AutomationDesk

Accessing Remote Calibration COM

For AutomationDesk 6.5

Release 2021-A – May 2021

How to Contact dSPACE

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

How to Contact dSPACE Support

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

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

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

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

Software Updates and Patches

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

Important Notice

This publication contains proprietary information that is protected by copyright. All rights are reserved. The publication may be printed for personal or internal use provided all the proprietary markings are retained on all printed copies. In all other cases, the publication must not be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of dSPACE GmbH.

This publication and the contents hereof are subject to change without notice.

Contents

About This Document

Content

This document gives you information on how to access a measurement and calibration system via AutomationDesk.

Required knowledge

Working with AutomationDesk requires:

- Basic knowledge in handling the PC and the Microsoft Windows operating system.
- Basic knowledge in developing applications or tests.
- Basic knowledge in handling the external device, which you control remotely via AutomationDesk.

dSPACE provides trainings for AutomationDesk. For more information, refer to https://www.dspace.com/go/trainings.

Symbols

dSPACE user documentation uses the following symbols:

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

Symbol	Description
📖	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

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

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

Special folders

Some software products use the following special folders:

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

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

or

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

Documents folder A standard folder for user-specific documents.

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

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

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

Accessing dSPACE Help and PDF Files

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

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

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

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

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

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

Basics and Instructions

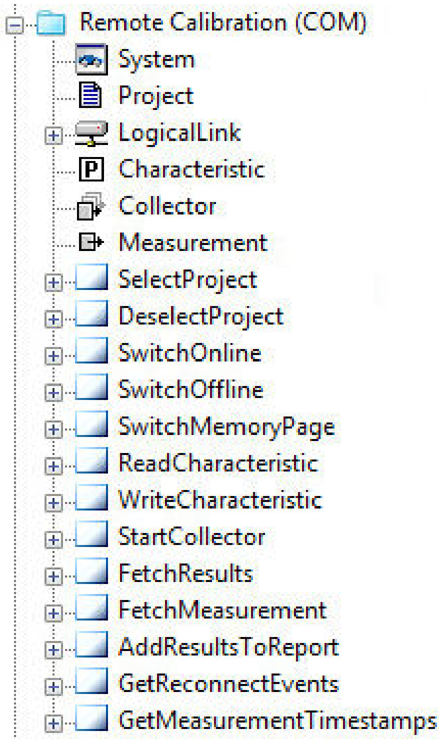
Where to go from here

Information in this section

Overview of the Remote Calibration (COM) Library Elements

Library overview

AutomationDesk can communicate with an ECU via a measurement and calibration system (MC system). The Remote Calibration (COM) library uses the COM/DCOM (ethernet) connection to read and write data from/to the remote system. It is also possible to work on a local machine.



- Remote Calibration (COM)
  - System
  - Project
  - LogicalLink
  - Characteristic
  - Collector
  - Measurement
  - SelectProject
  - DeselectProject
  - SwitchOnline
  - SwitchOffline
  - SwitchMemoryPage
  - ReadCharacteristic
  - WriteCharacteristic
  - StartCollector
  - FetchResults
  - FetchMeasurement
  - AddResultsToReport
  - GetReconnectEvents
  - GetMeasurementTimestamps

System

The System data object specifies connection settings of the MC system. AutomationDesk supports different MC systems. AutomationDesk supports the following MC systems that can be controlled remotely:

- ControlDesk
- INCA
- CANape

For further information, refer to System on page 60.

Project

The Project data object specifies the calibration project that you want to work with. For further information, refer to Project on page 53.

LogicalLink

The LogicalLink data object is the data container for the Collectors and Characteristics data objects. The LogicalLink specifies the connection settings to the physical ECU board. For further information, refer to LogicalLink on page 50.

Characteristics

The Characteristics data object is the data container for Characteristic data objects and specifies the characteristic values in an A2L file. In the ASAM MCD-3 MC object model, Characteristic is the generic term for all ECU variable types that can be calibrated. For further information, refer to Characteristics on page 37.

Collectors

The Collectors data object is the data container for Collector data objects and specifies the values of the measurement variables. For further information, refer to Collectors on page 43.

Measurement

The Measurement data object specifies a new measurement for the current project. For further information, refer to Measurement on page 52.

SelectProject

The SelectProject automation block loads the currently selected project. For further information, refer to SelectProject on page 55.

DeselectProject

The DeselectProject automation block is used to unload a project and stop all measurements within this project. For further information, refer to DeselectProject on page 44.

SwitchOnline

The SwitchOnline automation block is used to switch the connection between the logical link and MC system to online mode. For further information, refer to SwitchOnline on page 59.

SwitchOffline

The SwitchOffline automation block is used to switch the connection between the logical link and MC system to offline mode.For further information, refer to SwitchOffline on page 58.

SwitchMemoryPage

The SwitchMemoryPage automation block is used to switch between the working and reference page of the MC system. For further information, refer to SwitchMemoryPage on page 57.

ReadCharacteristic

The ReadCharacteristic automation block is used to read the characteristics via the remote MC system. For further information, refer to ReadCharacteristic on page 54.

WriteCharacteristic

The WriteCharacteristic automation block is used to write the characteristics via the remote MC system. For further information, refer to WriteCharacteristic on page 62.

StartCollector

The StartCollector automation block is used to start the event-based measurement of measurement variables. For further information, refer to StartCollector on page 56.

FetchResults

The FetchResults automation block is used to fetch the results of the values of the measured variables. For further information, refer to FetchResults on page 47.

FetchMeasurement

The FetchMeasurement automation block is used to start and stop the event-based measurement of measurement variables for a defined period of time. For further information, refer to FetchMeasurement on page 45.

AddResultsToReport

The AddResultsToReport automation block is used to add the results of the measurement to the report. For further information, refer to AddResultsToReport on page 33.

GetReconnectEvents

The GetReconnectEvents automation block is used to get the events that CalDesk is providing after an ECU has been unplugged or connected again. For further information, refer to GetReconnectEvents on page 48.

There is a demo project available for ControlDesk which contains an *AutomaticReconnect* sequence. This sequence shows how you can use the GetReconnectEvents block.

You find the demo project *Remote Calibration (COM) ControlDesk.zip* at `<DocumentsFolder>\Remote Calibration (COM)`.

GetMeasurementTimestamps

The GetMeasurementTimestamps automation block is used to get the start and stop time of a collector. For further information, refer to GetMeasurementTimestamps on page 49.

There is a demo project available for ControlDesk that contains an *AutomaticReconnect* sequence. This sequence shows how you can use the **GetMeasurementTimestamps** block.

You find the demo project *Remote Calibration (COM) ControlDesk.zip* at `<DocumentsFolder>\Remote Calibration (COM)`.

