Namespace NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Classes

CommonSteps

s

SetupAndCleanupSteps

Defines entry points for semiconductor setup and cleanup steps.

Structs

DMMMeasurementSettings

Defines DMM measurement settings.

Enums

SetupAndCleanupSteps.NIInstrumentType

Defines NI instrument types the NI Semiconductor Test Library supports.

Class CommonSteps

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

s

```
public static class CommonSteps
```

Inheritance

object ← CommonSteps

Inherited Members

object.ToString() , object.Equals(object) , object.Equals(object, object) , object.ReferenceEquals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone()

Methods

BurstPattern(ISemiconductorModuleContext, string[], string)

Bursts a pattern and publishes the pass/fail results as well as the fail count.

ContinuityTest(ISemiconductorModuleContext, string[], double[], string[], double[], double[], double[], double, double)

Performs a basic continuity test. It serially checks either upper or lower protection diodes DUT pins, regardless of if they are mapped to digital or SMU instruments. The test will first set 0V on all the pins and then source a small amount of current on the targeted continuity pins to validate the voltage drop across the protection diode. After current is applied, the targeted pin(s) will be forced back to 0V before continuing on to the next pin. Note that each continuity pin will be tested one pin at a time. Pins mapped to either an NI SMU or NI PPMU(s) instrument channel are supported.

DutPowerDown(ISemiconductorModuleContext, string[], double, bool, bool)

Forces DC voltage to Zero on the specified DUT supply pins. If the `powerDownSuppliesSerially` Boolean is set to True, the DUT supplies will be powered down sequentially in the order provided, and the `settlingTime` input will be used after power down each pin. If `forceLowestCurrentLimit` is set to false, the `dutSupplyPinsOrPinGroups` will be configured to force 0V at the currently programmed current limit and range, which may not be desirable. Otherwise, it will default to true, which will ensure the pins are configured to force 0V at the lowest possible current limit to mimic high-impedance. The actual

current limit selected is dependent on the mapped hardware channel. Both DCPower and Digital PPMU pins are supported.

DutPowerUp(ISemiconductorModuleContext, string[], double[], double[], double, bool)
Forces DC voltage on the specified DUT supply pins. Must provide voltages and current limit values for each of the DUT supply pins. If the `powerUpSuppliesSerially` Boolean is set to True, the DUT supplies will be powered up sequentially in the order provided, and the `settlingTime` input will be used after power up each pin. Both DCPower and Digital PPMU pins are supported.

ForceCurrentMeasureVoltage(ISemiconductorModuleContext, string[], double, double, double, double)
Forces the specified DC current on all pins and/or pin groups specified, waits the specified amount of settling time, and then measures the voltage on those pins and publishes the results to TestStand. Both DCPower and Digital PPMU pins are supported. Both the `settlingTime` and `apertureTime` inputs are expected to be provided in Seconds. By default the `apertureTime` input is set to -1, which will cause this input to be ignored and the device will use any pre-configured aperture time set by a proceeding set, such as the Setup NI-DCPower Instrumentation step.

ForceDcCurrent(ISemiconductorModuleContext, string[], double, double, double)
Forces the specified DC current on all pins and/or pin groups specified. Both DCPower and Digital PPMU pins are supported. If a value is provided to the settlingTimeInSeconds input, the method will wait the specified amount of settling time before continuing.

ForceDcVoltage(ISemiconductorModuleContext, string[], double, double, double)
Forces the specified DC voltage on all pins and/or pin groups specified. Both DCPower and Digital PPMU pins are supported. If a value is provided to the settlingTimeInSeconds input, the method will wait the specified amount of settling time before continuing.

ForceVoltageMeasureCurrent(ISemiconductorModuleContext, string[], double, double, double, double)
Forces the specified DC voltage on all pins and/or pin groups specified, waits the specified amount of settling time, and then measures the current on those pins and publishes the results to TestStand. Both DCPower and Digital PPMU pins are supported. Both the `settlingTime` and `apertureTime` inputs are expected to be provided in Seconds. By default the `apertureTime` input is set to -1, which will cause this input to be ignored and the device will use any pre-configured aperture time set by a proceeding set, such as the Setup NI-DCPower Instrumentation step.

