, Action<TSessionInformation, int, SitePinInfo> action)
```

### Parameters

**sessionsBundle** ISessionsBundle<TSessionInformation>

The sessions bundle object.

**action** Action<TSessionInformation, int, SitePinInfo>

The operation to happen on each session info. The second parameter is the session index. The third parameter is the SitePinInfo associates with a single channel.

### Type Parameters

**TSessionInformation**

The type of the session info.

# Method DoAndPublishResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**DoAndPublishResults<TSessionInformation>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string)**

Does an operation and publishes the results.

```
public static double[] DoAndPublishResults<TSessionInformation>(this  
ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, double>  
function, string publishedDataId = "")
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, double>](#)

The operation to happen on each session info.

**publishedDataId** [string](#)

The unique data id to use when publishing.

## Returns

[double](#)[]

The per-instrument operation results.

## Type Parameters

**TSessionInformation**

The type of the session info.

## DoAndPublishResults<TSessionInformation> (ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double[]>, string)

Does an operation and publishes the results.

```
public static double[][] DoAndPublishResults<TSessionInformation>(this  
ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, double[]>  
function, string publishedDataId = "")
```

### Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, double\[\]>](#)

The operation to happen on each session info.

**publishedDataId** [string](#)

The unique data id to use when publishing.

### Returns

[double\[\]\[\]](#)

The per-instrument per-channel operation results as a jagged array. Where, the first dimension represents the instrument sessions, and the second dimension represents the instrument channels within an instrument session.

### Type Parameters

**TSessionInformation**

The type of the session info.

## DoAndPublishResults<TSessionInformation> (ISessionsBundle<TSessionInformation>,

## Func<TSessionInformation, bool>, string)

Does an operation and publishes the results.

```
public static bool[] DoAndPublishResults<TSessionInformation>(this  
ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, bool>  
function, string publishedDataId = "")
```

### Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, bool>](#)

The operation to happen on each session info.

**publishedDataId** [string](#)

The unique data id to use when publishing.

### Returns

[bool](#)[]

The per-instrument operation results.

### Type Parameters

**TSessionInformation**

The type of the session info.

## DoAndPublishResults<TSessionInformation> (ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool[]>, string)

Does an operation and publishes the results.

```
public static bool[][] DoAndPublishResults<TSessionInformation>(this  
ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, bool[]>  
function, string publishedDataId = "")
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, bool\[\]>](#)

The operation to happen on each session info.

**publishedDataId** [string](#)

The unique data id to use when publishing.

## Returns

[bool\[\]\[\]](#)

The per-instrument per-channel operation results as a jagged array. Where, the first dimension represents the instrument sessions, and the second dimension represents the instrument channels within an instrument session.

## Type Parameters

**TSessionInformation**

The type of the session info.

**DoAndPublishResults<TSessionInformation>**  
(**ISessionsBundle<TSessionInformation>,**  
**Func<TSessionInformation, Tuple<double[], double[]>>, string,**  
**string**)

Does an operation and publishes the results.

```
public static Tuple<double[][], double[][]> DoAndPublishResults<TSessionInformation>  
(this ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation,
```

```
Tuple<double[], double[]>> function, string publishedDataId1 = "", string publishedDataId2 = "")
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, Tuple<double\[\], double\[\]>>](#)

The operation to happen on each session info. This operation returns two results.

**publishedDataId1** [string](#)

The unique data id to use when publishing the first operation result.

**publishedDataId2** [string](#)

The unique data id to use when publishing the second operation result.

## Returns

[Tuple<double\[\], double\[\]>](#)

The two per-instrument per-channel operation results as a tuple of two jagged arrays, one for each set of results. Where, the first dimension of each jagged array represents the instrument sessions, and the second dimension represents the instrument channels within an instrument session.

## Type Parameters

**TSessionInformation**

The type of the session info.

# Method DoAndReturnPerInstrumentPerChannelResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>)**

Does an operation on all sessions in parallel with the same inputs and returns per-instrument per-channel results.

```
public static TResult[] DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>(this ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, TResult> function)
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, TResult>](#)

The operation to happen on each session info.

## Returns

**TResult[]**

An array of the operation results for all sessions.

## Type Parameters

**TSessionInformation**

The type of the session info.

## TResult

The type of the operation result.

# DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>)

Does an operation on all sessions and channels in parallel and returns per-instrument per-channel results.

```
public static TResult[][] DoAndReturnPerInstrumentPerChannelResults<TSessionInformation,  
TResult>(this ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation,  
SitePinInfo, TResult> function)
```

## Parameters

### sessionsBundle [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

### function [Func<TSessionInformation, SitePinInfo, TResult>](#)

The operation to happen on each session info. The second parameter is the [SitePinInfo](#) associated with a single channel.

## Returns

### TResult[][]

An array of the operation results for all sessions.

## Type Parameters

### TSessionInformation

The type of the session info.

### TResult

The type of the operation result.

`DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>)`

Does an operation on all sessions in parallel with the same inputs and returns per-instrument per-channel results.

```
public static Tuple<TResult1[], TResult2[]>
DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>(this
ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation,
Tuple<TResult1, TResult2>> function)
```

## Parameters

`sessionsBundle ISessionsBundle<TSessionInformation>`

The sessions bundle object.

`function Func<TSessionInformation, Tuple<TResult1, TResult2>>`

The operation to happen on each session info.

## Returns

`Tuple<TResult1[], TResult2[]>`

Two arrays of the operation results for all sessions.

## Type Parameters

`TSessionInformation`

The type of the session info.

`TResult1`

The type of the first item of the operation result.

`TResult2`

The type of the second item of the operation result.



# Method DoAndReturnPerSitePerPinResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult[]>)**

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

```
public static PinSiteData<TResult> DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(this ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, TResult[]> function)
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, TResult\[\]>](#)

The operation to happen on each session info.

## Returns

[PinSiteData<TResult>](#)

The per-site per-pin results of the operation.

## Type Parameters

[TSessionInformation](#)

The type of the session info.

[TResult](#)

The type of the operation result.

## Remarks

The output parameter of the `function` is the **per-instrument all channels** result.

`DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>)`

Does an operation on all sessions and channels in parallel and returns per-site per-pin results.

```
public static PinSiteData<TResult> DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(this ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, SitePinInfo, TResult> function)
```

## Parameters

`sessionsBundle` [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

`function` [Func<TSessionInformation, SitePinInfo, TResult>](#)

The operation to happen on each session info. The second parameter is the [SitePinInfo](#) associated with a single channel.

## Returns

[PinSiteData<TResult>](#)

The per-site per-pin results of the operation.

## Type Parameters

`TSessionInformation`

The type of the session info.

`TResult`

The type of the operation result.

## Remarks

The output parameter of the `function` is the **per-instrument per channel** result.

`DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1[], TResult2[]>>)`

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

```
public static Tuple<PinSiteData<TResult1>, PinSiteData<TResult2>>
DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>(this
ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation,
Tuple<TResult1[], TResult2[]>> function)
```

## Parameters

`sessionsBundle` [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

`function` [Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>](#)

The operation to happen on each session info.

## Returns

[Tuple<PinSiteData<TResult1>, PinSiteData<TResult2>>](#)

The per-site per-pin results of the operation.

## Type Parameters

`TSessionInformation`

The type of the session info.

`TResult1`

The element type of the first array item of the operation result.

`TResult2`

The element type of the second array item of the operation result.

## Remarks

The output parameter of the **function** is the **per-instrument all channels** result.

**DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult[,]>)**

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

```
public static PinSiteData<TResult[]> DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(this ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, TResult[,]> function)
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

**function** [Func<TSessionInformation, TResult\[\]>](#)

The operation to happen on each session info.

## Returns

[PinSiteData<TResult\[\]>](#)

The per-site per-pin results of the operation.

## Type Parameters

**TSessionInformation**

The type of the session info.

**TResult**

The type of the operation result.

## Remarks

The output parameter of the `function` is the **per-instrument all channels** result.

## DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, int, TResult[]>)

Does an operation on all sessions in parallel with the same inputs and returns per-site per-pin results.

```
public static PinSiteData<TResult> DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>(this ISessionsBundle<TSessionInformation> sessionsBundle, Func<TSessionInformation, int, TResult[]> function)
```

## Parameters

### `sessionsBundle` [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

### `function` [Func<TSessionInformation, int, TResult\[\]>](#)

The operation to happen on each session info. The second parameter is the session index. The output parameter is the per-instrument result.

## Returns

### [PinSiteData<TResult>](#)

The per-site per-pin results of the operation.

## Type Parameters

### [TSessionInformation](#)

The type of the session info.

### [TResult](#)

The type of the operation result.

# Class Publish

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines measurement results publish methods.

```
public static class Publish
```

## Inheritance

[object](#) ← Publish

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[PublishResults<T>\(ISemiconductorModuleContext, PinSiteData<T>, string\)](#).

Publishes measurement results.

[PublishResults<T>\(ISemiconductorModuleContext, SiteData<T>, string, string\)](#).

Publishes measurement results for a specific pin.

[PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\], string\)](#).

Publishes measurement results.

[PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\]\[\], string\)](#).

Publishes measurement results.

[PublishSingleSiteResult<T>\(ISemiconductorModuleContext, T, string, string\)](#).

Publishes measurement result for a specific site-pin pair.

# Method PublishResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PublishResults<T>(ISemiconductorModuleContext, PinSiteData<T>, string)

Publishes measurement results.

```
public static void PublishResults<T>(this ISemiconductorModuleContext tsmContext,  
PinSiteData<T> results, string publishedDataId)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**results** [PinSiteData<T>](#)

The measurement results to publish.

**publishedDataId** [string](#)

The unique data id to use when publishing.

### Type Parameters

**T**

The type of the measurement data.

## PublishResults<T>(ISemiconductorModuleContext, SiteData<T>, string, string)

Publishes measurement results for a specific pin.

```
public static void PublishResults<T>(this ISemiconductorModuleContext tsmContext,  
SiteData<T> results, string publishedDataId, string pin = "")
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**results** [SiteData<T>](#)

The measurement results to publish.

**publishedDataId** [string](#)

The unique data id to use when publishing.

**pin** [string](#)

The pin to publish results for.

## Type Parameters

**T**

The type of the measurement data.

**PublishResults<TSessionInformation, TData>  
(ISessionsBundle<TSessionInformation>, TData[][], string)**

Publishes measurement results.

```
public static void PublishResults<TSessionInformation, TData>(this  
ISessionsBundle<TSessionInformation> sessionsBundle, TData[][] results,  
string publishedDataId)
```

## Parameters

**sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

#### **results** TData[][]

The multisite measurement results from multiple pins connected to multiple instrument sessions. The data is in the form of an array of arrays. Each element in the data array is an array that contains the measurement results for the channels of a single instrument session.

#### **publishedDataId** [string](#)

The unique data id to use when publishing.

### Type Parameters

#### **TSessionInformation**

The session information object.

#### **TData**

The type of the measurement data.

## PublishResults<TSessionInformation, TData> (ISessionsBundle<TSessionInformation>, TData[], string)

Publishes measurement results.

```
public static void PublishResults<TSessionInformation, TData>(this  
ISessionsBundle<TSessionInformation> sessionsBundle, TData[] results,  
string publishedDataId)
```

### Parameters

#### **sessionsBundle** [ISessionsBundle<TSessionInformation>](#)

The sessions bundle object.

#### **results** TData[]

The multisite measurement results from multiple pins connected to multiple instrument sessions. Each element in the array represents a measurement made for a single channel from each instrument session.

`publishedDataId` [string](#)

The unique data id to use when publishing.

## Type Parameters

**TSessionInformation**

The session information object.

**TData**

The type of the measurement data.

# Method PublishSingleSiteResult

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PublishSingleSiteResult<T>(ISemiconductorModuleContext, T, string, string)

Publishes measurement result for a specific site-pin pair.

```
public static void PublishSingleSiteResult<T>(this ISemiconductorModuleContext  
tsmSingleSiteContext, T result, string publishedDataId, string pin = "")
```

### Parameters

**tsmSingleSiteContext** ISemiconductorModuleContext

The single site NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**result** T

The measurement result to publish.

**publishedDataId** [string](#)

The unique data id to use when publishing.

**pin** [string](#)

The pin to publish results for.

### Type Parameters

T

The type of the measurement data.

### Remarks

Use this method with a single site context only.

# Class SitePinInfo

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains the information for an individual site-pin pair. Such as, the site number, pin name, instrument channel string used by the driver, instrument model, etc.

```
public class SitePinInfo
```

## Inheritance

[object](#) ← SitePinInfo

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[SitePinInfo\(int, string\)](#)

Constructs a site-pin pair.

[SitePinInfo\(int, string, string, string\)](#)

Constructs a site-pin pair.

[SitePinInfo\(string\)](#)

Constructs a site-pin pair.

## Fields

[SiteNumberNone](#)

Defines the value of site number for a system pin.

## Properties

[IndividualChannelString](#)

The individual channel string in "SMU\_4147\_C1\_S06/1" format.

## ModelString

The model string of the owning instrument of the channel associated with this site-pin pair.

## PinName

The pin name.

## SiteNumber

The site number.

## SitePinString

The site-pin pair in "site0/A" format.

# Constructor SitePinInfo

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SitePinInfo(string)

Constructs a site-pin pair.

```
public SitePinInfo(string sitePinString)
```

Parameters

**sitePinString** [string](#)

The site-pin pair in "site0/A" format.

## SitePinInfo(int, string)

Constructs a site-pin pair.

```
public SitePinInfo(int siteNumber, string pinName)
```

Parameters

**siteNumber** [int](#)

The site number. Specify -1 for a system pin.

**pinName** [string](#)

The pin name.

## SitePinInfo(int, string, string, string)

Constructs a site-pin pair.

```
public SitePinInfo(int siteNumber, string pinName, string individualChannelString, string modelString = "")
```

## Parameters

**siteNumber** [int](#)

The site number. Specify -1 for a system pin.

**pinName** [string](#)

The pin name.

**individualChannelString** [string](#)

The individual channel string.

**modelString** [string](#)

The model string of the owning instrument.

# Field SiteNumberNone

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines the value of site number for a system pin.

```
public const int SiteNumberNone = -1
```

Returns

[int↗](#)

Defines the value of site number for a system pin.

# Property IndividualChannelString

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## IndividualChannelString

The individual channel string in "SMU\_4147\_C1\_S06/1" format.

```
public string IndividualChannelString { get; }
```

### Property Value

[string](#) ↗

### Remarks

This is used to work with site-pin unaware drivers.

# Property ModelString

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ModelString

The model string of the owning instrument of the channel associated with this site-pin pair.

```
public string ModelString { get; }
```

## Property Value

[string](#) ↗

# Property PinName

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PinName

The pin name.

```
public string PinName { get; }
```

## Property Value

[string](#) ↗

# Property SiteNumber

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SiteNumber

The site number.

```
public int SiteNumber { get; }
```

## Property Value

[int](#)

## Remarks

Site number -1 means it's a system pin.

# Property SitePinString

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SitePinString

The site-pin pair in "site0/A" format.

```
public string SitePinString { get; }
```

### Property Value

[string](#) ↗

### Remarks

This is used to work with site-pin aware drivers.

# Class SystemSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines system settings.

```
public static class SystemSettings
```

## Inheritance

[object](#) ← SystemSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

[PowerLineFrequency](#)

The power line frequency of the system.

# Property PowerLineFrequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PowerLineFrequency

The power line frequency of the system.

```
public static double? PowerLineFrequency { get; }
```

### Property Value

[double](#)?

# Class Utilities

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Provides general helper methods.

```
public static class Utilities
```

## Inheritance

[object](#) ← Utilities

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

[ElementAtOrFirst<T>\(T\[\], int\)](#)

Returns the element at a specified index in a sequence or the first element if the index is out of range.

[InvokeInParallel\(params Action\[\]\)](#)

This method calls the Parallel.Invoke method to execute multiple methods in parallel. If any of the methods generates an exception, this method throws the first exception encountered. This allows the TestStand Semiconductor Module runtime error dialog to properly display the exception.

[PerInstrumentPerChannelResultsToPerSitePerPinResults<T>\(IEnumerable<ISessionInformation>, T\[\]\[\]\)](#)

Converts per-instrument per-channel results to per-site per-pin results.

[PerInstrumentPerChannelResultsToPinSiteData<T>\(IEnumerable<ISessionInformation>, T\[\]\[\]\)](#)

Converts per-instrument per-channel results to PinSiteData.

[PreciseWait\(double?\)](#)

This method blocks the thread and waits for the specified amount of time. This method uses the Stopwatch class to support shorter settling times.

[ToJaggedArray<T>\(T\[,\]\)](#)

Converts a two dimensional array to a jagged array.

## [TryDeterminePowerLineFrequency\(ref double\)](#)

Tries to determine the power line frequency of the PXI chassis using the SystemConfiguration API.

# Method ElementAtOrFirst

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ElementAtOrFirst<T>(T[], int)

Returns the element at a specified index in a sequence or the first element if the index is out of range.

```
public static T ElementAtOrFirst<T>(this T[] values, int index)
```

### Parameters

**values** T[]

The array to return an element from.

**index** [int](#)

The zero-based index of the element to retrieve.

### Returns

T

The first element if the index is outside the bounds of the source sequence. Otherwise, the element at the specified position in the source sequence.

### Type Parameters

T

The type of the elements of source array.

# Method InvokeInParallel

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InvokeInParallel(params Action[])

This method calls the Parallel.Invoke method to execute multiple methods in parallel. If any of the methods generates an exception, this method throws the first exception encountered. This allows the TestStand Semiconductor Module runtime error dialog to properly display the exception.

```
public static void InvokeInParallel(params Action[] actions)
```

### Parameters

actions [Action](#)[]

The methods to invoke in parallel.

# Method PerInstrumentPerChannelResultsToPerSitePerPinResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PerInstrumentPerChannelResultsToPerSitePerPinResults<T> (IEnumerable<ISessionInformation>, T[][])

Converts per-instrument per-channel results to per-site per-pin results.

```
public static IDictionary<int, IDictionary<string, T>>
PerInstrumentPerChannelResultsToPerSitePerPinResults<T>(this
IEnumerable<ISessionInformation> allSessionInfo, T[][] perInstrumentPerChannelResults)
```

### Parameters

**allSessionInfo** [IEnumerable](#)<ISessionInformation>

An enumerable of session info.

**perInstrumentPerChannelResults** T[][]

The per-instrument per-channel results to convert.

### Returns

[IDictionary](#)<int, [IDictionary](#)<string, T>>

The converted per-site per-pin results.

### Type Parameters

T

The type of the per-instrument per-channel result.

# Method PerInstrumentPerChannelResultsToPinSiteData

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PerInstrumentPerChannelResultsToPinSiteData<T> (IEnumerable<ISessionInformation>, T[][])

Converts per-instrument per-channel results to PinSiteData.

```
public static PinSiteData<T> PerInstrumentPerChannelResultsToPinSiteData<T>(this  
IEnumerable<ISessionInformation> allSessionInfo, T[][] perInstrumentPerChannelResults)
```

### Parameters

**allSessionInfo** [IEnumerable](#)<ISessionInformation>

An enumerable of session info.

**perInstrumentPerChannelResults** T[][]

The per-instrument per-channel results to convert.

### Returns

[PinSiteData](#)<T>

The converted PinSiteData.

### Type Parameters

T

The type of the per-instrument per-channel result.

# Method PreciseWait

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PreciseWait(double?)

This method blocks the thread and waits for the specified amount of time. This method uses the Stopwatch class to support shorter settling times.

```
public static void PreciseWait(double? timeInSeconds)
```

### Parameters

**timeInSeconds** [double?](#)

The time to precisely wait for.

### Remarks

This method passes through if the time specified is null.

# Method ToJaggedArray

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ToJaggedArray<T>(T[,])

Converts a two dimensional array to a jagged array.

```
public static T[][] ToJaggedArray<T>(this T[,] twoDimensionalArray)
```

### Parameters

**twoDimensionalArray** `T[,]`

The two dimensional array to be converted.

### Returns

`T[][],`

The converted jagged array.

### Type Parameters

**T**

The element type of the array.

# Method TryDeterminePowerLineFrequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.Common](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TryDeterminePowerLineFrequency(ref double)

Tries to determine the power line frequency of the PXI chassis using the SystemConfiguration API.

```
public static void TryDeterminePowerLineFrequency(ref double powerLineFrequency)
```

### Parameters

powerLineFrequency [double](#)

The power line frequency variable to be updated. It falls back to the original value when the try fails.

# Namespace NationalInstruments.SemiconductorTestLibrary.DataAbstraction

## Classes

### [PinSiteData<T>](#)

Defines an object containing values for one or more sites that is associated with a particular pin or set of pins, where T can be passed as any data type.

### [SiteData<T>](#)

Defines an object containing values for one or more sites, where T can be passed as any data type.

### [TSMContextExtensions](#)

Defines extension methods for NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

## Enums

### [ComparisonType](#)

Defines types of comparison results.

# Enum ComparisonType

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines types of comparison results.

```
public enum ComparisonType
```

## Fields

**EqualTo = 0**

The comparison result is equal.

**GreaterThan = 4**

The comparison result is greater than.

**GreaterThanOrEqualTo = 5**

The comparison result is greater than or equal to.

**LessThan = 2**

The comparison result is less than.

**LessThanOrEqualTo = 3**

The comparison result is less than or equal to.

**NotEqualTo = 1**

The comparison result is not equal.

# Class PinSiteData<T>

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object containing values for one or more sites that is associated with a particular pin or set of pins, where T can be passed as any data type.

```
public class PinSiteData<T>
```

## Type Parameters

T

## Inheritance

[object](#) ← PinSiteData<T>

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[PinSiteData\(Dictionary<string, IDictionary<int, T>>\)](#)

Initializes PinSiteData with multiple pins multiple sites data.

[PinSiteData\(string\[\], SiteData<T>\[\]\)](#)

Initializes PinSiteData with pin names and associated SiteData array.

## Properties

[PinNames](#)

Returns an array of pin names associated with current PinSiteData.

[SiteNumbers](#)

Returns an array of sites associated with current PinSiteData.

# Methods

## [Abs\(\)](#)

Performs Math.Abs operation on every element in current PinSiteData.

## [Add\(PinSiteData<T>\)](#)

Performs add operation between every element in current PinSiteData and the given PinSiteData.

## [Add\(SiteData<T>\)](#)

Performs add operation between every element in current PinSiteData and the given SiteData.

## [Add\(T\)](#)

Performs add operation between every element in current PinSiteData and the given value.

## [Compare\(ComparisonType, PinSiteData<T>\)](#)

Performs compare operation between every element in current PinSiteData and the given PinSiteData.

## [Compare\(ComparisonType, SiteData<T>\)](#)

Performs compare operation between every element in current PinSiteData and the given SiteData.

## [Compare\(ComparisonType, T\)](#)

Performs compare operation between every element in current PinSiteData and the given value.

## [Divide\(PinSiteData<T>\)](#)

Performs divide operation between every element in current PinSiteData and the given PinSiteData.

## [Divide\(SiteData<T>\)](#)

Performs divide operation between every element in current PinSiteData and the given SiteData.

## [Divide\(T\)](#)

Performs divide operation between every element in current PinSiteData and the given value.

## [ExtractPin\(string\)](#)

Gets the [SiteData<T>](#) for a given pin.

## [ExtractPins\(string\[\]\)](#)

Gets a new [PinSiteData<T>](#) object for the specified pins.

## [ExtractSite\(int\)](#)

Gets the data for a given site number.

## [GetValue\(int, string\)](#)

Gets the data for a given site number-pin name pair.

## [Invert\(\)](#)

Performs invert operation on every element in current PinSiteData.

#### [Log10\(\)](#)

Performs Math.Log10 operation on every element in current PinSiteData.

#### [Maximum\(PinSiteData<T>\)](#)

Returns the larger one of the element in current PinSiteData and the given PinSiteData.

#### [Maximum\(SiteData<T>\)](#)

Returns the larger one of the element in current PinSiteData and the given SiteData.

#### [Maximum\(T\)](#)

Returns the larger one of the element in current PinSiteData and the given value.

#### [Minimum\(PinSiteData<T>\)](#)

Returns the smaller one of the element in current PinSiteData and the given PinSiteData.

#### [Minimum\(SiteData<T>\)](#)

Returns the smaller one of the element in current PinSiteData and the given SiteData.

#### [Minimum\(T\)](#)

Returns the smaller one of the element in current PinSiteData and the given value.

#### [Multiply\(PinSiteData<T>\)](#)

Performs multiply operation on every element in current PinSiteData and the given PinSiteData.

#### [Multiply\(SiteData<T>\)](#)

Performs multiply operation on every element in current PinSiteData and the given SiteData.

#### [Multiply\(T\)](#)

Performs multiply operation on every element in current PinSiteData and the given value.

#### [Negate\(\)](#)

Returns the negative value of every element in current PinSiteData.

#### [Power\(PinSiteData<T>\)](#)

Raises every element in current PinSiteData to the power of every element in given PinSiteData.

#### [Power\(SiteData<T>\)](#)

Raises every element in current PinSiteData to the power of every element in given SiteData.

#### [Power\(T\)](#)

Raises every element in current PinSiteData to the power of the given value.

#### [SquareRoot\(\)](#)

Returns the square root of every element in current PinSiteData.

[Subtract\(PinSiteData<T>\)](#)

Subtracts every element in given PinSiteData from every element in current PinSiteData.

[Subtract\(SiteData<T>\)](#)

Subtracts every element in given PinSiteData from every element in current SiteData.

[Subtract\(T\)](#)

Subtracts the given value from every element in current PinSiteData.

[Truncate\(\)](#)

Returns integer portion of every element in current PinSiteData.

# Constructor PinSiteData

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PinSiteData(string[], SiteData<T> [])

Initializes PinSiteData with pin names and associated SiteData array.

```
public PinSiteData(string[] pinNames, SiteData<T>[] data)
```

### Parameters

pinNames [string](#)[]

The pin names array.

data [SiteData](#)<T>[]

The per-pin SiteData array.

## PinSiteData(Dictionary<string, IDictionary<int, T>>)

Initializes PinSiteData with multiple pins multiple sites data.

```
public PinSiteData(Dictionary<string, IDictionary<int, T>> pinSiteData)
```

### Parameters

pinSiteData [Dictionary](#)<[string](#), [IDictionary](#)<[int](#), T>>

The multiple pins multiple sites data.

# Property PinNames

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PinNames

Returns an array of pin names associated with current PinSiteData.

```
public string[] PinNames { get; }
```

## Property Value

[string](#) []

# Property SiteNumbers

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SiteNumbers

Returns an array of sites associated with current PinSiteData.

```
public int[] SiteNumbers { get; }
```

## Property Value

[int](#) []

# Method Abs

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Abs()

Performs Math.Abs operation on every element in current PinSiteData.

```
public PinSiteData<T> Abs()
```

## Returns

[PinSiteData<T>](#)

A new PinSiteData that contains absolute value for each element in current PinSiteData.

# Method Add

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Add(T)

Performs add operation between every element in current PinSiteData and the given value.

```
public PinSiteData<T> Add(T value)
```

### Parameters

**value** T

The scalar value to be added.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the result of the addition.

## Add(SiteData<T>)

Performs add operation between every element in current PinSiteData and the given SiteData.

```
public PinSiteData<T> Add(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) to be added.

### Returns

## [PinSiteData<T>](#)

A new PinSiteData that contains the result of the addition.

## Add(PinSiteData<T>)

Performs add operation between every element in current PinSiteData and the given PinSiteData.

```
public PinSiteData<T> Add(PinSiteData<T> other)
```

### Parameters

**other** [PinSiteData<T>](#)

The other [PinSiteData<T>](#) to be added.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the result of the addition.

# Method Compare

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Compare(ComparisonType, T)

Performs compare operation between every element in current PinSiteData and the given value.

```
public PinSiteData<bool> Compare(ComparisonType comparisonType, T value)
```

### Parameters

**comparisonType** [ComparisonType](#)

The comparison type to use.

**value** T

The scalar value to compare to.

### Returns

[PinSiteData<bool>](#)

A new PinSiteData that contains the compare results.

## Compare(ComparisonType, SiteData<T>)

Performs compare operation between every element in current PinSiteData and the given SiteData.

```
public PinSiteData<bool> Compare(ComparisonType comparisonType, SiteData<T> other)
```

### Parameters

**comparisonType** [ComparisonType](#)

The comparison type to use.

**other** [SiteData<T>](#)

The other [SiteData<T>](#) to compare to.

Returns

[PinSiteData<bool>](#)

A new PinSiteData that contains the compare results.

## Compare(ComparisonType, PinSiteData<T>)

Performs compare operation between every element in current PinSiteData and the given PinSiteData.

```
public PinSiteData<bool> Compare(ComparisonType comparisonType, PinSiteData<T> other)
```

Parameters

**comparisonType** [ComparisonType](#)

The comparison type to use.

**other** [PinSiteData<T>](#)

The other [PinSiteData<T>](#) to compare to.

Returns

[PinSiteData<bool>](#)

A new PinSiteData that contains the compare results.

# Method Divide

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Divide(T)

Performs divide operation between every element in current PinSiteData and the given value.

```
public PinSiteData<double> Divide(T value)
```

### Parameters

**value** T

The divisor value.

### Returns

[PinSiteData<double>](#)

A new PinSiteData that contains the results.

## Divide(SiteData<T>)

Performs divide operation between every element in current PinSiteData and the given SiteData.

```
public PinSiteData<double> Divide(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains the divisors.

### Returns

## [PinSiteData<double>](#)

A new PinSiteData that contains the results.

## Divide(PinSiteData<T>)

Performs divide operation between every element in current PinSiteData and the given PinSiteData.

```
public PinSiteData<double> Divide(PinSiteData<T> other)
```

### Parameters

other [PinSiteData<T>](#)

The other [PinSiteData<T>](#) that contains the divisors.

### Returns

[PinSiteData<double>](#)

A new PinSiteData that contains the results.

# Method ExtractPin

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ExtractPin(string)

Gets the [SiteData<T>](#) for a given pin.

```
public SiteData<T> ExtractPin(string pinName)
```

### Parameters

pinName [string](#) ↗

The specified pin name.

### Returns

[SiteData<T>](#)

The [SiteData<T>](#) for the specified pin.

# Method ExtractPins

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ExtractPins(string[])

Gets a new [PinSiteData<T>](#) object for the specified pins.

```
public PinSiteData<T> ExtractPins(string[] pinNames)
```

### Parameters

pinNames [string](#)[]

The specified pin names.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains data for given pins.

# Method ExtractSite

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ExtractSite(int)

Gets the data for a given site number.

```
public IDictionary<string, T> ExtractSite(int siteNumber)
```

### Parameters

**siteNumber** [int](#)

The specified site number.

### Returns

[IDictionary](#)<[string](#), T>

The data for the specified site number.

# Method GetValue

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GetValue(int, string)

Gets the data for a given site number-pin name pair.

```
public T GetValue(int siteNumber, string pinName)
```

### Parameters

**siteNumber** [int](#)

The specified site number.

**pinName** [string](#)

The specified pin name.

### Returns

T

The data for the specified site number and pin name.

# Method Invert

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Invert()

Performs invert operation on every element in current PinSiteData.

```
public PinSiteData<double> Invert()
```

Returns

[PinSiteData<double>](#)

A new PinSiteData that contains the inverted results.

# Method Log10

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Log10()

Performs Math.Log10 operation on every element in current PinSiteData.

```
public PinSiteData<double> Log10()
```

Returns

[PinSiteData<double>](#)

A new PinSiteData that contains the log10 results.

# Method Maximum

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Maximum(T)

Returns the larger one of the element in current PinSiteData and the given value.

```
public PinSiteData<T> Maximum(T value)
```

### Parameters

**value** T

The value to compare.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the larger values.

## Maximum(SiteData<T>)

Returns the larger one of the element in current PinSiteData and the given SiteData.

```
public PinSiteData<T> Maximum(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains the values to compare.

### Returns

## [PinSiteData<T>](#)

A new PinSiteData that contains the larger values.

## Maximum(PinSiteData<T>)

Returns the larger one of the element in current PinSiteData and the given PinSiteData.

```
public PinSiteData<T> Maximum(PinSiteData<T> other)
```

### Parameters

other [PinSiteData<T>](#)

The other [PinSiteData<T>](#) that contains the values to compare.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the larger values.

# Method Minimum

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Minimum(T)

Returns the smaller one of the element in current PinSiteData and the given value.

```
public PinSiteData<T> Minimum(T value)
```

### Parameters

**value** T

The value to compare.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the smaller values.

## Minimum(SiteData<T>)

Returns the smaller one of the element in current PinSiteData and the given SiteData.

```
public PinSiteData<T> Minimum(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains the values to compare.

### Returns

## [PinSiteData<T>](#)

A new PinSiteData that contains the smaller values.

## Minimum(PinSiteData<T>)

Returns the smaller one of the element in current PinSiteData and the given PinSiteData.

```
public PinSiteData<T> Minimum(PinSiteData<T> other)
```

### Parameters

other [PinSiteData<T>](#)

The other [PinSiteData<T>](#) that contains the values to compare.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the smaller values.

# Method Multiply

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Multiply(T)

Performs multiply operation on every element in current PinSiteData and the given value.

```
public PinSiteData<T> Multiply(T value)
```

### Parameters

**value** T

The multiplier.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the multiply results.

## Multiply(SiteData<T>)

Performs multiply operation on every element in current PinSiteData and the given SiteData.

```
public PinSiteData<T> Multiply(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains multipliers.

### Returns

## [PinSiteData<T>](#)

A new PinSiteData that contains the multiply results.

## Multiply(PinSiteData<T>)

Performs multiply operation on every element in current PinSiteData and the given PinSiteData.

```
public PinSiteData<T> Multiply(PinSiteData<T> other)
```

### Parameters

other [PinSiteData<T>](#)

The other [PinSiteData<T>](#) that contains multipliers.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the multiply results.

# Method Negate

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Negate()

Returns the negative value of every element in current PinSiteData.

```
public PinSiteData<T> Negate()
```

Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the negative values.

# Method Power

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Power(T)

Raises every element in current PinSiteData to the power of the given value.

```
public PinSiteData<T> Power(T value)
```

### Parameters

**value** T

The value to use as the power.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the results.

## Power(SiteData<T>)

Raises every element in current PinSiteData to the power of every element in given SiteData.

```
public PinSiteData<T> Power(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains power values.

### Returns

## [PinSiteData<T>](#)

A new PinSiteData that contains the results.

## Power(PinSiteData<T>)

Raises every element in current PinSiteData to the power of every element in given PinSiteData.

```
public PinSiteData<T> Power(PinSiteData<T> other)
```

### Parameters

**other** [PinSiteData<T>](#)

The other [PinSiteData<T>](#) that contains power values.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the results.

# Method SquareRoot

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SquareRoot()

Returns the square root of every element in current PinSiteData.

```
public PinSiteData<double> SquareRoot()
```

Returns

[PinSiteData<double>](#)

A new PinSiteData that contains the square root values.

# Method Subtract

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Subtract(T)

Subtracts the given value from every element in current PinSiteData.

```
public PinSiteData<T> Subtract(T value)
```

### Parameters

**value** T

The value to subtract.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the results.

## Subtract(SiteData<T>)

Subtracts every element in given PinSiteData from every element in current SiteData.

```
public PinSiteData<T> Subtract(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains values to subtract.

### Returns

## [PinSiteData<T>](#)

A new PinSiteData that contains the results.

## Subtract(PinSiteData<T>)

Subtracts every element in given PinSiteData from every element in current PinSiteData.

```
public PinSiteData<T> Subtract(PinSiteData<T> other)
```

### Parameters

other [PinSiteData<T>](#)

The other [PinSiteData<T>](#) that contains values to subtract.

### Returns

[PinSiteData<T>](#)

A new PinSiteData that contains the results.

# Method Truncate

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Truncate()

Returns integer portion of every element in current PinSiteData.

```
public PinSiteData<T> Truncate()
```

Returns

[PinSiteData<T>](#)

A new PinSiteData that contains integer values.

# Class SiteData<T>

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object containing values for one or more sites, where T can be passed as any data type.

```
public class SiteData<T>
```

## Type Parameters

T

The data type of the element data.

## Inheritance

[object](#) ← SiteData<T>

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

[SiteData\(IDictionary<int, T>\)](#)

Initializes SiteData with per-pin multiple sites data.

[SiteData\(int\[\], T\)](#)

Initializes a SiteData object with the same data value across all active sites. Each element in the siteNumbers array represents an active site number/value.

[SiteData\(T\[\]\)](#)

Initializes SiteData with multiple sites data array. The array index represents site number.

# Properties

[SiteNumbers](#)

Returns an array of sites associated with current SiteData.

# Methods

## [Abs\(\)](#)

Performs Math.Abs operation on every element in current SiteData.

## [Add\(SiteData<T>\)](#)

Performs add operation between every element in current SiteData and the given SiteData.

## [Add\(T\)](#)

Performs add operation between every element in current SiteData and the given value.

## [Compare\(ComparisonType, SiteData<T>\)](#)

Performs compare operation between every element in current SiteData and the given SiteData.

## [Compare\(ComparisonType, T\)](#)

Performs compare operation between every element in current SiteData and the given value.

## [Divide\(SiteData<T>\)](#)

Performs divide operation between every element in current SiteData and the given SiteData.

## [Divide\(T\)](#)

Performs divide operation between every element in current SiteData and the given value.

## [GetValue\(int\)](#)

Gets the data for a given site number.

## [Invert\(\)](#)

Performs invert operation on every element in current SiteData.

## [Log10\(\)](#)

Performs Math.Log10 operation on every element in current SiteData.

## [Maximum\(SiteData<T>\)](#)

Returns the larger of the element in current SiteData and the given SiteData.

## [Maximum\(T\)](#)

Returns the larger of the element in current SiteData and the given value.

## [Minimum\(SiteData<T>\)](#)

Returns the smaller of the element in current SiteData and the given SiteData.

## [Minimum\(T\)](#)

Returns the smaller of the element in current SiteData and the given value.

## [Multiply\(SiteData<T>\)](#)

Performs multiply operation on every element in current SiteData and the given SiteData.

### Multiply(T)

Performs multiply operation on every element in current SiteData and the given value.

### Negate()

Returns the negative value of every element in current SiteData.

### Power(SiteData <T>)

Raises every element in current SiteData to the power of every element in given SiteData.

### Power(T)

Raises every element in current SiteData to the power of the given value.

### SquareRoot()

Returns the square root of every element in current SiteData.

### Subtract(SiteData <T>)

Subtracts every element in given SiteData from every element in current SiteData.

### Subtract(T)

Subtracts the given value from every element in current SiteData.

### Truncate()

Returns integer portion of every element in current SiteData.

# Constructor SiteData

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SiteData(T[])

Initializes SiteData with multiple sites data array. The array index represents site number.

```
public SiteData(T[] data)
```

### Parameters

**data** T[]

The data array.

## SiteData(IDictionary<int, T>)

Initializes SiteData with per-pin multiple sites data.

```
public SiteData(IDictionary<int, T> data)
```

### Parameters

**data** [IDictionary](#)<[int](#), T>

The per-pin multiple sites data.

## SiteData(int[], T)

Initializes a SiteData object with the same data value across all active sites. Each element in the siteNumbers array represents an active site number/value.

```
public SiteData(int[] siteNumbers, T data)
```

## Parameters

**siteNumbers** [int\[\]](#)

The number of active sites.

**data** T

The data.

# Property SiteNumbers

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SiteNumbers

Returns an array of sites associated with current SiteData.

```
public int[] SiteNumbers { get; }
```

## Property Value

[int](#) []

# Method Abs

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Abs()

Performs Math.Abs operation on every element in current SiteData.

```
public SiteData<T> Abs()
```

## Returns

[SiteData](#)<T>

A new SiteData that contains absolute value for each element in current SiteData.

# Method Add

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Add(T)

Performs add operation between every element in current SiteData and the given value.

```
public SiteData<T> Add(T value)
```

### Parameters

**value** T

The scalar value to be added.

### Returns

[SiteData<T>](#)

A new SiteData that contains the sum of the addition.

## Add(SiteData<T>)

Performs add operation between every element in current SiteData and the given SiteData.

```
public SiteData<T> Add(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) to be added.

### Returns

## SiteData<T>

A new SiteData that contains the sum of the addition.

# Method Compare

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Compare(ComparisonType, T)

Performs compare operation between every element in current SiteData and the given value.

```
public SiteData<bool> Compare(ComparisonType comparisonType, T value)
```

### Parameters

**comparisonType** [ComparisonType](#)

The comparison type to use.

**value** T

The scalar value to compare.

### Returns

[SiteData<bool>](#)

A new SiteData that contains the compare results.

## Compare(ComparisonType, SiteData<T>)

Performs compare operation between every element in current SiteData and the given SiteData.

```
public SiteData<bool> Compare(ComparisonType comparisonType, SiteData<T> other)
```

### Parameters

**comparisonType** [ComparisonType](#)

The comparison type to use.

**other** [SiteData<T>](#)

The other [SiteData<T>](#) to compare.

Returns

[SiteData<bool>](#)

A new SiteData that contains the compare results.

# Method Divide

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Divide(T)

Performs divide operation between every element in current SiteData and the given value.

```
public SiteData<double> Divide(T value)
```

### Parameters

**value T**

The divisor value.

### Returns

[SiteData<double>](#)

A new SiteData that contains the results.

## Divide(SiteData<T>)

Performs divide operation between every element in current SiteData and the given SiteData.

```
public SiteData<double> Divide(SiteData<T> other)
```

### Parameters

**other [SiteData<T>](#)**

The other [SiteData<T>](#) that contains the divisors.

### Returns

## [SiteData<double>](#)

A new SiteData that contains the results.

# Method GetValue

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GetValue(int)

Gets the data for a given site number.

```
public T GetValue(int siteNumber)
```

### Parameters

siteNumber [int](#)

The site number.

### Returns

T

Single site data.

# Method Invert

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Invert()

Performs invert operation on every element in current SiteData.

```
public SiteData<double> Invert()
```

Returns

[SiteData<double>](#)

A new SiteData that contains the inverted results.

# Method Log10

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Log10()

Performs Math.Log10 operation on every element in current SiteData.

```
public SiteData<double> Log10()
```

Returns

[SiteData<double>](#)

A new SiteData that contains the log10 results.

# Method Maximum

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Maximum(T)

Returns the larger of the element in current SiteData and the given value.

```
public SiteData<T> Maximum(T value)
```

### Parameters

**value** T

The value to compare.

### Returns

[SiteData<T>](#)

A new SiteData that contains the larger values.

## Maximum(SiteData<T>)

Returns the larger of the element in current SiteData and the given SiteData.

```
public SiteData<T> Maximum(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains the values to compare.

### Returns

## SiteData<T>

A new SiteData that contains the larger values.

# Method Minimum

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Minimum(T)

Returns the smaller of the element in current SiteData and the given value.

```
public SiteData<T> Minimum(T value)
```

### Parameters

**value** T

The value to compare.

### Returns

[SiteData<T>](#)

A new SiteData that contains the smaller values.

## Minimum(SiteData<T>)

Returns the smaller of the element in current SiteData and the given SiteData.

```
public SiteData<T> Minimum(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains the values to compare.

### Returns

## SiteData<T>

A new SiteData that contains the smaller values.

# Method Multiply

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Multiply(T)

Performs multiply operation on every element in current SiteData and the given value.