Example of a Remote Calibration (COM) Sequence

Introduction

The following automation sequence shows you a program to access an MC system via the COM /DCOM interface. Measured data is fetched via the **FetchResults** block. The following blocks are used in the automation test sequence:

- SelectProject

  The desired project is selected and the connection to the MC system is set up.
- SwitchOnline

  The logical link to the MC system is switched to online mode. It is possible to either up- or download calibration data from / to the MC system.
- StartCollector

  The measurement is started. The measurement is performed in time-stamp mode. In time-stamp mode, the measurement is performed from one predefined time-stamp to the other.

  The **Collector** data object specifies the data acquisition rate and the number and names of measurement variables
- FetchResults

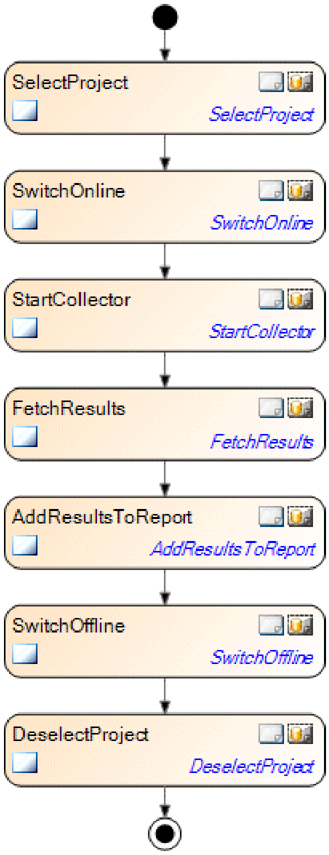
  The results of the current measurement. The values of the measurement variables are collected. After fetching the results, the result buffer is emptied.
- AddResultsToReport

  The results of the current measurement are added to the report. After referencing a data object, you can double-click the **AddResultsToReport** block to open the **Adaptor** dialog. In this dialog you can specify the signals and plots to be included.
- SwitchOffline

  The currently active logical link to the MC system is switched to offline mode.

- DeselectProject

  The project is deselected. The opened logical link is closed and the connection to the MC system is interrupted.



Result

When you execute the sequence, AutomationDesk collects all measured data from the MC system. The data acquisition rate and the number and names of measurement variables are specified and the online mode is set. The current values of a list of measured variables are read from the MC system.

Demo projects

Further AutomationDesk demo projects can be found at `<DocumentsFolder>\Remote Calibration (COM)`.

The demo folder for the Remote Calibration (COM) library contains the following files providing demo projects for the calibration tools ControlDesk, Vector CANape and ETAS INCA.

ControlDesk For working with the demo project `Remote Calibration(COM) ControlDesk.zip`, import `Remote Calibration(COM) ControlDesk.zip` in AutomationDesk.

The CalDemo ECU application (`CalDemo.exe`) is automatically started.

For further information about the ControlDesk-specific instructions, refer to the ControlDesk documentation.

Vector CANape For working with the `Remote Calibration(COM) CANape.zip` demo project, start the demo device *CCPsim*, and import the ZIP file in AutomationDesk.

ETAS INCA For working with the `Remote Calibration(COM) INCA.zip` demo project, import the ZIP file in AutomationDesk.

Related topics

HowTos

How to Create The Required Data Objects For a Calibration Task

Objective

Before you can start setting up a sequence for a measurement and calibration task, you have to create all required data objects for your automated calibration task in the Project Manager. The data objects are created hierarchically. It is necessary to reference one automation block to a specific data object.

Method

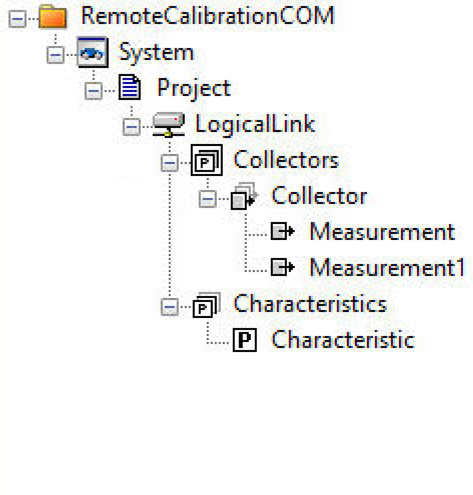
To create the required data objects for a calibration task

1. Drag a System data object from the Library Browser to your project in the Project Manager.
2. Drag a Project data object to the System data object.
3. Drag a LogicalLink data object to the Project data object.

   The LogicalLink data object contains the Collectors and Characteristics data containers for the Collector and Characteristic data objects.
4. Drag the required number of Collector and Characteristic data objects from the Library Browser to the Collectors and Characteristics data containers in the Project Manager.
5. Drag the required number of Measurement data objects to the Collectors data object in the Project Manager.

Result

All required data objects to perform a calibration task were added to the automation project in a hierarchical order in the Project Manager.



- RemoteCalibrationCOM
  - System
    - Project
      - LogicalLink
        - Collectors
          - Collector
            - Measurement
            - Measurement1
        - Characteristics
          - Characteristic

Next step

After you add all data objects to the Project Manager, you can proceed with How to Parameterize Data Objects to Access an MC System on page 14.

Related topics

HowTos

How to Parameterize Data Objects to Access an MC System........ 14

Examples

Example of a Remote Calibration (COM) Sequence........ 11

References

Characteristic........ 34
Collector........ 38
Measurement........ 52
Project........ 53
System........ 60

How to Parameterize Data Objects to Access an MC System

Objective

To perform a measurement and calibration task, you have to parameterize basic data objects. The variables of the MC system can only be read and displayed if a connection to the MC system was established successfully. You can change and customize the values of the basic data objects in order to provide a customized work frame for your automation task.

Note

If you are connected to an MC system you can download all parameters. Nevertheless it is possible to parameterize the data objects without connection. In this case, you can enter the parameters as a string for each data object.

Precondition

The following data objects must have been created:

- System
- Project
- LogicalLink

Method

To parameterize data objects to access an MC system

1. Double-click the **System** data object in the Project Manager and select the appropriate interface name from the drop-down list and enter a host (IP address or computer name).
2. Click **Connect** if you want to connect to the MC system.
3. Double-click on the Project data object. The **Project** dialog is opened. Enter a name for the calibration object or select a project from the drop-down list (available projects are listed only, if you are connected to the MC system).
4. Double-click on the **LogicalLink** data object to open the **LogicalLink** dialog and perform the required logical link settings. Select the appropriate logical link name and name of the binary file for your MC system. All available variables are read from the MC system and represented in the table of the dialog. The variables, variable types and descriptions are displayed in the same order as in ControlDesk.

Result

The **System**, **Project** and **LogicalLink** data objects were parameterized.