LeakageTest(ISemiconductorModuleContext, string[], double, double, double, double, bool)
Performs a single leakage measurement. The test configures the instruments connected to the pins specified, sources voltage on each pin, and then measures the current on all pins simultaneously. Provide input values to the method to specify the voltage to source and the current limit for the leakage measurement. The leakage current measurements are published separately for each pin, regardless of if a pin group is provided. Configure test evaluations on the Tests tab of the Step Settings pane in TestStand to evaluate the results of these measurements. If the forced voltage cannot

be applied to pins simultaneously, you can set the `serialOperationEnabled` Boolean input to true (false by default). When the `serialOperationEnabled` Boolean input is set to true and a pin group is provided, this will still perform parallel operations across all pins in the pin group. Take advantage of this functionality for when leakage can be performed on some DUT pins in parallel, but not others, and define those pins within separate pin groups (i.e. LeakageOddPins and LeakageEvenPins). Pins mapped to either an NI SMU or NI PPMU(s) instrument channel are supported.

Method BurstPattern

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

BurstPattern(ISemiconductorModuleContext, string[], string)

Bursts a pattern and publishes the pass/fail results as well as the fail count.

```
public static void BurstPattern(ISemiconductorModuleContext tsmContext, string[]
pinsOrPinGroups, string patternName)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`pinsOrPinGroups` string[]

The pins or pin groups.

`patternName` string

The name of the pattern to burst.

Method ContinuityTest

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

ContinuityTest(ISemiconductorModuleContext, string[], double[], string[], double[], double[], double[], double, double)

Performs a basic continuity test. It serially checks either upper or lower protection diodes DUT pins, regardless of if they are mapped to digital or SMU instruments. The test will first set 0V on all the pins and then source a small amount of current on the targeted continuity pins to validate the voltage drop across the protection diode. After current is applied, the targeted pin(s) will be forced back to 0V before continuing on to the next pin. Note that each continuity pin will be tested one pin at a time. Pins mapped to either an NI SMU or NI PPMU(s) instrument channel are supported.

```
public static void ContinuityTest(ISemiconductorModuleContext tsmContext, string[]
supplyPinsOrPinGroups, double[] currentLimitsPerSupplyPinOrPinGroup, string[]
continuityPinsOrPinGroups, double[] currentLevelPerContinuityPinOrPinGroup, double[]
voltageLimitHighPerContinuityPinOrPinGroup, double[]
voltageLimitLowPerContinuityPinOrPinGroup, double apertureTime, double settlingTime)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`supplyPinsOrPinGroups` string[]

The supply pins or pin groups.

`currentLimitsPerSupplyPinOrPinGroup` double[]

The current limits for every supply pin or pin group, in amperes.

`continuityPinsOrPinGroups` string[]

The continuity pins or pin groups.

`currentLevelPerContinuityPinOrPinGroup` double[]

The current levels for every continuity pin or pin group, in amperes.

`voltageLimitHighPerContinuityPinOrPinGroup` double[]

The voltage limit high for every continuity pin or pin group, in volts.

`voltageLimitLowPerContinuityPinOrPinGroup` double[]

The voltage limit low for every continuity pin or pin group, in volts.

`apertureTime` double

The measurement aperture time in seconds.

`settlingTime` double

The amount of time to wait before measuring the voltage, in seconds.

Method DutPowerDown

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

DutPowerDown(ISemiconductorModuleContext, string[], double, bool, bool)

Forces DC voltage to Zero on the specified DUT supply pins. If the `powerDownSuppliesSerially` Boolean is set to True, the DUT supplies will be powered down sequentially in the order provided, and the `settlingTime` input will be used after power down each pin. If `forceLowestCurrentLimit` is set to false, the `dutSupplyPinsOrPinGroups` will be configured to force 0V at the currently programmed current limit and range, which may not be desirable. Otherwise, it will default to true, which will ensure the pins are configured to force 0V at the lowest possible current limit to mimic high-impedance. The actual current limit selected is dependent on the mapped hardware channel. Both DCPower and Digital PPMU pins are supported.

```
public static void DutPowerDown(ISemiconductorModuleContext tsmContext, string[]
dutSupplyPinsOrPinGroups, double settlingTime = 0, bool powerDownSuppliesSerially = false,
bool forceLowestCurrentLimit = true)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`dutSupplyPinsOrPinGroups` string[]

The DUT supply pins or pin groups.

`settlingTime` double

The amount of time to wait before continuing, in seconds.

`powerDownSuppliesSerially` bool

Whether to power down DUT supplies sequentially.

`forceLowestCurrentLimit` bool

Whether to force lowest current limit (100nA) for safety considerations.

Method DutPowerUp

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

DutPowerUp(ISemiconductorModuleContext, string[], double[], double[], double, bool)

Forces DC voltage on the specified DUT supply pins. Must provide voltages and current limit values for each of the DUT supply pins. If the `powerUpSuppliesSerially` Boolean is set to True, the DUT supplies will be powered up sequentially in the order provided, and the `settlingTime` input will be used after power up each pin. Both DCPower and Digital PPMU pins are supported.