```
public SiteData<T> Multiply(T value)
```

### Parameters

**value** T

The multiplier.

### Returns

[SiteData<T>](#)

A new SiteData that contains the multiply results.

## Multiply(SiteData<T>)

Performs multiply operation on every element in current SiteData and the given SiteData.

```
public SiteData<T> Multiply(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains multipliers.

### Returns

## SiteData<T>

A new SiteData that contains the multiply results.

# Method Negate

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Negate()

Returns the negative value of every element in current SiteData.

```
public SiteData<T> Negate()
```

Returns

[SiteData](#)<T>

A new SiteData that contains the negative values.

# Method Power

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Power(T)

Raises every element in current SiteData to the power of the given value.

```
public SiteData<T> Power(T value)
```

### Parameters

**value** T

The value to use as the power.

### Returns

[SiteData<T>](#)

A new SiteData that contains the results.

## Power(SiteData<T>)

Raises every element in current SiteData to the power of every element in given SiteData.

```
public SiteData<T> Power(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains power values.

### Returns

## SiteData<T>

A new SiteData that contains the results.

# Method SquareRoot

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SquareRoot()

Returns the square root of every element in current SiteData.

```
public SiteData<double> SquareRoot()
```

Returns

[SiteData<double>](#)

A new SiteData that contains the square root values.

# Method Subtract

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Subtract(T)

Subtracts the given value from every element in current SiteData.

```
public SiteData<T> Subtract(T value)
```

### Parameters

**value** T

The value to subtract.

### Returns

[SiteData<T>](#)

A new SiteData that contains the results.

## Subtract(SiteData<T>)

Subtracts every element in given SiteData from every element in current SiteData.

```
public SiteData<T> Subtract(SiteData<T> other)
```

### Parameters

**other** [SiteData<T>](#)

The other [SiteData<T>](#) that contains values to subtract.

### Returns

## SiteData<T>

A new SiteData that contains the results.

# Method Truncate

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Truncate()

Returns integer portion of every element in current SiteData.

```
public SiteData<T> Truncate()
```

Returns

[SiteData](#)<T>

A new SiteData that contains integer values.

# Class TSMContextExtensions

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines extension methods for `NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext`.

```
public static class TSMContextExtensions
```

## Inheritance

[object](#) ← TSMContextExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[NewSiteData<T>\(ISemiconductorModuleContext, T\)](#).

Creates a new [SiteData<T>](#) object with the same value for all sites.

[NewSiteData<T>\(ISemiconductorModuleContext, T\[\]\)](#).

Creates a new [SiteData<T>](#) object with unique values for all sites.

# Method NewSiteData

Namespace: [NationalInstruments.SemiconductorTestLibrary.DataAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## NewSiteData<T>(ISemiconductorModuleContext, T)

Creates a new [SiteData<T>](#) object with the same value for all sites.

```
public static SiteData<T> NewSiteData<T>(this ISemiconductorModuleContext tsmContext,  
T value)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**value** T

Value to apply to all sites.

### Returns

[SiteData<T>](#)

A new [SiteData<T>](#) object.

### Type Parameters

T

Data Type of value to use with [SiteData<T>](#) object.

### Remarks

Example Usage:

```
var data = tsmContext.NewSiteData(4.0);
```

## NewSiteData<T>(ISemiconductorModuleContext, T[])

Creates a new [SiteData<T>](#) object with unique values for all sites.

```
public static SiteData<T> NewSiteData<T>(this ISemiconductorModuleContext tsmContext,  
T[] values)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**values** T[]

Values to apply to each site.

### Returns

[SiteData<T>](#)

A new [SiteData<T>](#) object.

### Type Parameters

T

Data Type of value to use with [SiteData<T>](#) object.

### Remarks

Note: the number of elements in value must equal to number of sites in NationalInstruments.TestStand. SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

### Example Usage:

```
var data = tsmContext.NewSiteData(new double[] { 1.0, 2.0, 3.0, 4.0, 5.0 });
```



# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction

## Classes

### [Initialization](#)

Defines entry points for semiconductor test steps.

### [TSMSessionManager](#)

Defines a session manager that manages device sessions through TSM.

## Interfaces

### [ISessionInformation](#)

Defines an interface for related information associated with an individual driver session.

### [ISessionsBundle<TSessionInformation>](#)

Defines an interface for bundling together multiple device session information.

# Interface ISessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an interface for related information associated with an individual driver session.

```
public interface ISessionInformation
```

## Remarks

Only the related information is defined by this interface. The driver-specific session should be declared within the class that implements this interface.

## Properties

### [AssociatedSitePinList](#)

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

```
IList<SitePinInfo> AssociatedSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Interface ISessionsBundle<TSessionInformation>

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an interface for bundling together multiple device session information.

```
public interface ISessionsBundle<TSessionInformation>
```

## Type Parameters

### TSessionInformation

The type of the information for a specific device session.

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\]>, string\)](#),  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#),  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#),  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#),  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>>, string, string\)](#),  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#),  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#),  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, int, TResult\[\]>\)](#),

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[,\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, SitePinInfo>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int, SitePinInfo>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation>\)](#),  
[Publish.PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\], string\)](#),  
[Publish.PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\]\[\], string\)](#).

## Properties

### [AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

### [InstrumentSessions](#)

An enumerable of [ISessionInformation](#).

### [TSMContext](#)

The associated `NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext`.

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

`IList<SitePinInfo> AggregateSitePinList { get; }`

Property Value

[`IList<SitePinInfo>`](#)

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [ISessionInformation](#).

```
IEnumerable<TSessionInformation> InstrumentSessions { get; }
```

### Property Value

[IEnumerable](#) <TSessionInformation>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
ISemiconductorModuleContext TSMContext { get; }
```

### Property Value

ISemiconductorModuleContext

# Class Initialization

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines entry points for semiconductor test steps.

```
public static class Initialization
```

## Inheritance

[object](#) ← Initialization

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[MapInitializedInstrumentSessions\(I SemiconductorModuleContext\)](#)

Maps already initialized instrument sessions.

# Method MapInitializedInstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## MapInitializedInstrumentSessions(ISemiconductorModuleContext)

Maps already initialized instrument sessions.

```
public static void MapInitializedInstrumentSessions(ISemiconductorModuleContext tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

### Remarks

Must be called when instruments initialization completes. This ensures any cached instrument session information is consistent between the NationalInstruments.SemiconductorTestLibrary.TestStandSteps assembly and the client assembly, as they may not share the same memory space.

# Class TMSessionManager

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines a session manager that manages device sessions through TSM.

```
public class TMSessionManager
```

## Inheritance

[object](#) ← TMSessionManager

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[TMSessionManager\(ISemiconductorModuleContext\)](#)

Constructs a TSM session manager.

## Methods

[DAQmx\(string\)](#)

Gets the [DAQmxTasksBundle](#) associated with the specified pin.

[DAQmx\(string\[\]\)](#)

Gets the [DAQmxTasksBundle](#) associated with the specified pins.

[DCPower\(string\)](#)

Gets the [DCPowerSessionsBundle](#) associated with the specified pin.

[DCPower\(string\[\]\)](#)

Gets the [DCPowerSessionsBundle](#) associated with the specified pins.

[DMM\(string\)](#)

Gets the [DMMSessionsBundle](#) associated with the specified pin.

## DMM(string[])

Gets the [DMMSessionsBundle](#) associated with the specified pins.

## Digital(string)

Gets the [DigitalSessionsBundle](#) associated with the specified pin.

## Digital(string[])

Gets the [DigitalSessionsBundle](#) associated with the specified pins.

## Egen(string)

Gets the [EgenSessionsBundle](#) associated with the specified pin.

## Egen(string[])

Gets the [EgenSessionsBundle](#) associated with the specified pins.

## Scope(string)

Gets the [ScopeSessionsBundle](#) associated with the specified pin.

## Scope(string[])

Gets the [ScopeSessionsBundle](#) associated with the specified pins.

## Sync()

Gets the [SyncSessionsBundle](#) associated with all pins.

## Sync(string)

Gets the [SyncSessionsBundle](#) associated with the specified pin.

## Sync(string[])

Gets the [SyncSessionsBundle](#) associated with the specified pins.

# Constructor TSMSessionManager

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMSessionManager(ISemiconductorModuleContext)

Constructs a TSM session manager.

```
public TSMSessionManager(ISemiconductorModuleContext tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context that associates with this manager.

# Method DAQmx

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DAQmx(string)

Gets the [DAQmxTasksBundle](#) associated with the specified pin.

```
public DAQmxTasksBundle DAQmx(string pin)
```

### Parameters

**pin** [string](#)

The given pin.

### Returns

[DAQmxTasksBundle](#)

The associated [DAQmxTasksBundle](#).

## DAQmx(string[])

Gets the [DAQmxTasksBundle](#) associated with the specified pins.

```
public DAQmxTasksBundle DAQmx(string[] pins)
```

### Parameters

**pins** [string](#)[]

The given pins.

### Returns

## [DAQmxTasksBundle](#)

The associated [DAQmxTasksBundle](#).

# Method DCPower

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DCPower(string)

Gets the [DCPowerSessionsBundle](#) associated with the specified pin.

```
public DCPowerSessionsBundle DCPower(string pin)
```

### Parameters

**pin** [string](#)

The given pin.

### Returns

[DCPowerSessionsBundle](#)

The associated [DCPowerSessionsBundle](#).

## DCPower(string[])

Gets the [DCPowerSessionsBundle](#) associated with the specified pins.

```
public DCPowerSessionsBundle DCPower(string[] pins)
```

### Parameters

**pins** [string](#)[]

The given pins.

### Returns

## [DCPowerSessionsBundle](#)

The associated [DCPowerSessionsBundle](#).

# Method DMM

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DMM(string)

Gets the [DMMSessionsBundle](#) associated with the specified pin.

```
public DMMSessionsBundle DMM(string pin)
```

### Parameters

**pin** [string](#)

The given pin.

### Returns

[DMMSessionsBundle](#)

The associated [DMMSessionsBundle](#).

## DMM(string[])

Gets the [DMMSessionsBundle](#) associated with the specified pins.

```
public DMMSessionsBundle DMM(string[] pins)
```

### Parameters

**pins** [string](#)[]

The given pins.

### Returns

## [DMMSessionsBundle](#)

The associated [DMMSessionsBundle](#).

# Method Digital

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Digital(string)

Gets the [DigitalSessionsBundle](#) associated with the specified pin.

```
public DigitalSessionsBundle Digital(string pin)
```

### Parameters

**pin** [string](#)

The given pin.

### Returns

[DigitalSessionsBundle](#)

The associated [DigitalSessionsBundle](#).

## Digital(string[])

Gets the [DigitalSessionsBundle](#) associated with the specified pins.

```
public DigitalSessionsBundle Digital(string[] pins)
```

### Parameters

**pins** [string](#)[]

The given pins.

### Returns

## [DigitalSessionsBundle](#)

The associated [DigitalSessionsBundle](#).

# Method Fgen

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Fgen(string)

Gets the [EgenSessionsBundle](#) associated with the specified pin.

```
public EgenSessionsBundle Fgen(string pin)
```

### Parameters

**pin** [string](#)

The given pin.

### Returns

[EgenSessionsBundle](#)

The associated [EgenSessionsBundle](#).

## Fgen(string[])

Gets the [EgenSessionsBundle](#) associated with the specified pins.

```
public EgenSessionsBundle Fgen(string[] pins)
```

### Parameters

**pins** [string](#)[]

The given pins.

### Returns

## [EgenSessionsBundle](#)

The associated [EgenSessionsBundle](#).

# Method Scope

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Scope(string)

Gets the [ScopeSessionsBundle](#) associated with the specified pin.

```
public ScopeSessionsBundle Scope(string pin)
```

### Parameters

**pin** [string](#)

The given pin.

### Returns

[ScopeSessionsBundle](#)

The associated [ScopeSessionsBundle](#).

## Scope(string[])

Gets the [ScopeSessionsBundle](#) associated with the specified pins.

```
public ScopeSessionsBundle Scope(string[] pins)
```

### Parameters

**pins** [string](#)[]

The given pins.

### Returns

## [ScopeSessionsBundle](#)

The associated [ScopeSessionsBundle](#).

# Method Sync

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Sync()

Gets the [SyncSessionsBundle](#) associated with all pins.

```
public SyncSessionsBundle Sync()
```

Returns

[SyncSessionsBundle](#)

The associated [SyncSessionsBundle](#).

## Sync(string)

Gets the [SyncSessionsBundle](#) associated with the specified pin.

```
public SyncSessionsBundle Sync(string pin)
```

Parameters

**pin** [string](#) ↗

The given pin.

Returns

[SyncSessionsBundle](#)

The associated [SyncSessionsBundle](#).

## Sync(string[])

Gets the [SyncSessionsBundle](#) associated with the specified pins.

```
public SyncSessionsBundle Sync(string[] pins)
```

## Parameters

**pins** [string](#)[]

The given pins.

## Returns

[SyncSessionsBundle](#)

The associated [SyncSessionsBundle](#).

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx

## Classes

### [AOFunctionGenerationSettings](#)

The class is used to configure the settings for an Analog Output Function Generation DAQmx task.

### [AnalogInput](#)

Defines operations on analog input channels.

### [AnalogOutput](#)

Defines operations on analog output channels.

### [AnalogOutputFunctionGeneration](#)

Defines operations on analog output channels configured under the Analog Output: Function Generation task type.

### [Configure](#)

Defines operations for configuring timing and other task properties.

### [DAQmxTaskInformation](#)

Defines an object that contains an individual NI-DAQmx task and its related information.

### [DAQmxTasksBundle](#)

Defines an object that contains one or more [DAQmxTaskInformation](#) objects.

### [DAQmxTimingSampleClockSettings](#)

The class is used to configure the timing settings for a DAQmx task.

### [DigitalInput](#)

Defines operations on digital input channels.

### [DigitalOutput](#)

Defines operations on digital output channels.

### [InitializeAndClose](#)

Defines NI-DAQmx NationalInstruments.DAQmx.Tasks initialize and close APIs.

### [SampleValuesCacher<T>](#)

Defines a class that caches sample values.

## [TaskControl](#)

Defines operations for performing low-level task control.

## [TriggersAndEvents](#)

Defines operations for configuring triggers.

## **Structs**

### [DefaultDAQmxTaskTypeStrings](#)

Defines the default DAQmx task type strings.

## **Enums**

### [DAQmxTaskType](#)

Defines the type of the DAQmx task.

### [DAQmxTerminalConfiguration](#)

Defines terminal configurations for analog input and output channels.

# Class AOFunctionGenerationSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

The class is used to configure the settings for an Analog Output Function Generation DAQmx task.

```
public class AOFunctionGenerationSettings
```

## Inheritance

[object](#) ← AOFunctionGenerationSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

Using this class is adventegous as it allows the user to set one or more properties in a single call.

## Constructors

[AOFunctionGenerationSettings\(\)](#)

## Properties

### [Amplitude](#)

Specifies the amplitude of the waveform to generate.

### [DutyCycle](#)

Specifies the duty cycle of the waveform. Only applicable when the FunctionType is set to AOFunctionGenerationType.Square.

### [Frequency](#)

Specifies the frequency of the waveform to generate.

### [FunctionType](#)

Specifies the kind of the waveform to generate (sine, square, triangle, or sawtooth).

## Offset

Specifies the offset of the waveform to generate.

## StartPhase

Specifies the start phase of the waveform to generate.

# Constructor AOFunctionGenerationSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## AOFunctionGenerationSettings()

```
public AOFunctionGenerationSettings()
```

# Property Amplitude

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Amplitude

Specifies the amplitude of the waveform to generate.

```
public double? Amplitude { get; set; }
```

## Property Value

[double](#)?

## Remarks

This value is used to get/set the value of the  
NationalInstruments.DAQmx.AOChannel.FunctionGenerationAmplitude.

# Property DutyCycle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DutyCycle

Specifies the duty cycle of the waveform. Only applicable when the FunctionType is set to AOFunctionGenerationType.Square.

```
public double? DutyCycle { get; set; }
```

### Property Value

[double](#)?

### Remarks

This value is used to get/set the value of the  
NationalInstruments.DAQmx.AOChannel.FunctionGenerationDutyCycle.

# Property Frequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Frequency

Specifies the frequency of the waveform to generate.

```
public double? Frequency { get; set; }
```

### Property Value

[double](#)?

### Remarks

This value is used to get/set the value of the  
NationalInstruments.DAQmx.AOChannel.FunctionGenerationFrequency.

# Property FunctionType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## FunctionType

Specifies the kind of the waveform to generate (sine, square, triangle, or sawtooth).

```
public AOFunctionGenerationType? FunctionType { get; set; }
```

### Property Value

AOFunctionGenerationType?

### Remarks

This value is used to get/set the value of the  
NationalInstruments.DAQmx.AOChannel.FunctionGenerationType.

# Property Offset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Offset

Specifies the offset of the waveform to generate.

```
public double? Offset { get; set; }
```

## Property Value

[double](#)?

## Remarks

This value is used to get/set the value of the  
NationalInstruments.DAQmx.AOChannel.FunctionGenerationOffset.

# Property StartPhase

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## StartPhase

Specifies the start phase of the waveform to generate.

```
public double? StartPhase { get; set; }
```

## Property Value

[double](#)?

## Remarks

This value is used to get/set the value of the  
NationalInstruments.DAQmx.AOChannel.FunctionGenerationStartPhase.

# Class AnalogInput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations on analog input channels.

```
public static class AnalogInput
```

## Inheritance

[object](#) ← AnalogInput

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ReadAnalogMultiSample\(DAQmxTasksBundle, int\)](#)

Reads multiple samples and returns pin- and site-aware objects of an array of doubles, where each element in the array represents one sample read. By default, the value of samplesToRead is -1. In this case, all available samples are read when this method is invoked.

[ReadAnalogWaveform\(DAQmxTasksBundle, int\)](#)

Reads the samples and returns pin- and site-aware objects of double AnalogWaveform, where each double value in the waveform represents one sample read. By default, all available samples are returned, unless otherwise specified by passing in value via the samplesToRead argument.

# Method ReadAnalogMultiSample

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadAnalogMultiSample(DAQmxTasksBundle, int)

Reads multiple samples and returns pin- and site-aware objects of an array of doubles, where each element in the array represents one sample read. By default, the value of samplesToRead is -1. In this case, all available samples are read when this method is invoked.

```
public static PinSiteData<double[]> ReadAnalogMultiSample(this DAQmxTasksBundle tasksBundle,  
    int samplesToRead = -1)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**samplesToRead** [int](#)

The number of samples to read.

### Returns

[PinSiteData<double\[\]>](#)

Per-site per-pin floating-point samples.

# Method ReadAnalogWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadAnalogWaveform(DAQmxTasksBundle, int)

Reads the samples and returns pin- and site-aware objects of double AnalogWaveform, where each double value in the waveform represents one sample read. By default, all available samples are returned, unless otherwise specified by passing in value via the samplesToRead argument.

```
public static PinSiteData<AnalogWaveform<double>> ReadAnalogWaveform(this DAQmxTasksBundle tasksBundle, int samplesToRead = -1)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**samplesToRead** [int](#)

The number of samples to read.

### Returns

[PinSiteData](#)<AnalogWaveform<[double](#)>>

Per-site per-pin analog waveform samples.

# Class AnalogOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations on analog output channels.

```
public static class AnalogOutput
```

## Inheritance

[object](#) ← AnalogOutput

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[WriteAnalogSingleSample\(DAQmxTasksBundle, PinSiteData<double>, bool\)](#)

Writes a static DC output state to the pin(s).

[WriteAnalogSingleSample\(DAQmxTasksBundle, SiteData<double>, bool\)](#)

Writes a static DC output state to the pin(s).

[WriteAnalogSingleSample\(DAQmxTasksBundle, double, bool\)](#)

Writes a static DC output state to the pin(s).

[WriteAnalogWaveform\(DAQmxTasksBundle, AnalogWaveform<double>, bool\)](#)

Writes the specified waveform to output on the pin(s).

[WriteAnalogWaveform\(DAQmxTasksBundle, PinSiteData<AnalogWaveform<double>>, bool\)](#)

Writes the specified waveform to output on the pin(s).

[WriteAnalogWaveform\(DAQmxTasksBundle, SiteData<AnalogWaveform<double>>, bool\)](#)

Writes the specified waveform to output on the pin(s).

# Method WriteAnalogSingleSample

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteAnalogSingleSample(DAQmxTasksBundle, double, bool)

Writes a static DC output state to the pin(s).

```
public static void WriteAnalogSingleSample(this DAQmxTasksBundle tasksBundle, double  
staticState, bool autoStart = true)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**staticState** [double](#)

The static state to write.

**autoStart** [bool](#)

Specifies whether to automatically start the tasks.

## WriteAnalogSingleSample(DAQmxTasksBundle, SiteData<double>, bool)

Writes a static DC output state to the pin(s).

```
public static void WriteAnalogSingleSample(this DAQmxTasksBundle tasksBundle,  
SiteData<double> siteData, bool autoStart = true)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**siteData** [SiteData<double>](#)

The per-site data to write.

**autoStart** [bool](#)

Specifies whether to automatically start the tasks.

## WriteAnalogSingleSample(DAQmxTasksBundle, PinSiteData<double>, bool)

Writes a static DC output state to the pin(s).

```
public static void WriteAnalogSingleSample(this DAQmxTasksBundle tasksBundle,  
PinSiteData<double> pinSiteData, bool autoStart = true)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**pinSiteData** [PinSiteData<double>](#)

The per-site per-pin data to write.

**autoStart** [bool](#)

Specifies whether to automatically start the tasks.

# Method WriteAnalogWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteAnalogWaveform(DAQmxTasksBundle, AnalogWaveform<double>, bool)

Writes the specified waveform to output on the pin(s).

```
public static void WriteAnalogWaveform(this DAQmxTasksBundle tasksBundle,  
AnalogWaveform<double> waveform, bool autoStart = true)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**waveform** AnalogWaveform<[double](#)>

The waveform to write.

**autoStart** [bool](#)

Specifies whether to automatically start the tasks.

## WriteAnalogWaveform(DAQmxTasksBundle, SiteData<AnalogWaveform<double>>, bool)

Writes the specified waveform to output on the pin(s).

```
public static void WriteAnalogWaveform(this DAQmxTasksBundle tasksBundle,  
SiteData<AnalogWaveform<double>> siteData, bool autoStart = true)
```

### Parameters

## **tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

## **siteData** [SiteData](#)<AnalogWaveform<[double](#)>>

The per-site waveform to write.

## **autoStart** [bool](#)

Specifies whether to automatically start the tasks.

# WriteAnalogWaveform(DAQmxTasksBundle, PinSiteData<AnalogWaveform<[double](#)>>, bool)

Writes the specified waveform to output on the pin(s).

```
public static void WriteAnalogWaveform(this DAQmxTasksBundle tasksBundle,  
PinSiteData<AnalogWaveform<double>> pinSiteData, bool autoStart = true)
```

## Parameters

### **tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### **pinSiteData** [PinSiteData](#)<AnalogWaveform<[double](#)>>

The per-site per-pin waveform to write.

### **autoStart** [bool](#)

Specifies whether to automatically start the tasks.

# Class AnalogOutputFunctionGeneration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations on analog output channels configured under the Analog Output: Function Generation task type.

```
public static class AnalogOutputFunctionGeneration
```

## Inheritance

[object](#) ← AnalogOutputFunctionGeneration

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ConfigureAOFunctionGeneration\(DAQmxTasksBundle, AOFunctionGenerationSettings\)](#)

Configures the analog output function generation for the task (also known as PureTune).

[StartAOFunctionGeneration\(DAQmxTasksBundle\)](#)

Transitions the task to the running state to begin the analog output function generation. This method will wait for the appropriate amount of settling time required by the instrument.

# Method ConfigureAOFunctionGeneration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureAOFunctionGeneration(DAQmxTasksBundle, AOFunctionGenerationSettings)

Configures the analog output function generation for the task (also known as PureTune).

```
public static void ConfigureAOFunctionGeneration(this DAQmxTasksBundle tasksBundle,  
AOFunctionGenerationSettings functionGenerationSettings)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**functionGenerationSettings** [AOFunctionGenerationSettings](#)

The function generation settings.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method StartAOFUNCTIONGeneration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## StartAOFUNCTIONGeneration(DAQmxTasksBundle)

Transitions the task to the running state to begin the analog output function generation. This method will wait for the appropriate amount of settling time required by the instrument.

```
public static void StartAOFUNCTIONGeneration(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Exceptions

DaqException

The underling driver session returned an error.

# Class Configure

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations for configuring timing and other task properties.

```
public static class Configure
```

## Inheritance

[object](#) ← Configure

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ConfigureAITerminalConfiguration\(DAQmxTasksBundle, AITerminalConfiguration\)](#)

Configures the terminal configuration for the analog input channels within the task. Refer to National Instruments.DAQmx.AITerminalConfiguration for more information.

[ConfigureAOTerminalConfiguration\(DAQmxTasksBundle, AOTerminalConfiguration\)](#)

Configures the terminal configuration for the analog output channels within the task, refer to National Instruments.DAQmx.AOTerminalConfiguration for more information.

[ConfigureTiming\(DAQmxTasksBundle, DAQmxTimingSampleClockSettings\)](#)

Configures the timing for the task. The capture mode (FiniteSamples/ContinuousSamples), rate of the Sample Clock, and the number of samples to acquire or generate.

[GetSampleClockRateDistinct\(DAQmxTasksBundle\)](#)

Specifies the sampling rate in samples per channel per second. If you use an external source for the Sample Clock, set this input to the maximum expected rate of that clock./>

[GetSampleClockRates\(DAQmxTasksBundle\)](#)

Gets the actual sample clock rate (Hz).

# Method ConfigureAITerminalConfiguration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureAITerminalConfiguration(DAQmxTasksBundle, AITerminalConfiguration)

Configures the terminal configuration for the analog input channels within the task. Refer to National Instruments.DAQmx.AITerminalConfiguration for more information.

```
public static void ConfigureAITerminalConfiguration(this DAQmxTasksBundle tasksBundle,  
AITerminalConfiguration terminalConfiguration)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**terminalConfiguration** AITerminalConfiguration

Specifies the terminal configuration mode.

### Exceptions

DaqException

The underling driver session returned an error.

# Method ConfigureAOTerminalConfiguration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureAOTerminalConfiguration(DAQmxTasksBundle, AOTerminalConfiguration)

Configures the terminal configuration for the analog output channels within the task, refer to National Instruments.DAQmx.AOTerminalConfiguration for more information.

```
public static void ConfigureAOTerminalConfiguration(this DAQmxTasksBundle tasksBundle,  
AOTerminalConfiguration terminalConfiguration)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**terminalConfiguration** AOTerminalConfiguration

Specifies the terminal configuration mode.

### Exceptions

DaqException

The underling driver session returned an error.

# Method ConfigureTiming

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTiming(DAQmxTasksBundle, DAQmxTimingSampleClockSettings)

Configures the timing for the task. The capture mode (FiniteSamples/ContinuousSamples), rate of the Sample Clock, and the number of samples to acquire or generate.

```
public static void ConfigureTiming(this DAQmxTasksBundle tasksBundle,  
DAQmxTimingSampleClockSettings timingSettings)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**timingSettings** [DAQmxTimingSampleClockSettings](#)

Specifies the timing settings.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method GetSampleClockRateDistinct

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetSampleClockRateDistinct(DAQmxTasksBundle)

Specifies the sampling rate in samples per channel per second. If you use an external source for the Sample Clock, set this input to the maximum expected rate of that clock./>

```
public static double GetSampleClockRateDistinct(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Returns

[double](#) ↗

Sample clock rate.

### Remarks

This method is the same as [GetSampleClockRates\(DAQmxTasksBundle\)](#), except it also checks if the flag state is the same values across all sessions in the bundle. If the values are the same, it returns the single double value. Otherwise, it throws an exception.

### Exceptions

[NI Semiconductor TestException](#)

The value for the sample clock rate is not the same for all underlying instrument sessions.

# Method GetSampleClockRates

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetSampleClockRates(DAQmxTasksBundle)

Gets the actual sample clock rate (Hz).

```
public static double[] GetSampleClockRates(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Returns

[double](#)[]

Sample clock rate, one value per underlying instrument session.

### Exceptions

[DaqException](#)

The underling driver session returned an error.

# Class DAQmxTaskInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains an individual NI-DAQmx task and its related information.

```
public class DAQmxTaskInformation : ISessionInformation
```

## Inheritance

[object](#) ← DAQmxTaskInformation

## Implements

[ISessionInformation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DAQmxTaskInformation\(Task, IList<SitePinInfo>\)](#)

Constructs a task information object.

[DAQmxTaskInformation\(Task, string\)](#)

Constructs a task information object that associates with an NI-DAQmx task.

## Properties

[AssociatedSitePinList](#)

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

[ChannelList](#)

The channels associated with the task.

[ChannelType](#)

The NI-DAQmx channel type.

## Task

The NI-DAQmx task.

## TaskType

The NI-DAQmx task type.

# Methods

## [BuildFullyQualifiedOutputTerminal\(ExportSignal\)](#)

Builds a fully qualified output terminal string for the specified signal.

## [GenerateInstrumentChannelToSitePinDictionary\(I SemiconductorModuleContext\)](#)

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

# Constructor DAQmxTaskInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DAQmxTaskInformation(Task, string)

Constructs a task information object that associates with an NI-DAQmx task.

```
public DAQmxTaskInformation(Task task, string channelList)
```

### Parameters

**task** Task

The NI-DAQmx task.

**channelList** [string](#)

The channels associated with the task.

## DAQmxTaskInformation(Task, IList<SitePinInfo>)

Constructs a task information object.

```
public DAQmxTaskInformation(Task task, IList<SitePinInfo> associatedSitePinList)
```

### Parameters

**task** Task

The NI-DAQmx task.

**associatedSitePinList** [IList](#)<[SitePinInfo](#)>

The specified subset of channels associated with the task.

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

```
public IList<SitePinInfo> AssociatedSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property ChannelList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ChannelList

The channels associated with the task.

```
public string ChannelList { get; }
```

### Property Value

[string](#) ↗

### Remarks

It's a comma-separated string of instrument-channel pair, e.g. "DAQmx\_6363\_C1\_S08/ai0:2, DAQmx\_6363\_C1\_S08/ai4".

# Property ChannelType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ChannelType

The NI-DAQmx channel type.

```
public ChannelType ChannelType { get; }
```

## Property Value

ChannelType

# Property Task

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Task

The NI-DAQmx task.

```
public Task Task { get; }
```

## Property Value

### Task

# Property TaskType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TaskType

The NI-DAQmx task type.

```
public DAQmxTaskType TaskType { get; }
```

## Property Value

[DAQmxTaskType](#)

# Method BuildFullyQualifiedOutputTerminal

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## BuildFullyQualifiedOutputTerminal(ExportSignal)

Builds a fully qualified output terminal string for the specified signal.

```
public string BuildFullyQualifiedOutputTerminal(ExportSignal signal)
```

### Parameters

**signal** ExportSignal

The given signal.

### Returns

[string](#)

The fully qualified output terminal string.

# Method GenerateInstrumentChannelToSitePinDictionary

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext )

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

```
public static void GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext  
tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Enum DAQmxTaskType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines the type of the DAQmx task.

```
public enum DAQmxTaskType
```

## Fields

**AnalogInput = 0**

Specifies the task is an analog input task.

**AnalogOutput = 1**

Specifies the task is an analog output task.

**AnalogOutputFunctionGeneration = 6**

Specifies the task is an analog output function generation task.

**CounterInput = 4**

Specifies the task is a counter input task.

**CounterOutput = 5**

Specifies the task is a counter output task.

**DigitalInput = 2**

Specifies the task is a digital input task.

**DigitalOutput = 3**

Specifies the task is a digital output task.

# Class DAQmxTasksBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains one or more [DAQmxTaskInformation](#) objects.

```
public class DAQmxTasksBundle : ISessionsBundle<DAQmxTaskInformation>
```

## Inheritance

[object](#) ← DAQmxTasksBundle

## Implements

[ISessionsBundle<DAQmxTaskInformation>](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>>, string, string\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, int, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[,\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation>](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]](#),  
[string](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]\[\]](#),  
[string](#)).

## Constructors

[DAQmxTasksBundle\(ISemiconductorModuleContext, IEnumerable<DAQmxTaskInformation>\)](#)

Constructs a tasks bundle object that represents a bunch of [DAQmxTaskInformations](#).

## Properties

[AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

[InstrumentSessions](#)

An enumerable of [ISessionInformation](#).

[TSMContext](#)

The associated `NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext`.

# Methods

## [FilterByPin\(string\)](#)

Filter current [DAQmxTasksBundle](#) and returns a new one with the requested pin.

## [FilterByPin\(string\[\]\)](#)

Filter current [DAQmxTasksBundle](#) and returns a new one with requested pins.

## [FilterBySite\(int\)](#)

Filters current [DAQmxTasksBundle](#) and returns a new one with the requested site.

## [FilterBySite\(int\[\]\)](#)

Filters current [DAQmxTasksBundle](#) and returns a new one with requested sites.

# Constructor DAQmxTasksBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**DAQmxTasksBundle(ISemiconductorModuleContext, IEnumerable<DAQmxTaskInformation>)**

Constructs a tasks bundle object that represents a bunch of [DAQmxTaskInformations](#).

```
public DAQmxTasksBundle(ISemiconductorModuleContext tsmContext,  
IEnumerable<DAQmxTaskInformation> allTaskInfo)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

**allTaskInfo** [IEnumerable](#)<[DAQmxTaskInformation](#)>

An enumerable of [DAQmxTaskInformations](#).

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

```
public IList<SitePinInfo> AggregateSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [ISessionInformation](#).

```
public IEnumerable<DAQmxTaskInformation> InstrumentSessions { get; }
```

## Property Value

[IEnumerable](#) <[DAQmxTaskInformation](#)>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public ISemiconductorModuleContext TSMContext { get; }
```

## Property Value

ISemiconductorModuleContext

# Method FilterByPin

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterByPin(string)

Filter current [DAQmxTasksBundle](#) and returns a new one with the requested pin.

```
public DAQmxTasksBundle FilterByPin(string requestedPin)
```

### Parameters

**requestedPin** [string](#) ↗

The requested pin.

### Returns

[DAQmxTasksBundle](#)

A new [DAQmxTasksBundle](#) object with the requested pin.

## FilterByPin(string[])

Filter current [DAQmxTasksBundle](#) and returns a new one with requested pins.

```
public DAQmxTasksBundle FilterByPin(string[] requestedPins)
```

### Parameters

**requestedPins** [string](#) ↗[]

The requested pins.

### Returns

## [DAQmxTasksBundle](#)

A new [DAQmxTasksBundle](#) object with requested pins.

# Method FilterBySite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterBySite(int)

Filters current [DAQmxTasksBundle](#) and returns a new one with the requested site.

```
public DAQmxTasksBundle FilterBySite(int requestedSite)
```

### Parameters

`requestedSite` [int](#)

The requested site.

### Returns

[DAQmxTasksBundle](#)

A new [DAQmxTasksBundle](#) object with the requested site.

## FilterBySite(int[])

Filters current [DAQmxTasksBundle](#) and returns a new one with requested sites.

```
public DAQmxTasksBundle FilterBySite(int[] requestedSites)
```

### Parameters

`requestedSites` [int](#)[]

The requested sites.

### Returns

## [DAQmxTasksBundle](#)

A new [DAQmxTasksBundle](#) object with requested sites.

# Enum DAQmxTerminalConfiguration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines terminal configurations for analog input and output channels.

```
public enum DAQmxTerminalConfiguration
```

## Fields

**Default = -1**

Uses device default terminal configuration.

**Differential = 10106**

Measures the potential difference between the AI/AO+ and AI/AO-.

**Nrse = 10078**

Measures the potential difference between the AI and the AI SENSE.

**Pseudodifferential = 12529**

Measures the potential difference between the AI/AO+ and AI/AO- while AI/AO- is tied to AI/AO GND through a relatively small impedance.

**Rse = 10083**

Measures the potential difference between the AI/AO and the AI/AO GND.

# Class DAQmxTimingSampleClockSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

The class is used to configure the timing settings for a DAQmx task.

```
public class DAQmxTimingSampleClockSettings
```

## Inheritance

[object](#) ← DAQmxTimingSampleClockSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

Using this class is advantageous as it allows the user to set one or more properties in a single call.

## Constructors

[DAQmxTimingSampleClockSettings\(\)](#)

## Properties

### [SampleClockActiveEdge](#)

Specifies on which edge of a clock pulse sampling takes place.

### [SampleClockRate](#)

Specifies the sampling rate in samples per channel per second.

### [SampleClockSource](#)

Specifies the terminal of the signal to use as the Sample Clock.

### [SampleQuantityMode](#)

Specifies if a task acquires or generates a finite number of samples or if it continuously acquires or generates samples.

## SampleTimingType

Specifies the type of sample timing to use for the task.

## SamplesPerChannel

Specifies the number of samples to acquire or generate for each channel in the task.

# Constructor

## DAQmxTimingSampleClockSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### DAQmxTimingSampleClockSettings()

```
public DAQmxTimingSampleClockSettings()
```

# Property SampleClockActiveEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SampleClockActiveEdge

Specifies on which edge of a clock pulse sampling takes place.

```
public SampleClockActiveEdge? SampleClockActiveEdge { get; set; }
```

### Property Value

SampleClockActiveEdge?

### Remarks

This value is used to get/set the value of the NationalInstruments.DAQmx.Timing.SampleClockActiveEdge driver property. This property is useful primarily when the signal you use as the Sample Clock is not a periodic clock.

# Property SampleClockRate

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SampleClockRate

Specifies the sampling rate in samples per channel per second.

```
public double? SampleClockRate { get; set; }
```

### Property Value

[double](#)?

### Remarks

If you use an external source for the Sample Clock, set the value to the maximum expected rate of that clock. This value is used to get/set the value of the NationalInstruments.DAQmx.Timing.SampleClockRate driver property.

# Property SampleClockSource

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SampleClockSource

Specifies the terminal of the signal to use as the Sample Clock.

```
public string SampleClockSource { get; set; }
```

### Property Value

[string](#) ↗

### Remarks

Use this property to specify an external source to use as the Sample Clock. This value is used to get/set the value of the NationalInstruments.DAQmx.Timing.SampleClockSource driver property.

# Property SampleQuantityMode

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SampleQuantityMode

Specifies if a task acquires or generates a finite number of samples or if it continuously acquires or generates samples.

```
public SampleQuantityMode? SampleQuantityMode { get; set; }
```

### Property Value

SampleQuantityMode?

### Remarks

This value is used to get/set the value of the NationalInstruments.DAQmx.Timing.SampleQuantityMode driver property.

# Property SampleTimingType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SampleTimingType

Specifies the type of sample timing to use for the task.

```
public SampleTimingType SampleTimingType { get; }
```

### Property Value

SampleTimingType

### Remarks

This value is used to get/set the value of the NationalInstruments.DAQmx.Timing.SampleTimingType driver property.

# Property SamplesPerChannel

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SamplesPerChannel

Specifies the number of samples to acquire or generate for each channel in the task.

```
public int? SamplesPerChannel { get; set; }
```

### Property Value

[int](#)?

### Remarks

This value is used to get/set the value of the NationalInstruments.DAQmx.Timing.SampleClockSource driver property.

# Struct DefaultDAQmxTaskTypeStrings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines the default DAQmx task type strings.

```
public struct DefaultDAQmxTaskTypeStrings
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#)

## Fields

### [AnalogInput](#)

The analog input task type string.

### [AnalogOutput](#)

The analog output task type string.

### [AnalogOutputFunctionGeneration](#)

The analog output function generation task type string.

### [CounterInput](#)

The counter input task type string.

### [CounterOutput](#)

The counter output task type string.

### [DigitalInput](#)

The digital input task type string.

### [DigitalOutput](#)

The digital output task type string.

# Field AnalogInput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

The analog input task type string.

```
public const string AnalogInput = "AI"
```

Returns

[string](#)

The analog input task type string.

# Field AnalogOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

The analog output task type string.

```
public const string AnalogOutput = "AO"
```

Returns

[string](#)

The analog output task type string.

# Field AnalogOutputFunctionGeneration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

The analog output function generation task type string.

```
public const string AnalogOutputFunctionGeneration = "AOFuncGen"
```

Returns

[string](#)

The analog output function generation task type string.

# Field CounterInput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

The counter input task type string.

```
public const string CounterInput = "CI"
```

Returns

[string](#)

The counter input task type string.

# Field CounterOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

The counter output task type string.

```
public const string CounterOutput = "CO"
```

Returns

[string](#)

The counter output task type string.

# Field DigitalInput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

The digital input task type string.

```
public const string DigitalInput = "DI"
```

Returns

[string](#)

The digital input task type string.

# Field DigitalOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

The digital output task type string.

```
public const string DigitalOutput = "DO"
```

Returns

[string](#)

The digital output task type string.

# Class DigitalInput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations on digital input channels.

```
public static class DigitalInput
```

## Inheritance

[object](#) ← DigitalInput

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ReadDigitalMultiSampleU32\(DAQmxTasksBundle, int\)](#)

Reads digital multiple U32 samples.

[ReadDigitalSingleSample\(DAQmxTasksBundle\)](#)

Reads a single sample and returns pin- and site-aware object of type Boolean.

[ReadDigitalWaveform\(DAQmxTasksBundle, int\)](#)

Reads the samples and returns pin- and site-aware object of type DigitalWaveform, where each element in the waveform represents one sample read. By default, all available samples are returned, unless otherwise specified by passing in value via the samplesToRead argument.

# Method ReadDigitalMultiSampleU32

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadDigitalMultiSampleU32(DAQmxTasksBundle, int)

Reads digital multiple U32 samples.

```
public static PinSiteData<uint[]> ReadDigitalMultiSampleU32(this DAQmxTasksBundle  
tasksBundle, int samplesToRead = -1)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**samplesToRead** [int](#)

The number of samples to read.

### Returns

[PinSiteData<uint\[\]>](#)

Per-site per-pin unsigned integer samples.

# Method ReadDigitalSingleSample

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadDigitalSingleSample(DAQmxTasksBundle)

Reads a single sample and returns pin- and site-aware object of type Boolean.

```
public static PinSiteData<bool> ReadDigitalSingleSample(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Returns

[PinSiteData<bool>](#)

Per-site per-pin boolean samples.

# Method ReadDigitalWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadDigitalWaveform(DAQmxTasksBundle, int)

Reads the samples and returns pin- and site-aware object of type DigitalWaveform, where each element in the waveform represents one sample read. By default, all available samples are returned, unless otherwise specified by passing in value via the samplesToRead argument.

```
public static PinSiteData<DigitalWaveform> ReadDigitalWaveform(this DAQmxTasksBundle  
tasksBundle, int samplesToRead = -1)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**samplesToRead** [int](#)

The number of samples to read.

### Returns

[PinSiteData](#)<DigitalWaveform>

Per-site per-pin digital waveform samples.

# Class DigitalOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations on digital output channels.