Next step

After you parameterized the System, Project and LogicalLink data objects, you can proceed with How to Parameterize Collector Data Objects on page 16 or How to Parameterize Characteristic Data Objects on page 17.

Related topics

HowTos

How to Create The Required Data Objects For a Calibration Task.......... 13

Examples

Example of a Remote Calibration (COM) Sequence.......... 11

References

LogicalLink.......... 50
Project.......... 53
System.......... 60

How to Parameterize Collector Data Objects

Objective

The Collector data object is used to configure and perform measurements and recordings. The values are collected with a common rate over a certain period of time. If you want to use different sample rates, you have to add further Collector data objects.

Preconditions

Your project must contain a Collectors data object as a data container for the subordinated Collector data objects.

Method

To parameterize a Collector data object

1. Double-click the Collector data object you want to parameterize. The Collector dialog opens.
2. Select the storage type for the Collector data object. The default value as specified in the ASAM MCD-3 MC specification is set to eST_AUSY. You can parameterize the Collector data object even if you are not connected to an MC system.
3. Select the buffer rate for the Collector data object. The buffer rate can only be used, if an event-based measurement was defined.
4. Enter an appropriate value for the downsampling rate. The value must be entered as integer. The default value is 10.
5. Enter the number of samples. The value must be entered as integer. The default value is 10.
6. Enter the buffer size for the Collector data object. The value must be entered as integer. The default value is 100.

7 Select the representation type according to the ASAM MCD-3 MC standard. The default value is eRT_PHYSICAL. Possible values are eRT_PHYSICAL and eRT_ECU.

Result

The selected Collector data object was parameterized.

Next step

When you have parameterized the Collector data object, you can proceed with How to Parameterize Characteristic Data Objects on page 17.

Related topics

HowTos

How to Parameterize Characteristic Data Objects.. 17

Examples

Example of a Remote Calibration (COM) Sequence.. 11

References

Collector.. 38

How to Parameterize Characteristic Data Objects

Objective

The Characteristics data object is the data container for the Characteristic data objects of an A2L or DBC file. You can define the Characteristics only in the corresponding A2L file. The measurement variables can be defined in both, A2L and DBC file.

Preconditions

Your project must contain a Characteristics data object as a data container for the subordinated Character data objects.

Method

To parameterize a Characteristic data object

1 Double-click the Characteristic data object you want to parameterize. The Characteristic dialog opens.

2 Select the characteristic type for the Characteristic data object. Possible values are Scalar, Curve and Map. Depending on the selected type, the read and write parameters vary.

3 Enter a name for the Characteristic data object.

4. Select the representation type according to the ASAM MCD-3 MC specification. The default value is set to eRT_PHYSICAL. Possible values are eRT_PHYSICAL and eRT_ECU.
5. Depending on the selected characteristic type, enter the start and stop index values for the x-axis and y-axis.
6. Select a value type for the Characteristic data object according to the ASAM MCD-3 MC specification. Possible values are:
   - eVT_VAL (default value)
   - eVT_OFFSET_NEG
   - eVT_OFFSET_POS
   - eVT_CONST

Result

The selected Characteristic data object was parameterized.

Next step

When you have parameterized the Characteristic data objects, you can proceed with How to Build Basic Sequences for Calibration and Measurement Tasks on page 18.

Related topics

HowTos

How to Parameterize Collector Data Objects... 16

Examples

Example of a Remote Calibration (COM) Sequence... 11

References

Characteristic.. 34

How to Build Basic Sequences for Calibration and Measurement Tasks

Objective

Within AutomationDesk you can create test automation sequences for your calibration and measurement task in the Sequence Builder.

Precondition

To build a basic sequence for a calibration and measurement task, you must already have created all necessary data objects in the Project Manager.

Method

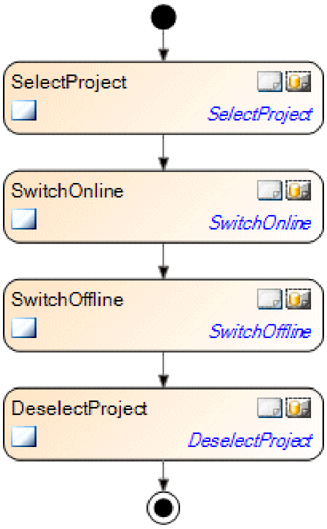
To build a basic sequence for a calibration task

1 Right-click the AutomationDesk project and choose New Sequence to add a sequence to the Project Manager.

2 Double-click the sequence element in the Project Manager to open the Sequence Builder.

3 Drag the following automation blocks from the Library Browser to the Sequence Builder:

- SelectProject
- SwitchOnline
- SwitchOffline
- DeselectProject



4 In the Sequence Builder, choose **View Data Objects** from the **SelectProject** automation block's context menu.

5 Click the **Reference Name** cell to set a reference to the desired Project data object in the Data Object Selector.

6 In the Sequence Builder, choose **View Data Objects** from the **SwitchOnline** automation block's context menu.

7 Select the **LogicalLink** data object in the Data Object Editor and click the **Reference Name** cell to set a reference to the desired device in the Data Object Selector.

8 Select the **CalibrationMemory** cell in the Data Object Editor and set the value to “Download”.

9 In the Sequence Builder, choose **View Data Objects** from the **SwitchOffline** automation block's context menu. Select the same device as for the **SwitchOnline** automation block in the Data Object Selector.

10 In the Sequence Builder, choose **View Data Objects** from the **DeselectProject** automation block's context menu. Select the same **Project** data object as referenced data object in the Data Object Selector.

Result

An automation sequence was created that connects to the MC system and performs a download of all available parameters. Within this sequence, the **LogicalLink** references the data objects that hold the information on the device to be connected to.

Next step

After you have built an automation sequence and referenced the corresponding data objects, you can proceed with How to Build Measurement Sequences Using FetchResults Blocks on page 20.

Related topics

HowTos

How to Create The Required Data Objects For a Calibration Task... 13

Examples

Example of a Remote Calibration (COM) Sequence.. 11

References

SelectProject.. 55
SwitchOffline.. 58
SwitchOnline.. 59

How to Build Measurement Sequences Using FetchResults Blocks

Objective

Using the Remote Calibration (COM) library in AutomationDesk there are two ways of performing a measurement task. This method describes how to measure data using the **FetchResults** block, for the second method, refer to How to Build Measurement Sequences Using FetchMeasurement Blocks on page 23.

Precondition

To build a sequence for a measurement task, you must already have created all necessary data objects in the Project Manager. Furthermore you must create a basic sequence as the framework for your task, refer to How to Build Basic Sequences for Calibration and Measurement Tasks on page 18.

Restrictions

It is not allowed to use the StartCollector block within a Parallel block.