```
public static void DutPowerUp(ISemiconductorModuleContext tsmContext, string[]
dutSupplyPinsOrPinGroups, double[] perSupplyPinOrPinGroupVoltages, double[]
perSupplyPinOrPinGroupCurrentLimits, double settlingTime = 0, bool powerUpSuppliesSerially
= false)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`dutSupplyPinsOrPinGroups` string[]

The DUT supply pins or pin groups.

`perSupplyPinOrPinGroupVoltages` double[]

The DC voltage level to force on each DUT supply pin or pin group, in volts.

`perSupplyPinOrPinGroupCurrentLimits` double[]

The current limit for each DUT supply pin or pin group, in amperes.

`settlingTime` double

The amount of time to wait before continuing, in seconds.

`powerUpSuppliesSerially` bool

Whether to power up DUT supplies sequentially.

Method ForceCurrentMeasureVoltage

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

ForceCurrentMeasureVoltage(ISemiconductorModuleContext, string[], double, double, double, double)

Forces the specified DC current on all pins and/or pin groups specified, waits the specified amount of settling time, and then measures the voltage on those pins and publishes the results to TestStand. Both DCPower and Digital PPMU pins are supported. Both the `settlingTime` and `apertureTime` inputs are expected to be provided in Seconds. By default the `apertureTime` input is set to -1, which will cause this input to be ignored and the device will use any pre-configured aperture time set by a proceeding set, such as the Setup NI-DCPower Instrumentation step.

```
public static void ForceCurrentMeasureVoltage(ISemiconductorModuleContext tsmContext,
string[] pinsOrPinGroups, double currentLevel, double voltageLimit, double settlingTime = 0,
double apertureTime = -1)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`pinsOrPinGroups` string[]

The pins or pin groups to force DC voltage on.

`currentLevel` double

The DC current level to force, in amperes.

`voltageLimit` double

The voltage limit in volts.

`settlingTime` double

The amount of time to wait before continuing, in seconds.

`apertureTime` double

The measurement aperture time in seconds.

Method ForceDcCurrent

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

ForceDcCurrent(ISemiconductorModuleContext, string[], double, double, double)

Forces the specified DC current on all pins and/or pin groups specified. Both DCPower and Digital PPMU pins are supported. If a value is provided to the settlingTimeInSeconds input, the method will wait the specified amount of settling time before continuing.

```
public static void ForceDcCurrent(ISemiconductorModuleContext tsmContext, string[]
pinsOrPinGroups, double currentLevel, double voltageLimit, double settlingTime = 0)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`pinsOrPinGroups` string[]

The pins or pin groups to force DC voltage on.

`currentLevel` double

The DC current level to force, in amperes.

`voltageLimit` double

The voltage limit in volts.

`settlingTime` double

The amount of time to wait before continuing, in seconds.

Method ForceDcVoltage

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

ForceDcVoltage(ISemiconductorModuleContext, string[], double, double, double)

Forces the specified DC voltage on all pins and/or pin groups specified. Both DCPower and Digital PPMU pins are supported. If a value is provided to the settlingTimeInSeconds input, the method will wait the specified amount of settling time before continuing.

```
public static void ForceDcVoltage(ISemiconductorModuleContext tsmContext, string[]
pinsOrPinGroups, double voltageLevel, double currentLimit, double settlingTime = 0)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`pinsOrPinGroups` string[]

The pins or pin groups to force DC voltage on.

`voltageLevel` double

The DC voltage level to force, in volts.

`currentLimit` double

The current limit in amperes.

`settlingTime` double

The amount of time to wait before continuing, in seconds.

Method ForceVoltageMeasureCurrent

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

ForceVoltageMeasureCurrent(ISemiconductorModuleContext, string[], double, double, double, double)

Forces the specified DC voltage on all pins and/or pin groups specified, waits the specified amount of settling time, and then measures the current on those pins and publishes the results to TestStand. Both DCPower and Digital PPMU pins are supported. Both the `settlingTime` and `apertureTime` inputs are expected to be provided in Seconds. By default the `apertureTime` input is set to -1, which will cause this input to be ignored and the device will use any pre-configured aperture time set by a proceeding set, such as the Setup NI-DCPower Instrumentation step.