```
public static class DigitalOutput
```

## Inheritance

[object](#) ← DigitalOutput

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[WriteDigital\(DAQmxTasksBundle, PinSiteData<bool>, bool\)](#)

Writes a single Boolean sample to the specified pin(s).

[WriteDigital\(DAQmxTasksBundle, SiteData<bool>, bool\)](#)

Writes a single Boolean sample to the specified pin(s).

[WriteDigital\(DAQmxTasksBundle, bool, bool\)](#)

Writes a single Boolean sample to the specified pin(s).

[WriteDigitalWaveform\(DAQmxTasksBundle, DigitalWaveform, bool\)](#)

Writes a DigitalWaveform (multiple Boolean samples over time) to the specified pin(s).

[WriteDigitalWaveform\(DAQmxTasksBundle, PinSiteData<DigitalWaveform>, bool\)](#)

Writes a DigitalWaveform (multiple Boolean samples over time) to the specified pin(s).

[WriteDigitalWaveform\(DAQmxTasksBundle, SiteData<DigitalWaveform>, bool\)](#)

Writes a DigitalWaveform (multiple Boolean samples over time) to the specified pin(s).

# Method WriteDigital

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteDigital(DAQmxTasksBundle, bool, bool)

Writes a single Boolean sample to the specified pin(s).

```
public static void WriteDigital(this DAQmxTasksBundle tasksBundle, bool staticState, bool  
autoStart = true)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**staticState** [bool](#)

The static state to write.

**autoStart** [bool](#)

Specifies whether to automatically start the tasks.

## WriteDigital(DAQmxTasksBundle, SiteData<bool>, bool)

Writes a single Boolean sample to the specified pin(s).

```
public static void WriteDigital(this DAQmxTasksBundle tasksBundle, SiteData<bool> siteData,  
bool autoStart = true)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

`siteData` [SiteData<bool>](#)

The per-site data to write.

`autoStart` [bool](#)

Specifies whether to automatically start the tasks.

## WriteDigital(DAQmxTasksBundle, PinSiteData<bool>, bool)

Writes a single Boolean sample to the specified pin(s).

```
public static void WriteDigital(this DAQmxTasksBundle tasksBundle, PinSiteData<bool>
pinSiteData, bool autoStart = true)
```

### Parameters

`tasksBundle` [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

`pinSiteData` [PinSiteData<bool>](#)

The per-site per-pin data to write.

`autoStart` [bool](#)

Specifies whether to automatically start the tasks.

# Method WriteDigitalWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteDigitalWaveform(DAQmxTasksBundle, DigitalWaveform, bool)

Writes a DigitalWaveform (multiple Boolean samples over time) to the specified pin(s).

```
public static void WriteDigitalWaveform(this DAQmxTasksBundle tasksBundle, DigitalWaveform
waveform, bool autoStart = true)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**waveform** DigitalWaveform

The per-site per-pin waveform to write.

**autoStart** [bool](#) ↗

Specifies whether to automatically start the tasks.

## WriteDigitalWaveform(DAQmxTasksBundle, SiteData<DigitalWaveform>, bool)

Writes a DigitalWaveform (multiple Boolean samples over time) to the specified pin(s).

```
public static void WriteDigitalWaveform(this DAQmxTasksBundle tasksBundle,
SiteData<DigitalWaveform> siteData, bool autoStart = true)
```

### Parameters

## **tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

## **siteData** [SiteData](#)<DigitalWaveform>

The per-site waveform to write.

## **autoStart** [bool](#) ↗

Specifies whether to automatically start the tasks.

# WriteDigitalWaveform(DAQmxTasksBundle, PinSiteData<DigitalWaveform>, bool)

Writes a DigitalWaveform (multiple Boolean samples over time) to the specified pin(s).

```
public static void WriteDigitalWaveform(this DAQmxTasksBundle tasksBundle,  
PinSiteData<DigitalWaveform> pinSiteData, bool autoStart = true)
```

## Parameters

### **tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### **pinSiteData** [PinSiteData](#)<DigitalWaveform>

The per-site per-pin waveform to write.

### **autoStart** [bool](#) ↗

Specifies whether to automatically start the tasks.

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines NI-DAQmx NationalInstruments.DAQmx.Tasks initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ClearAllDAQmxTasks\(ISemiconductorModuleContext\)](#).

Clears all NI-DAQmx tasks regardless of task type.

[ClearDAQmxAIVoltageTasks\(ISemiconductorModuleContext, string\)](#).

Clears NI-DAQmx analog input voltage tasks.

[ClearDAQmxAOFunctionGenerationTasks\(ISemiconductorModuleContext, string\)](#).

Clears NI-DAQmx analog output function generation voltage tasks.

[ClearDAQmxAOVoltageTasks\(ISemiconductorModuleContext, string\)](#).

Clears NI-DAQmx analog output voltage tasks.

[ClearDAQmxDITasks\(ISemiconductorModuleContext, string\)](#).

Clears NI-DAQmx digital input tasks.

[ClearDAQmxDOTasks\(ISemiconductorModuleContext, string\)](#).

Clears NI-DAQmx digital output tasks.

[CreateDAQmxAIVoltageTasks\(ISemiconductorModuleContext, string, double, double, DAQmxTerminal Configuration\)](#).

Creates NI-DAQmx analog input voltage tasks in the test system.

[CreateDAQmxAOFunctionGenerationTasks\(ISemiconductorModuleContext, string, AOFunctionGenerationType, double, double, double, DAQmxTerminalConfiguration\)](#)

Creates NI-DAQmx analog output function generation voltage tasks.

[CreateDAQmxAOVoltageTasks\(ISemiconductorModuleContext, string, double, double, DAQmxTerminalConfiguration\)](#)

Creates NI-DAQmx analog output voltage tasks in the test system.

[CreateDAQmxDITasks\(ISemiconductorModuleContext, string, ChannelLineGrouping\)](#)

Creates NI-DAQmx digital input tasks in the test system.

[CreateDAQmxDOTasks\(ISemiconductorModuleContext, string, ChannelLineGrouping\)](#)

Creates NI-DAQmx digital output tasks in the test system.

# Method ClearAllDAQmxTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ClearAllDAQmxTasks(ISemiconductorModuleContext)

Clears all NI-DAQmx tasks regardless of task type.

```
public static void ClearAllDAQmxTasks(ISemiconductorModuleContext tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Method ClearDAQmxAIVoltageTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ClearDAQmxAIVoltageTasks(ISemiconductorModuleContext, string)

Clears NI-DAQmx analog input voltage tasks.

```
public static void ClearDAQmxAIVoltageTasks(ISemiconductorModuleContext tsmContext, string  
analogInputVoltageTaskType = "AI")
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**analogInputVoltageTaskType** [string](#)

Specifies the task type.

# Method ClearDAQmxAOFunctionGenerationTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ClearDAQmxAOFunctionGenerationTasks(ISemiconductorModuleContext, string)

Clears NI-DAQmx analog output function generation voltage tasks.

```
public static void ClearDAQmxAOFunctionGenerationTasks(ISemiconductorModuleContext  
tsmContext, string analogOutputFunctionGenerationVoltageTaskType = "AOFuncGen")
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**analogOutputFunctionGenerationVoltageTaskType** [string](#)

Specifies the task type.

# Method ClearDAQmxAOVoltageTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ClearDAQmxAOVoltageTasks(ISemiconductorModuleContext, string)

Clears NI-DAQmx analog output voltage tasks.

```
public static void ClearDAQmxAOVoltageTasks(ISemiconductorModuleContext tsmContext, string  
analogOutputVoltageTaskType = "AO")
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**analogOutputVoltageTaskType** [string](#)

Specifies the task type.

# Method ClearDAQmxDITasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ClearDAQmxDITasks(ISemiconductorModuleContext, string)

Clears NI-DAQmx digital input tasks.

```
public static void ClearDAQmxDITasks(ISemiconductorModuleContext tsmContext, string  
digitalInputTaskType = "DI")
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**digitalInputTaskType** [string](#)

Specifies the task type.

# Method ClearDAQmxDOTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ClearDAQmxDOTasks(ISemiconductorModuleContext, string)

Clears NI-DAQmx digital output tasks.

```
public static void ClearDAQmxDOTasks(ISemiconductorModuleContext tsmContext, string  
digitalOutputTaskType = "DO")
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**digitalOutputTaskType** [string](#)

Specifies the task type.

# Method CreateDAQmxAIVoltageTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**CreateDAQmxAIVoltageTasks(ISemiconductorModuleContext, string, double, double, DAQmxTerminalConfiguration)**

Creates NI-DAQmx analog input voltage tasks in the test system.

```
public static void CreateDAQmxAIVoltageTasks(ISemiconductorModuleContext tsmContext, string  
analogInputVoltageTaskType = "AI", double minimumVoltage = -1, double maximumVoltage = 1,  
DAQmxTerminalConfiguration aiTerminalConfiguration = DAQmxTerminalConfiguration.Default)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**analogInputVoltageTaskType** [string](#)

Specifies the task type.

**minimumVoltage** [double](#)

Specifies minimum voltage value.

**maximumVoltage** [double](#)

Specifies maximum voltage value.

**aiTerminalConfiguration** [DAQmxTerminalConfiguration](#)

Specifies input terminal configuration.

# Method CreateDAQmxAOFunctionGenerationTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**CreateDAQmxAOFunctionGenerationTasks(ISemiconductorModuleContext, string, AOFunctionGenerationType, double, double, double, DAQmxTerminalConfiguration)**

Creates NI-DAQmx analog output function generation voltage tasks.

```
public static void CreateDAQmxAOFunctionGenerationTasks(ISemiconductorModuleContext tsmContext, string analogOutputFunctionGenerationVoltageTaskType = "AOFuncGen", AOFunctionGenerationType waveformType = AOFunctionGenerationType.Sine, double frequency = 1000, double amplitude = 5, double offset = 0, DAQmxTerminalConfiguration aoTerminalConfiguration = DAQmxTerminalConfiguration.Default)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**analogOutputFunctionGenerationVoltageTaskType** [string](#)

Specifies the task type.

**waveformType** AOFunctionGenerationType

Specifies waveform type.

**frequency** [double](#)

Specifies frequency.

**amplitude** [double](#)

Specifies amplitude value.

**offset** [double](#)

Specifies offset value.

**aoTerminalConfiguration** [DAQmxTerminalConfiguration](#)

Specifies output terminal configuration.

# Method CreateDAQmxAOVoltageTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**CreateDAQmxAOVoltageTasks(ISemiconductorModuleContext, string, double, double, DAQmxTerminalConfiguration)**

Creates NI-DAQmx analog output voltage tasks in the test system.

```
public static void CreateDAQmxAOVoltageTasks(ISemiconductorModuleContext tsmContext, string
analogOutputVoltageTaskType = "AO", double minimumVoltage = -1, double maximumVoltage = 1,
DAQmxTerminalConfiguration aoTerminalConfiguration = DAQmxTerminalConfiguration.Default)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**analogOutputVoltageTaskType** [string](#)

Specifies the task type.

**minimumVoltage** [double](#)

Specifies minimum voltage value.

**maximumVoltage** [double](#)

Specifies maximum voltage value.

**aoTerminalConfiguration** [DAQmxTerminalConfiguration](#)

Specifies output terminal configuration.

# Method CreateDAQmxDITasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## CreateDAQmxDITasks(ISemiconductorModuleContext, string, ChannelLineGrouping)

Creates NI-DAQmx digital input tasks in the test system.

```
public static void CreateDAQmxDITasks(ISemiconductorModuleContext tsmContext, string  
digitalInputTaskType = "DI", ChannelLineGrouping channelLineGrouping =  
ChannelLineGrouping.OneChannelForEachLine)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**digitalInputTaskType** [string](#)

Specifies the task type.

**channelLineGrouping** ChannelLineGrouping

Specifies channel line grouping method.

# Method CreateDAQmxDOTasks

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## CreateDAQmxDOTasks(ISemiconductorModuleContext, string, ChannelLineGrouping)

Creates NI-DAQmx digital output tasks in the test system.

```
public static void CreateDAQmxDOTasks(ISemiconductorModuleContext tsmContext, string  
digitalOutputTaskType = "DO", ChannelLineGrouping channelLineGrouping =  
ChannelLineGrouping.OneChannelForEachLine)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**digitalOutputTaskType** [string](#)

Specifies the task type.

**channelLineGrouping** ChannelLineGrouping

Specifies channel line grouping method.

# Class SampleValuesCacher<T>

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines a class that caches sample values.

```
public class SampleValuesCacher<T>
```

## Type Parameters

T

The type of the sample.

### Inheritance

[object](#) ← SampleValuesCacher<T>

### Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[SampleValuesCacher\(\)](#)

## Properties

### Instance

The singleton instance of the [SampleValuesCacher<T>](#).

## Methods

[GenerateDefaultSampleValues\(string\[\]\)](#)

Generates a dictionary that caches sample values.

[TryWriteAndRecoverCacheOnFailure\(DAQmxTaskInformation, PinSiteData<T>, Action<T\[\]>\)](#)

Tries to do the specified write operation and ensures that the sample values cache is updated on a successful write.

[TryWriteAndRecoverCacheOnFailure\(DAQmxTaskInformation, SiteData<T>, Action<T\[\]>\)](#)

Tries to do the specified write operation and ensures that the sample values cache is updated on a successful write.

[TryWriteAndRecoverCacheOnFailure\(DAQmxTaskInformation, T, Action<T\[\]>\)](#)

Tries to do the specified write operation and ensures that the sample values cache is updated on a successful write.

# Constructor SampleValuesCacher

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SampleValuesCacher()

```
public SampleValuesCacher()
```

# Property Instance

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Instance

The singleton instance of the [SampleValuesCacher<T>](#).

```
public static SampleValuesCacher<T> Instance { get; }
```

## Property Value

[SampleValuesCacher<T>](#)

# Method GenerateDefaultSampleValues

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GenerateDefaultSampleValues(string[])

Generates a dictionary that caches sample values.

```
public void GenerateDefaultSampleValues(string[] channelLists)
```

### Parameters

channelLists [string](#)[]

The channel lists each associates with a task.

# Method TryWriteAndRecoverCacheOnFailure

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TryWriteAndRecoverCacheOnFailure(DAQmxTaskInformation, T, Action<T[]>)

Tries to do the specified write operation and ensures that the sample values cache is updated on a successful write.

```
public void TryWriteAndRecoverCacheOnFailure(DAQmxTaskInformation taskInfo, T data,
Action<T[]> writeAction)
```

### Parameters

**taskInfo** [DAQmxTaskInformation](#)

The [DAQmxTaskInformation](#) object.

**data** T

The state to update to.

**writeAction** [Action](#)<T[]>

The driver write operation.

## TryWriteAndRecoverCacheOnFailure(DAQmxTaskInformation, SiteData<T>, Action<T[]>)

Tries to do the specified write operation and ensures that the sample values cache is updated on a successful write.

```
public void TryWriteAndRecoverCacheOnFailure(DAQmxTaskInformation taskInfo, SiteData<T>
data, Action<T[]> writeAction)
```

## Parameters

**taskInfo** [DAQmxTaskInformation](#)

The [DAQmxTaskInformation](#) object.

**data** [SiteData](#)<T>

The state to update to.

**writeAction** [Action](#)<T[]>

The driver write operation.

**TryWriteAndRecoverCacheOnFailure(DAQmxTaskInformation, PinSiteData<T>, Action<T[]>)**

Tries to do the specified write operation and ensures that the sample values cache is updated on a successful write.

```
public void TryWriteAndRecoverCacheOnFailure(DAQmxTaskInformation taskInfo, PinSiteData<T>
data, Action<T[]> writeAction)
```

## Parameters

**taskInfo** [DAQmxTaskInformation](#)

The [DAQmxTaskInformation](#) object.

**data** [PinSiteData](#)<T>

The state to update to.

**writeAction** [Action](#)<T[]>

The driver write operation.

# Class TaskControl

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations for performing low-level task control.

```
public static class TaskControl
```

## Inheritance

[object](#) ← TaskControl

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### [Abort\(DAQmxTasksBundle\)](#)

Aborts execution of the task, immediately terminating any currently active operation, such as a read or a write.

### [Commit\(DAQmxTasksBundle\)](#)

Programs the hardware with all parameters from the task properties previously configured.

### [Reserve\(DAQmxTasksBundle\)](#)

Reserves the hardware resources that are needed for the task.

### [Start\(DAQmxTasksBundle\)](#)

Starts the task.

### [Stop\(DAQmxTasksBundle\)](#)

Starts the task.

### [Unreserve\(DAQmxTasksBundle\)](#)

Releases all previously reserved resources.

### [Verify\(DAQmxTasksBundle\)](#)

Verifies that all task parameters are valid for the hardware.

## [WaitUntilDone\(DAQmxTasksBundle\)](#)

Waits for the measurement or generation to complete, regardless of the amount of time needed, and returns whether the execution is complete.

# Method Abort

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Abort(DAQmxTasksBundle)

Aborts execution of the task, immediately terminating any currently active operation, such as a read or a write.

```
public static void Abort(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

This is a low-level driver control method that is not recommended for general use. Aborting a task puts the task into an unstable but recoverable state. To recover the task, use [Start\(DAQmxTasksBundle\)](#), to restart the task or use [Stop\(DAQmxTasksBundle\)](#) to reset the task without starting it.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method Commit

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Commit(DAQmxTasksBundle)

Programs the hardware with all parameters from the task properties previously configured.

```
public static void Commit(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

This is a low-level driver control method that is not recommended for general use.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method Reserve

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Reserve(DAQmxTasksBundle)

Reserves the hardware resources that are needed for the task.

```
public static void Reserve(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

This is a low-level driver control method that is not recommended for general use. It marks the hardware resources that are needed for the task as in use. No other tasks can reserve these same resources.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method Start

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Start(DAQmxTasksBundle)

Starts the task.

```
public static void Start(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

This is a low-level driver control method that is not recommended for general use. It transitions the task to the running state, which begins device input or output.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method Stop

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Stop(DAQmxTasksBundle)

Starts the task.

```
public static void Stop(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

This is a low-level driver control method that is not recommended for general use. It transitions the task from the running state to the committed state, which ends device input or output.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method Unreserve

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Unreserve(DAQmxTasksBundle)

Releases all previously reserved resources.

```
public static void Unreserve(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

This is a low-level driver control method that is not recommended for general use.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method Verify

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Verify(DAQmxTasksBundle)

Verifies that all task parameters are valid for the hardware.

```
public static void Verify(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

This is a low-level driver control method that is not recommended for general use.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method WaitUntilDone

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WaitUntilDone(DAQmxTasksBundle)

Waits for the measurement or generation to complete, regardless of the amount of time needed, and returns whether the execution is complete.

```
public static void WaitUntilDone(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Remarks

`NationalInstruments.DAQmx.Task.WaitUntilDone` waits for the task to finish acquiring or generating the number of samples per channel specified on the `NationalInstruments.DAQmx.Timing` class.

`NationalInstruments.DAQmx.Task.WaitUntilDone` does not wait for pending asynchronous operations to complete. Use the methods and properties on `System.IAsyncResult` to verify that asynchronous reads and writes are complete.

### Exceptions

`DaqException`

The underling driver session returned an error.

# Class TriggersAndEvents

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines operations for configuring triggers.

```
public static class TriggersAndEvents
```

## Inheritance

[object](#) ← TriggersAndEvents

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ConfigureStartTriggerDigitalEdge\(DAQmxTasksBundle, string, DigitalEdgeStartTriggerEdge\)](#).

Configures the start trigger for the task.

[DisableStartTrigger\(DAQmxTasksBundle\)](#).

Disables the start trigger for the task.

[ExportSignal\(DAQmxTasksBundle, ExportSignal, string\)](#).

Routes a control signal to the specified terminal.

[GetFullyQualifiedOutputTerminals\(DAQmxTasksBundle, ExportSignal\)](#).

Gets the fully qualified terminal name for the specified signal.

# Method ConfigureStartTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureStartTriggerDigitalEdge(DAQmxTasksBundle, string, DigitalEdgeStartTriggerEdge)

Configures the start trigger for the task.

```
public static void ConfigureStartTriggerDigitalEdge(this DAQmxTasksBundle tasksBundle,  
    string source, DigitalEdgeStartTriggerEdge edge = DigitalEdgeStartTriggerEdge.Rising)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**source** [string](#) ↗

The name of the terminal to receive the digital signal to be used as the trigger source.

**edge** [DigitalEdgeStartTriggerEdge](#)

The edge of the digital signal to start acquiring or generating samples.

### Remarks

Configures the task to acquire or generate samples based on either the rising or falling edge of the digital signal source.

### Exceptions

[DaqException](#)

The underling driver session returned an error.

# Method DisableStartTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DisableStartTrigger(DAQmxTasksBundle)

Disables the start trigger for the task.

```
public static void DisableStartTrigger(this DAQmxTasksBundle tasksBundle)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

### Exceptions

**DaqException**

The underling driver session returned an error.

# Method ExportSignal

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ExportSignal(DAQmxTasksBundle, ExportSignal, string)

Routes a control signal to the specified terminal.

```
public static void ExportSignal(this DAQmxTasksBundle tasksBundle, ExportSignal signal,  
string outputTerminal)
```

### Parameters

#### `tasksBundle` [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

#### `signal` ExportSignal

The trigger, clock, or event to export.

#### `outputTerminal` [string](#) ↗

The destination of the exported signal. You can also specify a comma-delimited list for multiple terminal names.

### Remarks

Because the output terminal can reside on the device that generates the control signal or on a different device, you can use this method to share clocks and triggers among multiple devices.

### Exceptions

#### `DaqException`

The underling driver session returned an error.

# Method GetFullyQualifiedOutputTerminals

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DAQmx](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetFullyQualifiedOutputTerminals(DAQmxTasksBundle, ExportSignal)

Gets the fully qualified terminal name for the specified signal.

```
public static string[] GetFullyQualifiedOutputTerminals(this DAQmxTasksBundle tasksBundle,  
ExportSignal signal)
```

### Parameters

**tasksBundle** [DAQmxTasksBundle](#)

The [DAQmxTasksBundle](#) object.

**signal** ExportSignal

The trigger, clock, or event.

### Returns

[string](#)[]

Array of fully qualified output terminal strings, one per instrument session.

### Exceptions

DaqException

The underling driver session returned an error.

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower

## Classes

### [Control](#)

Defines methods for controlling the NI-DCPower session.

### [DCPowerMeasureSettings](#)

Defines DCPower measure settings.

### [DCPowerSessionInformation](#)

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIDCPower.NIDCPower session and its related information.

### [DCPowerSessionsBundle](#)

Defines an object that contains one or more [DCPowerSessionInformation](#) objects.

### [DCPowerSettings](#)

Defines DCPower settings.

### [DCPowerSourceSettings](#)

Defines DCPower source settings.

### [DCPowerWaveformAcquisitionSettings](#)

Defines DCPower waveform acquisition settings.

### [DCPowerWaveformResults](#)

Defines DCPower waveform results.

### [InitializeAndClose](#)

Defines NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions initialize and close APIs.

### [Measure](#)

Defines methods for DCPower measurements.

### [Source](#)

Defines methods for DCPower voltage/current sourcing.

### [TriggersAndEvents](#)

Defines methods for DCPower triggers and events.

## Structs

### [DCPowerModelStrings](#)

Defines DCPower model strings according to the driver definition below.

[https://dev.azure.com/ni/DevCentral/\\_git/ni-central?  
path=/src/nidcpower/base/nidcpowerSharedSource/source/nidcpowerSharedSource/productid/ProductID.h](https://dev.azure.com/ni/DevCentral/_git/ni-central?path=/src/nidcpower/base/nidcpowerSharedSource/source/nidcpowerSharedSource/productid/ProductID.h)



## [SingleDCPowerFetchResult](#)

Structure that defines the result of a single fetch operation.

## Enums

### [EventType](#)

Defines DCPower event type.

### [TriggerType](#)

Defines DCPower trigger type.

# Class Control

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for controlling the NI-DCPower session.

```
public static class Control
```

## Inheritance

[object](#) ← Control

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## [Abort\(DCPowerSessionsBundle\)](#)

Transitions the NI-DCPower session from the Running state to the Uncommitted state. If a sequence is running, the NI-DCPower session is stopped.

## [Commit\(DCPowerSessionsBundle\)](#)

Applies previously configured settings to the underlying device channel(s). Calling this methods moves the underlying channel(s) from the Uncommitted state into the Committed state. After calling this method, modifying any property reverts the underlying device channel(s) to the Uncommitted state.

## [Initiate\(DCPowerSessionsBundle\)](#)

Starts generation or acquisition and moves the underlying devices channel(s) from the Uncommitted state or Committed state to the Running state. To return to the Uncommitted state, call the Abort method.

# Method Abort

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Abort(DCPowerSessionsBundle)

Transitions the NI-DCPower session from the Running state to the Uncommitted state. If a sequence is running, the NI-DCPower session is stopped.

```
public static void Abort(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Remarks

Note: This is a lower level function for controlling over the NI-DCPower session. Any low level driver property updated after this method will not be applied until the next sourcing operation, or when the Commit method is explicitly called. If power output is enabled when you call the Abort method, the channels remain in their current state and continue providing power. Refer to the Programming States topic in the NI-DCPower User Manual and the document of your SMU model for information about the specific NI-DCPower software states. Use the OutputEnable method to disable power output per channel. Use the Reset method to disable output on all channels.

# Method Commit

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Commit(DCPowerSessionsBundle)

Applies previously configured settings to the underlying device channel(s). Calling this methods moves the underlying channel(s) from the Uncommitted state into the Committed state. After calling this method, modifying any property reverts the underlying device channel(s) to the Uncommitted state.

```
public static void Commit(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Remarks

Note: This is a lower level function for controlling over the NI-DCPower session. Refer to the Programming States topic in the NI-DCPower User Manual and the document of your SMU model for information about the specific NI-DCPower software states.

# Method Initiate

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Initiate(DCPowerSessionsBundle)

Starts generation or acquisition and moves the underlying devices channel(s) from the Uncommitted state or Committed state to the Running state. To return to the Uncommitted state, call the Abort method.

```
public static void Initiate(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Remarks

Note: This is a lower level function for controlling over the NI-DCPower session. Refer to the Programming States topic in the NI-DCPower User Manual and the document of your SMU model for information about the specific NI-DCPower software states.

# Class DCPowerMeasureSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines DC Power measure settings.

```
public class DC PowerMeasureSettings
```

## Inheritance

[object](#) ← DC PowerMeasureSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DCPowerMeasureSettings\(\)](#).

Default constructor.

## Properties

[ApertureTime](#)

The aperture time.

[ApertureTimeUnits](#)

The unit of aperture time.

[MeasureWhen](#)

The measure when.

[RecordLength](#)

The record length.

[Sense](#)

The measurement sense.

# Constructor DCPowerMeasureSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DCPowerMeasureSettings()

Default constructor.

```
public DCPowerMeasureSettings()
```

# Property ApertureTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApertureTime

The aperture time.

```
public double? ApertureTime { get; set; }
```

## Property Value

[double](#)?

# Property ApertureTimeUnits

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApertureTimeUnits

The unit of aperture time.

```
public DCPowerMeasureApertureTimeUnits? ApertureTimeUnits { get; set; }
```

## Property Value

DCPowerMeasureApertureTimeUnits?

# Property MeasureWhen

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureWhen

The measure when.

```
public DCPowerMeasurementWhen? MeasureWhen { get; set; }
```

## Property Value

DCPowerMeasurementWhen?

# Property RecordLength

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## RecordLength

The record length.

```
public int? RecordLength { get; set; }
```

## Property Value

[int](#)?

# Property Sense

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Sense

The measurement sense.

```
public DCPowerMeasurementSense? Sense { get; set; }
```

## Property Value

DCPowerMeasurementSense?

# Struct DCPowerModelStrings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines DCPower model strings according to the driver definition below.

[https://dev.azure.com/ni/DevCentral/\\_git/ni-central?  
path=/src/nidcpower/base/nidcpowerSharedSource/source/nidcpowerSharedSource/productid/ProductID.h](https://dev.azure.com/ni/DevCentral/_git/ni-central?path=/src/nidcpower/base/nidcpowerSharedSource/source/nidcpowerSharedSource/productid/ProductID.h)



```
public struct DCPowerModelStrings
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#)

## Fields

### [PXI\\_4110](#)

Indicates an NI PXI-4110 device.

### [PXI\\_4130](#)

Indicates an NI PXI-4130 device.

### [PXI\\_4131A](#)

Indicates an NI PXI-4131A device.

### [PXI\\_4132](#)

Indicates an NI PXI-4132 device.

### [PXIe\\_4051](#)

Indicates an NI PXIe-4051 device.

### [PXIe\\_4112](#)

Indicates an NI PXIe-4112 device.

### [PXIe\\_4113](#)

Indicates an NI PXIe-4113 device.

### [PXIe\\_4118](#)

Indicates an NI PXIe-4118 device.

## [PXIe 4135](#)

Indicates an NI PXIe-4135 device.

## [PXIe 4135 40W](#)

Indicates an NI PXIe-4135 high power device.

## [PXIe 4136](#)

Indicates an NI PXIe-4136 device.

## [PXIe 4137](#)

Indicates an NI PXIe-4137 device.

## [PXIe 4137 40W](#)

Indicates an NI PXIe-4137 high power device.

## [PXIe 4138](#)

Indicates an NI PXIe-4138 device.

## [PXIe 4139](#)

Indicates a NI PXIe-4139 device.

## [PXIe 4139 40W](#)

Indicates an NI PXIe-4139 high power device.

## [PXIe 4140](#)

Indicates an NI PXIe-4140 device.

## [PXIe 4141](#)

Indicates an NI PXIe-4141 device.

## [PXIe 4141 HSR](#)

Indicates an NI PXIe-4141 high sense resistance device.

## [PXIe 4142](#)

Indicates an NI PXIe-4142 device.

## [PXIe 4143](#)

Indicates an NI PXIe-4143 device.

## [PXIe 4144](#)

Indicates an NI PXIe-4144 device.

## [PXIe 4145](#)

Indicates an NI PXIe-4145 device.

## [PXIe 4147](#)

Indicates an NI PXIe-4147 device.

## [PXIe 4150](#)

Indicates an NI PXIe-4150 device.

## [PXIe 4151](#)

Indicates an NI PXIe-4151 device.

## [PXIe 4154](#)

Indicates an NI PXIe-4154 device.

## [PXIe 4161](#)

Indicates an NI PXIe-4161 device.

## [PXIe 4162](#)

Indicates an NI PXIe-4162 device.

## [PXIe 4162 10pA](#)

Indicates an NI PXIe-4162 10pA device.

## [PXIe 4163](#)

Indicates an NI PXIe-4163 device.

## [PXIe 4163 10pA](#)

Indicates an NI PXIe-4163 10pA device.

## [PXIe 4190](#)

Indicates an NI PXIe-4190 device.

## [PXIe 4190\\_500kHz](#)

Indicates an NI PXIe-4190 500kHz device.

# Field PXI\_4110

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXI-4110 device.

```
public const string PXI_4110 = "NI PXI-4110"
```

Returns

[string](#)

Indicates an NI PXI-4110 device.

# Field PXI\_4130

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXI-4130 device.

```
public const string PXI_4130 = "NI PXI-4130"
```

Returns

[string](#)

Indicates an NI PXI-4130 device.

# Field PXI\_4131A

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXI-4131A device.

```
public const string PXI_4131A = "NI PXI-4131A"
```

Returns

[string](#)

Indicates an NI PXI-4131A device.

# Field PXI\_4132

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXI-4132 device.

```
public const string PXI_4132 = "NI PXI-4132"
```

Returns

[string](#)

Indicates an NI PXI-4132 device.

# Field PXIe\_4051

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4051 device.

```
public const string PXIe_4051 = "NI PXIe-4051"
```

Returns

[string](#)

Indicates an NI PXIe-4051 device.

# Field PXIe\_4112

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4112 device.

```
public const string PXIe_4112 = "NI PXIe-4112"
```

Returns

[string](#)

Indicates an NI PXIe-4112 device.

# Field PXIe\_4113

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4113 device.

```
public const string PXIe_4113 = "NI PXIe-4113"
```

Returns

[string](#)

Indicates an NI PXIe-4113 device.

# Field PXIe\_4118

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4118 device.

```
public const string PXIe_4118 = "NI PXIe-4118"
```

Returns

[string](#)

Indicates an NI PXIe-4118 device.

# Field PXIe\_4135

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4135 device.

```
public const string PXIe_4135 = "NI PXIe-4135"
```

Returns

[string](#)

Indicates an NI PXIe-4135 device.

# Field PXIe\_4135\_40W

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4135 high power device.

```
public const string PXIe_4135_40W = "NI PXIe-4135 (40W)"
```

Returns

[string](#)

Indicates an NI PXIe-4135 high power device.

# Field PXIe\_4136

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4136 device.

```
public const string PXIe_4136 = "NI PXIe-4136"
```

Returns

[string](#)

Indicates an NI PXIe-4136 device.

# Field PXIe\_4137

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4137 device.

```
public const string PXIe_4137 = "NI PXIe-4137"
```

Returns

[string](#)

Indicates an NI PXIe-4137 device.

# Field PXIe\_4137\_40W

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4137 high power device.

```
public const string PXIe_4137_40W = "NI PXIe-4137 (40W)"
```

Returns

[string](#)

Indicates an NI PXIe-4137 high power device.

# Field PXIe\_4138

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4138 device.

```
public const string PXIe_4138 = "NI PXIe-4138"
```

Returns

[string](#)

Indicates an NI PXIe-4138 device.

# Field PXIe\_4139

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates a NI PXIe-4139 device.

```
public const string PXIe_4139 = "NI PXIe-4139"
```

Returns

[string](#)

Indicates a NI PXIe-4139 device.

# Field PXIe\_4139\_40W

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4139 high power device.

```
public const string PXIe_4139_40W = "NI PXIe-4139 (40W)"
```

Returns

[string](#)

Indicates an NI PXIe-4139 high power device.

# Field PXIe\_4140

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4140 device.

```
public const string PXIe_4140 = "NI PXIe-4140"
```

Returns

[string](#)

Indicates an NI PXIe-4140 device.

# Field PXIe\_4141

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4141 device.

```
public const string PXIe_4141 = "NI PXIe-4141"
```

Returns

[string](#)

Indicates an NI PXIe-4141 device.

# Field PXIe\_4141\_HSR

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4141 high sense resistance device.

```
public const string PXIe_4141_HSR = "NI PXIe-4141 (High Sense Resistance)"
```

Returns

[string](#)

Indicates an NI PXIe-4141 high sense resistance device.

# Field PXIe\_4142

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4142 device.

```
public const string PXIe_4142 = "NI PXIe-4142"
```

Returns

[string](#)

Indicates an NI PXIe-4142 device.

# Field PXIe\_4143

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4143 device.

```
public const string PXIe_4143 = "NI PXIe-4143"
```

Returns

[string](#)

Indicates an NI PXIe-4143 device.

# Field PXIe\_4144

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4144 device.

```
public const string PXIe_4144 = "NI PXIe-4144"
```

Returns

[string](#)

Indicates an NI PXIe-4144 device.

# Field PXIe\_4145

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4145 device.

```
public const string PXIe_4145 = "NI PXIe-4145"
```

Returns

[string](#)

Indicates an NI PXIe-4145 device.

# Field PXIe\_4147

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4147 device.

```
public const string PXIe_4147 = "NI PXIe-4147"
```

Returns

[string](#)

Indicates an NI PXIe-4147 device.

# Field PXIe\_4150

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4150 device.

```
public const string PXIe_4150 = "NI PXIe-4150"
```

Returns

[string](#)

Indicates an NI PXIe-4150 device.

# Field PXIe\_4151

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4151 device.

```
public const string PXIe_4151 = "NI PXIe-4151"
```

Returns

[string](#)

Indicates an NI PXIe-4151 device.

# Field PXIe\_4154

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4154 device.

```
public const string PXIe_4154 = "NI PXIe-4154"
```

Returns

[string](#)

Indicates an NI PXIe-4154 device.

# Field PXIe\_4161

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4161 device.

```
public const string PXIe_4161 = "NI PXIe-4161"
```

Returns

[string](#)

Indicates an NI PXIe-4161 device.

# Field PXIe\_4162

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4162 device.

```
public const string PXIe_4162 = "NI PXIe-4162"
```

Returns

[string](#)

Indicates an NI PXIe-4162 device.

# Field PXIe\_4162\_10pA

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4162 10pA device.

```
public const string PXIe_4162_10pA = "NI PXIe-4162 (10pA)"
```

Returns

[string](#)

Indicates an NI PXIe-4162 10pA device.

# Field PXIe\_4163

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4163 device.

```
public const string PXIe_4163 = "NI PXIe-4163"
```

Returns

[string](#)

Indicates an NI PXIe-4163 device.

# Field PXIe\_4163\_10pA

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4163 10pA device.

```
public const string PXIe_4163_10pA = "NI PXIe-4163 (10pA)"
```

Returns

[string](#)

Indicates an NI PXIe-4163 10pA device.

# Field PXIe\_4190

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXIe-4190 device.

```
public const string PXIe_4190 = "NI PXIe-4190"
```

Returns

[string](#)

Indicates an NI PXIe-4190 device.

# Field PXle\_4190\_500kHz

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Indicates an NI PXle-4190 500kHz device.

```
public const string PXle_4190_500kHz = "NI PXle-4190 (500kHz)"
```

Returns

[string](#)

Indicates an NI PXle-4190 500kHz device.

# Class DCPowerSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIDCPower.NIDCPower session and its related information.

```
public class DCPowerSessionInformation : ISessionInformation
```

## Inheritance

[object](#) ← DCPowerSessionInformation

## Implements

[ISessionInformation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DCPowerSessionInformation\(NIDCPower, IList<SitePinInfo>\)](#)

Constructs a session information object.

[DCPowerSessionInformation\(NIDCPower, string\)](#)

Constructs a session information object that associates with a NationalInstruments.ModularInstruments.NIDCPower.NIDCPower instrument session.

## Properties

[AllChannelsOutput](#)

The all channels NationalInstruments.ModularInstruments.NIDCPower.DCPowerOutput corresponds to the [AllChannelsString](#).

[AllChannelsString](#)

The all channels string associated with the [Session](#).

## [AllInstrumentsAreTheSameModel](#)

Indicates whether all instruments are the same model.

## [AssociatedSitePinList](#)

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

## [ModelString](#)

The DCPower model string if all instruments are the same model. Otherwise, it's empty string, please check [ModelString](#) for each element of the [AssociatedSitePinList](#) instead.

## [PowerLineFrequency](#)

The power line frequency value cache for PXI-4110, PXI-4130, and PXle-4154 since they don't support power line frequency property. The cached value is used to convert aperture time value between power line cycles and seconds, for these models.

## [Session](#)

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower session.

# Methods

## [GenerateInstrumentChannelToSitePinDictionary\(I SemiconductorModuleContext\)](#)

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

# Constructor DCPowerSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DCPowerSessionInformation(NIDCPower, string)

Constructs a session information object that associates with a NationalInstruments.ModularInstruments.NIDCPower.NIDCPower instrument session.

```
public DCPowerSessionInformation(NIDCPower session, string allChannelsString)
```

### Parameters

**session** NIDCPower

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower session.

**allChannelsString** [string](#)

The all channels string associated with the NationalInstruments.ModularInstruments.NIDCPower.NIDCPower session.

## DCPowerSessionInformation(NIDCPower, IList<SitePinInfo>)

Constructs a session information object.

```
public DCPowerSessionInformation(NIDCPower session, IList<SitePinInfo>
associatedSitePinList)
```

### Parameters

**session** NIDCPower

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower session.

**associatedSitePinList** [IList](#)<[SitePinInfo](#)>

The specified subset of channels associated with the session.

# Property AllChannelsOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AllChannelsOutput

The all channels NationalInstruments.ModularInstruments.NIDCPower.DCPowerOutput corresponds to the [AllChannelsString](#).

```
public DCPowerOutput AllChannelsOutput { get; }
```

Property Value

DCPowerOutput

# Property AllChannelsString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AllChannelsString

The all channels string associated with the [Session](#).

```
public string AllChannelsString { get; }
```

### Property Value

[string](#) ↗

### Remarks

It's a comma-separated string of instrument-channel pairs, e.g. "SMU\_4147\_C1\_S10/0, SMU\_4147\_C1\_S10/1".

# Property AllInstrumentsAreTheSameModel

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AllInstrumentsAreTheSameModel

Indicates whether all instruments are the same model.

```
public bool AllInstrumentsAreTheSameModel { get; }
```

### Property Value

[bool](#)

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

```
public IList<SitePinInfo> AssociatedSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property ModelString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ModelString

The DCPower model string if all instruments are the same model. Otherwise, it's empty string, please check [ModelString](#) for each element of the [AssociatedSitePinList](#) instead.

```
public string ModelString { get; }
```

### Property Value

[string](#) ↗

# Property PowerLineFrequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PowerLineFrequency

The power line frequency value cache for PXI-4110, PXI-4130, and PXIe-4154 since they don't support power line frequency property. The cached value is used to convert aperture time value between power line cycles and seconds, for these models.

```
public double PowerLineFrequency { get; set; }
```

## Property Value

[double](#)

# Property Session

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Session

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower session.

```
public NIDCPower Session { get; }
```

## Property Value

NIDCPower

# Method GenerateInstrumentChannelToSitePinDictionary

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext )

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

```
public static void GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext  
tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Class DCPowerSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains one or more [DCPowerSessionInformation](#) objects.

```
public class DC PowerSessionsBundle : ISessionsBundle<DCPowerSessionInformation>
```

## Inheritance

[object](#) ← DCPowerSessionsBundle

## Implements

[ISessionsBundle<DCPowerSessionInformation>](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>>, string, string\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, int, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[,\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation>](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]](#),  
[string](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]\[\]](#),  
[string](#)).

## Constructors

[DCPowerSessionsBundle\(ISemiconductorModuleContext, IEnumerable<DCPowerSessionInformation>\)](#)

Constructs a sessions bundle object that represents a bunch of [DCPowerSessionInformations](#).

## Properties

[AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

[InstrumentSessions](#)

An enumerable of [ISessionInformation](#).

[TSMContext](#)

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

# Methods

## [FilterByPin\(string\)](#)

Filter current [DCPowerSessionsBundle](#) and returns a new one with the requested pin.

## [FilterByPin\(string\[\]\)](#)

Filter current [DCPowerSessionsBundle](#) and returns a new one with requested pins.

## [FilterBySite\(int\)](#)

Filters current [DCPowerSessionsBundle](#) and returns a new one with the requested site.

## [FilterBySite\(int\[\]\)](#)

Filters current [DCPowerSessionsBundle](#) and returns a new one with requested sites.

# Constructor DCPowerSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**DCPowerSessionsBundle(ISemiconductorModuleContext, IEnumerable<DCPowerSessionInformation>)**

Constructs a sessions bundle object that represents a bunch of [DCPowerSessionInformations](#).

```
public DCPowerSessionsBundle(ISemiconductorModuleContext tsmContext,  
IEnumerable<DCPowerSessionInformation> allSessionInfo)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

**allSessionInfo** [IEnumerable](#)<DCPowerSessionInformation>

An enumerable of [DCPowerSessionInformations](#).

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

```
public IList<SitePinInfo> AggregateSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [ISessionInformation](#).