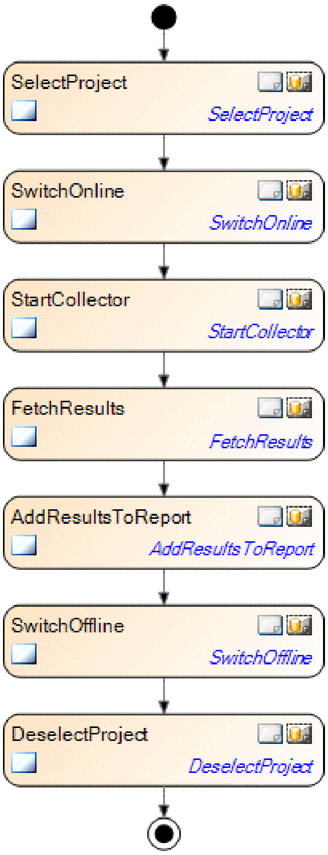
To start several measurements simultaneously, you can add as many Collector data objects to a single StartCollector block as required. You can add a Collector data object by dragging it from the Library Browser to the block. For each specified measurement, you must add a related FetchResults block to your sequence.

If your sequence contains many StartCollector blocks before a FetchResults block, only the first StartCollector block is executed.

Method

To build a measurement sequence using the FetchResults block

1 Drag the following automation blocks from the Library Browser to the basic sequence in the Sequence Builder:
   - StartCollector
   - FetchResults
   - AddResultsToReport (optional)



2 In the Sequence Builder, choose **View Data Objects** from the **StartCollector** automation block's context menu.

3 Click the **Reference Name** cell in the Data Object Editor and select the collector to be referenced in the Data Object Selector.

4 In the Sequence Builder, choose **View Data Objects** from the **FetchResults** automation block's context menu. Set the references and values via the Data Object Editor for the following data objects:
   - CollectorStop

     Set a boolean value (true or false) that determines whether the measurement is stopped or not.
   - WaitTime

     Determines the time to wait for results in seconds.
   - OverflowCount

     Counts the number of overflows that occurred during the measurement.
   - Results

     Determines the results to be included.

5 The **AddResultsToReport** block is optional and should be used to add the results of a Collector data object to the report. The **AddResultsToReport** block must be parameterized before you can execute the sequence. For information on how to parameterize the **AddResultsToReport** block, refer to How to Add Results of a Calibration Task to Reports on page 28.

Result

An automation sequence was created that measures data from the MC system via the **FetchResults** automation block. If you use this method to measure data, it is mandatory that you work with a **StartCollector** block.

Next step

After you have built an automation sequence to measure data via the **FetchResult** block, you can proceed with How to Build Measurement Sequences Using FetchMeasurement Blocks on page 23.

Related topics

HowTos

How to Add Results of a Calibration Task to Reports	28
How to Build Basic Sequences for Calibration and Measurement Tasks	18
How to Build Measurement Sequences Using FetchMeasurement Blocks	23
How to Create The Required Data Objects For a Calibration Task	13

Examples

Example of a Remote Calibration (COM) Sequence	11

References

FetchResults	47
StartCollector	56
SwitchOffline	58
SwitchOnline	59

How to Build Measurement Sequences Using FetchMeasurement Blocks

Objective

An alternative of performing a measurement task with a FetchResults block is to use the **FetchMeasurement** block If you use this method, the measurement is performed during a certain period of time and you do not need a **StopCollector** block to stop the measurement.

Precondition

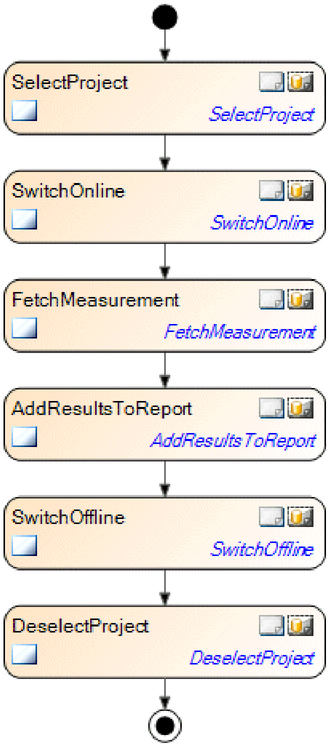
To build a sequence for a measurement task, you must already have created all necessary data objects in the Project Manager. Furthermore you must create a basic sequence as the framework for your task, refer to How to Build Basic Sequences for Calibration and Measurement Tasks on page 18.

Method

To build a measurement sequence using the FetchMeasurement block

1 Drag the following automation blocks from the Library Browser to the basic sequence in the Sequence Builder:

- FetchMeasurement
- AddResultsToReport (optional)



2 In the Sequence Builder, choose **View Data Objects** from the **FetchMeasurement** automation block's context menu. Set the references and values via the Data Object Editor for the following data objects:

- MeasurementTime

  Determines the absolute time for the measurement in seconds.
- OverflowCount

  Counts the number of overflows that occurred during the measurement.
- Results

  Determines the results to be included.

3 The **AddResultsToReport** block is optional and should be used to add the results of a Collector data object to the report. The **AddResultsToReport** block must be parameterized before you can execute the sequence. For information on how to parameterize the **AddResultsToReport** block, refer to How to Add Results of a Calibration Task to Reports on page 28.

Result

An automation sequence was created that measures data from the MC system via the **FetchMeasurement** automation block. If you use this method to measure data, you do not need a **StartCollector** and **StopCollector** data object.

Next step

After you have built an automation sequence to measure data via the **FetchResult** block, you can proceed with How to Add Results of a Calibration Task to Reports on page 28.

Related topics

HowTos

How to Add Results of a Calibration Task to Reports.. 28
How to Build Basic Sequences for Calibration and Measurement Tasks.. 18
How to Create The Required Data Objects For a Calibration Task.. 13

Examples

Example of a Remote Calibration (COM) Sequence.. 11

References

AddResultsToReport.. 33
FetchMeasurement.. 45
SwitchOffline.. 58
SwitchOnline.. 59

How to Build Sequences to Read Characteristics

Objective

Within AutomationDesk you can create test automation sequences to read the characteristics via the MC system. If you want to add the results of the Collector data object to a report, you have to use the automation blocks from the **Report** library.

Precondition

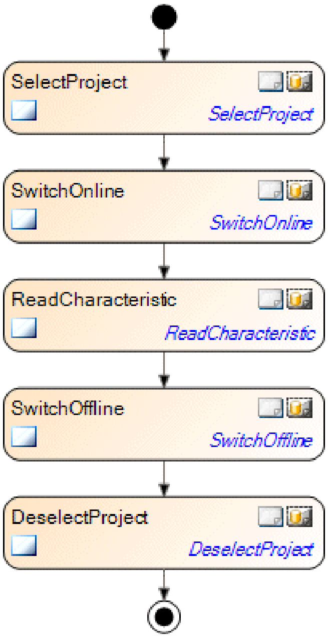
To build a sequence to read the characteristics, you must already have created all necessary data objects in the Project Manager. Furthermore you must create a basic sequence as the framework for your task, refer to How to Build Basic Sequences for Calibration and Measurement Tasks on page 18.