```
public static void ForceVoltageMeasureCurrent(ISemiconductorModuleContext tsmContext,
string[] pinsOrPinGroups, double voltageLevel, double currentLimit, double settlingTime = 0,
double apertureTime = -1)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`pinsOrPinGroups` string[]

The pins or pin groups to force DC voltage on.

`voltageLevel` double

The DC voltage level to force, in volts.

`currentLimit` double

The current limit in amperes.

`settlingTime` double

The amount of time to wait before continuing, in seconds.

`apertureTime` double

The measurement aperture time in seconds.

Method LeakageTest

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

LeakageTest(ISemiconductorModuleContext, string[], double, double, double, double, bool)

Performs a single leakage measurement. The test configures the instruments connected to the pins specified, sources voltage on each pin, and then measures the current on all pins simultaneously. Provide input values to the method to specify the voltage to source and the current limit for the leakage measurement. The leakage current measurements are published separately for each pin, regardless of if a pin group is provided. Configure test evaluations on the Tests tab of the Step Settings pane in TestStand to evaluate the results of these measurements. If the forced voltage cannot be applied to pins simultaneously, you can set the `serialOperationEnabled` Boolean input to true (false by default). When the `serialOperationEnabled` Boolean input is set to true and a pin group is provided, this will still perform parallel operations across all pins in the pin group. Take advantage of this functionality for when leakage can be performed on some DUT pins in parallel, but not others, and define those pins within separate pin groups (i.e. LeakageOddPins and LeakageEvenPins). Pins mapped to either an NI SMU or NI PPMU(s) instrument channel are supported.

```
public static void LeakageTest(ISemiconductorModuleContext tsmContext, string[]
pinsOrPinGroups, double voltageLevel, double currentLimit, double apertureTime, double
settlingTime, bool serialOperationEnabled = false)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`pinsOrPinGroups` string[]

The pins or pin groups.

`voltageLevel` double

The DC voltage level to force, in volts.

`currentLimit` double

The current limit in amperes.

`apertureTime` double

The measurement aperture time in seconds.

`settlingTime` double

The amount of time to wait before measuring the current, in seconds.

`serialOperationEnabled` bool

Whether to force voltage sequentially.

Struct DMMMeasurementSettings

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

Defines DMM measurement settings.

```
public struct DMMMeasurementSettings
```

Inherited Members
ValueType.Equals(object) , ValueType.GetHashCode() , ValueType.ToString() , object.Equals(object, object) , object.ReferenceEquals(object, object) , object.GetType()

Constructors

DMMMeasurementSettings(DmmMeasurementFunction, double, double)
Initializes a new instance of the DMMMeasurementSettings struct.

Properties

MeasurementFunction
The measurement function.

Range
The measurement range.

ResolutionDigits
The resolution in digits.

Constructor DMMMeasurementSettings

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

DMMMeasurementSettings(DmmMeasurementFunction, double, double)

Initializes a new instance of the DMMMeasurementSettings struct.

```
public DMMMeasurementSettings(DmmMeasurementFunction measurementFunction, double range,
double resolutionDigits)
```

Parameters

`measurementFunction` DmmMeasurementFunction

The measurement function.

`range` double

The measurement range.

`resolutionDigits` double

The resolution in digits.

Property MeasurementFunction

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

MeasurementFunction

The measurement function.

```
public DmmMeasurementFunction MeasurementFunction { get; }
```

Property Value

DmmMeasurementFunction

Property Range

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

Range

The measurement range.

```
public double Range { get; }
```

Property Value

double

Property ResolutionDigits

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

ResolutionDigits

The resolution in digits.

```
public double ResolutionDigits { get; }
```

Property Value

double

Class SetupAndCleanupSteps

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

Defines entry points for semiconductor setup and cleanup steps.