```
public IEnumerable<DCPowerSessionInformation> InstrumentSessions { get; }
```

## Property Value

[IEnumerable](#) <[DCPowerSessionInformation](#)>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public ISemiconductorModuleContext TSMContext { get; }
```

## Property Value

ISemiconductorModuleContext

# Method FilterByPin

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterByPin(string)

Filter current [DCPowerSessionsBundle](#) and returns a new one with the requested pin.

```
public DCPowerSessionsBundle FilterByPin(string requestedPin)
```

### Parameters

**requestedPin** [string](#) ↗

The requested pin.

### Returns

[DCPowerSessionsBundle](#)

A new [DCPowerSessionsBundle](#) object with the requested pin.

## FilterByPin(string[])

Filter current [DCPowerSessionsBundle](#) and returns a new one with requested pins.

```
public DCPowerSessionsBundle FilterByPin(string[] requestedPins)
```

### Parameters

**requestedPins** [string](#) ↗[]

The requested pins.

### Returns

## [DCPowerSessionsBundle](#)

A new [DCPowerSessionsBundle](#) object with requested pins.

# Method FilterBySite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterBySite(int)

Filters current [DCPowerSessionsBundle](#) and returns a new one with the requested site.

```
public DCPowerSessionsBundle FilterBySite(int requestedSite)
```

### Parameters

`requestedSite` [int](#)

The requested site.

### Returns

[DCPowerSessionsBundle](#)

A new [DCPowerSessionsBundle](#) object with the requested site.

## FilterBySite(int[])

Filters current [DCPowerSessionsBundle](#) and returns a new one with requested sites.

```
public DCPowerSessionsBundle FilterBySite(int[] requestedSites)
```

### Parameters

`requestedSites` [int](#)[]

The requested sites.

### Returns

## [DCPowerSessionsBundle](#)

A new [DCPowerSessionsBundle](#) object with requested sites.

# Class DCPowerSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines DC Power settings.

```
public class DC PowerSettings
```

## Inheritance

[object](#) ← DCPowerSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DCPowerSettings\(\)](#)

Default constructor.

## Properties

[MeasureSettings](#)

The measure settings.

[SourceSettings](#)

The source settings.

# Constructor DCPowerSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DCPowerSettings()

Default constructor.

```
public DCPowerSettings()
```

# Property MeasureSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureSettings

The measure settings.

```
public DCPowerMeasureSettings MeasureSettings { get; set; }
```

## Property Value

[DCPowerMeasureSettings](#)

# Property SourceSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SourceSettings

The source settings.

```
public DCPowerSourceSettings SourceSettings { get; set; }
```

## Property Value

[DCPowerSourceSettings](#)

# Class DCPowerSourceSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines DC Power source settings.

```
public class DC PowerSourceSettings
```

## Inheritance

[object](#) ← DC PowerSourceSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DCPowerSourceSettings\(\)](#)

Default constructor.

## Properties

### [Level](#)

The voltage or current level.

### [LevelRange](#)

The voltage or current level range.

### [Limit](#)

The current or voltage limit.

### [LimitHigh](#)

The current or voltage high limit.

### [LimitLow](#)

The current or voltage low limit.

## [LimitRange](#)

The current or voltage limit range.

## [LimitSymmetry](#)

The limit symmetry.

## [OutputFunction](#)

The output function.

## [SourceDelayInSeconds](#)

The source delay in seconds.

## [TransientResponse](#)

The transient response.

# Constructor DCPowerSourceSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DCPowerSourceSettings()

Default constructor.

```
public DCPowerSourceSettings()
```

# Property Level

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Level

The voltage or current level.

```
public double? Level { get; set; }
```

## Property Value

[double](#)?

# Property LevelRange

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## LevelRange

The voltage or current level range.

```
public double? LevelRange { get; set; }
```

### Property Value

[double](#)?

# Property Limit

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Limit

The current or voltage limit.

```
public double? Limit { get; set; }
```

## Property Value

[double](#)?

# Property LimitHigh

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## LimitHigh

The current or voltage high limit.

```
public double? LimitHigh { get; set; }
```

### Property Value

[double](#)?

# Property LimitLow

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## LimitLow

The current or voltage low limit.

```
public double? LimitLow { get; set; }
```

## Property Value

[double](#)?

# Property LimitRange

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## LimitRange

The current or voltage limit range.

```
public double? LimitRange { get; set; }
```

## Property Value

[double](#)?

# Property LimitSymmetry

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## LimitSymmetry

The limit symmetry.

```
public DCPowerComplianceLimitSymmetry? LimitSymmetry { get; set; }
```

## Property Value

DCPowerComplianceLimitSymmetry?

# Property OutputFunction

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## OutputFunction

The output function.

```
public DCPowerSourceOutputFunction? OutputFunction { get; set; }
```

## Property Value

DCPowerSourceOutputFunction?

# Property SourceDelayInSeconds

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SourceDelayInSeconds

The source delay in seconds.

```
public double? SourceDelayInSeconds { get; set; }
```

### Property Value

[double](#)?

# Property TransientResponse

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## TransientResponse

The transient response.

```
public DCPowerSourceTransientResponse? TransientResponse { get; set; }
```

## Property Value

DCPowerSourceTransientResponse?

# Class DCOPowerWaveformAcquisitionSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCOPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines DCOPower waveform acquisition settings.

```
public class DCOPowerWaveformAcquisitionSettings
```

## Inheritance

[object](#) ← DCOPowerWaveformAcquisitionSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DCOPowerWaveformAcquisitionSettings\(\)](#)

## Properties

[ApertureTime](#)

The aperture time.

[ApertureTimeUnits](#)

The aperture time units.

[MeasureTriggerType](#)

The measure trigger type.

[MeasureWhen](#)

The measure when.

# Constructor

## DCPowerWaveformAcquisitionSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### DCPowerWaveformAcquisitionSettings()

```
public DCPowerWaveformAcquisitionSettings()
```

# Property ApertureTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApertureTime

The aperture time.

```
public double ApertureTime { get; set; }
```

## Property Value

[double](#)

# Property ApertureTimeUnits

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApertureTimeUnits

The aperture time units.

```
public DCPowerMeasureApertureTimeUnits ApertureTimeUnits { get; set; }
```

## Property Value

DCPowerMeasureApertureTimeUnits

# Property MeasureTriggerType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureTriggerType

The measure trigger type.

```
public DCPowerMeasureTriggerType MeasureTriggerType { get; set; }
```

## Property Value

DCPowerMeasureTriggerType

# Property MeasureWhen

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureWhen

The measure when.

```
public DCPowerMeasurementWhen MeasureWhen { get; set; }
```

## Property Value

DCPowerMeasurementWhen

# Class DCPowerWaveformResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines DCPower waveform results.

```
public class DCPowerWaveformResults
```

## Inheritance

[object](#) ← DCPowerWaveformResults

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DCPowerWaveformResults\(DCPowerFetchResult, double\)](#)

Constructs a DCPower waveform results object.

## Properties

[DeltaTime](#)

The measurement record delta time.

[Result](#)

The DCPower fetch result.

# Constructor DCPowerWaveformResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DCPowerWaveformResults(DCPowerFetchResult, double)

Constructs a DCPower waveform results object.

```
public DCPOWERWAVEFORMRESULTS(DCPOWERFETCHRESULT result, double deltaTime)
```

### Parameters

**result** DCPowerFetchResult

The DCPower fetch result.

**deltaTime** [double](#)

The measurement record delta time.

# Property DeltaTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DeltaTime

The measurement record delta time.

```
public double DeltaTime { get; }
```

## Property Value

[double](#)

# Property Result

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Result

The DCPower fetch result.

```
public DCPowerFetchResult Result { get; }
```

## Property Value

DCPowerFetchResult

# Enum EventType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines DCPower event type.

```
public enum EventType
```

## Fields

**MeasureCompleteEvent = 0**

The measure complete event.

**PulseCompleteEvent = 1**

The pulse complete event.

**ReadyForPulseTriggerEvent = 2**

The ready for pulse trigger event.

**SequenceEngineDoneEvent = 3**

The sequence engine done event.

**SequenceIterationCompleteEvent = 4**

The sequence iteration complete event.

**SourceCompleteEvent = 5**

The source complete event.

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

[Close\(ISemiconductorModuleContext, bool\)](#)

Closes all NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions.

[Initialize\(ISemiconductorModuleContext, bool\)](#)

Initializes NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions in the test system.

[Reset\(ISemiconductorModuleContext, bool\)](#)

Resets all NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions.

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close(ISemiconductorModuleContext, bool)

Closes all NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions.

```
public static void Close(ISemiconductorModuleContext tsmContext, bool reset = true)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**reset** [bool](#)

Whether to reset the device sessions before closing.

# Method Initialize

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Initialize(ISemiconductorModuleContext, bool)

Initializes NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions in the test system.

```
public static void Initialize(ISemiconductorModuleContext tsmContext, bool resetDevice  
= false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset device during initialization.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset(ISemiconductorModuleContext, bool)

Resets all NationalInstruments.ModularInstruments.NIDCPower.NIDCPower sessions.

```
public static void Reset(ISemiconductorModuleContext tsmContext, bool resetDevice = false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**resetDevice** [bool](#)

Whether to perform a hard reset on the device.

# Class Measure

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for DCPower measurements.

```
public static class Measure
```

## Inheritance

[object](#) ← Measure

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[AcquireSynchronizedWaveforms\(DCPowerSessionsBundle, double, double\)](#).

Configures and acquires waveforms synchronized across the pins.

[ConfigureAndStartWaveformAcquisition\(DCPowerSessionsBundle, double, double\)](#).

Configures and starts a waveform acquisition.

[ConfigureApertureTime\(NIDCPower, string, string, double, double, DCPowerMeasureApertureTimeUnits?\)](#).

Configures the aperture time.

[ConfigureApertureTimeUnits\(NIDCPower, string, string, DCPowerMeasureApertureTimeUnits\)](#).

Configures the aperture time units.

[ConfigureMeasureSettings\(DCPowerSessionsBundle, PinSiteData<DCPowerMeasureSettings>\)](#).

Configures one or more measure settings based on the values populated within a [DCPowerMeasure Settings](#) object. Accepts a scalar input of type [DCPowerMeasureSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

[ConfigureMeasureSettings\(DCPowerSessionsBundle, SiteData<DCPowerMeasureSettings>\)](#).

Configures one or more measure settings based on the values populated within a [DCPowerMeasure Settings](#) object. Accepts a scalar input of type [DCPowerMeasureSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

## [ConfigureMeasureSettings\(DCPowerSessionsBundle, DCPowerMeasureSettings\)](#)

Configures one or more measure settings based on the values populated within a [DCPowerMeasureSettings](#) object. Accepts a scalar input of type [DCPowerMeasureSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

## [ConfigureMeasureSettings\(DCPowerSessionsBundle, IDictionary<string, DCPowerMeasureSettings>\)](#)

Configures [DCPowerMeasureSettings](#).

## [ConfigureMeasureWhen\(NIDCPower, string, string, DCPowerMeasurementWhen\)](#)

Configures the MeasurementWhen property.

## [ConfigureMeasureWhen\(DCPowerSessionsBundle, DCPowerMeasurementWhen\)](#)

Configures the DC Power MeasurementWhen property.

## [ConfigureMeasurementSense\(NIDCPower, string, string, DCPowerMeasurementSense\)](#)

Configures the measurement sense.

## [ConfigureMeasurementSense\(DCPowerSessionsBundle, DCPowerMeasurementSense\)](#)

Configures the measurement sense.

## [ConfigurePowerLineFrequency\(DCPowerSessionsBundle, double\)](#)

Configures the power line frequency in Hz (double).

## [FetchMeasurement\(DCPowerSessionsBundle, int, double\)](#)

Fetches results from a previous measurement.

## [FinishWaveformAcquisition\(DCPowerSessionsBundle, double,](#)

## [PinSiteData<DCPowerWaveformAcquisitionSettings>\)](#)

Fetches waveform acquisition results as a pin- and site-aware data object.

## [GetApertureTimeInSeconds\(DCPowerSessionsBundle, out double\)](#)

Gets aperture time in seconds.

## [GetPowerLineFrequency\(DCPowerSessionsBundle\)](#)

Gets the power line frequency.

## [MeasureAndPublishCurrent\(DCPowerSessionsBundle, string\)](#)

Measures the current on the target pin(s) and immediately publishes the results using the [publishedDataId](#) passed in.

## [MeasureAndPublishCurrent\(DCPowerSessionsBundle, string, out double\[\]\)](#)

Measures the current on the target pin(s) and immediately publishes the results using the [publishedDataId](#) passed in.

## [MeasureAndPublishVoltage\(DCPowerSessionsBundle, string\)](#)

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

[MeasureAndPublishVoltage\(DCPowerSessionsBundle, string, out double\[\]\[\]\)](#)

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

[MeasureAndReturnPerInstrumentPerChannelResults\(DCPowerSessionsBundle\)](#)

Measures and returns per-instrument per-channel results.

[MeasureAndReturnPerSitePerPinResults\(DCPowerSessionsBundle\)](#)

Measures and returns per-site per-pin results.

[MeasureCurrent\(DCPowerSessionsBundle\)](#)

Measures the current on the target pin(s) and returns a pin- and site-aware data object.

[MeasureVoltage\(DCPowerSessionsBundle\)](#)

Measures the voltage on the target pin(s) and returns a pin- and site-aware data object.

[MeasureVoltageAndCurrent\(DCPowerSessionInformation\)](#)

Measures the voltage and current.

# Method AcquireSynchronizedWaveforms

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## AcquireSynchronizedWaveforms(DCPowerSessionsBundle, double, double)

Configures and acquires waveforms synchronized across the pins.

```
public static PinSiteData<DCPowerFetchResult> AcquireSynchronizedWaveforms(this  
DCPowerSessionsBundle sessionsBundle, double apertureTimeInSeconds = 0, double  
measurementTimeInSeconds = 0)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**apertureTimeInSeconds** [double](#)

The measurement aperture time in seconds.

**measurementTimeInSeconds** [double](#)

The measurement time in seconds.

### Returns

[PinSiteData](#)<DCPowerFetchResult>

The per-site per-pin waveform results.

# Method

## ConfigureAndStartWaveformAcquisition

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### ConfigureAndStartWaveformAcquisition(DCPowerSessionsBundle, double, double)

Configures and starts a waveform acquisition.

```
public static PinSiteData<DCPowerWaveformAcquisitionSettings>
ConfigureAndStartWaveformAcquisition(this DCPowerSessionsBundle sessionsBundle, double
sampleRate, double bufferLength)
```

#### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**sampleRate** [double](#) ↗

The sample rate.

**bufferLength** [double](#) ↗

The buffer length in seconds.

#### Returns

[PinSiteData<DCPowerWaveformAcquisitionSettings>](#)

The original settings for the channels that do waveform acquisition.

# Method ConfigureApertureTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigureApertureTime(NIDCPower, string, string, double, double, DCPowerMeasureApertureTimeUnits?)**

Configures the aperture time.

```
public static void ConfigureApertureTime(this NIDCPower session, string channelString,  
string modelString, double powerLineFrequency, double apertureTime,  
DCPowerMeasureApertureTimeUnits? apertureTimeUnits)
```

## Parameters

**session** NIDCPower

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower object.

**channelString** [string](#)

The channel string.

**modelString** [string](#)

The DCPower instrument model.

**powerLineFrequency** [double](#)

The power line frequency used to calculate aperture time value from power line cycles to seconds. This is used just for PXI-4110, PXI-4130, and PXIe-4154 models since they don't support power line frequency property.

**apertureTime** [double](#)

The aperture time to set.

**apertureTimeUnits** DCPowerMeasureApertureTimeUnits?

The aperture time units to set.



# Method ConfigureApertureTimeUnits

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigureApertureTimeUnits(NIDCPower, string, string, DCPowerMeasureApertureTimeUnits)**

Configures the aperture time units.

```
public static void ConfigureApertureTimeUnits(this NIDCPower session, string channelString,  
    string modelString, DCPowerMeasureApertureTimeUnits apertureTimeUnits)
```

Parameters

**session** NIDCPower

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower object.

**channelString** [string](#)

The channel string.

**modelString** [string](#)

The DCPower instrument model.

**apertureTimeUnits** DCPowerMeasureApertureTimeUnits

The aperture time units to set.

# Method ConfigureMeasureSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureMeasureSettings(DCPowerSessionsBundle, DCPowerMeasureSettings)

Configures one or more measure settings based on the values populated within a [DCPowerMeasureSettings](#) object. Accepts a scalar input of type [DCPowerMeasureSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureMeasureSettings(this DCPowerSessionsBundle sessionsBundle,  
DCPowerMeasureSettings settings)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**settings** [DCPowerMeasureSettings](#)

The measure settings to configure.

## ConfigureMeasureSettings(DCPowerSessionsBundle, SiteData<DCPowerMeasureSettings>)

Configures one or more measure settings based on the values populated within a [DCPowerMeasureSettings](#) object. Accepts a scalar input of type [DCPowerMeasureSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureMeasureSettings(this DCPowerSessionsBundle sessionsBundle,  
SiteData<DCPowerMeasureSettings> settings)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`settings` [SiteData<DCPowerMeasureSettings>](#)

The measure settings to configure.

## ConfigureMeasureSettings(DCPowerSessionsBundle, PinSiteData<DCPowerMeasureSettings>)

Configures one or more measure settings based on the values populated within a [DCPowerMeasure Settings](#) object. Accepts a scalar input of type [DCPowerMeasureSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureMeasureSettings(this DCPowerSessionsBundle sessionsBundle,  
PinSiteData<DCPowerMeasureSettings> settings)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`settings` [PinSiteData<DCPowerMeasureSettings>](#)

The measure settings to configure.

## ConfigureMeasureSettings(DCPowerSessionsBundle, IDictionary<string, DCPowerMeasureSettings>)

Configures [DCPowerMeasureSettings](#).

```
public static void ConfigureMeasureSettings(this DCPowerSessionsBundle sessionsBundle,  
IDictionary<string, DCPowerMeasureSettings> settings)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**settings** [IDictionary](#)<[string](#), [DCPowerMeasureSettings](#)>

The specific settings to configure.

# Method ConfigureMeasureWhen

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureMeasureWhen(DCPowerSessionsBundle, DCPowerMeasurementWhen)

Configures the DCPower MeasurementWhen property.

```
public static void ConfigureMeasureWhen(this DCPowerSessionsBundle sessionsBundle,  
DCPowerMeasurementWhen measureWhen)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**measureWhen** DCPowerMeasurementWhen

The MeasurementWhen property to set.

## ConfigureMeasureWhen(NIDCPower, string, string, DCPowerMeasurementWhen)

Configures the MeasurementWhen property.

```
public static void ConfigureMeasureWhen(this NIDCPower session, string channelString, string  
modelString, DCPowerMeasurementWhen measureWhen)
```

### Parameters

**session** NIDCPower

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower object.

**channelString** [string](#)

The channel string.

**modelString** [string](#)

The DCPower instrument model.

**measureWhen** DCPowerMeasurementWhen

The measurement when to set.

# Method ConfigureMeasurementSense

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureMeasurementSense(DCPowerSessionsBundle, DCPowerMeasurementSense)

Configures the measurement sense.

```
public static void ConfigureMeasurementSense(this DCPowerSessionsBundle sessionsBundle,  
DCPowerMeasurementSense sense)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**sense** DCPowerMeasurementSense

The measurement sense to set.

## ConfigureMeasurementSense(NIDCPower, string, string, DCPowerMeasurementSense)

Configures the measurement sense.

```
public static void ConfigureMeasurementSense(this NIDCPower session, string channelString,  
string modelString, DCPowerMeasurementSense sense)
```

### Parameters

**session** NIDCPower

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower object.

**channelString** [string](#)

The channel string.

**modelString** [string](#)

The DCPower instrument model.

**sense** DCPowerMeasurementSense

The measurement sense to set.

# Method ConfigurePowerLineFrequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigurePowerLineFrequency(DCPowerSessionsBundle, double)

Configures the power line frequency in Hz (double).

```
public static void ConfigurePowerLineFrequency(this DCPowerSessionsBundle sessionsBundle,  
    double frequency)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**frequency** [double](#) ↗

The power line frequency in Hz to set.

### Remarks

This method should only be invoked once when driver sessions are first initialized.

# Method FetchMeasurement

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## FetchMeasurement(DCPowerSessionsBundle, int, double)

Fetches results from a previous measurement.

```
public static PinSiteData<SingleDCPowerFetchResult[]> FetchMeasurement(this
DCPowerSessionsBundle sessionsBundle, int pointsToFetch = 1, double timeoutInSeconds = 10)
```

### Parameters

[sessionsBundle](#) [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

[pointsToFetch](#) [int](#)

The number of points to Fetch.

[timeoutInSeconds](#) [double](#)

The time to wait before the operation is aborted.

### Returns

[PinSiteData](#)<[SingleDCPowerFetchResult](#)[]>

A [PinSiteData](#)<[T](#)> object that contains an array of [SingleDCPowerFetchResult](#) values, where each [SingleDCPowerFetchResult](#) object contains the voltage, current, and inCompliance result for a simple sample/point from the previous measurement.

### Remarks

This method should not be used when the MeasureWhen property is OnDemand.

# Method FinishWaveformAcquisition

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**FinishWaveformAcquisition(DCPowerSessionsBundle, double, PinSiteData<DCPowerWaveformAcquisitionSettings>)**

Fetches waveform acquisition results as a pin- and site-aware data object.

```
public static PinSiteData<DCPowerWaveformResults> FinishWaveformAcquisition(this  
DCPowerSessionsBundle sessionsBundle, double fetchWaveformLength,  
PinSiteData<DCPowerWaveformAcquisitionSettings> originalSettings)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**fetchWaveformLength** [double](#)

The waveform length in seconds to fetch.

**originalSettings** [PinSiteData<DCPowerWaveformAcquisitionSettings>](#)

The original settings for the channels that do waveform acquisition. This is the return value of the ConfigureAndStartWaveformAcquisition method.

## Returns

[PinSiteData<DCPowerWaveformResults>](#)

The per-site per-pin waveform results.

# Method GetApertureTimeInSeconds

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetApertureTimeInSeconds(DCPowerSessionsBundle, out double)

Gets aperture time in seconds.

```
public static PinSiteData<double> GetApertureTimeInSeconds(this DCPowerSessionsBundle sessionsBundle, out double maximumApertureTime)
```

### Parameters

sessionsBundle [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

maximumApertureTime [double](#)

Returns the maximum aperture time.

### Returns

[PinSiteData<double>](#)

The per-site per-pin aperture times.

# Method GetPowerLineFrequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetPowerLineFrequency(DCPowerSessionsBundle)

Gets the power line frequency.

```
public static PinSiteData<double> GetPowerLineFrequency(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Returns

[PinSiteData<double>](#)

The per-site per-pin power line frequencies.

# Method MeasureAndPublishCurrent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureAndPublishCurrent(DCPowerSessionsBundle, string, out double[][])

Measures the current on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static void MeasureAndPublishCurrent(this DCPowerSessionsBundle sessionsBundle,  
    string publishedDataId, out double[][] currentMeasurements)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`publishedDataId` [string](#)

The unique data id to use when publishing.

`currentMeasurements` [double](#)[][]

The returned current measurements.

### Remarks

Use this method for the fastest test time if the measurement results do not needed for any other operations. Otherwise, use the override for this method that returns `PinSiteData`.

## MeasureAndPublishCurrent(DCPowerSessionsBundle, string)

Measures the current on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static PinSiteData<double> MeasureAndPublishCurrent(this DCPowerSessionsBundle sessionsBundle, string publishedDataId)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**publishedDataId** [string](#)

The unique data id to use when publishing.

## Returns

[PinSiteData<double>](#)

The pin-site aware current measurements.

# Method MeasureAndPublishVoltage

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureAndPublishVoltage(DCPowerSessionsBundle, string, out double[][])

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static void MeasureAndPublishVoltage(this DCPowerSessionsBundle sessionsBundle,  
    string publishedDataId, out double[][] voltageMeasurements)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`publishedDataId` [string](#)

The unique data id to use when publishing.

`voltageMeasurements` [double](#)[][]

The returned voltage measurements.

### Remarks

Use this method for the fastest test time if the measurement results are not needed for any other operations. Otherwise, use the override for this method that returns `PinSiteData`.

## MeasureAndPublishVoltage(DCPowerSessionsBundle, string)

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static PinSiteData<double> MeasureAndPublishVoltage(this DCPowerSessionsBundle sessionsBundle, string publishedDataId)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**publishedDataId** [string](#)

The unique data id to use when publishing.

## Returns

[PinSiteData<double>](#)

The pin-site aware voltage measurements.

# Method

## MeasureAndReturnPerInstrumentPerChannelResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### MeasureAndReturnPerInstrumentPerChannelResults(DCPowerSessionsBundle)

Measures and returns per-instrument per-channel results.

```
public static Tuple<double[][][], double[][][]>
MeasureAndReturnPerInstrumentPerChannelResults(this DCPowerSessionsBundle sessionsBundle)
```

#### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

#### Returns

[Tuple](#)<[double](#)[], [double](#)[][], [double](#)[][][]>

The measurements in per-instrument per-channel format. Item1 is voltage measurements, Item2 is current measurements.

# Method

## MeasureAndReturnPerSitePerPinResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### MeasureAndReturnPerSitePerPinResults(DCPowerSessionsBundle)

Measures and returns per-site per-pin results.

```
public static Tuple<PinSiteData<double>, PinSiteData<double>>
MeasureAndReturnPerSitePerPinResults(this DCPowerSessionsBundle sessionsBundle)
```

#### Parameters

[sessionsBundle](#) [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

#### Returns

[Tuple](#)<[PinSiteData](#)<[double](#)>, [PinSiteData](#)<[double](#)>>

The measurements in per-site per-pin format. Item1 is voltage measurements, Item2 is current measurements.

# Method MeasureCurrent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureCurrent(DCPowerSessionsBundle)

Measures the current on the target pin(s) and returns a pin- and site-aware data object.

```
public static PinSiteData<double> MeasureCurrent(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Returns

[PinSiteData<double>](#)

The per-pin per-site voltage measurements.

# Method MeasureVoltage

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureVoltage(DCPowerSessionsBundle)

Measures the voltage on the target pin(s) and returns a pin- and site-aware data object.

```
public static PinSiteData<double> MeasureVoltage(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Returns

[PinSiteData<double>](#)

The per-pin per-site voltage measurements.

# Method MeasureVoltageAndCurrent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureVoltageAndCurrent(DCPowerSessionInformation)

Measures the voltage and current.

```
public static Tuple<double[], double[]> MeasureVoltageAndCurrent(this  
DCPowerSessionInformation sessionInfo)
```

### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

### Returns

[Tuple](#)<[double](#)[], [double](#)[]>

The measurements. Item1 is voltage measurements, Item2 is current measurements.

# Struct SingleDCPowerFetchResult

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Structure that defines the result of a single fetch operation.

```
public readonly struct SingleDCPowerFetchResult
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#)

## Constructors

[SingleDCPowerFetchResult\(double, double, bool\)](#)

Constructs a SingleDCPowerFetchResult object.

## Properties

[CurrentMeasurement](#)

Current measurement value.

[InCompliance](#)

Whether the output was in compliance mode when measurement was taken.

[VoltageMeasurement](#)

Voltage measurement value.

# Constructor SingleDCPowerFetchResult

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SingleDCPowerFetchResult(double, double, bool)

Constructs a SingleDCPowerFetchResult object.

```
public SingleDCPowerFetchResult(double voltageMeasurement, double currentMeasurement,  
bool inCompliance)
```

### Parameters

voltageMeasurement [double](#)

Voltage measurement value

currentMeasurement [double](#)

Current measurement value.

inCompliance [bool](#)

Whether the output is in compliance mode when measurement is taken.

# Property CurrentMeasurement

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CurrentMeasurement

Current measurement value.

```
public double CurrentMeasurement { get; }
```

## Property Value

[double](#) ↗

# Property InCompliance

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## InCompliance

Whether the output was in compliance mode when measurement was taken.

```
public bool InCompliance { get; }
```

## Property Value

[bool](#) ↗

# Property VoltageMeasurement

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## VoltageMeasurement

Voltage measurement value.

```
public double VoltageMeasurement { get; }
```

### Property Value

[double](#)

# Class Source

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for DCPower voltage/current sourcing.

```
public static class Source
```

## Inheritance

[object](#) ← Source

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[CheckDCVoltageModeAndLevels\(DCPowerSessionsBundle, out IEnumerable<string>, IDictionary<string, double>\)](#)

Checks if the output function is set to DCVoltage and the level(s) are set to the expected values.

[ConfigureCurrentLimit\(DCPowerOutput, double, double?\)](#)

Configures the current limit.

[ConfigureCurrentLimit\(DCPowerSessionsBundle, double, double?\)](#)

Configures the current limit.

[ConfigureOutputConnected\(DCPowerSessionsBundle, PinSiteData<bool>\)](#)

Configures the output relay of the underlying device channel (s) to either be connected (closed) or disconnected (open). Accepts a scalar input of type [bool](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

Pass this method a true value to physically disconnect the output terminal from the front panel.

Disconnect the output only if disconnecting is necessary for your application. For example, a battery connected to the output terminal might discharge unless the relay is disconnected. Excessive

connecting and disconnecting of the output can cause premature wear on the relay. This property is not supported by all devices. Refer to the Supported Properties by Device topic in the NI-DCPower LabVIEW VI Reference and the document of your SMU model for information about supported devices.

#### [ConfigureOutputConnected\(DCOPowerSessionsBundle, SiteData<bool>\)](#)

Configures the output relay of the underlying device channel (s) to either be connected (closed) or disconnected (open). Accepts a scalar input of type [bool](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

Pass this method a true value to physically disconnect the output terminal from the front panel.

Disconnect the output only if disconnecting is necessary for your application. For example, a battery connected to the output terminal might discharge unless the relay is disconnected. Excessive connecting and disconnecting of the output can cause premature wear on the relay.

This property is not supported by all devices. Refer to the Supported Properties by Device topic in the NI-DCPower LabVIEW VI Reference and the document of your SMU model for information about supported devices.

#### [ConfigureOutputConnected\(DCOPowerSessionsBundle, bool\)](#)

Configures the output relay of the underlying device channel (s) to either be connected (closed) or disconnected (open). Accepts a scalar input of type [bool](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

Pass this method a true value to physically disconnect the output terminal from the front panel.

Disconnect the output only if disconnecting is necessary for your application. For example, a battery connected to the output terminal might discharge unless the relay is disconnected. Excessive connecting and disconnecting of the output can cause premature wear on the relay.

This property is not supported by all devices. Refer to the Supported Properties by Device topic in the NI-DCPower LabVIEW VI Reference and the document of your SMU model for information about supported devices.

#### [ConfigureOutputEnabled\(DCOPowerSessionsBundle, PinSiteData<bool>\)](#)

Configures whether to enable (true) or disable (false) output generation on the underlying device channel(s).

#### [ConfigureOutputEnabled\(DCOPowerSessionsBundle, SiteData<bool>\)](#)

Configures whether to enable (true) or disable (false) output generation on the underlying device channel(s).

#### [ConfigureOutputEnabled\(DCOPowerSessionsBundle, bool\)](#)

Configures whether to enable (true) or disable (false) output generation on the underlying device channel(s).

[ConfigureSequence\(DCPowerOutput, double\[\], int, double?\)](#)

Configures a hardware-timed sequence of values.

[ConfigureSequence\(DCPowerSessionsBundle, double\[\], int, double?\)](#)

Configures a hardware-timed sequence of values.

[ConfigureSourceDelay\(DCPowerSessionsBundle, PinSiteData<double>\)](#)

Configures the source delay. With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

[ConfigureSourceDelay\(DCPowerSessionsBundle, SiteData<double>\)](#)

Configures the source delay. With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

[ConfigureSourceDelay\(DCPowerSessionsBundle, double\)](#)

Configures the source delay. With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

[ConfigureSourceSettings\(DCPowerSessionInformation, DCPowerSourceSettings, string\)](#)

Configures [DCPowerSourceSettings](#).

[ConfigureSourceSettings\(DCPowerSessionsBundle, PinSiteData<DCPowerSourceSettings>\)](#)

Configures one or more source settings based on values populated within a [DCPowerSourceSettings](#) object. Accepts a scalar input of type [DCPowerSourceSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

[ConfigureSourceSettings\(DCPowerSessionsBundle, SiteData<DCPowerSourceSettings>\)](#)

Configures one or more source settings based on values populated within a [DCPowerSourceSettings](#) object. Accepts a scalar input of type [DCPowerSourceSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

[ConfigureSourceSettings\(DCPowerSessionsBundle, DCPowerSourceSettings\)](#)

Configures one or more source settings based on values populated within a [DCPowerSourceSettings](#) object. Accepts a scalar input of type [DCPowerSourceSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

[ConfigureTransientResponse\(NIDCPower, string, string, DCPowerSourceTransientResponse\)](#)

Configures the transient response.

[ForceCurrent\(DCPowerSessionsBundle, IDictionary<string, DCPowerSourceSettings>, bool\)](#)

Forces current and specifies symmetric voltage limits.

[ForceCurrent\(DCPowerSessionsBundle, double, double?, double?, double?, bool\)](#)

Forces current on the target pins at the specified level. Must at least provide a level value, and the method will assume all other properties that have been previously set. Optionally, can also provide a specific voltage limit, current level range, voltage limit range values directly.

#### [ForceCurrentAsymmetricLimit\(DCPowerSessionsBundle, double, double, double, double?, double?, bool\)](#)

Behaves the same as the ForceCurrent() method, but has two voltage limit inputs for setting separate high and low voltage limits.

#### [ForceVoltage\(DCPowerSessionsBundle, PinSiteData<double>, double?, double?, double?, bool\)](#)

Forces voltage on the target pins at the specified level. Must at least provide a level value, and the method will assume all other properties that have been previously set. Optionally, can also provide a specific current limit, current limit range, voltage level range values directly.

#### [ForceVoltage\(DCPowerSessionsBundle, IDictionary<string, DCPowerSourceSettings>, bool\)](#)

Forces voltage and specifies symmetric current limit.

#### [ForceVoltage\(DCPowerSessionsBundle, double, double?, double?, double?, bool\)](#)

Forces voltage on the target pins at the specified level. Must at least provide a level value, and the method will assume all other properties that have been previously set. Optionally, can also provide a specific current limit, current limit range, voltage level range values directly.

#### [ForceVoltageAsymmetricLimit\(DCPowerSessionsBundle, double, double, double, double?, double?, bool\)](#)

Behaves the same as the ForceVoltage() method, but as two current limit inputs for setting separate high and low current limits.

#### [GetCurrentLimits\(DCPowerSessionsBundle\)](#)

Gets the current limits.

#### [GetSourceDelayInSeconds\(DCPowerSessionsBundle\)](#)

Gets the source delay in seconds for each of the underlying device channel(s), per-pin and per-site.

#### [PowerDown\(DCPowerSessionsBundle, double?\)](#)

Powers down the channel.

# Method CheckDCVoltageModeAndLevels

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**CheckDCVoltageModeAndLevels(DCPowerSessionsBundle, out  
IEnumerable<string>, IDictionary<string, double>)**

Checks if the output function is set to DCVoltage and the level(s) are set to the expected values.

```
public static bool CheckDCVoltageModeAndLevels(this DCPowerSessionsBundle sessionsBundle,  
    out IEnumerable<string> failedChannels, IDictionary<string, double> expectedVoltages = null)
```

Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**failedChannels** [IEnumerable<string>](#)

Returns the channels that fail the check.

**expectedVoltages** [IDictionary<string, double>](#)

The expected per-pin voltages.

Returns

[bool](#)

Whether all channels pass the check.

# Method ConfigureCurrentLimit

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureCurrentLimit(DCPowerSessionsBundle, double, double?)

Configures the current limit.

```
public static void ConfigureCurrentLimit(this DCPowerSessionsBundle sessionsBundle, double currentLimit, double? currentLimitRange = null)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`currentLimit` [double](#) ↗

The current limit to set.

`currentLimitRange` [double](#) ↗?

The current limit range to set. Use the absolute value of current limit to set current limit range when this parameter is not specified.

## ConfigureCurrentLimit(DCPowerOutput, double, double?)

Configures the current limit.

```
public static void ConfigureCurrentLimit(this DCPowerOutput output, double currentLimit, double? currentLimitRange = null)
```

### Parameters

`output` DCPowerOutput

The NationalInstruments.ModularInstruments.NIDCPower.DCPowerOutput object.

**currentLimit** [double](#)

The current limit to set.

**currentLimitRange** [double](#)?

The current limit range to set. Use the absolute value of current limit to set current limit range when this parameter is not specified.

# Method ConfigureOutputConnected

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureOutputConnected(DCPowerSessionsBundle, bool)

Configures the output relay of the underlying device channel (s) to either be connected (closed) or disconnected (open). Accepts a scalar input of type [bool](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

Pass this method a true value to physically disconnect the output terminal from the front panel.

Disconnect the output only if disconnecting is necessary for your application. For example, a battery connected to the output terminal might discharge unless the relay is disconnected. Excessive connecting and disconnecting of the output can cause premature wear on the relay.

This property is not supported by all devices. Refer to the Supported Properties by Device topic in the NI-DCPower LabVIEW VI Reference and the document of your SMU model for information about supported devices.

```
public static void ConfigureOutputConnected(this DCPowerSessionsBundle sessionsBundle,  
    bool connectOutput)
```

### Parameters

[sessionsBundle](#) [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

[connectOutput](#) [bool](#)

The boolean value to either connect (true) or disconnect (false) the output terminal.

## ConfigureOutputConnected(DCPowerSessionsBundle, SiteData<bool>)

Configures the output relay of the underlying device channel (s) to either be connected (closed) or disconnected (open). Accepts a scalar input of type [bool](#). With overrides for [SiteData<T>](#), and

[PinSiteData<T>](#) input.

Pass this method a true value to physically disconnect the output terminal from the front panel.

Disconnect the output only if disconnecting is necessary for your application. For example, a battery connected to the output terminal might discharge unless the relay is disconnected. Excessive connecting and disconnecting of the output can cause premature wear on the relay.

This property is not supported by all devices. Refer to the Supported Properties by Device topic in the NI-DCPower LabVIEW VI Reference and the document of your SMU model for information about supported devices.

```
public static void ConfigureOutputConnected(this DCPowerSessionsBundle sessionsBundle,  
SiteData<bool> connectOutput)
```

## Parameters

[sessionsBundle](#) [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

[connectOutput](#) [SiteData<bool>](#)

The boolean value to either connect (true) or disconnect (false) the output terminal.

## ConfigureOutputConnected(DCPowerSessionsBundle, PinSiteData<bool>)

Configures the output relay of the underlying device channel (s) to either be connected (closed) or disconnected (open). Accepts a scalar input of type [bool](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

Pass this method a true value to physically disconnect the output terminal from the front panel.

Disconnect the output only if disconnecting is necessary for your application. For example, a battery connected to the output terminal might discharge unless the relay is disconnected. Excessive connecting and disconnecting of the output can cause premature wear on the relay.

This property is not supported by all devices. Refer to the Supported Properties by Device topic in the NI-DCPower LabVIEW VI Reference and the document of your SMU model for information about supported devices.

```
public static void ConfigureOutputConnected(this DCPowerSessionsBundle sessionsBundle,
```

```
PinSiteData<bool> connectOutput)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**connectOutput** [PinSiteData<bool>](#)

The boolean value to either connect (true) or disconnect (false) the output terminal.

# Method ConfigureOutputEnabled

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureOutputEnabled(DCPowerSessionsBundle, bool)

Configures whether to enable (true) or disable (false) output generation on the underlying device channel(s).

```
public static void ConfigureOutputEnabled(this DCPowerSessionsBundle sessionsBundle,  
bool enableOutput)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**enableOutput** [bool](#)

The boolean value to either enable (true) or disable (false) the output .

### Remarks

Note: This property does not change the ConfigureOutputEnabled method. They are independent of each other.

## ConfigureOutputEnabled(DCPowerSessionsBundle, SiteData<bool>)

Configures whether to enable (true) or disable (false) output generation on the underlying device channel(s).

```
public static void ConfigureOutputEnabled(this DCPowerSessionsBundle sessionsBundle,  
SiteData<bool> enableOutput)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**enableOutput** [SiteData<bool>](#)

The boolean value to either enable (true) or disable (false) the output .

## Remarks

Note: This property does not change the ConfigureOutputEnabled method. They are independent of each other.

## ConfigureOutputEnabled(DCPowerSessionsBundle, PinSiteData<bool>)

Configures whether to enable (true) or disable (false) output generation on the underlying device channel(s).

```
public static void ConfigureOutputEnabled(this DCPowerSessionsBundle sessionsBundle,  
PinSiteData<bool> enableOutput)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**enableOutput** [PinSiteData<bool>](#)

The boolean value to either enable (true) or disable (false) the output .

## Remarks

Note: This property does not change the ConfigureOutputEnabled method. They are independent of each other.

# Method ConfigureSequence

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSequence(DCPowerSessionsBundle, double[], int, double?)

Configures a hardware-timed sequence of values.

```
public static void ConfigureSequence(this DCPowerSessionsBundle sessionsBundle, double[] sequence, int sequenceLoopCount, double? sequenceStepDeltaTimeInSeconds = null)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`sequence` [double\[\]](#)

The voltage or current sequence to set.

`sequenceLoopCount` [int](#)

The number of loops a sequence runs after initiation.

`sequenceStepDeltaTimeInSeconds` [double?](#)

The delta time between the start of two consecutive steps in a sequence.

## ConfigureSequence(DCPowerOutput, double[], int, double?)

Configures a hardware-timed sequence of values.

```
public static void ConfigureSequence(this DCPowerOutput output, double[] sequence, int sequenceLoopCount, double? sequenceStepDeltaTimeInSeconds = null)
```

## Parameters

**output** DCPowerOutput

The NationalInstruments.ModularInstruments.NIDCPower.DCPowerOutput object.

**sequence** [double](#)[]

The voltage or current sequence to set.

**sequenceLoopCount** [int](#)

The number of loops a sequence runs after initiation.

**sequenceStepDeltaTimeInSeconds** [double](#)?

The delta time between the start of two consecutive steps in a sequence.

# Method ConfigureSourceDelay

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSourceDelay(DCPowerSessionsBundle, double)

Configures the source delay. With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureSourceDelay(this DCPowerSessionsBundle sessionsBundle,  
double sourceDelayInSeconds)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**sourceDelayInSeconds** [double](#)

The double value of the source delay in seconds.

## ConfigureSourceDelay(DCPowerSessionsBundle, SiteData<double>)

Configures the source delay. With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureSourceDelay(this DCPowerSessionsBundle sessionsBundle,  
SiteData<double> sourceDelayInSeconds)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**sourceDelayInSeconds** [SiteData<double>](#)

The double value of the source delay in seconds.

## ConfigureSourceDelay(DCPowerSessionsBundle, PinSiteData<double>)

Configures the source delay. With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureSourceDelay(this DCPowerSessionsBundle sessionsBundle,  
PinSiteData<double> sourceDelayInSeconds)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**sourceDelayInSeconds** [PinSiteData<double>](#)

The double value of the source delay in seconds.

# Method ConfigureSourceSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSourceSettings(DCPowerSessionsBundle, DCPowerSourceSettings)

Configures one or more source settings based on values populated within a [DCPowerSourceSettings](#) object. Accepts a scalar input of type [DCPowerSourceSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureSourceSettings(this DCPowerSessionsBundle sessionsBundle,  
DCPowerSourceSettings settings)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**settings** [DCPowerSourceSettings](#)

The source settings to configure.

## ConfigureSourceSettings(DCPowerSessionsBundle, SiteData<DCPowerSourceSettings>)

Configures one or more source settings based on values populated within a [DCPowerSourceSettings](#) object. Accepts a scalar input of type [DCPowerSourceSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureSourceSettings(this DCPowerSessionsBundle sessionsBundle,  
SiteData<DCPowerSourceSettings> settings)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`settings` [SiteData<DCPowerSourceSettings>](#)

The source settings to configure.

## ConfigureSourceSettings(DCPowerSessionsBundle, PinSiteData<DCPowerSourceSettings>)

Configures one or more source settings based on values populated within a [DCPowerSourceSettings](#) object. Accepts a scalar input of type [DCPowerSourceSettings](#). With overrides for [SiteData<T>](#), and [PinSiteData<T>](#) input.