Method

To build a sequence to read characteristics

1 Drag the following automation blocks from the Library Browser to the basic sequence in the Sequence Builder:

- ReadCharacteristic



2 In the Sequence Builder, choose **View Data Objects** from the **ReadCharacteristics** automation block's context menu.

3 Select the **Characteristic** data object in the Data Object Editor and click the **Reference Name** cell to set a reference to the desired device in the Data Object Selector.

4 Select the **Results** cell in the Data Object Editor and click the **Reference Name** cell to set a reference to the desired data object in the Data Object Selector.

Result

An automation sequence was created that reads the characteristics via the MC system.

Next step

After you have built an automation sequence to read the characteristics, you can proceed with How to Add Results of a Calibration Task to Reports on page 28.

Related topics

HowTos

How to Build Basic Sequences for Calibration and Measurement Tasks.................................... 18
How to Build Sequences to Write Characteristics.. 26
How to Create The Required Data Objects For a Calibration Task... 13

Examples

Example of a Remote Calibration (COM) Sequence... 11

References

ReadCharacteristic.. 54

How to Build Sequences to Write Characteristics

Objective

Within AutomationDesk you can create test automation sequences to write the characteristics via the MC system.

Precondition

To build a sequence to write the characteristics, you must already have created all necessary data objects in the Project Manager. Furthermore you must create a basic sequence as the framework for your task, refer to How to Build Basic Sequences for Calibration and Measurement Tasks on page 18.

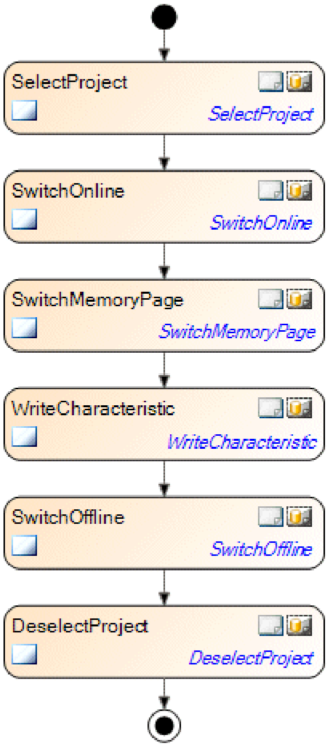
Method

To build a sequence to write characteristics

1 Drag the following automation blocks subordinated to the SelectProject block from the Library Browser to the basic sequence in the Sequence Builder:

- SwitchMemoryPage

- WriteCharacteristic



2 In the Sequence Builder, choose **View Data Objects** from the **SwitchMemoryPage** automation block's context menu.

3 Select the LogicalLink data object in the Data Object Editor and click the **Reference Name** cell to set a reference to the desired device in the Data Object Selector.

4 Select the **SwitchTo** cell in the Data Object Editor and select either **WorkingPage** or **ReferencePage** from the drop-down list in the **Value** cell. Values can only be written to the working page.

5 In the Sequence Builder, choose **View Data Objects** from the **WriteCharacteristic** automation block's context menu.

6 Select the Characteristic data object in the Data Object Editor and click the **Reference Name** cell to set a reference to the desired device in the Data Object Selector.

7 Select the **WriteValue** cell in the Data Object Editor and click the **Reference Name** cell to set a reference to the desired data object in the Data Object Selector.

Result

An automation sequence was created that writes the characteristics via the MC system.

Next step

After you have built an automation sequence to write the characteristics, you can proceed with How to Add Results of a Calibration Task to Reports on page 28.

Related topics

HowTos

How to Build Basic Sequences for Calibration and Measurement Tasks 18
How to Build Sequences to Read Characteristics 25
How to Create The Required Data Objects For a Calibration Task 13

Examples

Example of a Remote Calibration (COM) Sequence 11

References

WriteCharacteristic 62

How to Add Results of a Calibration Task to Reports

Objective

You can fetch the results of a calibration task within a sequence and add the results to a report by using an AddResultsToReport block.

Precondition

A sequence must exist that fetches the required data from the MC system.

Method

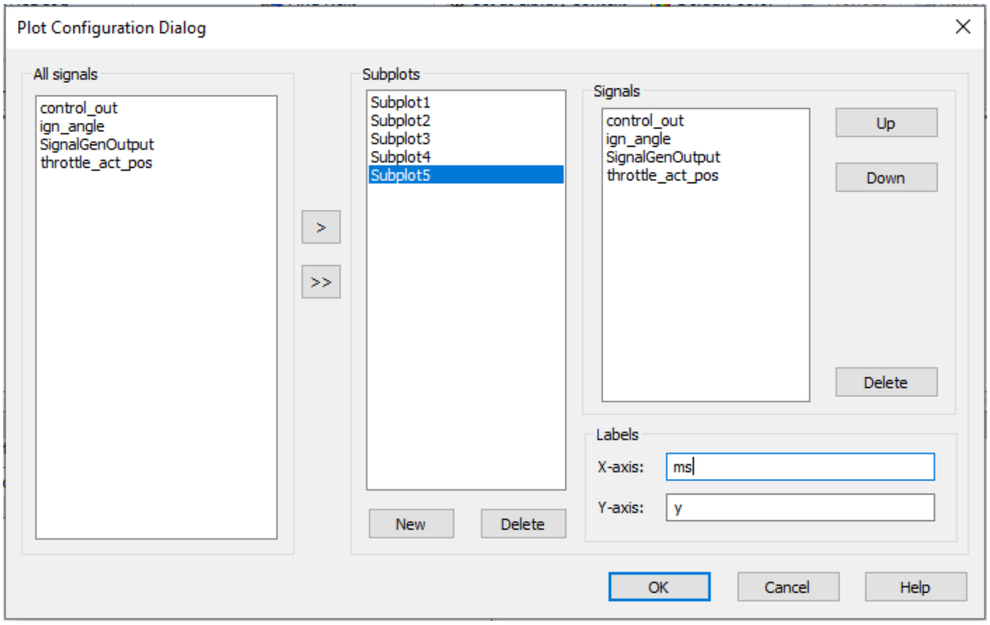
To add a result to a report

1. Drag the AddResultsToReport block to the sequence in the Sequence Builder.
2. Open the Data Object Editor to set a reference to a specified Collector data object.
3. Double-click the AddResultsToReport automation block to open the Adaptor dialog.

   Note

   The Adaptor dialog opens only, if the Collector data object of the block is parameterized.