```
public static class SetupAndCleanupSteps
```

Inheritance

object ← SetupAndCleanupSteps

Inherited Members

object.ToString() , object.Equals(object) , object.Equals(object, object) , object.ReferenceEquals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone()

Methods

CleanupInstrumentation(ISemiconductorModuleContext, bool, NIInstrumentType)

Closes any open instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset before closing the session (default = False). The sessions will always be closed in parallel. By default, the `instrumentType` input is set to All, which closes sessions for all instrument types in parallel. This can be configured to target a specific instrument type, which can be useful for debugging purposes and/or if there is a need to ensure sessions are closed sequentially (requiring multiple instances of this step). Note that the following types are supported: niDCPower, niDigitalPattern, niRelayDriver, niDAQmx, niDMM, niScope, niFGen, and niSync.

ResetInstrumentation(ISemiconductorModuleContext, bool, NIInstrumentType)

Resets the instrument sessions for the specified `instrumentType` associated with the pin map by invoking the Reset() method of the supported instrument driver. By default, the `instrumentType` input is set to All, which targets all supported instrument types in parallel. This can be configured to target a specific instrument type, which can be useful for debugging purposes and/or if there is a need to ensure instruments are reset individually or sequentially (requiring multiple instances of this step). Note that the following types are supported: niDCPower, niDigitalPattern, niRelayDriver, niDMM, niScope, niFGen, and niSync. For instrumentation that also have the ResetDevice() method (hard reset), this can optionally be invoked instead of the Reset() method (soft-reset) if the `resetDevice` input is set True (default = False): niDCPower, niDigitalPattern, niScope, and niFGen. Refer to the individual instrument driver documentation for more details.

SetupNIDCPowerInstrumentation(ISemiconductorModuleContext, bool, double, DCPowerMeasureApertureTimeUnits, DCPowerMeasurementWhen, DCPowerMeasurementSense, double, double)

Initializes the NI DCPower instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False). If the `sourceDelay` is set to -1 the method will not set the source delay property, and will assume the initialized default value from the driver, which is expected to be the inverse of the power line frequency. If the `powerLineFrequency` is set to -1, the method will attempt to automatically determine the power line frequency and set the power line frequency property for the respective driver sessions. If the power line frequency cannot be determined, the property will not be set and the driver will use the default value of this property (60 Hz). This is currently only supported by systems that use a PXIe-109x chassis or newer.

SetupNIDMMInstrumentation(ISemiconductorModuleContext, bool, DmmApertureTimeUnits, double, double, double, DMMMeasurementSettings?)

Initializes NI DMM instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False). If the `powerLineFrequency` is set to -1, the method will attempt to automatically determine the power line frequency and set the power line frequency property for the respective driver sessions. If the power line frequency cannot be determined, the property will not be set and the driver will use the default value of this property (60Hz). This is currently only supported by systems that use a PXIe-109x chassis or newer. If the `initialMeasurmentSettings` is specified, all DMM sessions will be configured to the specified measurement function (not configured by default).

SetupNIDigitalPatternInstrumentation(ISemiconductorModuleContext, bool, string, string)

Initializes an NI Digital Pattern instrument sessions associated with the pin map. It loads in all digital project files associated with the digital project configured for the test program, this includes the: pin map, specifications, patterns, source waveforms, capture waveforms, timing sheets, and levels sheets. No sheets will be applied during this setup step unless specified by the `levelsSheetToApply` and/or `timingSheetToApply` inputs. Otherwise, the program will not assume to know loaded sheet to apply and will expect that the users program will apply the appropriate sheet(s) within a proceeding step. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

SetupNIFGenInstrumentation(ISemiconductorModuleContext, bool)

Initializes NI FGEN instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

SetupNIRelayModules(ISemiconductorModuleContext, bool, string)

Initializes NI Switch instrument sessions and NI DAQmx tasks associated with relay modules within the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False). If the `initialRelayConfigurationToApply` input is provided, the step will

apply the specified relay configuration. Note that the relay configuration must be defined within the pin map, otherwise the step will throw an exception. Supported devices: PXI-2567 and PXIe-6368.

SetupNIScopeInstrumentation(ISemiconductorModuleContext, bool)

Initializes NI Scope instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

SetupNISyncInstrumentation(ISemiconductorModuleContext, bool)

Initializes the NI Sync instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

Method CleanupInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

CleanupInstrumentation(ISemiconductorModuleContext, bool, NIInstrumentType)

Closes any open instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset before closing the session (default = False). The sessions will always be closed in parallel. By default, the `instrumentType` input is set to All, which closes sessions for all instrument types in parallel. This can be configured to target a specific instrument type, which can be useful for debugging purposes and/or if there is a need to ensure sessions are closed sequentially (requiring multiple instances of this step). Note that the following types are supported: niDCPower, niDigitalPattern, niRelayDriver, niDAQmx, niDMM, niScope, niFGen, and niSync.