```
public static void ConfigureSourceSettings(this DCPowerSessionsBundle sessionsBundle,  
PinSiteData<DCPowerSourceSettings> settings)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`settings` [PinSiteData<DCPowerSourceSettings>](#)

The source settings to configure.

## ConfigureSourceSettings(DCPowerSessionInformation, DCPowerSourceSettings, string)

Configures [DCPowerSourceSettings](#).

```
public static void ConfigureSourceSettings(this DCPowerSessionInformation sessionInfo,  
DCPowerSourceSettings settings, string channelString = "")
```

### Parameters

`sessionInfo` [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

### **settings** [DCPowerSourceSettings](#)

The source settings to configure.

### **channelString** [string](#) ↗

The channel string. Empty string means all channels in the session.

# Method ConfigureTransientResponse

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTransientResponse(NIDCPower, string, string, DCPowerSourceTransientResponse)

Configures the transient response.

```
public static void ConfigureTransientResponse(this NIDCPower session, string channelString,  
    string modelString, DCPowerSourceTransientResponse transientResponse)
```

### Parameters

**session** NIDCPower

The NationalInstruments.ModularInstruments.NIDCPower.NIDCPower object.

**channelString** [string](#)

The channel string.

**modelString** [string](#)

The DCPower instrument model [DCPowerModelStrings](#).

**transientResponse** DCPowerSourceTransientResponse

The transient response to set.

# Method ForceCurrent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ForceCurrent(DCPowerSessionsBundle, double, double?, double?, double?, bool)

Forces current on the target pins at the specified level. Must at least provide a level value, and the method will assume all other properties that have been previously set. Optionally, can also provide a specific voltage limit, current level range, voltage limit range values directly.

```
public static void ForceCurrent(this DCPowerSessionsBundle sessionsBundle, double
currentLevel, double? voltageLimit = null, double? currentLevelRange = null, double?
voltageLimitRange = null, bool waitForSourceCompletion = false)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**currentLevel** [double](#)?

The current level to force.

**voltageLimit** [double](#)?

The voltage limit to use.

**currentLevelRange** [double](#)?

The current level range to use.

**voltageLimitRange** [double](#)?

The voltage limit range to use.

**waitForSourceCompletion** [bool](#)?

Setting this to True will wait until sourcing is complete before continuing, which includes the set amount of source delay. Otherwise, the source delay amount is not directly accounted for by this method and the WaitForEvent must be manually invoked in proceeding code.

## ForceCurrent(DCPowerSessionsBundle, IDictionary<string, DCPowerSourceSettings>, bool)

Forces current and specifies symmetric voltage limits.

```
public static void ForceCurrent(this DCPowerSessionsBundle sessionsBundle,  
IDictionary<string, DCPowerSourceSettings> settings, bool waitForSourceCompletion = false)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**settings** [IDictionary<string, DCPowerSourceSettings>](#)

The per-pin settings to use.

**waitForSourceCompletion** [bool](#)

Setting this to True will wait until sourcing is complete before continuing, which includes the set amount of source delay. Otherwise, the source delay amount is not directly accounted for by this method and the WaitForEvent must be manually invoked in proceeding code.

# Method ForceCurrentAsymmetricLimit

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ForceCurrentAsymmetricLimit(DCPowerSessionsBundle, double, double, double?, double?, bool)**

Behaves the same as the ForceCurrent() method, but has two voltage limit inputs for setting separate high and low voltage limits.

```
public static void ForceCurrentAsymmetricLimit(this DCPowerSessionsBundle sessionsBundle,
    double currentLevel, double voltageLimitHigh, double voltageLimitLow, double?
    currentLevelRange = null, double? voltageLimitRange = null, bool waitForSourceCompletion
    = false)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**currentLevel** [double](#)

The current level to force.

**voltageLimitHigh** [double](#)

The voltage high limit to use.

**voltageLimitLow** [double](#)

The voltage low limit to use.

**currentLevelRange** [double](#)?

The current level range to use.

**voltageLimitRange** [double](#)?

The voltage limit range to use.

## `waitForSourceCompletion` [bool](#)

Setting this to True will wait until sourcing is complete before continuing, which includes the set amount of source delay. Otherwise, the source delay amount is not directly accounted for by this method and the `WaitForEvent` must be manually invoked in proceeding code.

# Method ForceVoltage

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ForceVoltage(DCPowerSessionsBundle, double, double?, double?, double?, bool)

Forces voltage on the target pins at the specified level. Must at least provide a level value, and the method will assume all other properties that have been previously set. Optionally, can also provide a specific current limit, current limit range, voltage level range values directly.

```
public static void ForceVoltage(this DCPowerSessionsBundle sessionsBundle, double
voltageLevel, double? currentLimit = null, double? voltageLevelRange = null, double?
currentLimitRange = null, bool waitForSourceCompletion = false)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**voltageLevel** [double](#)?

The voltage level to force.

**currentLimit** [double](#)?

The current limit to use.

**voltageLevelRange** [double](#)?

The voltage level range to use.

**currentLimitRange** [double](#)?

The current limit range to use.

**waitForSourceCompletion** [bool](#)?

Setting this to True will wait until sourcing is complete before continuing, which includes the set amount of source delay. Otherwise, the source delay amount is not directly accounted for by this method and the WaitForEvent must be manually invoked in proceeding code.

## ForceVoltage(DCPowerSessionsBundle, PinSiteData<double>, double?, double?, double?, bool)

Forces voltage on the target pins at the specified level. Must at least provide a level value, and the method will assume all other properties that have been previously set. Optionally, can also provide a specific current limit, current limit range, voltage level range values directly.

```
public static void ForceVoltage(this DCPowerSessionsBundle sessionsBundle,  
PinSiteData<double> voltageLevels, double? currentLimit = null, double? voltageLevelRange =  
null, double? currentLimitRange = null, bool waitForSourceCompletion = false)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**voltageLevels** [PinSiteData<double>](#)

The voltage levels to force for different site-pin pairs.

**currentLimit** [double?](#)

The current limit to use.

**voltageLevelRange** [double?](#)

The voltage level range to use.

**currentLimitRange** [double?](#)

The current limit range to use.

**waitForSourceCompletion** [bool](#)

Setting this to True will wait until sourcing is complete before continuing, which includes the set amount of source delay. Otherwise, the source delay amount is not directly accounted for by this method and the WaitForEvent must be manually invoked in proceeding code.

## ForceVoltage(DCPowerSessionsBundle, IDictionary<string, DCPowerSourceSettings>, bool)

Forces voltage and specifies symmetric current limit.

```
public static void ForceVoltage(this DCPowerSessionsBundle sessionsBundle,  
IDictionary<string, DCPowerSourceSettings> settings, bool waitForSourceCompletion = false)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**settings** [IDictionary](#)<[string](#), [DCPowerSourceSettings](#)>

The per-pin settings to use.

**waitForSourceCompletion** [bool](#)

Setting this to True will wait until sourcing is complete before continuing, which includes the set amount of source delay. Otherwise, the source delay amount is not directly accounted for by this method and the [WaitForEvent](#) must be manually invoked in proceeding code.

# Method ForceVoltageAsymmetricLimit

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ForceVoltageAsymmetricLimit(DCPowerSessionsBundle, double, double, double?, double?, bool)**

Behaves the same as the ForceVoltage() method, but as two current limit inputs for setting separate high and low current limits.

```
public static void ForceVoltageAsymmetricLimit(this DCPowerSessionsBundle sessionsBundle,
    double voltageLevel, double currentLimitHigh, double currentLimitLow, double?
    voltageLevelRange = null, double? currentLimitRange = null, bool waitForSourceCompletion
    = false)
```

## Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**voltageLevel** [double](#)

The voltage level to force.

**currentLimitHigh** [double](#)

The current high limit to use.

**currentLimitLow** [double](#)

The current low limit to use.

**voltageLevelRange** [double](#)?

The voltage level range to use.

**currentLimitRange** [double](#)?

The current limit range to use.

## `waitForSourceCompletion` [bool](#)

Setting this to True will wait until sourcing is complete before continuing, which includes the set amount of source delay. Otherwise, the source delay amount is not directly accounted for by this method and the `WaitForEvent` must be manually invoked in proceeding code.

# Method GetCurrentLimits

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetCurrentLimits(DCPowerSessionsBundle)

Gets the current limits.

```
public static PinSiteData<double> GetCurrentLimits(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Returns

[PinSiteData<double>](#)

The per-site per-pin current limits.

# Method GetSourceDelayInSeconds

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetSourceDelayInSeconds(DCPowerSessionsBundle)

Gets the source delay in seconds for each of the underlying device channel(s), per-pin and per-site.

```
public static PinSiteData<double> GetSourceDelayInSeconds(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

### Returns

[PinSiteData<double>](#)

The source delay in seconds ([PinSiteData<T>](#), where T is of type [double](#)).

# Method PowerDown

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## PowerDown(DCPowerSessionsBundle, double?)

Powers down the channel.

```
public static void PowerDown(this DCPowerSessionsBundle sessionsBundle, double? settlingTime = null)
```

### Parameters

`sessionsBundle` [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

`settlingTime` [double](#)?

The settling time. Null means no need to wait for the turn off operation to settle.

# Enum TriggerType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines DCPower trigger type.

```
public enum TriggerType
```

## Fields

**MeasureTrigger = 0**

The measure trigger.

**PulseTrigger = 1**

The pulse trigger.

**SequenceAdvanceTrigger = 2**

The sequence advance trigger.

**SourceTrigger = 3**

The source trigger.

**StartTrigger = 4**

The start trigger.

# Class TriggersAndEvents

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for DCPower triggers and events.

```
public static class TriggersAndEvents
```

## Inheritance

[object](#) ← TriggersAndEvents

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ClearTriggers\(DCPowerSessionsBundle\)](#).

Clears all trigger types.

[ConfigureMeasureTriggerDigitalEdge\(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string\)](#).

Configures a digital edge trigger for the MeasureTrigger.

[ConfigureMeasureTriggerSoftwareEdge\(DCPowerSessionInformation, string, string\)](#).

Configures a software edge trigger for the MeasureTrigger.

[ConfigurePulseTriggerDigitalEdge\(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string\)](#).

Configures a digital edge trigger for the PulseTrigger.

[ConfigurePulseTriggerDisable\(DCPowerSessionInformation, string, string\)](#).

Disables the PulseTrigger.

[ConfigurePulseTriggerSoftwareEdge\(DCPowerSessionInformation, string, string\)](#).

Configures a software edge trigger for the PulseTrigger.

[ConfigureSequenceAdvanceTriggerDigitalEdge\(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string\)](#)

Configures a digital edge trigger for the SequenceAdvanceTrigger.

[ConfigureSequenceAdvanceTriggerDisable\(DCPowerSessionInformation, string, string\)](#)

Disables the SequenceAdvanceTrigger.

[ConfigureSequenceAdvanceTriggerSoftwareEdge\(DCPowerSessionInformation, string, string\)](#)

Configures a software edge trigger for the SequenceAdvanceTrigger.

[ConfigureSourceTriggerDigitalEdge\(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string\)](#)

Configures a digital edge trigger for the SourceTrigger.

[ConfigureSourceTriggerDisable\(DCPowerSessionInformation, string, string\)](#)

Disables the SourceTrigger.

[ConfigureSourceTriggerSoftwareEdge\(DCPowerSessionInformation, string, string\)](#)

Configures a software edge trigger for the SourceTrigger.

[ConfigureStartTriggerDigitalEdge\(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string\)](#)

Configures a digital edge trigger for the StartTrigger.

[ConfigureStartTriggerDisable\(DCPowerSessionInformation, string, string\)](#)

Disables the StartTrigger.

[ConfigureStartTriggerSoftwareEdge\(DCPowerSessionInformation, string, string\)](#)

Configures a software edge trigger for the StartTrigger.

[ConfigureTriggerDigitalEdge\(DCPowerSessionsBundle, TriggerType, string, DCPowerTriggerEdge\)](#)

Configures a digital edge trigger for the selected TriggerType: MeasureTrigger, PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, and StartTrigger.

[ConfigureTriggerSoftwareEdge\(DCPowerSessionsBundle, TriggerType\)](#)

Configures a software edge trigger for the selected TriggerType: MeasureTrigger, PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, and StartTrigger.

[DisableTriggers\(DCPowerSessionsBundle\)](#)

Disables all triggers by configuring them to None: PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, and StartTrigger.

Note that MeasureTrigger is not supported. It does not need to be disabled.

[ExportSignal\(DCPowerSessionsBundle, DCPowerSignalSource, string\)](#)

Exports the selected DCPowerSignalSource to the target output terminal.

[SendSoftwareEdgeTrigger\(DCPowerSessionsBundle, TriggerType\)](#)

Sends a software trigger as the selected TriggerType.

[WaitForEvent\(DCPowerSessionsBundle, EventType, double\)](#)

Waits for the selected EventType to occur.

# Method ClearTriggers

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ClearTriggers(DCPowerSessionsBundle)

Clears all trigger types.

```
public static void ClearTriggers(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

# Method ConfigureMeasureTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureMeasureTriggerDigitalEdge(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string)

Configures a digital edge trigger for the MeasureTrigger.

```
public static void ConfigureMeasureTriggerDigitalEdge(this DCPowerSessionInformation sessionInfo, string triggerTerminal, string modelString, DCPowerTriggerEdge triggerEdge = DCPowerTriggerEdge.Rising, string channelString = "")
```

### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**triggerTerminal** [string](#)

The input terminal to configure the trigger to look for a Digital Edge.

This is the fully qualified terminal string, which should be in the form of

`"/Dev1/PXI_Trig0"`

, where Dev1 is the instrument generating the trigger and PXI\_Trig0 is the trigger line the trigger is being sent on.

Note that the input terminal can also be a terminal from another instrument or channel.

For example, you can set the input terminal on Dev1 to be /Dev2/Engine0/SourceCompleteEvent, where Engine0 is channel 0.

**modelString** [string](#)

The model string of the associated instrument.

**triggerEdge** [DCPowerTriggerEdge](#)

The digital edge to look for, either NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Rising or NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Falling.

#### channelString [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

#### Remarks

This method does not abort the underlying driver session.

# Method

## ConfigureMeasureTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### ConfigureMeasureTriggerSoftwareEdge(DCPowerSessionInformation, string, string)

Configures a software edge trigger for the MeasureTrigger.

```
public static void ConfigureMeasureTriggerSoftwareEdge(this DCPowerSessionInformation  
sessionInfo, string modelString, string channelString = "")
```

#### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**modelString** [string](#)

The model string of the associated instrument.

**channelString** [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

#### Remarks

This method does not abort the underlying driver session.

# Method ConfigurePulseTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigurePulseTriggerDigitalEdge(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string)**

Configures a digital edge trigger for the PulseTrigger.

```
public static void ConfigurePulseTriggerDigitalEdge(this DCPowerSessionInformation sessionInfo, string triggerTerminal, string modelString, DCPowerTriggerEdge triggerEdge = DCPowerTriggerEdge.Rising, string channelString = "")
```

## Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**triggerTerminal** [string](#)

The input terminal to configure the trigger to look for a Digital Edge.

This is the fully qualified terminal string, which should be in the form of

`"/Dev1/PXI_Trig0"`

, where Dev1 is the instrument generating the trigger and PXI\_Trig0 is the trigger line the trigger is being sent on.

Note that the input terminal can also be a terminal from another instrument or channel.

For example, you can set the input terminal on Dev1 to be /Dev2/Engine0/SourceCompleteEvent, where Engine0 is channel 0.

**modelString** [string](#)

The model string of the associated instrument.

**triggerEdge** [DCPowerTriggerEdge](#)

The digital edge to look for, either NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Rising or NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Falling.

#### channelString [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

#### Remarks

This method does not abort the underlying driver session.

# Method ConfigurePulseTriggerDisable

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigurePulseTriggerDisable(DCPowerSessionInformation, string, string)

Disables the PulseTrigger.

```
public static void ConfigurePulseTriggerDisable(this DCPowerSessionInformation sessionInfo,
    string modelString, string channelString = "")
```

### Parameters

`sessionInfo` [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

`modelString` [string](#)

The model string of the associated instrument.

`channelString` [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

### Remarks

This method does not abort the underlying driver session.

# Method ConfigurePulseTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigurePulseTriggerSoftwareEdge(DCPowerSessionInformation, string, string)

Configures a software edge trigger for the PulseTrigger.

```
public static void ConfigurePulseTriggerSoftwareEdge(this DCPowerSessionInformation sessionInfo, string modelString, string channelString = "")
```

### Parameters

`sessionInfo` [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

`modelString` [string](#)

The model string of the associated instrument.

`channelString` [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

### Remarks

This method does not abort the underlying driver session.

# Method

## ConfigureSequenceAdvanceTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigureSequenceAdvanceTriggerDigitalEdge(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string)**

Configures a digital edge trigger for the SequenceAdvanceTrigger.

```
public static void ConfigureSequenceAdvanceTriggerDigitalEdge(this DCPowerSessionInformation sessionInfo, string triggerTerminal, string modelString, DCPowerTriggerEdge triggerEdge = DCPowerTriggerEdge.Rising, string channelString = "")
```

### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**triggerTerminal** [string](#)

The input terminal to configure the trigger to look for a Digital Edge.

This is the fully qualified terminal string, which should be in the form of

`"/Dev1/PXI_Trig0"`

, where Dev1 is the instrument generating the trigger and PXI\_Trig0 is the trigger line the trigger is being sent on.

Note that the input terminal can also be a terminal from another instrument or channel.

For example, you can set the input terminal on Dev1 to be /Dev2/Engine0/SourceCompleteEvent, where Engine0 is channel 0.

**modelString** [string](#)

The model string of the associated instrument.

## **triggerEdge** DCPowerTriggerEdge

The digital edge to look for, either NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Rising or NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Falling.

## **channelString** [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

## Remarks

This method does not abort the underlying driver session.

# Method

## ConfigureSequenceAdvanceTriggerDisable

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### ConfigureSequenceAdvanceTriggerDisable(DCPowerSessionInformation, string, string)

Disables the SequenceAdvanceTrigger.

```
public static void ConfigureSequenceAdvanceTriggerDisable(this DCPowerSessionInformation sessionInfo, string modelString, string channelString = "")
```

#### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**modelString** [string](#)

The model string of the associated instrument.

**channelString** [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

#### Remarks

This method does not abort the underlying driver session.

# Method ConfigureSequenceAdvanceTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSequenceAdvanceTriggerSoftwareEdge(DCPowerSessionInformation, string, string)

Configures a software edge trigger for the SequenceAdvanceTrigger.

```
public static void ConfigureSequenceAdvanceTriggerSoftwareEdge(this  
DCPowerSessionInformation sessionInfo, string modelString, string channelString = "")
```

### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**modelString** [string](#)

The model string of the associated instrument.

**channelString** [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

### Remarks

This method does not abort the underlying driver session.

# Method ConfigureSourceTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigureSourceTriggerDigitalEdge(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string)**

Configures a digital edge trigger for the SourceTrigger.

```
public static void ConfigureSourceTriggerDigitalEdge(this DCPowerSessionInformation sessionInfo, string triggerTerminal, string modelString, DCPowerTriggerEdge triggerEdge = DCPowerTriggerEdge.Rising, string channelString = "")
```

## Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**triggerTerminal** [string](#)

The input terminal to configure the trigger to look for a Digital Edge.

This is the fully qualified terminal string, which should be in the form of

`"/Dev1/PXI_Trig0"`

, where Dev1 is the instrument generating the trigger and PXI\_Trig0 is the trigger line the trigger is being sent on.

Note that the input terminal can also be a terminal from another instrument or channel.

For example, you can set the input terminal on Dev1 to be /Dev2/Engine0/SourceCompleteEvent, where Engine0 is channel 0.

**modelString** [string](#)

The model string of the associated instrument.

**triggerEdge** [DCPowerTriggerEdge](#)

The digital edge to look for, either NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Rising or NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Falling.

#### channelString [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

#### Remarks

This method does not abort the underlying driver session.

# Method ConfigureSourceTriggerDisable

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSourceTriggerDisable(DCPowerSessionInformation, string, string)

Disables the SourceTrigger.

```
public static void ConfigureSourceTriggerDisable(this DCPowerSessionInformation sessionInfo,  
string modelString, string channelString = "")
```

### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**modelString** [string](#)

The model string of the associated instrument.

**channelString** [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

### Remarks

This method does not abort the underlying driver session.

# Method ConfigureSourceTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSourceTriggerSoftwareEdge(DCPowerSessionInformation, string, string)

Configures a software edge trigger for the SourceTrigger.

```
public static void ConfigureSourceTriggerSoftwareEdge(this DCPowerSessionInformation sessionInfo, string modelString, string channelString = "")
```

### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**modelString** [string](#)

The model string of the associated instrument.

**channelString** [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

### Remarks

This method does not abort the underlying driver session.

# Method ConfigureStartTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigureStartTriggerDigitalEdge(DCPowerSessionInformation, string, string, DCPowerTriggerEdge, string)**

Configures a digital edge trigger for the StartTrigger.

```
public static void ConfigureStartTriggerDigitalEdge(this DCPowerSessionInformation sessionInfo, string triggerTerminal, string modelString, DCPowerTriggerEdge triggerEdge = DCPowerTriggerEdge.Rising, string channelString = "")
```

Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**triggerTerminal** [string](#)

The input terminal to configure the trigger to look for a Digital Edge.

This is the fully qualified terminal string, which should be in the form of

`"/Dev1/PXI_Trig0"`

, where Dev1 is the instrument generating the trigger and PXI\_Trig0 is the trigger line the trigger is being sent on.

Note that the input terminal can also be a terminal from another instrument or channel.

For example, you can set the input terminal on Dev1 to be /Dev2/Engine0/SourceCompleteEvent, where Engine0 is channel 0.

**modelString** [string](#)

The model string of the associated instrument.

**triggerEdge** [DCPowerTriggerEdge](#)

The digital edge to look for, either NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Rising or NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Falling.

#### channelString [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

#### Remarks

This method does not abort the underlying driver session.

# Method ConfigureStartTriggerDisable

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureStartTriggerDisable(DCPowerSessionInformation, string, string)

Disables the StartTrigger.

```
public static void ConfigureStartTriggerDisable(this DCPowerSessionInformation sessionInfo,
    string modelString, string channelString = "")
```

### Parameters

`sessionInfo` [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

`modelString` [string](#)

The model string of the associated instrument.

`channelString` [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

### Remarks

This method does not abort the underlying driver session.

# Method ConfigureStartTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureStartTriggerSoftwareEdge(DCPowerSessionInformation, string, string)

Configures a software edge trigger for the StartTrigger.

```
public static void ConfigureStartTriggerSoftwareEdge(this DCPowerSessionInformation sessionInfo, string modelString, string channelString = "")
```

### Parameters

**sessionInfo** [DCPowerSessionInformation](#)

The [DCPowerSessionInformation](#) object.

**modelString** [string](#)

The model string of the associated instrument.

**channelString** [string](#)

The channel string containing one or more instrument channels. For Example: "SMU\_4147\_C2\_16/0", or "SMU\_4147\_C2\_16/3, SMU\_4137\_C2\_17/0".

### Remarks

This method does not abort the underlying driver session.

# Method ConfigureTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTriggerDigitalEdge(DCPowerSessionsBundle, TriggerType, string, DCPowerTriggerEdge)

Configures a digital edge trigger for the selected TriggerType: MeasureTrigger, PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, and StartTrigger.

```
public static void ConfigureTriggerDigitalEdge(this DCPowerSessionsBundle sessionsBundle,  
TriggerType triggerType, string triggerTerminal, DCPowerTriggerEdge triggerEdge  
= DCPowerTriggerEdge.Rising)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**triggerType** [TriggerType](#)

Type of trigger, either MeasureTrigger, PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, StartTrigger.

**triggerTerminal** [string](#)

The input terminal to configure the trigger to look for a Digital Edge.

This is the fully qualified terminal string, which should be in the form of

`" /Dev1/PXI_Trig0"`

, where Dev1 is the instrument generating the trigger and PXI\_Trig0 is the trigger line the trigger is being sent on.

Note that the input terminal can also be a terminal from another instrument or channel.

For example, you can set the input terminal on Dev1 to be /Dev2/Engine0/SourceCompleteEvent, where Engine0 is channel 0.

#### **triggerEdge** DCPowerTriggerEdge

The digital edge to look for, either NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Rising or NationalInstruments.ModularInstruments.NIDCPower.DCPowerTriggerEdge.Falling.

# Method ConfigureTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTriggerSoftwareEdge(DCPowerSessionsBundle, TriggerType)

Configures a software edge trigger for the selected TriggerType: MeasureTrigger, PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, and StartTrigger.

```
public static void ConfigureTriggerSoftwareEdge(this DCPowerSessionsBundle sessionsBundle,  
TriggerType triggerType)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**triggerType** [TriggerType](#)

Type of trigger, either MeasureTrigger, PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, or StartTrigger.

# Method DisableTriggers

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DisableTriggers(DCPowerSessionsBundle)

Disables all triggers by configuring them to None: PulseTrigger, SequenceAdvanceTrigger, SourceTrigger, and StartTrigger.

Note that MeasureTrigger is not supported. It does not need to be disabled.

```
public static void DisableTriggers(this DCPowerSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

# Method ExportSignal

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ExportSignal(DCPowerSessionsBundle, DCPowerSignalSource, string)

Exports the selected DCPowerSignalSource to the target output terminal.

```
public static void ExportSignal(this DCPowerSessionsBundle sessionsBundle,  
DCPowerSignalSource signalSource, string outputTerminal)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**signalSource** DCPowerSignalSource

The signal source to export.

**outputTerminal** [string](#)

The output terminal the signal routes to.

# Method SendSoftwareEdgeTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SendSoftwareEdgeTrigger(DCPowerSessionsBundle, TriggerType)

Sends a software trigger as the selected TriggerType.

```
public static void SendSoftwareEdgeTrigger(this DCPowerSessionsBundle sessionsBundle,  
TriggerType triggerType)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**triggerType** [TriggerType](#)

The type of the trigger.

# Method WaitForEvent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DCPower](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WaitForEvent(DCPowerSessionsBundle, EventType, double)

Waits for the selected EventType to occur.

```
public static void WaitForEvent(this DCPowerSessionsBundle sessionsBundle, EventType
eventType, double timeout = 5)
```

### Parameters

**sessionsBundle** [DCPowerSessionsBundle](#)

The [DCPowerSessionsBundle](#) object.

**eventType** [EventType](#)

The type of event to wait for.

**timeout** [double](#) ↗

The maximum time to wait. Default is 5 seconds.

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM

## Classes

### [DMMSessionInformation](#)

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIDmm.NIDmm session and its related information.

### [DMMSessionsBundle](#)

Defines an object that contains one or more [DMMSessionInformation](#) objects.

### [InitializeAndClose](#)

Defines NIDMM sessions initialize and close APIs.

### [Measurement](#)

Defines methods for data acquisition.

### [PropertiesConfiguration](#)

Defines methods to configure NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions.

# Class DMMSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIDmm.NIDmm session and its related information.

```
public class DMMSessionInformation : ISessionInformation
```

## Inheritance

[object](#) ← DMMSessionInformation

## Implements

[ISessionInformation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DMMSessionInformation\(NIDmm\)](#).

Constructs a session information object that associates with a NationalInstruments.Modular Instruments.NIDmm.NIDmm instrument session.

[DMMSessionInformation\(NIDmm, IList<SitePinInfo>\)](#).

Constructs a session information object.

## Properties

[AssociatedSitePinList](#)

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

[Session](#)

The NationalInstruments.ModularInstruments.NIDmm.NIDmm session.

## Methods

### [GenerateResourceDescriptorToSitePinDictionary\(ISemiconductorModuleContext\)](#)

Generates a dictionary that takes a resource descriptor and returns the associated site-pin pair information.

# Constructor DMMSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DMMSessionInformation(NIDmm)

Constructs a session information object that associates with a NationalInstruments.ModularInstruments.NIDmm.NIDmm instrument session.

```
public DMMSessionInformation(NIDmm session)
```

### Parameters

**session** NIDmm

The NationalInstruments.ModularInstruments.NIDmm.NIDmm session.

## DMMSessionInformation(NIDmm, IList<SitePinInfo>)

Constructs a session information object.

```
public DMMSessionInformation(NIDmm session, IList<SitePinInfo> associatedSitePinList)
```

### Parameters

**session** NIDmm

The NationalInstruments.ModularInstruments.NIDmm.NIDmm session.

**associatedSitePinList** [IList](#)<SitePinInfo>

The specified subset of channels associated with the session.

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

```
public IList<SitePinInfo> AssociatedSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property Session

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Session

The NationalInstruments.ModularInstruments.NIDmm.NIDmm session.

```
public NIDmm Session { get; }
```

## Property Value

NIDmm

# Method GenerateResourceDescriptorToSitePinDictionary

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GenerateResourceDescriptorToSitePinDictionary(ISemiconductorModuleContext)

Generates a dictionary that takes a resource descriptor and returns the associated site-pin pair information.

```
public static void GenerateResourceDescriptorToSitePinDictionary(ISemiconductorModuleContext tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Class DMMSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains one or more [DMMSessionInformation](#) objects.

```
public class DMMSessionsBundle : ISessionsBundle<DMMSessionInformation>
```

## Inheritance

[object](#) ← DMMSessionsBundle

## Implements

[ISessionsBundle<DMMSessionInformation>](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>>, string, string\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, int, TResult\[\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[,\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, SitePinInfo>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int, SitePinInfo>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation>\)](#),  
[Publish.PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\], string\)](#),  
[Publish.PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\]\[\], string\)](#).

## Constructors

[DMMSessionsBundle\(ISemiconductorModuleContext, IEnumerable<DMMSessionInformation>\)](#)

Constructs a sessions bundle object that represents a bunch of [DMMSessionInformations](#).

## Properties

[AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

[InstrumentSessions](#)

An enumerable of [ISessionInformation](#).

[TSMContext](#)

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

# Methods

## [FilterByPin\(string\)](#)

Filter current [DMMSessionsBundle](#) and returns a new one with the requested pin.

## [FilterByPin\(string\[\]\)](#)

Filters current [DMMSessionsBundle](#) and returns a new one with requested pins.

## [FilterBySite\(int\)](#)

Filters current [DMMSessionsBundle](#) and returns a new one with the requested site.

## [FilterBySite\(int\[\]\)](#)

Filters current [DMMSessionsBundle](#) and returns a new one with requested sites.

# Constructor DMMSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DMMSessionsBundle(ISemiconductorModuleContext, IEnumerable<DMMSessionInformation>)

Constructs a sessions bundle object that represents a bunch of [DMMSessionInformations](#).

```
public DMMSessionsBundle(ISemiconductorModuleContext tsmContext,  
IEnumerable<DMMSessionInformation> allSessionInfo)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

**allSessionInfo** [IEnumerable](#)<DMMSessionInformation>

An enumerable of [DMMSessionInformations](#).

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

```
public IList<SitePinInfo> AggregateSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [ISessionInformation](#).

```
public IEnumerable<DMMSessionInformation> InstrumentSessions { get; }
```

## Property Value

[IEnumerable](#) <[DMMSessionInformation](#)>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public ISemiconductorModuleContext TSMContext { get; }
```

## Property Value

ISemiconductorModuleContext

# Method FilterByPin

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterByPin(string)

Filter current [DMMSessionsBundle](#) and returns a new one with the requested pin.

```
public DMMSessionsBundle FilterByPin(string requestedPin)
```

### Parameters

**requestedPin** [string](#) ↗

The requested pin.

### Returns

[DMMSessionsBundle](#)

A new [DMMSessionsBundle](#) object with the requested pin.

## FilterByPin(string[])

Filters current [DMMSessionsBundle](#) and returns a new one with requested pins.

```
public DMMSessionsBundle FilterByPin(string[] requestedPins)
```

### Parameters

**requestedPins** [string](#) ↗[]

The requested pins.

### Returns

## [DMMSessionsBundle](#)

A new [DMMSessionsBundle](#) object with requested pins.

# Method FilterBySite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterBySite(int)

Filters current [DMMSessionsBundle](#) and returns a new one with the requested site.

```
public DMMSessionsBundle FilterBySite(int requestedSite)
```

### Parameters

`requestedSite` [int](#)

The requested site.

### Returns

[DMMSessionsBundle](#)

A new [DMMSessionsBundle](#) object with the requested site.

## FilterBySite(int[])

Filters current [DMMSessionsBundle](#) and returns a new one with requested sites.

```
public DMMSessionsBundle FilterBySite(int[] requestedSites)
```

### Parameters

`requestedSites` [int](#)[]

The requested sites.

### Returns

## [DMMSessionsBundle](#)

A new [DMMSessionsBundle](#) object with requested sites.

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines NIDMM sessions initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[Close\(ISemiconductorModuleContext, bool\)](#)

Closes all NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions.

[Initialize\(ISemiconductorModuleContext, bool\)](#)

Initializes NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions in the test system.

[Reset\(ISemiconductorModuleContext\)](#)

Resets all NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions.

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close(ISemiconductorModuleContext, bool)

Closes all NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions.

```
public static void Close(ISemiconductorModuleContext tsmContext, bool resetDevice = true)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset the device sessions before closing.

# Method Initialize

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Initialize(ISemiconductorModuleContext, bool)

Initializes NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions in the test system.

```
public static void Initialize(ISemiconductorModuleContext tsmContext, bool resetDevice  
= false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset device during initialization.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset(ISemiconductorModuleContext)

Resets all NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions.

```
public static void Reset(ISemiconductorModuleContext tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Class Measurement

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for data acquisition.

```
public static class Measurement
```

## Inheritance

[object](#) ← Measurement

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[Abort\(DMMSessionsBundle\)](#)

Aborts an acquisition.

[Fetch\(DMMSessionsBundle, double\)](#)

Fetches the values from previously initiated measurements.

[FetchAndPublish\(DMMSessionsBundle, double, string\)](#)

Fetches the values from previously initiated measurements and publishes measurement results.

[FetchMultiPoint\(DMMSessionsBundle, int, double\)](#)

Fetches the specified number of values from previously initiated measurements.

[Initiate\(DMMSessionsBundle\)](#)

Initiates an acquisition.

[Read\(DMMSessionsBundle, double\)](#)

Reads measurement results.

[ReadAndPublish\(DMMSessionsBundle, double, string\)](#)

Reads measurement results and publishes measurement results.

[ReadMultiPoint\(DMMSessionsBundle, int, double\)](#)

Reads specified number of measurement results.

# Method Abort

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Abort(DMMSessionsBundle)

Aborts an acquisition.

```
public static void Abort(this DMMSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

# Method Fetch

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Fetch(DMMSessionsBundle, double)

Fetches the values from previously initiated measurements.

```
public static PinSiteData<double> Fetch(this DMMSessionsBundle sessionsBundle,  
double maximumTimeInMilliseconds)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**maximumTimeInMilliseconds** [double](#)

The maximum time for the fetch to complete in milliseconds.

### Returns

[PinSiteData<double>](#)

The measurement results in per-site per-pin format.

# Method FetchAndPublish

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## FetchAndPublish(DMMSessionsBundle, double, string)

Fetches the values from previously initiated measurements and publishes measurement results.

```
public static double[] FetchAndPublish(this DMMSessionsBundle sessionsBundle, double maximumTimeInMilliseconds, string publishedDataId = "")
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**maximumTimeInMilliseconds** [double](#)

The maximum time for the fetch to complete in milliseconds.

**publishedDataId** [string](#)

The unique data id to be used when publishing.

### Returns

[double](#)[]

The measurement results in per-instrument format.

# Method FetchMultiPoint

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## FetchMultiPoint(DMMSessionsBundle, int, double)

Fetches the specified number of values from previously initiated measurements.

```
public static double[][] FetchMultiPoint(this DMMSessionsBundle sessionsBundle, int
numberToRead, double maximumTimeInMilliseconds)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**numberToRead** [int](#)

The number of values to fetch.

**maximumTimeInMilliseconds** [double](#)

The maximum time for the fetch to complete in milliseconds.

### Returns

[double](#)[][]

The measurement results in per-instrument per-point format.

# Method Initiate

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Initiate(DMMSessionsBundle)

Initiates an acquisition.

```
public static void Initiate(this DMMSessionsBundle sessionsBundle)
```

### Parameters

sessionsBundle [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

# Method Read

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Read(DMMSessionsBundle, double)

Reads measurement results.

```
public static PinSiteData<double> Read(this DMMSessionsBundle sessionsBundle,  
double maximumTimeInMilliseconds)
```

### Parameters

`sessionsBundle` [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

`maximumTimeInMilliseconds` [double](#)

The maximum time for the fetch to complete in milliseconds.

### Returns

[PinSiteData<double>](#)

The measurement results in per-site per-pin format.

# Method ReadAndPublish

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadAndPublish(DMMSessionsBundle, double, string)

Reads measurement results and publishes measurement results.

```
public static double[] ReadAndPublish(this DMMSessionsBundle sessionsBundle, double maximumTimeInMilliseconds, string publishedDataId = "")
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**maximumTimeInMilliseconds** [double](#)

The maximum time for the fetch to complete in milliseconds.

**publishedDataId** [string](#)

The unique data id to use when publishing.

### Returns

[double](#)[]

The measurement results in per-instrument format.

# Method ReadMultiPoint

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadMultiPoint(DMMSessionsBundle, int, double)

Reads specified number of measurement results.

```
public static double[][] ReadMultiPoint(this DMMSessionsBundle sessionsBundle, int
numberToRead, double maximumTimeInMilliseconds)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**numberToRead** [int](#)

The number of values to fetch.

**maximumTimeInMilliseconds** [double](#)

The maximum time for the fetch to complete in milliseconds.

### Returns

[double](#)[][]

The measurement results in per-instrument per-value format.

# Class PropertiesConfiguration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods to configure NationalInstruments.ModularInstruments.NIDmm.NIDmm sessions.

```
public static class PropertiesConfiguration
```

## Inheritance

[object](#) ← PropertiesConfiguration

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ConfigureACBandwidth\(DMMSessionsBundle, double, double\)](#)

Configures AC bandwidth for AC measurements.

[ConfigureADCCalibration\(DMMSessionsBundle, DmmAdcCalibration\)](#)

Allows the DMM to compensate for gain drift since the last external or self-calibration.

[ConfigureApertureTime\(DMMSessionsBundle, DmmApertureTimeUnits, double\)](#)

Configures aperture time.

[ConfigureAutoZero\(DMMSessionsBundle, DmmAuto\)](#)

Configures the DMM for auto zero.

[ConfigureMeasurementAbsolute\(DMMSessionsBundle, DmmMeasurementFunction, double, double\)](#)

Configures measurements using absolute resolution.

[ConfigureMeasurementDigits\(DMMSessionsBundle, DmmMeasurementFunction, double, double\)](#)

Configures measurements using number of digits resolution.

[ConfigureMultiPoint\(DMMSessionsBundle, int, int, string, double\)](#)

Configures multi-point acquisition.

[ConfigurePowerlineFrequency\(DMMSessionsBundle, double\)](#)

Configures power line frequency.

[ConfigureSettleTime\(DMMSessionsBundle, double\)](#)

Configures settle time.

[ConfigureTrigger\(DMMSessionsBundle, DmmTriggerSource, double\)](#)

Configures trigger.

[SendSoftwareTrigger\(DMMSessionsBundle\)](#)

Sends out software trigger.

# Method ConfigureACBandwidth

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureACBandwidth(DMMSessionsBundle, double, double)

Configures AC bandwidth for AC measurements.

```
public static void ConfigureACBandwidth(this DMMSessionsBundle sessionsBundle, double  
minimumFrequency = 20, double maximumFrequency = 25000)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**minimumFrequency** [double](#)

The minimum frequency value in Hz to use.

**maximumFrequency** [double](#)

The maximum frequency value in Hz to use.

# Method ConfigureADCCalibration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureADCCalibration(DMMSessionsBundle, DmmAdcCalibration)

Allows the DMM to compensate for gain drift since the last external or self-calibration.

```
public static void ConfigureADCCalibration(this DMMSessionsBundle sessionsBundle,  
DmmAdcCalibration dmmAdcCalibrationMode)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**dmmAdcCalibrationMode** DmmAdcCalibration

The ADC calibration mode to be used.

### Remarks

When ADC Calibration is AUTO, the DMM enables or disables ADC calibration.

When ADC Calibration is ON, the DMM measures an internal reference to calculate the correct gain for the measurement.

When ADC Calibration is OFF, the DMM does not compensate for changes to the gain.

# Method ConfigureApertureTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureApertureTime(DMMSessionsBundle, DmmApertureTimeUnits, double)

Configures aperture time.

```
public static void ConfigureApertureTime(this DMMSessionsBundle sessionsBundle,  
DmmApertureTimeUnits apertureTimeUnits = DmmApertureTimeUnits.Seconds, double apertureTime  
= 0)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**apertureTimeUnits** [DmmApertureTimeUnits](#)

The unit of the aperture time.

**apertureTime** [double](#) ↗

The aperture time to use.

# Method ConfigureAutoZero

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureAutoZero(DMMSessionsBundle, DmmAuto)

Configures the DMM for auto zero.

```
public static void ConfigureAutoZero(this DMMSessionsBundle sessionsBundle,  
DmmAuto autoZeroMode)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**autoZeroMode** DmmAuto

The auto zero mode to be used: AUTO, OFF, ON, or ONCE.

### Remarks

When Auto Zero is AUTO, the DMM chooses the AutoZero setting based on the configured function and resolution.

When Auto Zero is OFF, the DMM does not compensate for zero reading offset. Not Supported on 4065 DMM models.

When Auto Zero is ONCE, the DMM takes a zero reading once and then turns off Auto Zero. Not Supported on 4065 DMM models.

When Auto Zero is ON, the DMM internally disconnects the input, takes a zero reading, and then subtracts the zero reading from the measurement. This prevents offset voltages present on the input circuitry of the DMM from affecting measurement accuracy.

### Exceptions

[NISemiconductorTestException](#)

A device in an underlying session does not support configuring Auto Zero.