4. In the Adaptor dialog, select the desired signals and subplots that you want to add to the report. You can arrange the signals to be included via the Up and Down buttons in the Adaptor dialog.



All signals: control_out, ign_angle, SignalGenOutput, throttle_act_pos

Subplots: Subplot1, Subplot2, Subplot3, Subplot4, Subplot5

Signals: control_out, ign_angle, SignalGenOutput, throttle_act_pos

X-axis: ms

Y-axis: y

5 As an option, you can enter label names for the x-axis and y-axis.

6 Reference the desired data object that shall deliver the results for the report in the Data Object Selector.

Result

If you execute the sequence, the AddResultsToReport block generates a plot with the specified plot settings in the report.

Related topics

HowTos

How to Build Basic Sequences for Calibration and Measurement Tasks.. 18
How to Create The Required Data Objects For a Calibration Task.. 13

Examples

Example of a Remote Calibration (COM) Sequence.. 11

References

AddResultsToReport... 33

Reference Information

Where to go from here

Information in this section

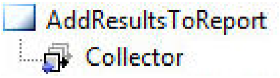
Automation Blocks

Where to go from here

Information in this section

AddResultsToReport

Graphical representation



AddResultsToReport
Collector

Purpose

To add the results of a Collector data object to the report.

Description

The AddResultsToReport block allows to add the results of the measured signals to the report. The measurement results are available after executing a FetchMeasurement block or a FetchResults block. You can select the results of the measured signals that you want to add to the report, define plots and subplots and state labels for the x-axis and y-axis.

Note

Do not specify more than 7 subplots within one block if you want to use PDF reports.

Data objects

This automation block provides the following data object:

Name	In / Out	Type	Default Value	Description
Collector	In	Extended Data Object	None	Specifies the collector from which the results are to be used.The Collector data object must be referenced.

Related topics

HowTos

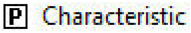
How to Add Results of a Calibration Task to Reports.. 28

References

FetchMeasurement.. 45
FetchResults.. 47

Characteristic

Graphical representation



Purpose

To represent and edit the characteristic values of an A2L file.

Description

The Characteristic data object is subordinated below the LogicalLink data object, that belongs to each Characteristics data container. You can view and customize the characteristic values of an A2L file. Depending on the characteristic type, different internal data types and settings are available. The following characteristic types are supported.

- Scalar
- Curve
- Map

If you write CurveCharacteristics or MapCharacteristics, you can specify a start and a stop index to specify the first and the last value to be changed in a range of values. In these cases the start index begins at 0 and the stop index is the first position that is not changed. A stop index of -1 means that all the values of a range can be changed. The following parameters are specified:

Name	In / Out	Type	Default Value	Description
CharacteristicType	In	String	" "	Represents the characteristic type. Possible methods: ▪ Scalar ▪ Curve ▪ Map
CharacteristicName	In / Out	String	" "	Represents the selected characteristic name as selected from the A2L file
RepresentationType	In / Out	Int	eRT_PHYSICAL	You can specify the conversion mode for each value you read or write by selecting a representation type. Possible methods:

Name	In / Out	Type	Default Value	Description
				▪ eRT_ECU Source value in its original format on the hardware ▪ eRT_PHYSICAL Value in a converted form
ValueType	In / Out	Int	eVT_VAL	States the value type as used from the object model. This data object is available for WriteCharacteristic blocks only. Possible methods: ▪ eVT_CONST One value (a constant) is used ▪ eVT_OFFSET_NEG A negative offset value is used. ▪ eVT_OFFSET_POS A positive offset value is used. ▪ eVT_VAL (default value) n values are used.
XStartIndex	In	Int	0	Start index for the x-axis.
XStopIndex	In	Int	-1	Stop index for the x-axis.
YStartIndex	In	Int	0	Start index for the y-axis.
YStopIndex	In	Int	-1	Stop index for the y-axis.
Results	In	List	[]	Depends on the selected type. This data object is available for ReadCharacteristic blocks only. The following formats are predefined: ▪ Scalar [value] ▪ Curve [[XAxisValue], [Value]] ▪ Map [[XAxisValue], [YAxisValue], [Value]]
WriteValue	In / Out	Variant	None	Depends on the selected type. This data object is available for WriteCharacteristic blocks only. The following formats are predefined: ▪ Scalar: [Value] ▪ Curve: [[XAxisValue], [Value] ▪ Map: [[XAxisValue], [YAxisValue], [Value]]

Access via Exec block

This data object provides some methods that you can access via Python script in an Exec block. The following methods are available:

Method	Description
GetCharacteristicName()	To get the specified characteristic name.
SetCharacteristicName(CharacteristicName)	To set the characteristic name.
GetCharacteristicType()	To get the specified characteristic type.

Method	Description
SetCharacteristicType(CharacteristicType)	To set the characteristic type. The value that you want to set, must be available in the list of characteristic types (see GetAvailableCharacteristicTypeNames() method).
GetAvailableCharacteristicTypeNames()	To get a list of strings providing the characteristic types from which you can choose one.
GetRepresentationType()	To get the specified representation type.
SetRepresentationType(RepresentationType)	To set the representation type. The value that you want to set, must be available in the list of representation types (see GetAvailableRepresentationTypeNames() method).
GetAvailableRepresentationTypeNames()	To get a list of strings providing the representation types from which you can choose one.
GetXStartIndex()	To get the specified start index for the x-axis.
SetXStartIndex(XStartIndex)	To set start index for the x-axis.
GetXStopIndex()	To get the specified stop index for the x-axis.
SetXStopIndex(XStopIndex)	To set stop index for the x-axis.
GetYStartIndex()	To get the specified start index for the y-axis.
SetYStartIndex(YStartIndex)	To set start index for the y-axis.
GetYStopIndex()	To get the specified stop index for the y-axis.
SetYStopIndex(YStopIndex)	To set stop index for the y-axis.
GetValueType()	To get the specified value type.
SetValueType(ValueType)	To set the value type. The value that you want to set, must be available in the list of value types (see GetAvailableValueTypeNames() method).
GetAvailableValueTypeNames()	To get a list of strings providing the value types from which you can choose one.

Example

```
_AD_.MyCharacteristic.SetCharacteristicName('SignalAmplitude')
_AD_.MyCharacteristic.SetCharacteristicType('Scalar')
_AD_.MyCharacteristic.SetRepresentationType('eRT_PHYSICAL')
_AD_.MyCharacteristic.SetXStartIndex(0)
_AD_.MyCharacteristic.SetXStopIndex(-1)
_AD_.MyCharacteristic.SetYStartIndex(0)
_AD_.MyCharacteristic.SetYStopIndex(-1)
_AD_.MyCharacteristic.SetValueType('eVT_VAL')
```

This example shows you how to access a Characteristic data object that is already referenced to a Characteristic data object from the calibration project's data structure. If you want to access a Characteristic data object directly from the project's data structure, you have to specify the entire path: for example, `_AD_.ControlDesk.XCPOnCAN.XCP.MC3Characteristics.Characteristic1.SetValueType('eVT_VAL')`.