```
public static void CleanupInstrumentation(ISemiconductorModuleContext tsmContext,
bool resetDevice = false, SetupAndCleanupSteps.NIInstrumentType instrumentType
= NIInstrumentType.All)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

`instrumentType` SetupAndCleanupSteps.NIInstrumentType

The type of instrument to close.

Method ResetInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

ResetInstrumentation(ISemiconductorModuleContext, bool, NIInstrumentType)

Resets the instrument sessions for the specified `instrumentType` associated with the pin map by invoking the Reset() method of the supported instrument driver. By default, the `instrumentType` input is set to All, which targets all supported instrument types in parallel. This can be configured to target a specific instrument type, which can be useful for debugging purposes and/or if there is a need to ensure instruments are reset individually or sequentially (requiring multiple instances of this step). Note that the following types are supported: niDCPower, niDigitalPattern, niRelayDriver, niDMM, niScope, niFGen, and niSync. For instrumentation that also have the ResetDevice() method (hard reset), this can optionally be invoked instead of the Reset() method (soft-reset) if the `resetDevice` input is set True (default = False): niDCPower, niDigitalPattern, niScope, and niFGen. Refer to the individual instrument driver documentation for more details.

```
public static void ResetInstrumentation(ISemiconductorModuleContext tsmContext,
bool resetDevice = false, SetupAndCleanupSteps.NIInstrumentType instrumentType
= NIInstrumentType.All)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to perform a hard reset on the device.

`instrumentType` SetupAndCleanupSteps.NIInstrumentType

The type of instrument to reset.

Method SetupNIDCPowerInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

SetupNIDCPowerInstrumentation(ISemiconductorModuleContext, bool, double, DCPowerMeasureApertureTimeUnits, DCPowerMeasurementWhen, DCPowerMeasurementSense, double, double)

Initializes the NI DCPower instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False). If the `sourceDelay` is set to -1 the method will not set the source delay property, and will assume the initialized default value from the driver, which is expected to be the inverse of the power line frequency. If the `powerLineFrequency` is set to -1, the method will attempt to automatically determine the power line frequency and set the power line frequency property for the respective driver sessions. If the power line frequency cannot be determined, the property will not be set and the driver will use the default value of this property (60 Hz). This is currently only supported by systems that use a PXIe-109x chassis or newer.

```
public static void SetupNIDCPowerInstrumentation(ISemiconductorModuleContext tsmContext,
bool resetDevice = false, double apertureTime = 1, DCPowerMeasureApertureTimeUnits
apertureTimeUnits = DCPowerMeasureApertureTimeUnits.PowerLineCycles, DCPowerMeasurementWhen
measureWhen = DCPowerMeasurementWhen.OnDemand, DCPowerMeasurementSense measurementSense =
DCPowerMeasurementSense.Remote, double sourceDelay = -1, double powerLineFrequency = -1)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

`apertureTime` double

The aperture time.

`apertureTimeUnits` DCPowerMeasureApertureTimeUnits

The aperture time units.

`measureWhen` DCPowerMeasurementWhen

When to do measurement.

`measurementSense` DCPowerMeasurementSense

The measurement sense to use.

`sourceDelay` double

The source delay in seconds.

`powerLineFrequency` double

The power line frequency.

Method SetupNIDMMInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

SetupNIDMMInstrumentation(ISemiconductorModuleContext, bool, DmmApertureTimeUnits, double, double, double, DMMMeasurementSettings?)