# Method ConfigureMeasurementAbsolute

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureMeasurementAbsolute(DMMSessionsBundle, DmmMeasurementFunction, double, double)

Configures measurements using absolute resolution.

```
public static void ConfigureMeasurementAbsolute(this DMMSessionsBundle sessionsBundle,  
DmmMeasurementFunction measurementFunction = DmmMeasurementFunction.DCVolts, double range =  
0.02, double resolutionAbsolute = 0.001)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**measurementFunction** DmmMeasurementFunction

The measurement function to use.

**range** [double](#)

The range to use.

**resolutionAbsolute** [double](#)

The absolute resolution to use.

# Method ConfigureMeasurementDigits

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureMeasurementDigits(DMMSessionsBundle, DmmMeasurementFunction, double, double)

Configures measurements using number of digits resolution.

```
public static void ConfigureMeasurementDigits(this DMMSessionsBundle sessionsBundle,  
DmmMeasurementFunction measurementFunction = DmmMeasurementFunction.DCVolts, double range =  
0.02, double resolutionDigits = 5.5)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**measurementFunction** DmmMeasurementFunction

The measurement function to use.

**range** [double](#)

The range to use.

**resolutionDigits** [double](#)

The resolution number of digits to use.

# Method ConfigureMultiPoint

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureMultiPoint(DMMSessionsBundle, int, int, string, double)

Configures multi-point acquisition.

```
public static void ConfigureMultiPoint(this DMMSessionsBundle sessionsBundle, int triggerCount, int sampleCount, string sampleTrigger, double sampleIntervalInSeconds)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**triggerCount** [int](#)

The number of the triggers the device receives before returning to the idle state.

**sampleCount** [int](#)

The number of measurements the device makes in each measurement sequence initiated by a trigger.

**sampleTrigger** [string](#)

The name of the trigger source that initiates the acquisition.

**sampleIntervalInSeconds** [double](#)

The interval in seconds that the device waits between measurements.

# Method ConfigurePowerlineFrequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigurePowerlineFrequency(DMMSessionsBundle, double)

Configures power line frequency.

```
public static void ConfigurePowerlineFrequency(this DMMSessionsBundle sessionsBundle, double powerlineFrequency = 60)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**powerlineFrequency** [double](#)

The frequency in Hz to use.

# Method ConfigureSettleTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSettleTime(DMMSessionsBundle, double)

Configures settle time.

```
public static void ConfigureSettleTime(this DMMSessionsBundle sessionsBundle, double  
settleTime = 0.01)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**settleTime** [double](#) ↗

The settle time to use.

# Method ConfigureTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTrigger(DMMSessionsBundle, DmmTriggerSource, double)

Configures trigger.

```
public static void ConfigureTrigger(this DMMSessionsBundle sessionsBundle, DmmTriggerSource triggerSource, double triggerDelayInSeconds)
```

### Parameters

**sessionsBundle** [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

**triggerSource** DmmTriggerSource

The name of the trigger source that initiates the acquisition.

**triggerDelayInSeconds** [double](#)

The interval in seconds that the device waits after it has received a trigger before taking a measurement.

# Method SendSoftwareTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.DMM](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SendSoftwareTrigger(DMMSessionsBundle)

Sends out software trigger.

```
public static void SendSoftwareTrigger(this DMMSessionsBundle sessionsBundle)
```

### Parameters

sessionsBundle [DMMSessionsBundle](#)

The [DMMSessionsBundle](#) object.

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital

## Classes

### [DigitalSessionInformation](#)

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIDigital.NIDigital session and its related information.

### [DigitalSessionsBundle](#)

Defines an object that contains one or more [DigitalSessionInformation](#) objects.

### [FrequencyCounter](#)

Defines methods for frequency measurements.

### [HistoryRAM](#)

Defines methods for History RAM.

### [HistoryRAMResults](#)

Defines History RAM results type.

### [HistoryRAMSettings](#)

Defines settings of History RAM.

### [HistoryRAMTriggerSettings](#)

Defines History RAM trigger settings.

### [InitializeAndClose](#)

Defines NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions initialize and close APIs.

### [LevelsAndTiming](#)

Defines methods for levels and timing.

### [PPMU](#)

Defines methods for PPMU measurements.

### [PPMUSettings](#)

Defines settings for PPMU.

### [Pattern](#)

Defines methods to deal with digital patterns.

## [SequencerFlagsAndRegisters](#)

Defines methods to deal with digital pattern sequencer flags and registers.

## [SourceAndCapture](#)

Defines methods for sourcing and capturing waveforms.

## [StaticState](#)

Defines methods for static state read/write.

## [TriggersAndEvents](#)

Defines methods to deal with digital pattern triggers and events.

# Enums

## [LevelsAndTiming.LevelType](#)

Defines voltage level types.

# Class DigitalSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIDigital.NIDigital session and its related information.

```
public class DigitalSessionInformation : ISessionInformation
```

## Inheritance

[object](#) ← DigitalSessionInformation

## Implements

[ISessionInformation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[DigitalSessionInformation\(NIDigital, IList<SitePinInfo>\)](#).

Constructs a session information object.

[DigitalSessionInformation\(NIDigital, string, string\)](#)

Constructs a session information object that associates with a NationalInstruments.Modular Instruments.NIDigital.NIDigital instrument session.

## Properties

[AssociatedSiteList](#)

The site numbers corresponding to [SiteListString](#) in a more accessible format.

[AssociatedSitePinList](#)

The pin set info corresponding to [PinSetString](#) in a more accessible format.

[PinSet](#)

The NationalInstruments.ModularInstruments.NIDigital.DigitalPinSet according to [PinSetString](#) on the [Session](#).

### [PinSetString](#)

The pin set string associated with the [Session](#).

### [Session](#)

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

### [SiteListString](#)

The site list string associated with the [Session](#).

## Methods

### [GetPerSitePinSetStrings\(\)](#)

Gets pin set strings on per-site basis.

# Constructor DigitalSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DigitalSessionInformation(NIDigital, string, string)

Constructs a session information object that associates with a NationalInstruments.ModularInstruments.NIDigital.NIDigital instrument session.

```
public DigitalSessionInformation(NIDigital session, string pinSetString, string siteList)
```

### Parameters

**session** NIDigital

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

**pinSetString** [string](#)

The pin set string associated with the NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

**siteList** [string](#)

The site list associated with the NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

## DigitalSessionInformation(NIDigital, IList<SitePinInfo>)

Constructs a session information object.

```
public DigitalSessionInformation(NIDigital session, IList<SitePinInfo>
associatedSitePinList)
```

### Parameters

**session** NIDigital

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

**associatedSitePinList** [IList](#) <SitePinInfo>

The specified subset of channels associated with the session.

# Property AssociatedSiteList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSiteList

The site numbers corresponding to [SiteListString](#) in a more accessible format.

```
public IList<int> AssociatedSiteList { get; }
```

## Property Value

[IList](#) <[int](#)>

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

The pin set info corresponding to [PinSetString](#) in a more accessible format.

```
public IList<SitePinInfo> AssociatedSitePinList { get; }
```

### Property Value

[IList](#) <[SitePinInfo](#)>

# Property PinSet

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PinSet

The NationalInstruments.ModularInstruments.NIDigital.DigitalPinSet according to [PinSetString](#) on the [Session](#).

```
public DigitalPinSet PinSet { get; }
```

## Property Value

DigitalPinSet

# Property PinSetString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PinSetString

The pin set string associated with the [Session](#).

```
public string PinSetString { get; }
```

## Property Value

[string](#) ↗

## Remarks

It's a comma-separated string of site-pin pairs, e.g. "site0/A, site0/B".

# Property Session

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Session

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

```
public NIDigital Session { get; }
```

## Property Value

NIDigital

# Property SiteListString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SiteListString

The site list string associated with the [Session](#).

```
public string SiteListString { get; }
```

### Property Value

[string](#) ↗

### Remarks

It's a comma-separated string of sites, e.g. "site0, site1".

# Method GetPerSitePinSetStrings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GetPerSitePinSetStrings()

Gets pin set strings on per-site basis.

```
public IDictionary<string, string> GetPerSitePinSetStrings()
```

Returns

[IDictionary](#) <[string](#), [string](#)>

A dictionary that contains per-site pin set strings.

# Class DigitalSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains one or more [DigitalSessionInformation](#) objects.

```
public class DigitalSessionsBundle : ISessionsBundle<DigitalSessionInformation>
```

## Inheritance

[object](#) ← DigitalSessionsBundle

## Implements

[ISessionsBundle<DigitalSessionInformation>](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\], string>\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>, string, string\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, int, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[,\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation>](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]](#),  
[string](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]\[\]](#),  
[string](#)) .

## Constructors

[DigitalSessionsBundle\( ISemiconductorModuleContext, IEnumerable<string>,  
IEnumerable<DigitalSessionInformation> \)](#)

Constructs a sessions bundle object that represents a bunch of [DigitalSessionInformations](#).

## Properties

[AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

[InstrumentSessions](#)

An enumerable of [DigitalSessionInformations](#).

[Pins](#)

[TSMContext](#)

The associated `NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext`.

## Methods

[`DoAndReturnPerSiteResults<T>\(Func<DigitalSessionInformation, SiteData<T>>\)`](#)

Does an operation on each [DigitalSessionInformation](#) in parallel and returns per-site results.

[`DoAndReturnPerSiteResults<T>\(Func<DigitalSessionInformation, T\[\]>\)`](#)

Does an operation on each [DigitalSessionInformation](#) in parallel and returns per-site results.

[`Do<T>\(SiteData<T>, Action<DigitalSessionInformation, T\[\]>\)`](#)

Does an operation on each [DigitalSessionInformation](#) in parallel with the given per-site data.

[`FilterByPin\(string\)`](#)

Filters current [DigitalSessionsBundle](#) and returns a new one with the requested pin.

[`FilterByPin\(string\[\]\)`](#)

Filters current [DigitalSessionsBundle](#) and returns a new one with requested pins.

[`FilterBySite\(int\)`](#)

Filters current [DigitalSessionsBundle](#) and returns a new one with the requested site.

[`FilterBySite\(int\[\]\)`](#)

Filters current [DigitalSessionsBundle](#) and returns a new one with requested sites.

[`PerInstrumentPerSiteResultsToPerSiteResults<T>\(T\[\]\[\]\[\]\)`](#)

Converts per-instrument per-site results to per-site results.

[`PublishPatternResults\(bool\[\]\[\]\[\], string\)`](#)

Publishes measurement results.

# Constructor DigitalSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

DigitalSessionsBundle(ISemiconductorModuleContext,  
IEnumerable<string>,  
IEnumerable<DigitalSessionInformation>)

Constructs a sessions bundle object that represents a bunch of [DigitalSessionInformations](#).

```
public DigitalSessionsBundle(ISemiconductorModuleContext tsmContext, IEnumerable<string>  
pins, IEnumerable<DigitalSessionInformation> allSessionInfo)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

**pins** [IEnumerable](#)<string>

The pins associated with this sessions bundle.

**allSessionInfo** [IEnumerable](#)<DigitalSessionInformation>

An enumerable of [DigitalSessionInformations](#).

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

```
public IList<SitePinInfo> AggregateSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [DigitalSessionInformations](#).

```
public IEnumerable<DigitalSessionInformation> InstrumentSessions { get; }
```

## Property Value

[IEnumerable](#) <[DigitalSessionInformation](#)>

# Property Pins

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Pins

```
public IEnumerable<string> Pins { get; }
```

### Property Value

[IEnumerable](#) <[string](#)>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public ISemiconductorModuleContext TSMContext { get; }
```

## Property Value

ISemiconductorModuleContext

# Method Do

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Do<T>(SiteData<T>, Action<DigitalSessionInformation, T[]>)

Does an operation on each [DigitalSessionInformation](#) in parallel with the given per-site data.

```
public void Do<T>(SiteData<T> data, Action<DigitalSessionInformation, T[]> action)
```

### Parameters

**data** [SiteData](#)<T>

The per-site data the action relies on.

**action** [Action](#)<[DigitalSessionInformation](#), T[]>

The operation to do. This operation takes [DigitalSessionInformation](#) and per-site data as inputs.

### Type Parameters

**T**

The element type of the per-site data.

# Method DoAndReturnPerSiteResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DoAndReturnPerSiteResults<T> (Func<DigitalSessionInformation, T[]>)

Does an operation on each [DigitalSessionInformation](#) in parallel and returns per-site results.

```
public SiteData<T> DoAndReturnPerSiteResults<T>(Func<DigitalSessionInformation,  
T[]> function)
```

### Parameters

function [Func<DigitalSessionInformation, T\[\]>](#)

The operation to do. This operation takes [DigitalSessionInformation](#) as input and returns per-site array result.

### Returns

[SiteData<T>](#)

The results for all sites.

### Type Parameters

T

The element type of the per-site result.

## DoAndReturnPerSiteResults<T> (Func<DigitalSessionInformation, SiteData<T>>)

Does an operation on each [DigitalSessionInformation](#) in parallel and returns per-site results.

```
public SiteData<T> DoAndReturnPerSiteResults<T>(Func<DigitalSessionInformation,  
SiteData<T>> function)
```

## Parameters

**function** [Func<DigitalSessionInformation, SiteData<T>>](#)

The operation to do. This operation takes [DigitalSessionInformation](#) as input and returns per-site dictionary result.

## Returns

[SiteData<T>](#)

The results for all sites.

## Type Parameters

**T**

The element type of the per-site result.

# Method FilterByPin

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterByPin(string)

Filters current [DigitalSessionsBundle](#) and returns a new one with the requested pin.

```
public DigitalSessionsBundle FilterByPin(string requestedPin)
```

### Parameters

**requestedPin** [string](#) ↗

The requested pin.

### Returns

[DigitalSessionsBundle](#)

A new [DigitalSessionsBundle](#) object with the requested pin.

## FilterByPin(string[])

Filters current [DigitalSessionsBundle](#) and returns a new one with requested pins.

```
public DigitalSessionsBundle FilterByPin(string[] requestedPins)
```

### Parameters

**requestedPins** [string](#) ↗[]

The requested pins.

### Returns

## [DigitalSessionsBundle](#)

A new [DigitalSessionsBundle](#) object with requested pins.

# Method FilterBySite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterBySite(int)

Filters current [DigitalSessionsBundle](#) and returns a new one with the requested site.

```
public DigitalSessionsBundle FilterBySite(int requestedSite)
```

### Parameters

`requestedSite int`

The requested site.

### Returns

[DigitalSessionsBundle](#)

A new [DigitalSessionsBundle](#) object with the requested site.

## FilterBySite(int[])

Filters current [DigitalSessionsBundle](#) and returns a new one with requested sites.

```
public DigitalSessionsBundle FilterBySite(int[] requestedSites)
```

### Parameters

`requestedSites int[]`

The requested sites.

### Returns

## [DigitalSessionsBundle](#)

A new [DigitalSessionsBundle](#) object with requested sites.

# Method PerInstrumentPerSiteResultsToPerSiteResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PerInstrumentPerSiteResultsToPerSiteResults<T>(T[][])

Converts per-instrument per-site results to per-site results.

```
public IDictionary<int, T> PerInstrumentPerSiteResultsToPerSiteResults<T>(T[][]
[] perInstrumentPerSiteResults)
```

### Parameters

**perInstrumentPerSiteResults** `T[][]`

The per-instrument per-site results to convert.

### Returns

[`IDictionary`](#)<[`int`](#), `T`>

The converted per-site results.

### Type Parameters

`T`

The type of the per-instrument per-site result.

# Method PublishPatternResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## PublishPatternResults(bool[][], string)

Publishes measurement results.

```
public void PublishPatternResults(bool[][] results, string publishedDataId)
```

### Parameters

**results** [bool](#)[][]

The pattern result data from multiple pins connected to multiple NI-Digital Pattern instruments. Each element in the burst results array is an array that contains pattern results for the sites of a single instrument session.

**publishedDataId** [string](#)

The unique data id to use when publishing.

# Class FrequencyCounter

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for frequency measurements.

```
public static class FrequencyCounter
```

## Inheritance

[object](#) ← FrequencyCounter

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[MeasureFrequency\(DigitalSessionsBundle, FrequencyMeasurementMode?, double?\)](#)

Measures frequency and returns a pin- and site-aware data object.

# Method MeasureFrequency

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureFrequency(DigitalSessionsBundle, FrequencyMeasurementMode?, double?)

Measures frequency and returns a pin- and site-aware data object.

```
public static PinSiteData<double> MeasureFrequency(this DigitalSessionsBundle  
sessionsBundle, FrequencyMeasurementMode? measurementMode = null, double? measurementTime  
= null)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**measurementMode** FrequencyMeasurementMode?

The frequency measurement mode.

**measurementTime** [double](#)?

The measurement time in seconds.

### Returns

[PinSiteData<double>](#)

The measurements in per-site per-pin format.

# Class HistoryRAM

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for History RAM.

```
public static class HistoryRAM
```

## Inheritance

[object](#) ← HistoryRAM

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[Configure\(NIDigital, HistoryRAMSettings\)](#).

Configures History RAM.

[ConfigureHistoryRAM\(DigitalSessionsBundle, HistoryRAMSettings\)](#).

Configures History RAM.

[FetchHistoryRAMResults\(NIDigital, IDictionary<string, string>\)](#).

Fetches History RAM results.

[FetchHistoryRAMResults\(DigitalSessionsBundle\)](#)

Fetches results from the History RAM and returns as a site-aware object of type DigitalHistoryRamCycleInformation.

[GetConfiguration\(NIDigital\)](#).

Gets History RAM configuration.

[GetHistoryRAMConfiguration\(DigitalSessionsBundle\)](#)

Gets History RAM configuration.

[LogResultsToFiles\(DigitalSessionsBundle, SiteData<List<DigitalHistoryRamCycleInformation>>, SiteData<List<long>>, string, string\)](#).

Logs History RAM results to CSV files. This method should be used for debug only.

# Method Configure

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Configure(NIDigital, HistoryRAMSettings)

Configures History RAM.

```
public static void Configure(this NIDigital session, HistoryRAMSettings settings)
```

### Parameters

**session** NIDigital

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

**settings** [HistoryRAMSettings](#)

The configuration to apply.

# Method ConfigureHistoryRAM

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureHistoryRAM(DigitalSessionsBundle, HistoryRAMSettings)

Configures History RAM.

```
public static void ConfigureHistoryRAM(this DigitalSessionsBundle sessionsBundle,  
HistoryRAMSettings settings)
```

### Parameters

*sessionsBundle* [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

*settings* [HistoryRAMSettings](#)

The configuration to apply.

# Method FetchHistoryRAMResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## FetchHistoryRAMResults(DigitalSessionsBundle)

Fetches results from the History RAM and returns as a site-aware object of type DigitalHistoryRamCycleInformation.

```
public static SiteData<HistoryRAMResults> FetchHistoryRAMResults(this  
DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### Returns

[SiteData<HistoryRAMResults>](#)

The per-site History RAM cycle information and scan cycle numbers.

## FetchHistoryRAMResults(NIDigital, IDictionary<string, string>)

Fetches History RAM results.

```
public static SiteData<HistoryRAMResults> FetchHistoryRAMResults(this NIDigital session,  
IDictionary<string, string> perSitePinSetStrings)
```

### Parameters

**session** NIDigital

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

`perSitePinSetStrings` [IDictionary<string, string>](#)

The per-site pin set strings associated with the session.

Returns

[SiteData<HistoryRAMResults>](#)

The single session per-site History RAM cycle information and scan cycle numbers.

# Method GetConfiguration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetConfiguration(NIDigital)

Gets History RAM configuration.

```
public static HistoryRAMSettings GetConfiguration(this NIDigital session)
```

### Parameters

**session** NIDigital

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

### Returns

[HistoryRAMSettings](#)

Current History RAM configuration.

# Method GetHistoryRAMConfiguration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetHistoryRAMConfiguration(DigitalSessionsBundle)

Gets History RAM configuration.

```
public static HistoryRAMSettings GetHistoryRAMConfiguration(this
DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### Returns

[HistoryRAMSettings](#)

Current History RAM configuration.

# Method LogResultsToFiles

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**LogResultsToFiles(DigitalSessionsBundle,  
SiteData<List<DigitalHistoryRamCycleInformation>>,  
SiteData<List<long>>, string, string)**

Logs History RAM results to CSV files. This method should be used for debug only.

```
public static void LogResultsToFiles(this DigitalSessionsBundle sessionsBundle,  
SiteData<List<DigitalHistoryRamCycleInformation>> cycleInformation, SiteData<List<long>>  
scanCycleNumbers, string patternName, string outputDirectory)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**cycleInformation** [SiteData<List<DigitalHistoryRamCycleInformation>>](#)

The per-site cycle information.

**scanCycleNumbers** [SiteData<List<long>>](#)

The per-site scan cycle numbers.

**patternName** [string](#)

The name of the pattern.

**outputDirectory** [string](#)

The directory to save the CSV files.

# Class HistoryRAMResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines History RAM results type.

```
public class HistoryRAMResults
```

## Inheritance

[object](#) ← HistoryRAMResults

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[HistoryRAMResults\(List<DigitalHistoryRamCycleInformation>, List<long>\)](#).

Constructs a History RAM results object.

## Properties

[CycleInformation](#)

The per-site cycle information.

[ScanCycleNumbers](#)

The per-site scan cycle numbers.

# Constructor HistoryRAMResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## HistoryRAMResults(List<DigitalHistoryRamCycleInformation>, List<long>)

Constructs a History RAM results object.

```
public HistoryRAMResults(List<DigitalHistoryRamCycleInformation> cycleInformation,  
List<long> scanCycleNumbers)
```

### Parameters

**cycleInformation** [List](#)<DigitalHistoryRamCycleInformation>

The cycle information.

**scanCycleNumbers** [List](#)<long>

The scan cycle numbers.

# Property CycleInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CycleInformation

The per-site cycle information.

```
public List<DigitalHistoryRamCycleInformation> CycleInformation { get; }
```

## Property Value

[List](#) <DigitalHistoryRamCycleInformation>

# Property ScanCycleNumbers

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ScanCycleNumbers

The per-site scan cycle numbers.

```
public List<long> ScanCycleNumbers { get; }
```

## Property Value

[List](#) <[long](#)>

# Class HistoryRAMSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines settings of History RAM.

```
public class HistoryRAMSettings
```

## Inheritance

[object](#) ← HistoryRAMSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[HistoryRAMSettings\(\)](#)

## Properties

[BufferSizePerSite](#)

The in-memory History RAM buffer size in samples.

[CyclesToAcquire](#)

The cycles to acquire after the trigger conditions are met.

[MaximumSamplesToAcquirePerSite](#)

The maximum samples to acquire for each site.

[NumberOfSamplesIsFinite](#)

Whether the number of samples is finite.

[TriggerSettings](#)

The History RAM trigger settings.

# Constructor HistoryRAMSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## HistoryRAMSettings()

```
public HistoryRAMSettings()
```

# Property BufferSizePerSite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## BufferSizePerSite

The in-memory History RAM buffer size in samples.

```
public long BufferSizePerSite { get; set; }
```

### Property Value

[long](#)

# Property CyclesToAcquire

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CyclesToAcquire

The cycles to acquire after the trigger conditions are met.

```
public HistoryRamCycle CyclesToAcquire { get; set; }
```

## Property Value

HistoryRamCycle

# Property MaximumSamplesToAcquirePerSite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MaximumSamplesToAcquirePerSite

The maximum samples to acquire for each site.

```
public int MaximumSamplesToAcquirePerSite { get; set; }
```

### Property Value

[int](#)

# Property NumberOfSamplesIsFinite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## NumberOfSamplesIsFinite

Whether the number of samples is finite.

```
public bool NumberOfSamplesIsFinite { get; set; }
```

### Property Value

[bool](#) ↗

# Property TriggerSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## TriggerSettings

The History RAM trigger settings.

```
public HistoryRAMTriggerSettings TriggerSettings { get; set; }
```

### Property Value

[HistoryRAMTriggerSettings](#)

# Class HistoryRAMTriggerSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines History RAM trigger settings.

```
public class HistoryRAMTriggerSettings
```

## Inheritance

[object](#) ← HistoryRAMTriggerSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[HistoryRAMTriggerSettings\(\)](#)

## Properties

### [CycleNumber](#)

The number of cycles to execute. Use this property only when [TriggerType](#) is set to National Instruments.ModularInstruments.NIDigital.HistoryRamTriggerType.CycleNumber.

### [CycleOffset](#)

The cycle offset. Use this property only when [TriggerType](#) is set to NationalInstruments.Modular Instruments.NIDigital.HistoryRamTriggerType.PatternLabel.

### [PatternLabel](#)

The pattern label. Use this property only when [TriggerType](#) is set to NationalInstruments.Modular Instruments.NIDigital.HistoryRamTriggerType.PatternLabel.

### [PretriggerSamples](#)

The number of samples to acquire before the History RAM trigger.

### [TriggerType](#)

The trigger type.

### [VectorOffset](#)

The vector offset. Use this property only when [TriggerType](#) is set to NationalInstruments.ModularInstruments.NIDigital.HistoryRamTriggerType.PatternLabel.

# Constructor HistoryRAMTriggerSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## HistoryRAMTriggerSettings()

```
public HistoryRAMTriggerSettings()
```

# Property CycleNumber

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CycleNumber

The number of cycles to execute. Use this property only when [TriggerType](#) is set to `NationalInstruments.ModularInstruments.NIDigital.HistoryRamTriggerType.CycleNumber`.

```
public long CycleNumber { get; set; }
```

## Property Value

[long](#) ↗

# Property CycleOffset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CycleOffset

The cycle offset. Use this property only when [TriggerType](#) is set to NationalInstruments.Modular Instruments.NIDigital.HistoryRamTriggerType.PatternLabel.

```
public long CycleOffset { get; set; }
```

### Property Value

[long↗](#)

# Property PatternLabel

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## PatternLabel

The pattern label. Use this property only when [TriggerType](#) is set to NationalInstruments.Modular Instruments.NIDigital.HistoryRamTriggerType.PatternLabel.

```
public string PatternLabel { get; set; }
```

## Property Value

[string](#) ↗

# Property PretriggerSamples

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## PretriggerSamples

The number of samples to acquire before the History RAM trigger.

```
public int PretriggerSamples { get; set; }
```

### Property Value

[int](#)

# Property TriggerType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## TriggerType

The trigger type.

```
public HistoryRamTriggerType TriggerType { get; set; }
```

## Property Value

HistoryRamTriggerType

# Property VectorOffset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## VectorOffset

The vector offset. Use this property only when [TriggerType](#) is set to NationalInstruments.Modular Instruments.NIDigital.HistoryRamTriggerType.PatternLabel.

```
public long VectorOffset { get; set; }
```

## Property Value

[long↗](#)

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[Close\(ISemiconductorModuleContext, bool\)](#)

Closes all NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions.

[Initialize\(ISemiconductorModuleContext, string, string, bool, bool\)](#)

Initializes NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions in the test system.

[Reset\(ISemiconductorModuleContext, bool\)](#)

Resets all NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions.

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close(ISemiconductorModuleContext, bool)

Closes all NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions.

```
public static void Close(ISemiconductorModuleContext tsmContext, bool reset = true)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**reset** [bool](#)

Whether to reset the device sessions before closing.

# Method Initialize

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Initialize(ISemiconductorModuleContext, string, string, bool, bool)

Initializes NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions in the test system.

```
public static void Initialize(ISemiconductorModuleContext tsmContext, string  
levelsSheetToApply = "", string timingSheetToApply = "", bool resetDevice = false, bool  
applySourceWaveformData = false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**levelsSheetToApply** [string](#)

The name of the levels sheet to apply.

**timingSheetToApply** [string](#)

The name of the timing sheet to apply.

**resetDevice** [bool](#)

Whether to reset device during initialization.

**applySourceWaveformData** [bool](#)

Whether to apply the data in waveform files to source waveforms.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset(ISemiconductorModuleContext, bool)

Resets all NationalInstruments.ModularInstruments.NIDigital.NIDigital sessions.

```
public static void Reset(ISemiconductorModuleContext tsmContext, bool resetDevice = false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**resetDevice** [bool](#)

Whether to perform a hard reset on the device.

# Class LevelsAndTiming

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for levels and timing.

```
public static class LevelsAndTiming
```

## Inheritance

[object](#) ← LevelsAndTiming

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ApplyLevelsAndTiming\(DigitalSessionsBundle, string, string\)](#)

Applies digital levels and timing defined in the loaded levels and timing sheets.

[ApplyTDROffsets\(DigitalSessionsBundle, PrecisionTimeSpan\[\]\)](#)

Applies the correction for propagation delay offsets to a digital pattern instrument. Use this method to apply per-instrument session per-pin offsets.

[ApplyTDROffsets\(DigitalSessionsBundle, PinSiteData<PrecisionTimeSpan>\)](#)

Applies the correction for propagation delay offsets to a digital pattern instrument. Use this method to apply per-site per-pin offsets.

[ConfigureSingleLevel\(DigitalPinSet, LevelType, double\)](#)

Configures a single level.

[ConfigureSingleLevel\(DigitalSessionsBundle, LevelType, SiteData<double>\)](#)

Configures a single level. Use this method to configure different levels for different sites.

[ConfigureSingleLevel\(DigitalSessionsBundle, LevelType, double\)](#)

Configures a single level. Use this method to configure the same level for all sessions.

[ConfigureTdrEndpointTermination\(DigitalSessionsBundle, TdrEndpointTermination\)](#)

## [ConfigureTerminationMode\(DigitalSessionsBundle, TerminationMode\)](#)

Configures termination mode.

## [ConfigureTimeSetCompareEdgesStrobe\(DigitalSessionsBundle, string, PinSiteData<double>\)](#)

Configures the strobe edge time. Use this method to configure different strobe edge times for different site-pin pairs.

## [ConfigureTimeSetCompareEdgesStrobe\(DigitalSessionsBundle, string, SiteData<double>\)](#)

Configures the strobe edge time. Use this method to configure different strobe edge times for different sites.

## [ConfigureTimeSetCompareEdgesStrobe\(DigitalSessionsBundle, string, double\)](#)

Configures the strobe edge time. Use this method to configure the same strobe edge time to all sessions.

## [ConfigureTimeSetDriveEdges\(DigitalSessionsBundle, string, DriveFormat, double, double, double, double\)](#)

Configures the drive format and drive edge placement.

## [ConfigureTimeSetDriveEdges\(DigitalSessionsBundle, string, DriveFormat, double, double, double, double, double, double\)](#)

Configures the drive format and drive edge placement.

## [ConfigureTimeSetPeriod\(DigitalSessionsBundle, string, double\)](#)

Configures the period of a time set.

## [ConfigureVoltageLevels\(DigitalSessionsBundle, double, double, double, double, double\)](#)

Configures multiple types of levels.

## [LoadTDROffsetsFromFile\(DigitalSessionsBundle, string, out PrecisionTimeSpan\[\]\[\], bool\)](#)

Loads TDR offsets from a file.

## [LoadTDROffsetsFromFile\(DigitalSessionsBundle, string, bool\)](#)

Loads TDR offsets from a file.

## [MeasureTDROffsets\(DigitalSessionsBundle, out PrecisionTimeSpan\[\]\[\], bool\)](#)

Measures propagation delays through cables, connectors, and load boards using Time-Domain Reflectometry (TDR). You can optionally apply the offsets to the pins. Use this method to measure per-instrument session per-pin offsets.

## [MeasureTDROffsets\(DigitalSessionsBundle, bool\)](#)

Measures propagation delays through cables, connectors, and load boards using Time-Domain Reflectometry (TDR). You can optionally apply the offsets to the pins. Use this method to measure per-site per-pin offsets.

[SaveTDROffsetsToFile\(DigitalSessionsBundle, PrecisionTimeSpan\[\], string\)](#)

Saves TDR offsets to a file.

[SaveTDROffsetsToFile\(DigitalSessionsBundle, PinSiteData<PrecisionTimeSpan>, string\)](#)

Saves TDR offsets to a file.

# Method ApplyLevelsAndTiming

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApplyLevelsAndTiming(DigitalSessionsBundle, string, string)

Applies digital levels and timing defined in the loaded levels and timing sheets.

```
public static void ApplyLevelsAndTiming(this DigitalSessionsBundle sessionsBundle, string  
levelsSheet, string timingSheet)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**levelsSheet** [string](#) ↗

The name of the levels sheet to apply.

**timingSheet** [string](#) ↗

The name of the timing sheet to apply.

# Method ApplyTDROffsets

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApplyTDROffsets(DigitalSessionsBundle, PinSiteData<PrecisionTimeSpan>)

Applies the correction for propagation delay offsets to a digital pattern instrument. Use this method to apply per-site per-pin offsets.

```
public static void ApplyTDROffsets(this DigitalSessionsBundle sessionsBundle,  
PinSiteData<PrecisionTimeSpan> offsets)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**offsets** [PinSiteData](#)<PrecisionTimeSpan>

The per-site per-pin offsets to apply.

## ApplyTDROffsets(DigitalSessionsBundle, PrecisionTimeSpan[][][])

Applies the correction for propagation delay offsets to a digital pattern instrument. Use this method to apply per-instrument session per-pin offsets.

```
public static void ApplyTDROffsets(this DigitalSessionsBundle sessionsBundle,  
PrecisionTimeSpan[][][] offsets)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**offsets** PrecisionTimeSpan[][]

Offsets to apply. Where the first dimension represents instrument sessions and the second dimension represents pins.

# Method ConfigureSingleLevel

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureSingleLevel(DigitalSessionsBundle, LevelType, double)

Configures a single level. Use this method to configure the same level for all sessions.

```
public static void ConfigureSingleLevel(this DigitalSessionsBundle sessionsBundle,  
    LevelsAndTiming.LevelType levelType, double levelValue)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**levelType** [LevelsAndTiming.LevelType](#)

The type of level to configure.

**levelValue** [double](#)

The value of level to configure.

## ConfigureSingleLevel(DigitalSessionsBundle, LevelType, SiteData<double>)

Configures a single level. Use this method to configure different levels for different sites.

```
public static void ConfigureSingleLevel(this DigitalSessionsBundle sessionsBundle,  
    LevelsAndTiming.LevelType levelType, SiteData<double> perSiteLevelValues)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

#### **levelType** [LevelsAndTiming.LevelType](#)

The type of level to configure.

#### **perSiteLevelValues** [SiteData<double>](#)

The per-site value of level to configure.

## ConfigureSingleLevel(DigitalPinSet, LevelType, double)

Configures a single level.

```
public static void ConfigureSingleLevel(this DigitalPinSet pinSet, LevelsAndTiming.LevelType  
levelType, double levelValue)
```

### Parameters

#### **pinSet** DigitalPinSet

The NationalInstruments.ModularInstruments.NIDigital.DigitalPinSet object.

#### **levelType** [LevelsAndTiming.LevelType](#)

The type of the level to configure.

#### **levelValue** [double](#)

The value of the level to configure.

# Method ConfigureTdrEndpointTermination

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTdrEndpointTermination(DigitalSessionsBundle, TdrEndpointTermination)

```
public static void ConfigureTdrEndpointTermination(this DigitalSessionsBundle sessionsBundle, TdrEndpointTermination tdrEndpointTermination)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**tdrEndpointTermination** [TdrEndpointTermination](#)

TDR Endpoint Termination type. The default value is `NationalInstruments.ModularInstruments.NIDigital.TdrEndpointTermination.TdrToOpenCircuit`.

# Method ConfigureTerminationMode

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTerminationMode(DigitalSessionsBundle, TerminationMode)

Configures termination mode.

```
public static void ConfigureTerminationMode(this DigitalSessionsBundle sessionsBundle,  
TerminationMode terminationMode)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**terminationMode** TerminationMode

The termination mode to configure.

# Method

## ConfigureTimeSetCompareEdgesStrobe

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### ConfigureTimeSetCompareEdgesStrobe(DigitalSessionsBundle, string, double)

Configures the strobe edge time. Use this method to configure the same strobe edge time to all sessions.

```
public static void ConfigureTimeSetCompareEdgesStrobe(this DigitalSessionsBundle  
sessionsBundle, string timeSet, double compareEdge)
```

#### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**timeSet** [string](#)

The name of the time set.

**compareEdge** [double](#)

The strobe edge time to configure.

### ConfigureTimeSetCompareEdgesStrobe(DigitalSessionsBundle, string, SiteData<double>)

Configures the strobe edge time. Use this method to configure different strobe edge times for different sites.

```
public static void ConfigureTimeSetCompareEdgesStrobe(this DigitalSessionsBundle  
sessionsBundle, string timeSet, SiteData<double> compareEdges)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**timeSet** [string](#)

The name of the time set.

**compareEdges** [SiteData<double>](#)

The per-site strobe edge time to configure.

## ConfigureTimeSetCompareEdgesStrobe(DigitalSessionsBundle, string, PinSiteData<double>)

Configures the strobe edge time. Use this method to configure different strobe edge times for different site-pin pairs.

```
public static void ConfigureTimeSetCompareEdgesStrobe(this DigitalSessionsBundle  
sessionsBundle, string timeSet, PinSiteData<double> compareEdges)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**timeSet** [string](#)

The name of the time set.

**compareEdges** [PinSiteData<double>](#)

The strobe edge time for all site-pin pairs to configure.

# Method ConfigureTimeSetDriveEdges

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigureTimeSetDriveEdges(DigitalSessionsBundle, string, DriveFormat, double, double, double, double)**

Configures the drive format and drive edge placement.

```
public static void ConfigureTimeSetDriveEdges(this DigitalSessionsBundle sessionsBundle,
    string timeSet, DriveFormat format, double driveOn, double driveData, double driveReturn,
    double driveOff)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**timeSet** [string](#)

The name of the time set.

**format** [DriveFormat](#)

The drive format of the time set.

**driveOn** [double](#)

The delay from the beginning of the vector period for turning on the pin driver.

**driveData** [double](#)

The delay from the beginning of the vector period until the pattern data is driven to the pattern value.

**driveReturn** [double](#)

The delay from the beginning of the vector period until the pin changes from the pattern data to the return value.

## **driveOff** [double](#)

The delay from the beginning of the vector period to turn off the pin driver.

# ConfigureTimeSetDriveEdges(DigitalSessionsBundle, string, DriveFormat, double, double, double, double, double)

Configures the drive format and drive edge placement.

```
public static void ConfigureTimeSetDriveEdges(this DigitalSessionsBundle sessionsBundle,  
    string timeSet, DriveFormat format, double driveOn, double driveData, double driveReturn,  
    double driveOff, double driveData2, double driveReturn2)
```

## Parameters

### **sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### **timeSet** [string](#)

The name of the time set.

### **format** [DriveFormat](#)

The drive format of the time set.

### **driveOn** [double](#)

The delay from the beginning of the vector period for turning on the pin driver.

### **driveData** [double](#)

The delay from the beginning of the vector period until the pattern data is driven to the pattern value.

### **driveReturn** [double](#)

The delay from the beginning of the vector period until the pin changes from the pattern data to the return value.

### **driveOff** [double](#)

The delay from the beginning of the vector period to turn off the pin driver.

## **driveData2** [double](#)

The delay from the beginning of the vector period until the pattern data is driven to the second pattern value.

## **driveReturn2** [double](#)

The delay from the beginning of the vector period until the pin changes from the second pattern data to the return value.

# Method ConfigureTimeSetPeriod

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureTimeSetPeriod(DigitalSessionsBundle, string, double)

Configures the period of a time set.

```
public static void ConfigureTimeSetPeriod(this DigitalSessionsBundle sessionsBundle, string
timeSet, double period)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**timeSet** [string](#) ↗

The name of the time set.

**period** [double](#) ↗

The period to configure.

# Method ConfigureVoltageLevels

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**ConfigureVoltageLevels(DigitalSessionsBundle, double, double, double, double, double)**

Configures multiple types of levels.

```
public static void ConfigureVoltageLevels(this DigitalSessionsBundle sessionsBundle, double vil, double vih, double vol, double voh, double vterm)
```

Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**vil** [double](#)

The vil value to configure.

**vih** [double](#)

The vih value to configure.

**vol** [double](#)

The vol value to configure.

**voh** [double](#)

The voh value to configure.

**vterm** [double](#)

The vterm value to configure.

# Method LoadTDROffsetsFromFile

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## LoadTDROffsetsFromFile(DigitalSessionsBundle, string, bool)

Loads TDR offsets from a file.

```
public static PinSiteData<PrecisionTimeSpan> LoadTDROffsetsFromFile(this  
DigitalSessionsBundle sessionsBundle, string filePath, bool throwOnMissingChannels = true)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**filePath** [string](#)

The path of the file to load the offsets.

**throwOnMissingChannels** [bool](#)

Whether to throw an exception if the offset for any channel is missing.

### Returns

[PinSiteData](#)<PrecisionTimeSpan>

TDR offset values retrieved from the file.

### Exceptions

[ArgumentException](#)

This exception will be thrown if `throwOnMissingChannels` is true and an offset value was not found in the file for one or more of channels in the sessions bundle.

## LoadTDROffsetsFromFile(DigitalSessionsBundle, string, out PrecisionTimeSpan[][], bool)

Loads TDR offsets from a file.

```
public static void LoadTDROffsetsFromFile(this DigitalSessionsBundle sessionsBundle, string filePath, out PrecisionTimeSpan[][] offsets, bool throwOnMissingChannels = true)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**filePath** [string](#)

The path of the file to load the offsets.

**offsets** [PrecisionTimeSpan\[\]\[\]](#)

TDR offset values retrieved from the file. Where the first dimension represents instrument sessions and the second dimension represents pins.

**throwOnMissingChannels** [bool](#)

Whether to throw a message if the offset for any channel is missing.

### Exceptions

[ArgumentException](#)

This exception will be thrown if throwOnMissingChannels is true and an offset value was not found in the file for one or more of channels in the sessions bundle.

# Method MeasureTDROffsets

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureTDROffsets(DigitalSessionsBundle, bool)

Measures propagation delays through cables, connectors, and load boards using Time-Domain Reflectometry (TDR). You can optionally apply the offsets to the pins. Use this method to measure per-site per-pin offsets.

```
public static PinSiteData<PrecisionTimeSpan> MeasureTDROffsets(this DigitalSessionsBundle sessionsBundle, bool apply = false)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**apply** [bool](#) ↗

Whether to apply the offsets to the pins.

### Returns

[PinSiteData](#)<PrecisionTimeSpan>

## MeasureTDROffsets(DigitalSessionsBundle, out PrecisionTimeSpan[][], bool)

Measures propagation delays through cables, connectors, and load boards using Time-Domain Reflectometry (TDR). You can optionally apply the offsets to the pins. Use this method to measure per-instrument session per-pin offsets.

```
public static void MeasureTDROffsets(this DigitalSessionsBundle sessionsBundle, out PrecisionTimeSpan[][] offsets, bool apply = false)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**offsets** PrecisionTimeSpan[][]

The measured TDR offsets. Where the first dimension represents instrument sessions, and the second dimension represents pins.

**apply** [bool](#)

Whether to apply the offsets to the pins.

# Method SaveTDROffsetsToFile

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**SaveTDROffsetsToFile(DigitalSessionsBundle,  
PinSiteData<PrecisionTimeSpan>, string)**

Saves TDR offsets to a file.

```
public static void SaveTDROffsetsToFile(this DigitalSessionsBundle sessionsBundle,  
PinSiteData<PrecisionTimeSpan> offsets, string filePath)
```

Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**offsets** [PinSiteData](#)<PrecisionTimeSpan>

The per-site per-pin offsets to save.

**filePath** [string](#) ↗

The path of the file to save the offsets to.