Note

Independently of the specified name for the Characteristics structure element, you have to specify MC3Characteristics as its internal name.

Related topics

HowTos

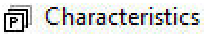
How to Create The Required Data Objects For a Calibration Task.. 13

References

Characteristics.. 37
ReadCharacteristic... 54
WriteCharacteristic... 62

Characteristics

Graphical representation



Characteristics

Purpose

To group Characteristic data objects.

Description

The Characteristics data container, that is subordinated below the LogicalLink data object, is used to group characteristic values of an A2L file. You can add and remove Characteristic data objects via the context menu or via drag and drop. If you want to exchange a characteristic of an A2L file, you can drag and drop the desired characteristic to an existing characteristic. The configuration parameters of the original characteristic in the Characteristics block are retained. The name of the data object is not modified.

Data objects

None

Related topics

HowTos

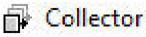
How to Create The Required Data Objects For a Calibration Task... 13

References

Characteristic.. 34
LogicalLink.. 50
ReadCharacteristic.. 54
WriteCharacteristic... 62

Collector

Graphical representation



Collector

Purpose

To collect the values of measurement variables.

Description

The Collector data object is subordinated below a LogicalLink data object that belongs to the Collectors data container. The Collector data object contains the collected values of the measurement variables. You can have more than one Collector data object in one Collectors data container. The values are collected with a common rate over a defined period of time. The Collector data object contains the trigger conditions for start and stop during the data acquisition.

Tip

- Avoiding overflows

  An overflow can occur if the buffer is to small to take new samples. Note the following calculation formula:

  ```
  MinBufferSize = DataAcquisitionTime / (BufferRate · DownSampling)
  ```

  The data acquisition time depends on the calibration tool.
- Minimum measurement time

  The measurement results are not collected from the ring buffer if the measurement time is too short. Note the following calculation formula:

  ```
  MinMeasurementTime = BufferRate · DownSampling · NumberOfSamples
  ```

The following parameters are specified:

Name	In / Out	Type	Default Value	Description
StorageType	In / Out	Int	eST_AUSY	Specifies the storage destination for the collector results. It can be the internal data storage of the calibration tool (eST_AUSY) or a file (eST_FILE).
FileName	In / Out	String	" "	Specifies the file to store the collector results in, e.g., `C:\CalibrationProjects\results.mf4`. It is only required if you specified eST_FILE as the storage type. AutomationDesk supports any file types of the connected calibration tool, for example, MAT, CSV, and MF4. See also Basics on Automating ControlDesk's Measurement and Recording Features (ControlDesk MCD-3 Automation 📖).
NumberOfSamples	In / Out	Int	10	States the number of samples to be collected. The measurement values of the collector are only returned if the specified number of samples has been reached. If you read the result before the collector is completed, you will get an empty dictionary. If the buffer of the calibration server contains as many samples as specified, a result event is sent.
DownSampling	In / Out	Int	10	States the downsampling rate of the values to be collected.
BufferSize	In / Out	Int	100	Specifies the number of sample times to be recorded in the measurement buffer of the Collector. As the buffer is a ring buffer, earlier values are overwritten by later values when the buffer capacity is exceeded. The buffer size must be greater than the number of samples to allow reading the sampled values before they are overwritten with the next measurement values.
BufferRate	In / Out	String	" "	States the buffer rate for the values to be collected and influences the measurement on the device (ECU). For example, a buffer rate of 5 ms and a downsampling value of 10 results in writing measurement values after 50 ms.
RepresentationType	In / Out	Int	eRT_PHYSICAL	States the representation type as used by the object model. The possible values are: ▪ eRT_ECU Source value in its original format on the hardware ▪ eRT_PHYSICAL Value in a converted form
OverflowCount	Out	Int	0	Internal parameter that can be read by a FetchResults or FetchMeasurement block. It indicates the number of buffer overflows occurred at the collector. Each time, an overflow occurs, the value is increased by 1.

Name	In / Out	Type	Default Value	Description
Results	Out	Dictionary	{}	Internal parameter that can be read by a FetchResults or FetchMeasurement block. The results are represented in the following syntax: {MeasurementName1:[[Values]],MeasurementName2:[[Values]], "xAxis":[Values] } The xAxis determines the time stamps for the collector.

Note

Automated recording with ControlDesk

If you store a recording via the MC system, some of the recording settings in your automation script may be ignored. For example, ControlDesk ignores downsampling and representation type settings, because it does not perform DAQ downsampling and it defines the representation type according to the measurement data file type. If the file type supports the storing of conversion formulas (for example, MF4 files), source values are stored. If the file type does not support the storing of conversion formulas (for example, MAT and CSV files), physical values are stored.

Access via Exec block

This data object provides some methods that you can access via Python script in an Exec block. The following methods are available:

Method	Description
GetStorageType()	To get the specified storage type.
SetStorageType(StorageType)	To set the storage type. The value that you want to set, must be available in the list of storage type names (see GetAvailableStorageTypeNames() method). The collector must be stopped beforehand.
GetAvailableStorageTypeNames()	To get a list of strings providing the storage types from which you can choose one.
GetFileName()	To get the name of the specified file.
SetFileName(FileName)	To set the name of the file. The collector must be stopped beforehand.
GetAbsolutePath()	To get the absolute path of the specified file.
SetAbsolutePath(AbsolutePath)	To set the absolute path of the specified file. The collector must be stopped beforehand.
GetBufferRate()	To get the specified buffer rate.
SetBufferRate(BufferRate)	To set the buffer rate. The value that you want to set, must be available in the list of buffer rate names (see GetAvailableBufferRateNames() method). The collector must be stopped beforehand.