Initializes NI DMM instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False). If the `powerLineFrequency` is set to -1, the method will attempt to automatically determine the power line frequency and set the power line frequency property for the respective driver sessions. If the power line frequency cannot be determined, the property will not be set and the driver will use the default value of this property (60Hz). This is currently only supported by systems that use a PXIe-109x chassis or newer. If the `initialMeasurmentSettings` is specified, all DMM sessions will be configured to the specified measurement function (not configured by default).

```
public static void SetupNIDMMInstrumentation(ISemiconductorModuleContext tsmContext, bool
resetDevice = false, DmmApertureTimeUnits apertureTimeUnits =
DmmApertureTimeUnits.PowerLineCycles, double apertureTime = 1, double settleTime = 0.01,
double powerLineFrequency = -1, DMMMeasurementSettings? initialMeasurmentSettings = null)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

`apertureTimeUnits` DmmApertureTimeUnits

The aperture time units.

`apertureTime` double

The aperture time.

`settleTime` double

The settle time.

`powerLineFrequency` double

The power line frequency.

`initialMeasurmentSettings` DMMMeasurementSettings?

The initial measurement settings.

Method SetupNIDigitalPatternInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

SetupNIDigitalPatternInstrumentation(ISemiconductorModuleContext, bool, string, string)

Initializes an NI Digital Pattern instrument sessions associated with the pin map. It loads in all digital project files associated with the digital project configured for the test program, this includes the: pin map, specifications, patterns, source waveforms, capture waveforms, timing sheets, and levels sheets. No sheets will be applied during this setup step unless specified by the `levelsSheetToApply` and/or `timingSheetToApply` inputs. Otherwise, the program will not assume to know loaded sheet to apply and will expect that the users program will apply the appropriate sheet(s) within a proceeding step. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

```
public static void SetupNIDigitalPatternInstrumentation(ISemiconductorModuleContext
tsmContext, bool resetDevice = false, string levelsSheetToApply = "", string
timingSheetToApply = "")
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

`levelsSheetToApply` string

The name of the levels sheet to apply.

`timingSheetToApply` string

The name of the timing sheet to apply.

Method SetupNIFGenInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

SetupNIFGenInstrumentation(ISemiconductorModuleContext, bool)

Initializes NI FGEN instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

```
public static void SetupNIFGenInstrumentation(ISemiconductorModuleContext tsmContext, bool
resetDevice = false)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

Method SetupNIRelayModules

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

SetupNIRelayModules(ISemiconductorModuleContext, bool, string)

Initializes NI Switch instrument sessions and NI DAQmx tasks associated with relay modules within the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False). If the `initialRelayConfigurationToApply` input is provided, the step will apply the specified relay configuration. Note that the relay configuration must be defined within the pin map, otherwise the step will throw an exception. Supported devices: PXI-2567 and PXIe-6368.

```
public static void SetupNIRelayModules(ISemiconductorModuleContext tsmContext, bool
resetDevice = false, string initialRelayConfigurationToApply = "")
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

`initialRelayConfigurationToApply` string

The initial relay configuration to apply.

Method SetupNIScopeInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

SetupNIScopeInstrumentation(ISemiconductorModuleContext, bool)

Initializes NI Scope instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

```
public static void SetupNIScopeInstrumentation(ISemiconductorModuleContext tsmContext, bool
resetDevice = false)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

Method SetupNISyncInstrumentation

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

SetupNISyncInstrumentation(ISemiconductorModuleContext, bool)

Initializes the NI Sync instrument sessions associated with the pin map. If the `resetDevice` input is set to True, then the instrument will be reset as the session is initialized (default = False).

```
public static void SetupNISyncInstrumentation(ISemiconductorModuleContext tsmContext, bool
resetDevice = false)
```

Parameters

`tsmContext` ISemiconductorModuleContext

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

`resetDevice` bool

Whether to reset device during initialization.

Enum SetupAndCleanupSteps.NIInstrumentType

Namespace: NationalInstruments.SemiconductorTestLibrary.TestStandSteps

Assembly: NationalInstruments.SemiconductorTestLibrary.TestStandSteps.dll

Defines NI instrument types the NI Semiconductor Test Library supports.

```
public enum SetupAndCleanupSteps.NIInstrumentType
```

Fields

`All = 0`

All NI instruments.

`NIDAQmx = 4`

An NI-DAQmx task.

`NIDCPower = 1`

An NI-DCPower instrument.

`NIDMM = 5`

An NI-DMM instrument.

`NIDigitalPattern = 2`

An NI-Digital Pattern instrument.

`NIFGen = 6`

An NI-FGEN instrument.

`NIRelayDriver = 3`

A relay driver module (NI-SWITCH instrument).

`NIScope = 7`

An NI-SCOPE instrument.

`NISync = 8`

An NI-Sync instrument.