**SaveTDROffsetsToFile(DigitalSessionsBundle,  
PrecisionTimeSpan[][], string)**

Saves TDR offsets to a file.

```
public static void SaveTDROffsetsToFile(this DigitalSessionsBundle sessionsBundle,  
PrecisionTimeSpan[][] offsets, string filePath)
```

Parameters

## **sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

## **offsets** PrecisionTimeSpan[][]

The per-instrument session per-pin offsets to save. Where the first dimension represents instrument sessions and the second dimension represents pins.

## **filePath** [string](#)

The path of the file to save the offsets to.

## Remarks

The resulting file is pinmap specific. It is recommended that the filename provided contains the same name as the pinmap, as well as timestamp.

# Enum LevelsAndTiming.LevelType

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines voltage level types.

```
public enum LevelsAndTiming.LevelType
```

## Fields

**Ioh = 6**

The current that the DUT sources to the active load while outputting a voltage above [Vcom](#).

**Iol = 5**

The current that the DUT sinks from the active load while outputting a voltage below [Vcom](#).

**Vcom = 7**

The commutating voltage at which the active load circuit switches between sourcing current and sinking current.

**Vih = 1**

The input voltage that the digital pattern instrument applies to the input of the DUT when the test instrument drives a logic high (1).

**Vil = 0**

The input voltage that the digital pattern instrument applies to the input of the DUT when the test instrument drives a logic low (0).

**Voh = 3**

The output voltage from the DUT above which the comparator on the test instrument interprets a logic high (H).

**Vol = 2**

The output voltage from the DUT below which the comparator on the test instrument interprets a logic low (L).

**Vterm = 4**

The termination voltage the instrument applies during non-drive cycles.

# Class PPMU

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for PPMU measurements.

```
public static class PPMU
```

## Inheritance

[object](#) ← PPMU

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ConfigureApertureTime\(DigitalSessionsBundle, double\)](#)

Configures the aperture time for the PPMU measurement.

[DisconnectOutput\(DigitalSessionsBundle, double?\)](#)

The pin is electrically disconnected from instrument functions.

[Force\(NIDigital, string, PPMUSettings\)](#)

Forces on digital devices.

[ForceCurrent\(DigitalSessionsBundle, IDictionary<string, PPMUSettings>\)](#)

Forces current on the target pin(s).

[ForceCurrent\(DigitalSessionsBundle, double, double?, double?, double?, double?, double?\)](#)

Forces current on the target pin(s) at the specified level. You must provide a current level value, and the method will assume all other properties that have been previously set. Optionally, you can also provide a specific voltage limit and current level range values directly.

[ForceVoltage\(DigitalSessionsBundle, PinSiteData<double>, double?, double?, double?\)](#)

Forces voltage on the target pin(s) at the specified level. You must provide the voltage level values, and the method will assume all other properties that have been previously set. Optionally, you can also provide a specific current limit, current limit range, and voltage level range values directly.

## [ForceVoltage\(DigitalSessionsBundle, SiteData<double>, double?, double?, double?\)](#)

Forces voltage on the target pin(s) at the specified level. You must provide the voltage level values, and the method will assume all other properties that have been previously set. Optionally, you can also provide a specific current limit, current limit range, and voltage level range values directly.

## [ForceVoltage\(DigitalSessionsBundle, IDictionary<string, PPMUSettings>\)](#)

Forces voltage on the target pin(s).

## [ForceVoltage\(DigitalSessionsBundle, double, double?, double?, double?\)](#)

Forces voltage on the target pin(s) at the specified level. You must provide the voltage level value, and the method will assume all other properties that have been previously set. Optionally, you can also provide a specific current limit, current limit range, and voltage level range values directly.

## [Measure\(DigitalSessionInformation, MeasurementType\)](#)

Measures on digital devices.

## [MeasureAndPublishCurrent\(DigitalSessionsBundle, string\)](#)

Measures the current on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

## [MeasureAndPublishCurrent\(DigitalSessionsBundle, string, out double\[\]\[\]\)](#)

Measures the current on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

## [MeasureAndPublishVoltage\(DigitalSessionsBundle, string\)](#)

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

## [MeasureAndPublishVoltage\(DigitalSessionsBundle, string, out double\[\]\[\]\)](#)

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

## [MeasureAndReturnPerInstrumentPerChannelResults\(DigitalSessionsBundle, MeasurementType\)](#)

Measures and returns per-instrument per-channel results.

## [MeasureCurrent\(DigitalSessionsBundle\)](#)

Measures the current on the target pin(s) and returns a pin- and site-aware data object.

## [MeasureVoltage\(DigitalSessionsBundle\)](#)

Measures the voltage on the target pin(s) and returns a pin- and site-aware data object.

## [SelectDigital\(DigitalSessionsBundle, double?\)](#)

Sets the Selected Function mode to Digital, such that the pattern sequencer controls the specified pin(s).

[SelectPPMU\(DigitalSessionsBundle, double?\)](#)

Sets the Selected Function mode to PPMU, such that the PPMU controls the specified pin(s) and connects the PPMU. The pin driver is in a non-drive state, and the active load is disabled.

[TurnOffOutput\(DigitalSessionsBundle, double?\)](#)

Sets the Selected Function mode to Off to put the digital driver into a non-drive state, disables the active load, disconnects the PPMU, and closes the I/O switch connecting the instrument channel.

# Method ConfigureApertureTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureApertureTime(DigitalSessionsBundle, double)

Configures the aperture time for the PPMU measurement.

```
public static void ConfigureApertureTime(this DigitalSessionsBundle sessionsBundle,  
double apertureTimeInSeconds)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**apertureTimeInSeconds** [double](#) ↗

The aperture time in seconds.

# Method DisconnectOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DisconnectOutput(DigitalSessionsBundle, double?)

The pin is electrically disconnected from instrument functions.

```
public static void DisconnectOutput(this DigitalSessionsBundle sessionsBundle, double?
settlingTimeInSeconds = null)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**settlingTimeInSeconds** [double](#)?

The settling time in seconds. Passing a Null value (default) bypasses the wait operation (No-op).

### Remarks

Remarks:

- Selecting this function causes the PPMU to stop sourcing prior to disconnecting the pin.
- CAUTION: In the Disconnect state, some I/O protection and sensing circuitry remain exposed. Do not subject the instrument to voltage beyond its operating range.

Example:

```
DigitalSessionsBundle myDutPin = new TSMSessionManager().Digital("MyDutPin");
myDutPin.DisconnectOutput(settlingTimeInSeconds: 0.1);
```

# Method Force

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Force(NIDigital, string, PPMUSettings)

Forces on digital devices.

```
public static void Force(this NIDigital session, string pinSetString, PPMUSettings settings)
```

### Parameters

**session** NIDigital

The NationalInstruments.ModularInstruments.NIDigital.NIDigital session.

**pinSetString** [string](#)

The pin set string.

**settings** [PPMUSettings](#)

The forcing settings.

# Method ForceCurrent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ForceCurrent(DigitalSessionsBundle, double, double?, double?, double?, double?, double?)

Forces current on the target pin(s) at the specified level. You must provide a current level value, and the method will assume all other properties that have been previously set. Optionally, you can also provide a specific voltage limit and current level range values directly.

```
public static void ForceCurrent(this DigitalSessionsBundle sessionsBundle, double
    currentLevel, double? currentLevelRange = null, double? voltageLimitLow = null, double?
    voltageLimitHigh = null, double? apertureTime = null, double? settlingTime = null)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**currentLevel** [double](#)?

The current level.

**currentLevelRange** [double](#)?

The current level range.

**voltageLimitLow** [double](#)?

The low voltage limit.

**voltageLimitHigh** [double](#)?

The high voltage limit.

**apertureTime** [double](#)?

The aperture time.

`settlingTime` [double](#)?

The settling time.

## Remarks

Use this method to force the same current level on all sites.

**ForceCurrent(DigitalSessionsBundle, IDictionary<string, PPMUSettings>)**

Forces current on the target pin(s).

```
public static void ForceCurrent(this DigitalSessionsBundle sessionsBundle,  
IDictionary<string, PPMUSettings> settings)
```

## Parameters

`sessionsBundle` [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

`settings` [IDictionary](#)<[string](#), [PPMUSettings](#)>

The per-pin settings to use.

# Method ForceVoltage

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ForceVoltage(DigitalSessionsBundle, double, double?, double?, double?)

Forces voltage on the target pin(s) at the specified level. You must provide the voltage level value, and the method will assume all other properties that have been previously set. Optionally, you can also provide a specific current limit, current limit range, and voltage level range values directly.

```
public static void ForceVoltage(this DigitalSessionsBundle sessionsBundle, double voltageLevel, double? currentLimitRange = null, double? apertureTime = null, double? settlingTime = null)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**voltageLevel** [double](#)?

The voltage level.

**currentLimitRange** [double](#)?

The current limit range.

**apertureTime** [double](#)?

The aperture Time.

**settlingTime** [double](#)?

The settling time.

### Remarks

Use this method to force the same voltage level on all sessions.

## ForceVoltage(DigitalSessionsBundle, SiteData<double>, double?, double?, double?)

Forces voltage on the target pin(s) at the specified level. You must provide the voltage level values, and the method will assume all other properties that have been previously set. Optionally, you can also provide a specific current limit, current limit range, and voltage level range values directly.

```
public static void ForceVoltage(this DigitalSessionsBundle sessionsBundle, SiteData<double>
voltageLevels, double? currentLimitRange = null, double? apertureTime = null, double?
settlingTime = null)
```

### Parameters

`sessionsBundle` [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

`voltageLevels` [SiteData<double>](#)

The voltage levels for all sites.

`currentLimitRange` [double](#)?

The current limit range.

`apertureTime` [double](#)?

The aperture Time.

`settlingTime` [double](#)?

The settling time.

### Remarks

Use this method to force different voltage levels on different sites.

## ForceVoltage(DigitalSessionsBundle, PinSiteData<double>, double?, double?, double?)

Forces voltage on the target pin(s) at the specified level. You must provide the voltage level values, and the method will assume all other properties that have been previously set. Optionally, you can also

provide a specific current limit, current limit range, and voltage level range values directly.

```
public static void ForceVoltage(this DigitalSessionsBundle sessionsBundle,  
PinSiteData<double> voltageLevels, double? currentLimitRange = null, double? apertureTime =  
null, double? settlingTime = null)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**voltageLevels** [PinSiteData<double>](#)

The voltage levels for all site-pin pairs.

**currentLimitRange** [double?](#)

The current limit range.

**apertureTime** [double?](#)

The aperture Time.

**settlingTime** [double?](#)

The settling time.

## Remarks

Use this method to force different voltage levels for different site-pin pairs.

## ForceVoltage(DigitalSessionsBundle, IDictionary<string, PPMUSettings>)

Forces voltage on the target pin(s).

```
public static void ForceVoltage(this DigitalSessionsBundle sessionsBundle,  
IDictionary<string, PPMUSettings> settings)
```

## Parameters

`sessionsBundle` [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

`settings` [IDictionary](#)<[string](#), [PPMUSettings](#)>

The per-pin settings to use.

# Method Measure

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## Measure(DigitalSessionInformation, MeasurementType)

Measures on digital devices.

```
public static double[] Measure(this DigitalSessionInformation sessionInfo,  
MeasurementType measurementType)
```

### Parameters

**sessionInfo** [DigitalSessionInformation](#)

The [DigitalSessionInformation](#) object.

**measurementType** [MeasurementType](#)

The type of the measurement, could be either voltage or current.

### Returns

[double](#)[]

The measurements.

# Method MeasureAndPublishCurrent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureAndPublishCurrent(DigitalSessionsBundle, string, out double[][])

Measures the current on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static void MeasureAndPublishCurrent(this DigitalSessionsBundle sessionsBundle,  
    string publishedDataId, out double[][] currentMeasurements)
```

### Parameters

`sessionsBundle` [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

`publishedDataId` [string](#)

The unique data id to be used when publishing.

`currentMeasurements` [double](#)[][]

The returned current measurements.

### Remarks

Use this method to save test time if the measurement results are not needed for any other operations. Otherwise, use the override for this method that returns PinSiteData.

## MeasureAndPublishCurrent(DigitalSessionsBundle, string)

Measures the current on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static PinSiteData<double> MeasureAndPublishCurrent(this DigitalSessionsBundle sessionsBundle, string publishedDataId)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**publishedDataId** [string](#)

The unique data id to use when publishing.

## Returns

[PinSiteData<double>](#)

The pin-site aware current measurements.

# Method MeasureAndPublishVoltage

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureAndPublishVoltage(DigitalSessionsBundle, string, out double[][])

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static void MeasureAndPublishVoltage(this DigitalSessionsBundle sessionsBundle,  
    string publishedDataId, out double[][] voltageMeasurements)
```

### Parameters

`sessionsBundle` [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

`publishedDataId` [string](#)

The unique data id to use when publishing.

`voltageMeasurements` [double](#)[][]

The returned voltage measurements.

### Remarks

Use this method to save test time if the measurement results are not needed for any other operations. Otherwise, use the override for this method that returns PinSiteData.

## MeasureAndPublishVoltage(DigitalSessionsBundle, string)

Measures the voltage on the target pin(s) and immediately publishes the results using the `publishedDataId` passed in.

```
public static PinSiteData<double> MeasureAndPublishVoltage(this DigitalSessionsBundle sessionsBundle, string publishedDataId)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**publishedDataId** [string](#)

The unique data id to use when publishing.

## Returns

[PinSiteData<double>](#)

The pin-site aware voltage measurements.

# Method

## MeasureAndReturnPerInstrumentPerChannelResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### MeasureAndReturnPerInstrumentPerChannelResults(DigitalSessionsBundle, MeasurementType)

Measures and returns per-instrument per-channel results.

```
public static double[][] MeasureAndReturnPerInstrumentPerChannelResults(this
    DigitalSessionsBundle sessionsBundle, MeasurementType measurementType)
```

#### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**measurementType** [MeasurementType](#)

The type of the measurement, could be either voltage or current.

#### Returns

[double](#)[][]

The measurements in per-instrument per-channel format.

# Method MeasureCurrent

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureCurrent(DigitalSessionsBundle)

Measures the current on the target pin(s) and returns a pin- and site-aware data object.

```
public static PinSiteData<double> MeasureCurrent(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### Returns

[PinSiteData<double>](#)

The current measurements.

# Method MeasureVoltage

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## MeasureVoltage(DigitalSessionsBundle)

Measures the voltage on the target pin(s) and returns a pin- and site-aware data object.

```
public static PinSiteData<double> MeasureVoltage(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### Returns

[PinSiteData<double>](#)

The voltage measurements.

# Method SelectDigital

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SelectDigital(DigitalSessionsBundle, double?)

Sets the Selected Function mode to Digital, such that the pattern sequencer controls the specified pin(s).

```
public static void SelectDigital(this DigitalSessionsBundle sessionsBundle, double?  
settlingTimeInSeconds = null)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**settlingTimeInSeconds** [double](#)?

The settling time in seconds. Passing a Null value (default) bypasses the wait operation (No-op).

### Remarks

Remarks:

- If a pattern is being bursted, the pin immediately switches to bursting the pattern.
- The PPMU stops sourcing and turns off when the Digital function is selected. Despite this, you can still make voltage measurements.
- Internally within the instrument the pin electrics are now connected to the driver, comparator, and active load functions.
- The state of the digital pin driver when you change the selected function to Digital is determined by the most recent call to WriteStatic or the last vector of the most recently bursted pattern, whichever is latter.

Example:

```
DigitalSessionsBundle myDutPin = new TSMSessionManager().Digital("MyDutPin");  
myDutPin.SelectDigital();
```

# Method SelectPPMU

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SelectPPMU(DigitalSessionsBundle, double?)

Sets the Selected Function mode to PPMU, such that the PPMU controls the specified pin(s) and connects the PPMU. The pin driver is in a non-drive state, and the active load is disabled.

```
public static void SelectPPMU(this DigitalSessionsBundle sessionsBundle, double?  
settlingTimeInSeconds = null)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**settlingTimeInSeconds** [double](#)?

The settling time in seconds. Passing a Null value (default) bypasses the wait operation (No-op).

### Remarks

Remarks:

- The PPMU does not start sourcing or measuring until ForceVoltage(), ForceCurrent(), MeasureVoltage(), or MeasureCurrent() is called.
- The driver, comparator, and active load are off while this function is selected.
- If you change the Selected Function mode to PPMU using this method, the PPMU is initially not sourcing.
- Note: you can make PPMU voltage measurements calling MeasureVoltage or MeasureCurrent from within any selected function.

Example:

```
DigitalSessionsBundle myDutPin = new TSMSessionManager().Digital("MyDutPin");  
myDutPin.SelectPPMU();
```

# Method TurnOffOutput

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## TurnOffOutput(DigitalSessionsBundle, double?)

Sets the Selected Function mode to Off to put the digital driver into a non-drive state, disables the active load, disconnects the PPMU, and closes the I/O switch connecting the instrument channel.

```
public static void TurnOffOutput(this DigitalSessionsBundle sessionsBundle, double?  
settlingTimeInSeconds = null)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**settlingTimeInSeconds** [double](#)?

The settling time in seconds. Null means immediately turning off operation to settle.

### Remarks

Note that the output channel is still physically connected after calling this method. To physically disconnect the output channel, call [DisconnectOutput\(\)](#) method.

### Example:

```
DigitalSessionsBundle myDutPin = new TSMSessionManager().Digital("MyDutPin");  
myDutPin.TurnOffOutput(settlingTimeInSeconds: 0.1);
```

# Class PPMUSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines settings for PPMU.

```
public class PPMUSettings
```

## Inheritance

[object](#) ← PPMUSettings

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[PPMUSettings\(\)](#)

Default constructor.

## Properties

[ApertureTime](#)

The aperture time. Null value will be ignored.

[CurrentLevel](#)

The current level when forcing current.

[CurrentLevelRange](#)

The current level range when forcing current. Null value will be ignored.

[CurrentLimitRange](#)

The current limit range when forcing voltage.

[OutputFunction](#)

The output function.

## SettlingTime

The settling time. Null value will be ignored.

## VoltageLevel

The voltage level when forcing voltage.

## VoltageLimitHigh

The high voltage limit when forcing current. Null value will be ignored.

## VoltageLimitLow

The low voltage limit when forcing current. Null value will be ignored.

# Constructor PPMUSettings

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## PPMUSettings()

Default constructor.

```
public PPMUSettings()
```

# Property ApertureTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApertureTime

The aperture time. Null value will be ignored.

```
public double? ApertureTime { get; set; }
```

## Property Value

[double](#)?

# Property CurrentLevel

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CurrentLevel

The current level when forcing current.

```
public double CurrentLevel { get; set; }
```

## Property Value

[double](#) ↗

# Property CurrentLevelRange

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CurrentLevelRange

The current level range when forcing current. Null value will be ignored.

```
public double? CurrentLevelRange { get; set; }
```

### Property Value

[double](#)?

# Property CurrentLimitRange

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CurrentLimitRange

The current limit range when forcing voltage.

```
public double? CurrentLimitRange { get; set; }
```

Property Value

[double](#)?

# Property OutputFunction

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## OutputFunction

The output function.

```
public PpmuOutputFunction OutputFunction { get; set; }
```

## Property Value

PpmuOutputFunction

# Property SettlingTime

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SettlingTime

The settling time. Null value will be ignored.

```
public double? SettlingTime { get; set; }
```

## Property Value

[double](#)?

# Property VoltageLevel

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## VoltageLevel

The voltage level when forcing voltage.

```
public double VoltageLevel { get; set; }
```

### Property Value

[double](#) ↗

# Property VoltageLimitHigh

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## VoltageLimitHigh

The high voltage limit when forcing current. Null value will be ignored.

```
public double? VoltageLimitHigh { get; set; }
```

### Property Value

[double](#)?

# Property VoltageLimitLow

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## VoltageLimitLow

The low voltage limit when forcing current. Null value will be ignored.

```
public double? VoltageLimitLow { get; set; }
```

### Property Value

[double](#)?

# Class Pattern

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods to deal with digital patterns.

```
public static class Pattern
```

## Inheritance

[object](#) ← Pattern

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[AbortKeepAlivePattern\(DigitalSessionsBundle\)](#)

Stops the keep alive pattern if it is currently running.

[AbortPattern\(DigitalSessionsBundle\)](#)

Stops bursting the pattern.

[BurstPattern\(DigitalSessionsBundle, string, bool, bool, double\)](#)

Bursts a digital pattern.

[BurstPatternAndPublishResults\(DigitalSessionsBundle, string, bool, double, string\)](#)

Bursts a digital pattern and publishes pass/fail comparison results.

[BurstPatternSynchronized\(DigitalSessionsBundle, string, bool, bool, double\)](#)

Starts a pattern burst on digital pattern instruments that you have previously synchronized using NI-TClk.

[GetFailCount\(DigitalSessionsBundle\)](#)

Gets fail count on a per-pin per-site basis of last burst pattern (long).

[GetSitePassFail\(DigitalSessionsBundle\)](#)

Gets the per-site pass/fail comparison results of last burst pattern (long) as a SiteData object of type Bool.

[WaitUntilDone\(DigitalSessionsBundle, double\)](#)

Waits until the pattern burst is complete. This method is a blocking call, but will timeout after the specified time.

# Method AbortKeepAlivePattern

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## AbortKeepAlivePattern(DigitalSessionsBundle)

Stops the keep alive pattern if it is currently running.

```
public static void AbortKeepAlivePattern(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

# Method AbortPattern

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## AbortPattern(DigitalSessionsBundle)

Stops bursting the pattern.

```
public static void AbortPattern(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

# Method BurstPattern

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## BurstPattern(DigitalSessionsBundle, string, bool, bool, double)

Bursts a digital pattern.

```
public static void BurstPattern(this DigitalSessionsBundle sessionsBundle, string
startLabel, bool selectDigitalFunction = true, bool waitUntilDone = true, double
timeoutInSeconds = 5)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**startLabel** [string](#)

The pattern name or exported pattern label from which to start bursting the pattern.

**selectDigitalFunction** [bool](#)

Whether to set selected function to digital.

**waitUntilDone** [bool](#)

Whether to wait for pattern burst to complete.

**timeoutInSeconds** [double](#)

The maximum time interval allowed for the pattern burst to complete.

# Method BurstPatternAndPublishResults

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## BurstPatternAndPublishResults(DigitalSessionsBundle, string, bool, double, string)

Bursts a digital pattern and publishes pass/fail comparison results.

```
public static bool[][] BurstPatternAndPublishResults(this DigitalSessionsBundle sessionsBundle, string startLabel, bool selectDigitalFunction = true, double timeoutInSeconds = 5, string publishedDataId = "")
```

### Parameters

#### **sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

#### **startLabel** [string](#)

The pattern name or exported pattern label from which to start bursting the pattern.

#### **selectDigitalFunction** [bool](#)

Whether to set selected function to digital.

#### **timeoutInSeconds** [double](#)

The maximum time interval allowed for the pattern burst to complete.

#### **publishedDataId** [string](#)

The unique data id to be used when publishing.

### Returns

#### [bool](#)[][]

The pass/fail comparison results.



# Method BurstPatternSynchronized

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

**BurstPatternSynchronized(DigitalSessionsBundle, string, bool, bool, double)**

Starts a pattern burst on digital pattern instruments that you have previously synchronized using NI-TClk.

```
public static void BurstPatternSynchronized(this DigitalSessionsBundle sessionsBundle,  
    string startLabel, bool selectDigitalFunction = true, bool waitUntilDone = true, double  
    timeoutInSeconds = 5)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**startLabel** [string](#)

The pattern name or exported pattern label from which to start bursting the pattern.

**selectDigitalFunction** [bool](#)

Whether to set selected function to digital.

**waitUntilDone** [bool](#)

Whether to wait for pattern burst to complete.

**timeoutInSeconds** [double](#)

The maximum time interval allowed for the pattern burst to complete.

# Method GetFailCount

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetFailCount(DigitalSessionsBundle)

Gets fail count on a per-pin per-site basis of last burst pattern (long).

```
public static PinSiteData<long> GetFailCount(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

`sessionsBundle` [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### Returns

[PinSiteData<long>](#)

The per-site per-pin fail count.

# Method GetSitePassFail

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## GetSitePassFail(DigitalSessionsBundle)

Gets the per-site pass/fail comparison results of last burst pattern (long) as a SiteData object of type Bool.

```
public static SiteData<bool> GetSitePassFail(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

[sessionsBundle](#) [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### Returns

[SiteData<bool>](#)

The per-site pass/fail comparison results.

# Method WaitUntilDone

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WaitUntilDone(DigitalSessionsBundle, double)

Waits until the pattern burst is complete. This method is a blocking call, but will timeout after the specified time.

```
public static void WaitUntilDone(this DigitalSessionsBundle sessionsBundle, double  
timeoutInSeconds = 10)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**timeoutInSeconds** [double](#) ↗

Timeout in seconds for which to abort this wait operation.

### Exceptions

[ArgumentException](#) ↗

The value for timeoutInSeconds is invalid.

[MaxTimeExceededException](#)

The pattern burst takes longer than the specified timeoutInSeconds.

# Class SequencerFlagsAndRegisters

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods to deal with digital pattern sequencer flags and registers.

```
public static class SequencerFlagsAndRegisters
```

## Inheritance

[object](#) ← SequencerFlagsAndRegisters

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ReadSequencerFlag\(DigitalSessionsBundle, string\)](#)

Reads the Boolean value of a pattern sequencer flag.

[ReadSequencerFlagDistinct\(DigitalSessionsBundle, string\)](#)

Reads the Boolean state of a pattern sequencer flag.

[ReadSequencerRegister\(DigitalSessionsBundle, string\)](#)

Reads the numeric state of a pattern sequencer register.

[ReadSequencerRegisterDistinct\(DigitalSessionsBundle, string\)](#)

Reads the numeric state of a pattern sequencer flag.

[WriteSequencerFlag\(DigitalSessionsBundle, string, bool\)](#)

Writes a Boolean state of a pattern sequencer flag.

[WriteSequencerFlagSynchronized\(DigitalSessionsBundle, string, bool\)](#)

Writes a Boolean state to a specified pattern sequencer flag for synchronized instruments.

[WriteSequencerRegister\(DigitalSessionsBundle, string, int\)](#)

Writes a value to a pattern sequencer register.

# Method ReadSequencerFlag

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadSequencerFlag(DigitalSessionsBundle, string)

Reads the Boolean value of a pattern sequencer flag.

```
public static bool[] ReadSequencerFlag(this DigitalSessionsBundle sessionsBundle,  
string flag)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**flag** [string](#)

The name of the pattern sequencer flag to read. Possible values include "seqflag0", "seqflag1", "seqflag2", or "seqflag3".

### Returns

[bool](#)

An array of the states for the specified pattern sequencer flag, one state value per session.

# Method ReadSequencerFlagDistinct

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadSequencerFlagDistinct(DigitalSessionsBundle, string)

Reads the Boolean state of a pattern sequencer flag.

```
public static bool ReadSequencerFlagDistinct(this DigitalSessionsBundle sessionsBundle,  
    string flag)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**flag** [string](#)

The name of the pattern sequencer flag to read. Possible values include "seqflag0", "seqflag1", "seqflag2", or "seqflag3".

### Returns

[bool](#)

An array of the states for the specified pattern sequencer flag, one state value per session.

### Remarks

This method is the same as [ReadSequencerFlag\(DigitalSessionsBundle, string\)](#), except it also checks to confirm if the flag state is the same across all sessions in the bundle. If the states are the same, it will return the single boolean state value. Otherwise, it will throw an exception.

### Exceptions

[NISemiconductorTestException](#)

The state of the sequence flag is not the same between instrument sessions.



# Method ReadSequencerRegister

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadSequencerRegister(DigitalSessionsBundle, string)

Reads the numeric state of a pattern sequencer register.

```
public static int[] ReadSequencerRegister(this DigitalSessionsBundle sessionsBundle,  
string registerName)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**registerName** [string](#)

Specifies pattern sequencer register to read. Possible values include "reg0" through "reg15".

### Returns

[int](#)[]

An array of the values for the specified pattern sequencer register, one integer value per session.

# Method ReadSequencerRegisterDistinct

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadSequencerRegisterDistinct(DigitalSessionsBundle, string)

Reads the numeric state of a pattern sequencer flag.

```
public static int ReadSequencerRegisterDistinct(this DigitalSessionsBundle sessionsBundle,  
string registerName)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**registerName** [string](#)

Specifies pattern sequencer register to read. Possible values include "reg0" through "reg15".

### Returns

[int](#)

An single int value for the specified pattern sequencer register.

### Remarks

This method is the same as [ReadSequencerRegister\(DigitalSessionsBundle, string\)](#), except it also checks to confirm if the register values are the same across all sessions in the bundle. If the states are indeed the same, it will return the single integer value. Otherwise, it will throw an exception.

### Exceptions

[NI Semiconductor Test Exception](#)

The state of the sequence register is not the same between instrument sessions.

# Method WriteSequencerFlag

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteSequencerFlag(DigitalSessionsBundle, string, bool)

Writes a Boolean state of a pattern sequencer flag.

```
public static void WriteSequencerFlag(this DigitalSessionsBundle sessionsBundle, string
flag, bool value)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**flag** [string](#) ↗

The name of the pattern sequencer flag to read. Possible values include "seqflag0", "seqflag1", "seqflag2", or "seqflag3".

**value** [bool](#) ↗

The value to assign to the specified pattern sequencer flag.

# Method WriteSequencerFlagSynchronized

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteSequencerFlagSynchronized(DigitalSessionsBundle, string, bool)

Writes a Boolean state to a specified pattern sequencer flag for synchronized instruments.

```
public static void WriteSequencerFlagSynchronized(this DigitalSessionsBundle sessionsBundle,  
string flag, bool value)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**flag** [string](#)

The name of the pattern sequencer flag to read. Possible values include "seqflag0", "seqflag1", "seqflag2", or "seqflag3".

**value** [bool](#)

The value to assign to the specified pattern sequencer flag.

# Method WriteSequencerRegister

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteSequencerRegister(DigitalSessionsBundle, string, int)

Writes a value to a pattern sequencer register.

```
public static void WriteSequencerRegister(this DigitalSessionsBundle sessionsBundle, string
register, int value)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**register** [string](#)

Specifies the sequencer register to which you would like to write the specified value. Possible values include "reg0" through "reg15".

**value** [int](#)

The value to write to the specified pattern sequence register.

# Class SourceAndCapture

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for sourcing and capturing waveforms.

```
public static class SourceAndCapture
```

## Inheritance

[object](#) ← SourceAndCapture

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[CreateParallelCaptureWaveform\(DigitalSessionsBundle, string\)](#)

Creates the capture waveform settings for parallel acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[CreateParallelCaptureWaveform\(DigitalSessionsBundle, string, string\)](#)

Creates the capture waveform settings for parallel acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[CreateParallelCaptureWaveform\(DigitalSessionsBundle, string\[\], string\)](#)

Creates the capture waveform settings for parallel acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[CreateParallelSourceWaveform\(DigitalSessionsBundle, string\[\], string, SourceDataMapping\)](#)

Creates source waveform settings required for serial sourcing. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[CreateSerialCaptureWaveform\(DigitalSessionsBundle, string, string, uint, BitOrder\)](#)

Creates capture waveform settings for serial acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[CreateSerialCaptureWaveform\(DigitalSessionsBundle, string, uint, BitOrder\)](#)

Creates capture waveform settings for serial acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[CreateSerialSourceWaveform\(DigitalSessionsBundle, string, SourceDataMapping, uint, BitOrder\)](#)

Creates source waveform settings required for serial sourcing. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[CreateSerialSourceWaveform\(DigitalSessionsBundle, string, string, SourceDataMapping, uint, BitOrder\)](#)

Creates source waveform settings required for serial sourcing. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

[FetchCaptureWaveform\(DigitalSessionsBundle, string, int, double\)](#)

Fetches the capture waveform and returns a pin- and site-aware object of uint values.

[WriteSourceWaveformBroadcast\(DigitalSessionsBundle, string, uint\[\], bool, int\)](#)

Writes source waveform. Use this method to write the same waveform data to all sites.

[WriteSourceWaveformSiteUnique\(DigitalSessionsBundle, string, SiteData<uint\[\]>, bool, int\)](#)

Writes source waveform. Use this method to write different waveform data to different sites.

# Method CreateParallelCaptureWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CreateParallelCaptureWaveform(DigitalSessionsBundle, string)

Creates the capture waveform settings for parallel acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateParallelCaptureWaveform(this DigitalSessionsBundle sessionsBundle,  
    string waveformName)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**waveformName** [string](#) ↗

The name of the capture waveform.

## CreateParallelCaptureWaveform(DigitalSessionsBundle, string, string)

Creates the capture waveform settings for parallel acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateParallelCaptureWaveform(this DigitalSessionsBundle sessionsBundle,  
    string pin, string waveformName)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**pin** [string](#)

The pin for which to create the capture waveform.

**waveformName** [string](#)

The name of the capture waveform.

## CreateParallelCaptureWaveform(DigitalSessionsBundle, string[], string)

Creates the capture waveform settings for parallel acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateParallelCaptureWaveform(this DigitalSessionsBundle sessionsBundle,  
string[] pins, string waveformName)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**pins** [string](#)[]

The pins for which to create the capture waveform.

**waveformName** [string](#)

The name of the capture waveform.

# Method CreateParallelSourceWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CreateParallelSourceWaveform(DigitalSessionsBundle, string[], string, SourceDataMapping)

Creates source waveform settings required for serial sourcing. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateParallelSourceWaveform(this DigitalSessionsBundle sessionsBundle,  
    string[] pins, string waveformName, SourceDataMapping sourceDataMapping)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**pins** [string](#)[]

The pins for which to source the source waveform.

**waveformName** [string](#)

The name of the source waveform.

**sourceDataMapping** [SourceDataMapping](#)

The source data mapping: [NationalInstruments.ModularInstruments.NIDigital.SourceDataMapping.Broadcast](#) or [NationalInstruments.ModularInstruments.NIDigital.SourceDataMapping.SiteUnique](#)

### Exceptions

[IviCDriverException](#)

The NI-Digital Pattern Driver returned an error.

[SelectorNameException](#)

The pinSet contains a pin or pin group name not loaded in the pin map.

#### [InvalidOperationException](#)

The pinSet contains a system pin

#### [ArgumentException](#)

The value for waveformName is an empty string or contains an invalid character.

#### [OutOfRangeException](#)

The number of waveforms in capture memory exceeds the maximum number of waveforms allowed.

# Method CreateSerialCaptureWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CreateSerialCaptureWaveform(DigitalSessionsBundle, string, uint, BitOrder)

Creates capture waveform settings for serial acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateSerialCaptureWaveform(this DigitalSessionsBundle sessionsBundle,  
    string waveformName, uint sampleWidth, BitOrder bitOrder = BitOrder.MostSignificantBitFirst)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**waveformName** [string](#)

The name of the capture waveform.

**sampleWidth** [uint](#)

The width in bits of each serial sample. The value must be between 1 and 32.

**bitOrder** [BitOrder](#)

The bit order significance.

### Remarks

The number of waveforms is limited to 512.

## CreateSerialCaptureWaveform(DigitalSessionsBundle, string, string, uint, BitOrder)

Creates capture waveform settings for serial acquisition. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateSerialCaptureWaveform(this DigitalSessionsBundle sessionsBundle,  
    string pin, string waveformName, uint sampleWidth, BitOrder bitOrder =  
    BitOrder.MostSignificantBitFirst)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**pin** [string](#)

The pin for which to create the capture waveform.

**waveformName** [string](#)

The name of the capture waveform.

**sampleWidth** [uint](#)

The width in bits of each serial sample. The value must be between 1 and 32.

**bitOrder** [BitOrder](#)

The bit order significance.

## Remarks

The number of waveforms is limited to 512.

# Method CreateSerialSourceWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## CreateSerialSourceWaveform(DigitalSessionsBundle, string, SourceDataMapping, uint, BitOrder)

Creates source waveform settings required for serial sourcing. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateSerialSourceWaveform(this DigitalSessionsBundle sessionsBundle,  
    string waveformName, SourceDataMapping dataMapping, uint sampleWidth, BitOrder bitOrder)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**waveformName** [string](#)

The name of the source waveform.

**dataMapping** [SourceDataMapping](#)

Specifies whether the waveform is broadcasted to all sites or a unique waveform is sourced per site.

**sampleWidth** [uint](#)

The width in bits of each serial sample. The value must be between 1 and 32.

**bitOrder** [BitOrder](#)

The bit order significance.

## CreateSerialSourceWaveform(DigitalSessionsBundle, string, SourceDataMapping, uint, BitOrder)

Creates source waveform settings required for serial sourcing. Settings apply across all sites if multiple sites are configured in the pin map. You cannot reconfigure settings after waveforms are created.

```
public static void CreateSerialSourceWaveform(this DigitalSessionsBundle sessionsBundle,  
    string pin, string waveformName, SourceDataMapping dataMapping, uint sampleWidth,  
    BitOrder bitOrder)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**pin** [string](#)

The pin for which to create the source waveform.

**waveformName** [string](#)

The name of the source waveform.

**dataMapping** [SourceDataMapping](#)

Specifies whether the waveform is broadcasted to all sites or a unique waveform is sourced per site.

**sampleWidth** [uint](#)

The width in bits of each serial sample. The value must be between 1 and 32.

**bitOrder** [BitOrder](#)

The bit order significance.

# Method FetchCaptureWaveform

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## FetchCaptureWaveform(DigitalSessionsBundle, string, int, double)

Fetches the capture waveform and returns a pin- and site-aware object of uint values.

```
public static SiteData<uint[]> FetchCaptureWaveform(this DigitalSessionsBundle sessionsBundle, string waveformName, int samplesToRead, double timeoutInSeconds = 5)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**waveformName** [string](#)

The name of the capture waveform.

**samplesToRead** [int](#)

The number of samples to read.

**timeoutInSeconds** [double](#)

The maximum time to read for waveform samples.

### Returns

[SiteData<uint\[\]>](#)

The captured data.

# Method WriteSourceWaveformBroadcast

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteSourceWaveformBroadcast(DigitalSessionsBundle, string, uint[], bool, int)

Writes source waveform. Use this method to write the same waveform data to all sites.

```
public static void WriteSourceWaveformBroadcast(this DigitalSessionsBundle sessionsBundle,
string waveformName, uint[] waveformData, bool expandToMinimumSize = false, int minimumSize
= 128)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**waveformName** [string](#)

The name of the source waveform.

**waveformData** [uint](#)[]

The waveform data.

**expandToMinimumSize** [bool](#)

Whether to make the size of the waveform data at least a specified value.

**minimumSize** [int](#)

The minimum size of the waveform data.

# Method WriteSourceWaveformSiteUnique

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteSourceWaveformSiteUnique(DigitalSessionsBundle, string, SiteData<uint[]>, bool, int)

Writes source waveform. Use this method to write different waveform data to different sites.

```
public static void WriteSourceWaveformSiteUnique(this DigitalSessionsBundle sessionsBundle,
    string waveformName, SiteData<uint[]> perSiteWaveformData, bool expandToMinimumSize = false,
    int minimumSize = 128)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**waveformName** [string](#)

The name of the source waveform.

**perSiteWaveformData** [SiteData<uint\[\]>](#)

The per-site waveform data.

**expandToMinimumSize** [bool](#)

Whether to make the size of the waveform data at least a specified value.

**minimumSize** [int](#)

The minimum size of the waveform data.

### Remarks

This method takes per-site source waveform.

# Class StaticState

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for static state read/write.

```
public static class StaticState
```

## Inheritance

[object](#) ← StaticState

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ReadStatic\(DigitalSessionsBundle\)](#)

Reads the current state to target the pin(s) and returns as pin- and site-aware data object.

[WriteStatic\(DigitalSessionsBundle, PinState\)](#)

Writes static state to target the pin(s) that will take effect immediately.

[WriteStatic\(DigitalSessionsBundle, PinSiteData<PinState>\)](#)

Writes static state to target the pin(s) that will take effect immediately.

[WriteStatic\(DigitalSessionsBundle, SiteData<PinState>\)](#)

Writes static state to target the pin(s) that will take effect immediately.

# Method ReadStatic

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ReadStatic(DigitalSessionsBundle)

Reads the current state to target the pin(s) and returns as pin- and site-aware data object.

```
public static PinSiteData<PinState> ReadStatic(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

`sessionsBundle` [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

### Returns

[PinSiteData](#)<PinState>

The states in per-site per-pin format.

# Method WriteStatic

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## WriteStatic(DigitalSessionsBundle, PinState)

Writes static state to target the pin(s) that will take effect immediately.

```
public static void WriteStatic(this DigitalSessionsBundle sessionsBundle, PinState state)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**state** PinState

The state to write.

### Remarks

Use this method to write the same state on all sessions.

## WriteStatic(DigitalSessionsBundle, SiteData<PinState>)

Writes static state to target the pin(s) that will take effect immediately.

```
public static void WriteStatic(this DigitalSessionsBundle sessionsBundle,  
SiteData<PinState> states)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**states** [SiteData<PinState>](#)

The states to write for all sites.

## Remarks

Use this method to write different states on different sites.

## WriteStatic(DigitalSessionsBundle, PinSiteData<PinState>)

Writes static state to target the pin(s) that will take effect immediately.

```
public static void WriteStatic(this DigitalSessionsBundle sessionsBundle,  
PinSiteData<PinState> states)
```

## Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**states** [PinSiteData<PinState>](#)

The states to write for all site-pin pairs.

## Remarks

Use this method to write different states for different site-pin pairs.

# Class TriggersAndEvents

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods to deal with digital pattern triggers and events.

```
public static class TriggersAndEvents
```

## Inheritance

[object](#) ← TriggersAndEvents

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ConfigureConditionalJumpTriggerDigitalEdge\(DigitalSessionsBundle, string, int, DigitalEdge\)](#)

Configures a digital edge trigger for the specified Conditional Jump Trigger.

[ConfigureConditionalJumpTriggerSoftwareEdge\(DigitalSessionsBundle, int\)](#)

Configures a software edge trigger for the specified Conditional Jump Trigger.

[ConfigureStartTriggerDigitalEdge\(DigitalSessionsBundle, string, DigitalEdge\)](#)

Configures a digital edge trigger for the Start Trigger.

[ConfigureStartTriggerSoftwareEdge\(DigitalSessionsBundle\)](#)

Configures a software edge trigger for the Start Trigger.

[DisableConditionalJumpTrigger\(DigitalSessionsBundle, int\)](#)

Disables the specified ConditionalJumpTrigger by configuring it to None.

[DisableConditionalJumpTriggers\(DigitalSessionsBundle\)](#)

Disables all ConditionalJumpTriggers by configuring them to None.

[DisableStartTrigger\(DigitalSessionsBundle\)](#)

Disables StartTrigger trigger by configuring it to None.

[ExportSignal\(DigitalSessionsBundle, SignalType, string, string\)](#)

Routes trigger and event signals to the specified outputTerminal.

[SendSoftwareEdgeConditionalJumpTrigger\(DigitalSessionsBundle, int\)](#)

Sends the Software Conditional Jump Trigger to a digital pattern instrument, forcing the Conditional Jump Trigger to assert, regardless of how the Conditional Jump Trigger is configured.

[SendSoftwareEdgeStartTrigger\(DigitalSessionsBundle\)](#)

Sends the software start trigger to a digital pattern instrument, forcing the start trigger to assert, regardless of how the start trigger is configured.

# Method

## ConfigureConditionalJumpTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### ConfigureConditionalJumpTriggerDigitalEdge(DigitalSessionsBundle, string, int, DigitalEdge)

Configures a digital edge trigger for the specified Conditional Jump Trigger.

```
public static void ConfigureConditionalJumpTriggerDigitalEdge(this DigitalSessionsBundle sessionsBundle, string source, int conditionalJumpTriggerId, DigitalEdge digitalEdge = DigitalEdge.Rising)
```

#### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**source** [string](#)

The source terminal for the start trigger. Possible values include but are not limited to "PXI\_Trig0", "PXI\_Trig1", "PXI\_Trig2", "PXI\_Trig3", "PXI\_Trig4", "PXI\_Trig5", "PXI\_Trig6", or "PXI\_Trig7".

**conditionalJumpTriggerId** [int](#)

The ID of the of ConditionalJumpTrigger to configure. Valid values include: 0, 1, 2, and 3, which all equate to conditionalJumpTrigger0, conditionalJumpTrigger1, conditionalJumpTrigger2, and conditionalJumpTrigger3, respectively.

**digitalEdge** [DigitalEdge](#)

The edge of the digital signal that should be triggered on: Rising edge (default) or Falling edge.

#### Exceptions

[OutOfRangeException](#)

The value for type is invalid.

# Method ConfigureConditionalJumpTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureConditionalJumpTriggerSoftwareEdge(DigitalSessionsBundle, int)

Configures a software edge trigger for the specified Conditional Jump Trigger.

```
public static void ConfigureConditionalJumpTriggerSoftwareEdge(this DigitalSessionsBundle sessionsBundle, int conditionalJumpTriggerId)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**conditionalJumpTriggerId** [int](#)

The ID of the ConditionalJumpTrigger to configure. Valid values include: 0, 1, 2, and 3, which all equate to conditionalJumpTrigger0, conditionalJumpTrigger1, conditionalJumpTrigger2, and conditionalJumpTrigger3, respectively.

# Method ConfigureStartTriggerDigitalEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureStartTriggerDigitalEdge(DigitalSessionsBundle, string, DigitalEdge)

Configures a digital edge trigger for the Start Trigger.

```
public static void ConfigureStartTriggerDigitalEdge(this DigitalSessionsBundle sessionsBundle, string source, DigitalEdge digitalEdge = DigitalEdge.Rising)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**source** [string](#)

The source terminal for the start trigger. Possible values include but are not limited to "PXI\_Trig0", "PXI\_Trig1", "PXI\_Trig2", "PXI\_Trig3", "PXI\_Trig4", "PXI\_Trig5", "PXI\_Trig6", or "PXI\_Trig7".

**digitalEdge** [DigitalEdge](#)

The edge of the digital signal that should be triggered on: Rising edge (default) or Falling edge.

### Exceptions

[OutOfRangeException](#)

The value for type is invalid.

# Method ConfigureStartTriggerSoftwareEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ConfigureStartTriggerSoftwareEdge(DigitalSessionsBundle)

Configures a software edge trigger for the Start Trigger.

```
public static void ConfigureStartTriggerSoftwareEdge(this DigitalSessionsBundle  
sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

# Method DisableConditionalJumpTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DisableConditionalJumpTrigger(DigitalSessionsBundle, int)

Disables the specified ConditionalJumpTrigger by configuring it to None.

```
public static void DisableConditionalJumpTrigger(this DigitalSessionsBundle sessionsBundle,  
int conditionalJumpTriggerId)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**conditionalJumpTriggerId** [int](#)

The ID of the of ConditionalJumpTrigger to configure. Valid values include: 0, 1, 2, and 3, which all equate to conditionalJumpTrigger0, conditionalJumpTrigger1, conditionalJumpTrigger2, and conditionalJumpTrigger3, respectively.

# Method DisableConditionalJumpTriggers

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DisableConditionalJumpTriggers(DigitalSessionsBundle)

Disables all ConditionalJumpTriggers by configuring them to None.

```
public static void DisableConditionalJumpTriggers(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

[sessionsBundle](#) [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

# Method DisableStartTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## DisableStartTrigger(DigitalSessionsBundle)

Disables StartTrigger trigger by configuring it to None.

```
public static void DisableStartTrigger(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

# Method ExportSignal

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ExportSignal(DigitalSessionsBundle, SignalType, string, string)

Routes trigger and event signals to the specified outputTerminal.

```
public static void ExportSignal(this DigitalSessionsBundle sessionsBundle, SignalType  
signalType, string signalID, string outputTerminal)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**signalType** SignalType

The type of signal to export.

**signalID** string ↗

The instance of the selected signal to export. Possible values include "patternOpcodeEvent0", "patternOpcodeEvent1", "patternOpcodeEvent2", or "patternOpcodeEvent3".

**outputTerminal** string ↗

The terminal to which to export the signal. Possible values include but are not limited to "PXI\_Trig0", "PXI\_Trig1", "PXI\_Trig2", "PXI\_Trig3", "PXI\_Trig4", "PXI\_Trig5", "PXI\_Trig6", or "PXI\_Trig7".

### Exceptions

[ArgumentException](#) ↗

The value for one or all of the parameters, signalType, signalID, or outputTerminal, is invalid.

[IviCDriverException](#)

The underlying NI-Digital Pattern Driver returned an error.



# Method

## SendSoftwareEdgeConditionalJumpTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

### SendSoftwareEdgeConditionalJumpTrigger(DigitalSessionsBundle, int)

Sends the Software Conditional Jump Trigger to a digital pattern instrument, forcing the Conditional Jump Trigger to assert, regardless of how the Conditional Jump Trigger is configured.

```
public static void SendSoftwareEdgeConditionalJumpTrigger(this DigitalSessionsBundle sessionsBundle, int conditionalJumpTriggerId)
```

#### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

**conditionalJumpTriggerId** [int](#)

The ID of the of ConditionalJumpTrigger to configure. Valid values include: 0, 1, 2, and 3, which all equate to conditionalJumpTrigger0, conditionalJumpTrigger1, conditionalJumpTrigger2, and conditionalJumpTrigger3, respectively.

# Method SendSoftwareEdgeStartTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Digital](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## SendSoftwareEdgeStartTrigger(DigitalSessionsBundle)

Sends the software start trigger to a digital pattern instrument, forcing the start trigger to assert, regardless of how the start trigger is configured.

```
public static void SendSoftwareEdgeStartTrigger(this DigitalSessionsBundle sessionsBundle)
```

### Parameters

**sessionsBundle** [DigitalSessionsBundle](#)

The [DigitalSessionsBundle](#) object.

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen

## Classes

### [FgenSessionInformation](#)

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIFgen.NIFgen session and its related information.

### [FgenSessionsBundle](#)

Defines an object that contains one or more [FgenSessionInformation](#) objects.

### [InitializeAndClose](#)

Defines NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions initialize and close APIs.

# Class FgenSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIFgen.NIFgen session and its related information.

```
public class FgenSessionInformation : ISessionInformation
```

## Inheritance

[object](#) ← FgenSessionInformation

## Implements

[ISessionInformation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[FgenSessionInformation\(NIFgen, IList<SitePinInfo>\)](#)

Constructs a session information object.

[FgenSessionInformation\(NIFgen, string\)](#)

Constructs a session information object that associates with a NationalInstruments.Modular Instruments.NIFgen.NIFgen instrument session.

## Properties

[AllChannelsString](#)

The all channels string associated with the [Session](#).

[AssociatedSitePinList](#)

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

## ModelString

The NI-FGEN instrument model string.

## Session

The NationalInstruments.ModularInstruments.NIFgen.NIFgen session.

# Methods

## [GenerateInstrumentChannelToSitePinDictionary\(I SemiconductorModuleContext\)](#)

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

# Constructor FgenSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FgenSessionInformation(NIFgen, string)

Constructs a session information object that associates with a NationalInstruments.ModularInstruments.NIFgen.NIFgen instrument session.

```
public FgenSessionInformation(NIFgen session, string channelList)
```

### Parameters

**session** NIFgen

The NationalInstruments.ModularInstruments.NIFgen.NIFgen session.

**channelList** [string](#)

The channels list associated with the NationalInstruments.ModularInstruments.NIFgen.NIFgen session.

## FgenSessionInformation(NIFgen, IList<SitePinInfo>)

Constructs a session information object.

```
public FgenSessionInformation(NIFgen session, IList<SitePinInfo> associatedSitePinList)
```

### Parameters

**session** NIFgen

The NationalInstruments.ModularInstruments.NIFgen.NIFgen session.

**associatedSitePinList** [IList](#)<[SitePinInfo](#)>

The specified subset of channels associated with the session.

# Property AllChannelsString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AllChannelsString

The all channels string associated with the [Session](#).

```
public string AllChannelsString { get; }
```

## Property Value

[string](#) ↗

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

```
public IList<SitePinInfo> AssociatedSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property ModelString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ModelString

The NI-FGEN instrument model string.

```
public string ModelString { get; }
```

### Property Value

[string](#) ↗

# Property Session

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Session

The NationalInstruments.ModularInstruments.NIFgen.NIFgen session.

```
public NIFgen Session { get; }
```

## Property Value

NIFgen

# Method GenerateInstrumentChannelToSitePinDictionary

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext )

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

```
public static void GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext  
tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Class FgenSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains one or more [FgenSessionInformation](#) objects.

```
public class FgenSessionsBundle : ISessionsBundle<FgenSessionInformation>
```

## Inheritance

[object](#) ← FgenSessionsBundle

## Implements

[ISessionsBundle<FgenSessionInformation>](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>>, string, string\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, int, TResult\[\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[,\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult\[\]>\)](#),  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, SitePinInfo>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int, SitePinInfo>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation, int>\)](#),  
[ParallelExecution.Do<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Action<TSessionInformation>\)](#),  
[Publish.PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\], string\)](#),  
[Publish.PublishResults<TSessionInformation, TData>\(ISessionsBundle<TSessionInformation>, TData\[\]\[\], string\)](#).

## Constructors

[EgenSessionsBundle\(ISemiconductorModuleContext, IEnumerable<FgenSessionInformation>\)](#)

Constructs a sessions bundle object that represents a bunch of [FgenSessionInformation](#)s.

## Properties

[AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

[InstrumentSessions](#)

An enumerable of [ISessionInformation](#).

[TSMContext](#)

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

# Methods

## [FilterByPin\(string\)](#)

Filters current [EgenSessionsBundle](#) and returns a new one with the requested pin.

## [FilterByPin\(string\[\]\)](#)

Filters current [EgenSessionsBundle](#) and returns a new one with requested pins.

## [FilterBySite\(int\)](#)

Filters current [EgenSessionsBundle](#) and returns a new one with the requested site.

## [FilterBySite\(int\[\]\)](#)

Filters current [EgenSessionsBundle](#) and returns a new one with requested sites.

# Constructor FgenSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FgenSessionsBundle(ISemiconductorModuleContext, IEnumerable<FgenSessionInformation>)

Constructs a sessions bundle object that represents a bunch of [FgenSessionInformations](#).

```
public FgenSessionsBundle(ISemiconductorModuleContext tsmContext,  
IEnumerable<FgenSessionInformation> allSessionInfo)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

**allSessionInfo** [IEnumerable](#)<[FgenSessionInformation](#)>

An enumerable of [FgenSessionInformations](#).

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

```
public IList<SitePinInfo> AggregateSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [ISessionInformation](#).

```
public IEnumerable<FgenSessionInformation> InstrumentSessions { get; }
```

## Property Value

[IEnumerable](#) <[FgenSessionInformation](#)>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public ISemiconductorModuleContext TSMContext { get; }
```

## Property Value

ISemiconductorModuleContext

# Method FilterByPin

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterByPin(string)

Filters current [FgenSessionsBundle](#) and returns a new one with the requested pin.

```
public FgenSessionsBundle FilterByPin(string requestedPin)
```

### Parameters

**requestedPin** [string](#)

The requested pin.

### Returns

[FgenSessionsBundle](#)

A new [FgenSessionsBundle](#) object with the requested pin.

## FilterByPin(string[])

Filters current [FgenSessionsBundle](#) and returns a new one with requested pins.

```
public FgenSessionsBundle FilterByPin(string[] requestedPins)
```

### Parameters

**requestedPins** [string](#)[]

The requested pins.

### Returns

## [EgenSessionsBundle](#)

A new [EgenSessionsBundle](#) object with requested pins.

# Method FilterBySite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterBySite(int)

Filters current [FgenSessionsBundle](#) and returns a new one with the requested site.

```
public FgenSessionsBundle FilterBySite(int requestedSite)
```

### Parameters

requestedSite [int](#)

The requested site.

### Returns

[FgenSessionsBundle](#)

A new [FgenSessionsBundle](#) object with the requested site.

## FilterBySite(int[])

Filters current [FgenSessionsBundle](#) and returns a new one with requested sites.

```
public FgenSessionsBundle FilterBySite(int[] requestedSites)
```

### Parameters

requestedSites [int](#)[]

The requested sites.

### Returns

## [EgenSessionsBundle](#)

A new [EgenSessionsBundle](#) object with requested sites.

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[Close\(ISemiconductorModuleContext, bool\)](#)

Closes all NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions.

[Initialize\(ISemiconductorModuleContext, bool\)](#)

Initializes NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions in the test system.

[Reset\(ISemiconductorModuleContext, bool\)](#)

Resets all NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions.

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close(ISemiconductorModuleContext, bool)

Closes all NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions.

```
public static void Close(ISemiconductorModuleContext tsmContext, bool reset = true)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**reset** [bool](#)

Whether to reset the device sessions before closing.

# Method Initialize

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Initialize(ISemiconductorModuleContext, bool)

Initializes NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions in the test system.

```
public static void Initialize(ISemiconductorModuleContext tsmContext, bool resetDevice  
= false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset device during initialization.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Fgen](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset(ISemiconductorModuleContext, bool)

Resets all NationalInstruments.ModularInstruments.NIFgen.NIFgen sessions.

```
public static void Reset(ISemiconductorModuleContext tsmContext, bool resetDevice = false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to perform a hard reset on the device.

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay

## Classes

### [Control](#)

Defines methods for relay control.

### [InitializeAndClose](#)

Defines NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions initialize and close APIs.

# Class Control

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

Defines methods for relay control.

```
public static class Control
```

## Inheritance

[object](#) ← Control

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[ApplyRelayConfiguration\(ISemiconductorModuleContext, string\)](#)

Performs the relay actions on the relays in the relay configuration.

[ControlRelay\(ISemiconductorModuleContext, IDictionary<string, RelayDriverAction>\)](#)

Performs the relay actions on the relays.

[ControlRelay\(ISemiconductorModuleContext, string, RelayDriverAction\)](#)

Performs the relay action on the relay.

[ControlRelay\(ISemiconductorModuleContext, string\[\], SiteData<RelayDriverAction>\)](#)

Performs the relay actions on the relays.

[ControlRelay\(ISemiconductorModuleContext, string\[\], RelayDriverAction\)](#)

Performs the same relay action on multiple relays.

[ControlRelay\(ISemiconductorModuleContext, string\[\], RelayDriverAction\[\]\)](#)

Performs the relay actions on the relays.

# Method ApplyRelayConfiguration

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ApplyRelayConfiguration(ISemiconductorModuleContext, string)

Performs the relay actions on the relays in the relay configuration.

```
public static void ApplyRelayConfiguration(ISemiconductorModuleContext tsmContext,  
    string relayConfiguration)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**relayConfiguration** [string](#)

The name of the relay configuration to apply.

# Method ControlRelay

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Extensions.dll

## ControlRelay(ISemiconductorModuleContext, string, RelayDriverAction)

Performs the relay action on the relay.

```
public static void ControlRelay(ISemiconductorModuleContext tsmContext, string relay,  
RelayDriverAction relayAction)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**relay** [string](#)

The name of the relay to control.

**relayAction** RelayDriverAction

The relay action to perform.

## ControlRelay(ISemiconductorModuleContext, string[], RelayDriverAction)

Performs the same relay action on multiple relays.

```
public static void ControlRelay(ISemiconductorModuleContext tsmContext, string[] relays,  
RelayDriverAction relayAction)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**relays** [string](#)[]

The name of the relays to control.

**relayAction** RelayDriverAction

The relay action to perform.

## ControlRelay(ISemiconductorModuleContext, string[], RelayDriverAction[])

Performs the relay actions on the relays.

```
public static void ControlRelay(ISemiconductorModuleContext tsmContext, string[] relays,  
RelayDriverAction[] relayActions)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**relays** [string](#)[]

The name of the relays to control.

**relayActions** RelayDriverAction[]

The relay actions to perform.

## ControlRelay(ISemiconductorModuleContext, IDictionary<string, RelayDriverAction>)

Performs the relay actions on the relays.

```
public static void ControlRelay(ISemiconductorModuleContext tsmContext, IDictionary<string,  
RelayDriverAction> relayNameToActionDictionary)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**relayNameToActionDictionary** [IDictionary](#)<[string](#), RelayDriverAction>

A dictionary that maps the relay name to the relay action to apply on that relay.

## ControlRelay(ISemiconductorModuleContext, string[], SiteData<RelayDriverAction>)

Performs the relay actions on the relays.

```
public static void ControlRelay(ISemiconductorModuleContext tsmContext, string[] relays,  
SiteData<RelayDriverAction> perSiteRelayActions)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**relays** [string](#)[]

The name of the relays to control.

**perSiteRelayActions** [SiteData](#)<RelayDriverAction>

The per-site relay actions to perform.

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[Close\(ISemiconductorModuleContext, bool, bool\)](#)

Closes all NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions.

[Initialize\(ISemiconductorModuleContext, bool\)](#)

Initializes NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions in the test system.

[Reset\(ISemiconductorModuleContext\)](#)

Resets all NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions.

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close(ISemiconductorModuleContext, bool, bool)

Closes all NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions.

```
public static void Close(ISemiconductorModuleContext tsmContext, bool resetDevice = true,  
bool disable = true)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset the device sessions before closing.

**disable** [bool](#)

Whether to disable the device sessions before closing.

# Method Initialize

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Initialize(ISemiconductorModuleContext, bool)

Initializes NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions in the test system.

```
public static void Initialize(ISemiconductorModuleContext tsmContext, bool resetDevice  
= false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset device during initialization.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Relay](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset(ISemiconductorModuleContext)

Resets all NationalInstruments.ModularInstruments.NISwitch.NISwitch sessions.

```
public static void Reset(ISemiconductorModuleContext tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope

## Classes

### [InitializeAndClose](#)

Defines NationalInstruments.ModularInstruments.NIScope.NIScope sessions initialize and close APIs.

### [ScopeSessionInformation](#)

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIScope.NIScope session and its related information.

### [ScopeSessionsBundle](#)

Defines an object that contains one or more [ScopeSessionInformation](#) objects.

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines NationalInstruments.ModularInstruments.NIScope.NIScope sessions initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

[Close\(ISemiconductorModuleContext, bool\)](#)

Closes all NationalInstruments.ModularInstruments.NIScope.NIScope sessions.

[Initialize\(ISemiconductorModuleContext, bool\)](#)

Initializes NationalInstruments.ModularInstruments.NIScope.NIScope sessions in the test system.

[Reset\(ISemiconductorModuleContext, bool\)](#)

Resets all NationalInstruments.ModularInstruments.NIScope.NIScope sessions.

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close(ISemiconductorModuleContext, bool)

Closes all NationalInstruments.ModularInstruments.NIScope.NIScope sessions.

```
public static void Close(ISemiconductorModuleContext tsmContext, bool reset = true)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**reset** [bool](#)

Whether to reset the device sessions before closing.

# Method Initialize

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Initialize(ISemiconductorModuleContext, bool)

Initializes NationalInstruments.ModularInstruments.NIScope.NIScope sessions in the test system.

```
public static void Initialize(ISemiconductorModuleContext tsmContext, bool resetDevice  
= false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset device during initialization.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset(ISemiconductorModuleContext, bool)

Resets all NationalInstruments.ModularInstruments.NIScope.NIScope sessions.

```
public static void Reset(ISemiconductorModuleContext tsmContext, bool resetDevice = false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context object.

**resetDevice** [bool](#)

Whether to perform a hard reset on the device.

# Class ScopeSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains an individual NationalInstruments.ModularInstruments.NIScope.NIScope session and its related information.

```
public class ScopeSessionInformation : ISessionInformation
```

## Inheritance

[object](#) ← ScopeSessionInformation

## Implements

[ISessionInformation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[ScopeSessionInformation\(NIScope, IList<SitePinInfo>\)](#).

Constructs a session information object.

[ScopeSessionInformation\(NIScope, string\)](#).

Constructs a session information object that associates with a NationalInstruments.Modular Instruments.NIScope.NIScope instrument session.

## Properties

[AllChannelsString](#)

The all channels string associated with the [Session](#).

[AssociatedSitePinList](#)

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

## ModelString

The NI-Scope instrument model string.

## Session

The NationalInstruments.ModularInstruments.NIScope.NIScope session.

# Methods

## [GenerateInstrumentChannelToSitePinDictionary\(I SemiconductorModuleContext\)](#)

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

# Constructor ScopeSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ScopeSessionInformation(NIScope, string)

Constructs a session information object that associates with a NationalInstruments.ModularInstruments.NIScope.NIScope instrument session.

```
public ScopeSessionInformation(NIScope session, string channelList)
```

### Parameters

**session** NIScope

The NationalInstruments.ModularInstruments.NIScope.NIScope session.

**channelList** [string](#)

The channels list associated with the NationalInstruments.ModularInstruments.NIScope.NIScope session.

## ScopeSessionInformation(NIScope, IList<SitePinInfo>)

Constructs a session information object.

```
public ScopeSessionInformation(NIScope session, IList<SitePinInfo> associatedSitePinList)
```

### Parameters

**session** NIScope

The NationalInstruments.ModularInstruments.NIScope.NIScope session.

**associatedSitePinList** [IList](#)<[SitePinInfo](#)>

The specified subset of channels associated with the session.



# Property AllChannelsString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AllChannelsString

The all channels string associated with the [Session](#).

```
public string AllChannelsString { get; }
```

### Property Value

[string](#) ↗

### Remarks

It's a comma-separated string of channel numbers (when the associated [Session](#) is a single instrument session) or instrument-channel pairs (when the associated [Session](#) is a multi-instrument session. For example, "0, 1", "Scope\_5622\_C1\_S10/0, Scope\_5622\_C1\_S11/1", etc.

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

```
public IList<SitePinInfo> AssociatedSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property ModelString

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ModelString

The NI-Scope instrument model string.

```
public string ModelString { get; }
```

### Property Value

[string](#) ↗

# Property Session

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Session

The NationalInstruments.ModularInstruments.NIScope.NIScope session.

```
public NIScope Session { get; }
```

## Property Value

NIScope

# Method GenerateInstrumentChannelToSitePinDictionary

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext )

Generates a dictionary that takes a channel string and returns the associated site-pin pair information.

```
public static void GenerateInstrumentChannelToSitePinDictionary( ISemiconductorModuleContext  
tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Class ScopeSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains one or more [ScopeSessionInformation](#) objects.

```
public class ScopeSessionsBundle : ISessionsBundle<ScopeSessionInformation>
```

## Inheritance

[object](#) ← ScopeSessionsBundle

## Implements

[ISessionsBundle<ScopeSessionInformation>](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\], string>\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>, string, string\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, int, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[,\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation>](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]](#),  
[string](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]\[\]](#),  
[string](#)).

## Constructors

[ScopeSessionsBundle\(ISemiconductorModuleContext, IEnumerable<ScopeSessionInformation>\)](#)

Constructs a sessions bundle object that represents a bunch of [ScopeSessionInformations](#).

## Properties

[AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

[InstrumentSessions](#)

An enumerable of [ISessionInformation](#).

[TSMContext](#)

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

# Methods

## [FilterByPin\(string\)](#)

Filters current [ScopeSessionsBundle](#) and returns a new one with the requested pin.

## [FilterByPin\(string\[\]\)](#)

Filters current [ScopeSessionsBundle](#) and returns a new one with requested pins.

## [FilterBySite\(int\)](#)

Filters current [ScopeSessionsBundle](#) and returns a new one with the requested site.

## [FilterBySite\(int\[\]\)](#)

Filters current [ScopeSessionsBundle](#) and returns a new one with requested sites.

# Constructor ScopeSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**ScopeSessionsBundle(ISemiconductorModuleContext,  
IEnumerable<ScopeSessionInformation>)**

Constructs a sessions bundle object that represents a bunch of [ScopeSessionInformations](#).

```
public ScopeSessionsBundle(ISemiconductorModuleContext tsmContext,  
IEnumerable<ScopeSessionInformation> allSessionInfo)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

**allSessionInfo** [IEnumerable](#)<[ScopeSessionInformation](#)>

An enumerable of [ScopeSessionInformations](#).

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

```
public IList<SitePinInfo> AggregateSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [ISessionInformation](#).

```
public IEnumerable<ScopeSessionInformation> InstrumentSessions { get; }
```

## Property Value

[IEnumerable](#) <[ScopeSessionInformation](#)>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public ISemiconductorModuleContext TSMContext { get; }
```

## Property Value

ISemiconductorModuleContext

# Method FilterByPin

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterByPin(string)

Filters current [ScopeSessionsBundle](#) and returns a new one with the requested pin.

```
public ScopeSessionsBundle FilterByPin(string requestedPin)
```

### Parameters

**requestedPin** [string](#) ↗

The requested pin.

### Returns

[ScopeSessionsBundle](#)

A new [ScopeSessionsBundle](#) object with the requested pin.

## FilterByPin(string[])

Filters current [ScopeSessionsBundle](#) and returns a new one with requested pins.

```
public ScopeSessionsBundle FilterByPin(string[] requestedPins)
```

### Parameters

**requestedPins** [string](#) ↗[]

The requested pins.

### Returns

## [ScopeSessionsBundle](#)

A new [ScopeSessionsBundle](#) object with requested pins.

# Method FilterBySite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Scope](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterBySite(int)

Filters current [ScopeSessionsBundle](#) and returns a new one with the requested site.

```
public ScopeSessionsBundle FilterBySite(int requestedSite)
```

### Parameters

`requestedSite` [int](#)

The requested site.

### Returns

[ScopeSessionsBundle](#)

A new [ScopeSessionsBundle](#) object with the requested site.

## FilterBySite(int[])

Filters current [ScopeSessionsBundle](#) and returns a new one with requested sites.

```
public ScopeSessionsBundle FilterBySite(int[] requestedSites)
```

### Parameters

`requestedSites` [int](#)[]

The requested sites.

### Returns

## [ScopeSessionsBundle](#)

A new [ScopeSessionsBundle](#) object with requested sites.

# Namespace NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync

## Classes

### [InitializeAndClose](#)

Defines [NI Sync](#) sessions initialize and close APIs.

### [NI Sync](#)

Wraps the NI-Sync C driver.

### [SyncSessionInformation](#)

Defines an object that contains an individual [NI Sync](#) session and its related information.

### [SyncSessionsBundle](#)

Defines an object that contains one or more [SyncSessionInformation](#) objects.

## Enums

### [SynchronizationClock](#)

Specifies the synchronization clock options.

### [UpdateEdge](#)

Specifies the update edge options.

# Class InitializeAndClose

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines [NISync](#) sessions initialize and close APIs.

```
public static class InitializeAndClose
```

## Inheritance

[object](#) ← InitializeAndClose

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Fields

### [NISyncInstrumentTypeld](#)

Defines an instrument type ID to identify [NISync](#) instruments when get/set sessions from/to the NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModule Context.

## Methods

### [Close\(ISemiconductorModuleContext, bool\)](#)

Closes all [NISync](#) sessions.

### [Initialize\(ISemiconductorModuleContext, bool\)](#)

Initializes [NISync](#) sessions in the test system.

### [Reset\(ISemiconductorModuleContext\)](#)

Resets all [NISync](#) sessions.

# Field NI SyncInstrumentTypeId

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an instrument type ID to identify [NI Sync](#) instruments when get/set sessions from/to the National Instruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public const string NI SyncInstrumentTypeId = "Sync"
```

Returns

[string](#)

Defines an instrument type ID to identify instruments when get/set sessions from/to the .

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close(ISemiconductorModuleContext, bool)

Closes all [NISync](#) sessions.

```
public static void Close(ISemiconductorModuleContext tsmContext, bool resetDevice = true)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset the device sessions before closing.

# Method Initialize

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Initialize(ISemiconductorModuleContext, bool)

Initializes [NISync](#) sessions in the test system.

```
public static void Initialize(ISemiconductorModuleContext tsmContext, bool resetDevice  
= false)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

**resetDevice** [bool](#)

Whether to reset device during initialization.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset(ISemiconductorModuleContext)

Resets all [NISync](#) sessions.

```
public static void Reset(ISemiconductorModuleContext tsmContext)
```

### Parameters

**tsmContext** ISemiconductorModuleContext

The NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext object.

# Class NI Sync

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Wraps the NI-Sync C driver.

```
public class NI Sync
```

## Inheritance

[object](#) ← NI Sync

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[NI Sync\(string, bool, bool\)](#)

Creates a new NI-Sync instrument driver session.

## Properties

[ResourceName](#)

Indicates the name of the NI-Sync device.

## Methods

[Close\(\)](#)

Ends an NI-Sync instrument driver session and frees the device for other operations.

[ConnectClkTerminals\(string, string\)](#)

Connects a source clock terminal to a destination clock terminal. A clock terminal connection is characterized by its source terminal and destination terminal.

[ConnectSoftwareTrigger\(string, SynchronizationClock, bool, UpdateEdge, double\)](#)

Connects the global software trigger to a destination trigger terminal. Once you connect the global software trigger, you can fire the trigger using SendSoftwareTrigger(string).

#### [ConnectTriggerTerminals\(string, string, SynchronizationClock, bool, UpdateEdge\)](#)

Routes triggers through the PXI backplane, between devices, or between multiple chassis. Once a terminal route is connected, you can invert the trigger signal at the destination terminal, synchronize the trigger to the rising or falling edge of a synchronization clock, fire the trigger asynchronously, or route the trigger to other trigger terminals. You can also route clocks along some trigger lines by setting a full speed or divided synchronization clock as the source terminal.

#### [DisconnectClkTerminals\(string, string\)](#)

Closes a route between a source clock terminal and a destination clock terminal.

#### [DisconnectSoftwareTrigger\(string\)](#)

Closes a route between the global software trigger and a destination trigger terminal.

#### [DisconnectTriggerTerminals\(string, string\)](#)

Closes a route between a source trigger terminal and a destination trigger terminal.

#### [Reset\(\)](#)

Resets the specified device. You can also reset a device before the program starts by using the reset device parameter of niSync Initialize.

#### [SendSoftwareTrigger\(\)](#)

Sends a trigger pulse using the global software trigger. You must first route the global software trigger to at least one destination terminal using ConnectSoftwareTrigger(string, string, string, int, int, double) for this operation to have any effect.

# Constructor NI Sync

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## NI Sync(string, bool, bool)

Creates a new NI-Sync instrument driver session.

```
public NI Sync(string resourceName, bool idQuery = true, bool resetDevice = false)
```

### Parameters

**resourceName** [string](#)

The name of the NI-Sync device.

**idQuery** [bool](#)

Specifies whether to query the instrument ID to determine if the instrument is compatible with the NI-Sync driver. The default is TRUE.

**resetDevice** [bool](#)

Specifies whether to reset the NI-Sync module to its default state—including the selected time reference, any connected terminals, and all scheduled future time events—during the initialization procedure. The default is FALSE.

# Property ResourceName

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ResourceName

Indicates the name of the NI-Sync device.

```
public string ResourceName { get; }
```

## Property Value

[string](#) ↗

# Method Close

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Close()

Ends an NI-Sync instrument driver session and frees the device for other operations.

```
public void Close()
```

# Method ConnectClkTerminals

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ConnectClkTerminals(string, string)

Connects a source clock terminal to a destination clock terminal. A clock terminal connection is characterized by its source terminal and destination terminal.

```
public void ConnectClkTerminals(string sourceTerminal, string destinationTerminal)
```

### Parameters

**sourceTerminal** [string](#)

Specifies the source terminal of the clock you would like to connect.

**destinationTerminal** [string](#)

Specifies where to route the clock signal specified in the source terminal.

# Method ConnectSoftwareTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ConnectSoftwareTrigger(string, SynchronizationClock, bool, UpdateEdge, double)

Connects the global software trigger to a destination trigger terminal. Once you connect the global software trigger, you can fire the trigger using SendSoftwareTrigger(string).

```
public void ConnectSoftwareTrigger(string destinationTerminal, SynchronizationClock synchronizationClock = SynchronizationClock.FullSpeedClock, bool invertSignal = false, UpdateEdge updateEdge = UpdateEdge.RisingEdge, double delay = 0)
```

### Parameters

#### destinationTerminal [string](#)

Specifies the terminal where the Global Software Trigger will fire.

#### synchronizationClock [SynchronizationClock](#)

Specifies whether to use the full-speed or a divided synchronization clock to control when the global software trigger fires. The global software trigger will be synchronized with the rising or falling edge of the clock you select in this terminal.

#### invertSignal [bool](#)

Specifies whether to invert the digital signal of the Global Software Trigger when it reaches the destination terminal:

#### updateEdge [UpdateEdge](#)

Specifies on which update edge of the synchronization clock to propagate the global software trigger.

#### delay [double](#)

Specifies the amount of time, in seconds, to delay the global software trigger pulse. The delay must be a multiple of the synchronization clock period. The global software trigger can be delayed up to 15

clock cycles for each route.

# Method ConnectTriggerTerminals

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## ConnectTriggerTerminals(string, string, SynchronizationClock, bool, UpdateEdge)

Routes triggers through the PXI backplane, between devices, or between multiple chassis. Once a terminal route is connected, you can invert the trigger signal at the destination terminal, synchronize the trigger to the rising or falling edge of a synchronization clock, fire the trigger asynchronously, or route the trigger to other trigger terminals. You can also route clocks along some trigger lines by setting a full speed or divided synchronization clock as the source terminal.

```
public void ConnectTriggerTerminals(string sourceTerminal, string destinationTerminal,  
SynchronizationClock synchronizationClock = SynchronizationClock.Asynchronous, bool  
invertSignal = false, UpdateEdge updateEdge = UpdateEdge.RisingEdge)
```

### Parameters

**sourceTerminal** [string](#)

Specifies the source of the trigger you want to connect to the destination terminal.

**destinationTerminal** [string](#)

Specifies the destination trigger terminal that the source terminal will connect to.

**synchronizationClock** [SynchronizationClock](#)

Specifies whether to use the full-speed or a divided synchronization clock to control when the trigger fires. The trigger will be synchronized with the rising or falling edge of the clock you select in this terminal.

**invertSignal** [bool](#)

Specifies whether to invert the source terminal signal at the destination terminal.

**updateEdge** [UpdateEdge](#)

Specifies on which update edge of the synchronization clock to propagate the trigger.

# Method DisconnectClkTerminals

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DisconnectClkTerminals(string, string)

Closes a route between a source clock terminal and a destination clock terminal.

```
public void DisconnectClkTerminals(string sourceTerminal, string destinationTerminal)
```

### Parameters

**sourceTerminal** [string](#)

Specifies the source clock terminal to disconnect from the destination terminal.

**destinationTerminal** [string](#)

Specifies the destination clock terminal to disconnect from the source terminal.

# Method DisconnectSoftwareTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DisconnectSoftwareTrigger(string)

Closes a route between the global software trigger and a destination trigger terminal.

```
public void DisconnectSoftwareTrigger(string destinationTerminal)
```

### Parameters

**destinationTerminal** [string](#)

Specifies the destination trigger terminal to disconnect the global software trigger terminal from. The global software trigger must be connected to the destination terminal for this operation to succeed.

# Method DisconnectTriggerTerminals

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## DisconnectTriggerTerminals(string, string)

Closes a route between a source trigger terminal and a destination trigger terminal.

```
public void DisconnectTriggerTerminals(string sourceTerminal, string destinationTerminal)
```

### Parameters

**sourceTerminal** [string](#)

Specifies the source terminal you would like to disconnect from the destination terminal. The source and destination terminals must be connected for this operation to succeed.

**destinationTerminal** [string](#)

Specifies the destination trigger terminal to disconnect from the source terminal. The source and destination terminals must be connected for this operation to succeed.

# Method Reset

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Reset()

Resets the specified device. You can also reset a device before the program starts by using the reset device parameter of niSync Initialize.

```
public void Reset()
```

# Method SendSoftwareTrigger

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SendSoftwareTrigger()

Sends a trigger pulse using the global software trigger. You must first route the global software trigger to at least one destination terminal using ConnectSoftwareTrigger(string, string, string, int, int, double) for this operation to have any effect.

```
public void SendSoftwareTrigger()
```

# Class SyncSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains an individual [NISync](#) session and its related information.

```
public class SyncSessionInformation : ISessionInformation
```

## Inheritance

[object](#) ← SyncSessionInformation

## Implements

[ISessionInformation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

[SyncSessionInformation\(NISync\)](#)

Constructs a session information object that associates with a [NISync](#) instrument session.

## Properties

[AssociatedSitePinList](#)

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

[Session](#)

The [NISync](#) session.

# Constructor SyncSessionInformation

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## SyncSessionInformation(NISync)

Constructs a session information object that associates with a [NISync](#) instrument session.

```
public SyncSessionInformation(NISync session)
```

### Parameters

session [NISync](#)

The [NISync](#) session.

# Property AssociatedSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AssociatedSitePinList

Gets a list of [SitePinInfo](#) objects containing information for the individual site-pin pairs associated with the driver session.

```
public IList<SitePinInfo> AssociatedSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property Session

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## Session

The [NISync](#) session.

```
public NISync Session { get; }
```

## Property Value

[NISync](#)

# Class SyncSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Defines an object that contains one or more [SyncSessionInformation](#) objects.

```
public class SyncSessionsBundle : ISessionsBundle<SyncSessionInformation>
```

## Inheritance

[object](#) ← SyncSessionsBundle

## Implements

[ISessionsBundle<SyncSessionInformation>](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool\[\], string>\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, bool>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double\[\]>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, double>, string\)](#) ,  
[ParallelExecution.DoAndPublishResults<TSessionInformation>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<double\[\], double\[\]>, string, string\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, TResult>\)](#) ,  
[ParallelExecution.DoAndReturnPerInstrumentPerChannelResults<TSessionInformation, TResult1, TResult2>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, Tuple<TResult1, TResult2>>\)](#) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>\(ISessionsBundle<TSessionInformation>, Func<TSessionInformation, SitePinInfo, TResult>\)](#) ,

[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, int, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[,\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, TResult\[\]>](#)) ,  
[ParallelExecution.DoAndReturnPerSitePerPinResults<TSessionInformation, TResult1, TResult2>](#)  
([ISessionsBundle<TSessionInformation>](#), [Func<TSessionInformation, Tuple<TResult1\[\], TResult2\[\]>>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int, SitePinInfo>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation, int>](#)) ,  
[ParallelExecution.Do<TSessionInformation>](#)([ISessionsBundle<TSessionInformation>](#),  
[Action<TSessionInformation>](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]](#),  
[string](#)) ,  
[Publish.PublishResults<TSessionInformation, TData>](#)([ISessionsBundle<TSessionInformation>](#), [TData\[\]\[\]](#),  
[string](#)).

## Constructors

[SyncSessionsBundle\(ISemiconductorModuleContext, IEnumerable<SyncSessionInformation>\)](#)

Constructs a sessions bundle object that represents a bunch of [SyncSessionInformation](#)s.

## Properties

[AggregateSitePinList](#)

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

[InstrumentSessions](#)

An enumerable of [ISessionInformation](#).

[TSMContext](#)

The associated `NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext`.

# Methods

## [FilterBySite\(int\)](#)

Filters current [SyncSessionsBundle](#) and returns a new one with the requested site.

## [FilterBySite\(int\[\]\)](#)

Filters current [SyncSessionsBundle](#) and returns a new one with requested sites.

# Constructor SyncSessionsBundle

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

**SyncSessionsBundle(ISemiconductorModuleContext,  
IEnumerable<SyncSessionInformation>)**

Constructs a sessions bundle object that represents a bunch of [SyncSessionInformations](#).

```
public SyncSessionsBundle(ISemiconductorModuleContext tsmContext,  
IEnumerable<SyncSessionInformation> allSessionInfo)
```

## Parameters

**tsmContext** ISemiconductorModuleContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

**allSessionInfo** [IEnumerable](#)<SyncSessionInformation>

An enumerable of [SyncSessionInformations](#).

# Property AggregateSitePinList

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## AggregateSitePinList

An aggregated list of [SitePinInfo](#) objects containing information for all of the individual site-pin pairs associated with each of the session information objects within the bundle.

```
public IList<SitePinInfo> AggregateSitePinList { get; }
```

Property Value

[IList](#)<[SitePinInfo](#)>

# Property InstrumentSessions

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## InstrumentSessions

An enumerable of [ISessionInformation](#).

```
public IEnumerable<SyncSessionInformation> InstrumentSessions { get; }
```

## Property Value

[IEnumerable](#) <[SyncSessionInformation](#)>

# Property TSMContext

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## TSMContext

The associated NationalInstruments.TestStand.SemiconductorModule.CodeModuleAPI.ISemiconductorModuleContext.

```
public ISemiconductorModuleContext TSMContext { get; }
```

## Property Value

ISemiconductorModuleContext

# Method FilterBySite

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

## FilterBySite(int)

Filters current [SyncSessionsBundle](#) and returns a new one with the requested site.

```
public SyncSessionsBundle FilterBySite(int requestedSite)
```

### Parameters

`requestedSite int`

The requested site.

### Returns

[SyncSessionsBundle](#)

A new [SyncSessionsBundle](#) object with the requested site.

## FilterBySite(int[])

Filters current [SyncSessionsBundle](#) and returns a new one with requested sites.

```
public SyncSessionsBundle FilterBySite(int[] requestedSites)
```

### Parameters

`requestedSites int[]`

The requested sites.

### Returns

## [SyncSessionsBundle](#)

A new [SyncSessionsBundle](#) object with requested sites.

# Enum SynchronizationClock

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Specifies the synchronization clock options.

```
public enum SynchronizationClock
```

## Fields

**Asynchronous = 0**

The trigger is not synchronized to any clock.

**DividedClock1 = 2**

Divides the synchronization clock by the value specified in the Clock Divisor 1 parameter and uses the frequency of the divided clock to synchronize the trigger.

**DividedClock2 = 3**

Divides the synchronization clock by the value specified in the Clock Divisor 2 parameter and uses the frequency of the divided clock to synchronize the trigger.

**FullSpeedClock = 1**

Uses the full speed frequency of the synchronization clock to synchronize the trigger.

# Enum UpdateEdge

Namespace: [NationalInstruments.SemiconductorTestLibrary.InstrumentAbstraction.Sync](#)

Assembly: NationalInstruments.SemiconductorTestLibrary.Abstractions.dll

Specifies the update edge options.

```
public enum UpdateEdge
```

## Fields

**FallingEdge = 1**

Propagates the trigger when the digital signal of the synchronization clock transitions from high to low.

**RisingEdge = 0**

Propagates the trigger when the digital signal of the synchronization clock transitions from low to high.