Method	Description
GetAvailableBufferRateNames()	To get a list of strings providing the buffer rates from which you can choose one. The project must be selected and the LogicalLinkName must be set beforehand.
GetDownSampling()	To get the specified downsampling rate.
SetDownSampling(DownSampling)	To set the downsampling rate. The collector must be stopped beforehand.
GetNumberOfSamples()	To get the specified number of samples.
SetNumberOfSamples(NumberOfSamples)	To set the number of samples. The collector must be stopped beforehand.
GetBufferSize()	To get the specified buffer size.
SetBufferSize(BufferSize)	To set the buffer size. The collector must be stopped beforehand.
GetRepresentationType()	To get the specified representation type.
SetRepresentationType(RepresentationType)	To set the representation type. The value that you want to set, must be available in the list of representation type names (see GetAvailableRepresentationTypeNames() method). The collector must be stopped beforehand.
GetAvailableRepresentationTypeNames()	To get a list of strings providing the representation types from which you can choose one.
Activate()	To activate a collector (according to ASAM MCD3 standard). Precondition: The collector exists in the selected project and is in the state CONFIGURED.
Change()	To change a collector (according to ASAM MCD3 standard). Precondition: The collector exists in the selected project and is in the state CONFIGURED or ACTIVATED.
Check()	To check a collector (according to ASAM MCD3 standard). Preconditions: If the collector does not exist in the selected project it will be created and configured. An already existing collector must be in the state CREATED.
Results,OverflowCount=\ DynamicFetchMeasurement(MeasurementTime)	To start the FetchMeasurement in the context of the other dynamic collector features. The measurement is finished at the end of the defined MeasurementTime. Preconditions: ▪ The value of MeasurementTime must be of type float or an integer value and it must be positiv. ▪ The collector exists in the selected project and is in the state ACTIVATED.
Results,OverflowCount=\ DynamicFetchResults(WaitTime,CollectorStop)	To fetch the Results in the context of the other dynamic collector features. The measurement is continued beyond the WaitTime. If CollectorStop comes true the measurement will be stopped. Preconditions: ▪ The value of WaitTime must be of type float or an integer value and it must be positiv. ▪ The value of CollectorStop must be of type boolean or an integer value. If it is an integer value it must be 0 or positiv.

Method	Description
	▪ The collector exists in the selected project and is in the state STARTED.
DynamicSetMeasurements(Measurements)	To set the updated Measurements list during run time. Measurements must be a plain list filled with strings. Precondition: If the collector does not exist in the selected project it will be created and configured. An already existing collector must be in the state CREATED.
DynamicStartCollector()	To start the collector in the context of the other dynamic collector features. Precondition: The collector exists in the selected project and is in the state ACTIVATED.
DynamicUpdateConfiguration()	To update the configuration of the Collector during run time. Precondition: If the collector does not exist in the selected project it will be created and configured. An already existing collector must be in the state CREATED.

Note

- Do not use the dynamic collector methods (DynamicFetchMeasurement, DynamicFetchResults, etc.) with the Remote Calibration (COM) library automation blocks FetchMeasurement, StartCollector and FetchResults together in one project. It could cause conflicts.
- You cannot use the internal data storage for the collector results if you work with a 32-bit INCA MC system, because it does not provide 64-bit marshalling.

Note

To plot results in a report when using the dynamic collector methods (DynamicFetchMeasurement, DynamicFetchResults) in your project, use the Add2DPlot (Report Library) or AddSignalToReport (Evaluation Library) automation block instead of the AddResultsToReport (Remote Calibration (COM)) automation block.

For further information, refer to `<DocumentsFolder>\Remote Calibration (COM)\DynamicCollectorExamples.zip`.

Example

```
_AD_.MyCollector.SetStorageType('eST_FILE')
_AD_.MyCollector.SetFileName('D:\Work\MyCalProject.mf4')
_AD_.MyCollector.SetAbsolutePath(0)
_AD_.MyCollector.SetBufferRate('5ms')
_AD_.MyCollector.SetDownSampling(10)
_AD_.MyCollector.SetNumberOfSamples(10)
_AD_.MyCollector.SetBufferSize(100)
_AD_.MyCollector.SetRepresentationType('eRT_ECU')
_AD_.MyCollector.DynamicUpdateConfiguration()
_AD_.MyCollector.DynamicSetMeasurements(['throttle_act_pos', 'ign_angle'])
_AD_.MyCollector.Check()
_AD_.MyCollector.Activate()
_AD_.ResultDict, _AD_.OverflowCount = \
        _AD_.MyCollector.DynamicFetchMeasurement(3.0)
_AD_.MyCollector.Change()
```

This example shows you how to access a Collector data object that is already referenced to a Collector data object from the calibration project's data structure. If you want to access a Collector data object directly from the project's data structure, you have to specify the entire path: for example, `_AD_.ControlDesk.XCPOnCAN.XCP.MC3Collectors.Collector1.SetStorageType ('eST_FILE')`.

Note

Independently of the specified name for the Collectors structure element, you have to specify MC3Collectors as its internal name.

Related topics

HowTos

How to Create The Required Data Objects For a Calibration Task.. 13

References

Clear Value (AutomationDesk Basic Practices 🕮)
Collectors.. 43
StartCollector.. 56

Collectors

Graphical representation

Collectors

Purpose

To group Collector data objects that collect the values of measurement variables.

Description

The Collectors data container is used to group Collector data objects. The Collector data objects store the values of the measurement variables. You can add and remove Collector data objects via the context menu or via drag and drop.

Data objects

None

Related topics

HowTos

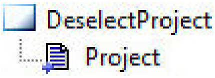
How to Create The Required Data Objects For a Calibration Task... 13

References

Collector.. 38
LogicalLink.. 50
StartCollector... 56

DeselectProject

Graphical representation



DeselectProject
Project

Purpose

To deselect the currently selected project.

Description

After deselecting the currently selected project, all eventually started collectors are stopped. All logical links are destroyed and the connection to the MC system is interrupted.

> **Note**
>
> Within a sequence it is not allowed to use this block parallel to other blocks.

Data objects

This automation block provides the following data object:

Name	In / Out	Type	Default Value	Description
Project	In	Project	None	States the name of the project to be deselected.

Related topics

HowTos

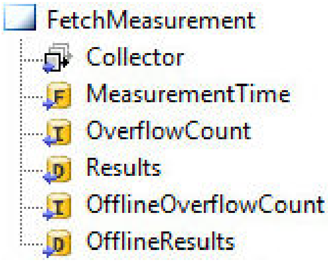
How to Build Basic Sequences for Calibration and Measurement Tasks.. 18

References

Project.. 53
SelectProject.. 55
System... 60

FetchMeasurement

Graphical representation



FetchMeasurement
- Collector
- MeasurementTime
- OverflowCount
- Results
- OfflineOverflowCount
- OfflineResults

Purpose

To start and stop the measurement of measurement values.

Description

The time-based measurement of values is started on execution of this block. Time-based measurement means, that the measurement stops automatically when the specified MeasurementTime duration has been reached. The measurement results can be added to a report by using the AddResultsToReport block.

> **Note**
>
> - Within a sequence it is not allowed to use this block parallel to other blocks.
> - You cannot use the internal data storage for the collector results if you work with a 32-bit INCA MC system, because it does not provide 64-bit marshalling.

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
Collector	In	Extended Data Object	None	Specifies the Collector data object for which the measuring is to be done.The Collector data object must be referenced.
MeasurementTime	In	Float	10.0	Specifies the time of measurement. The elapsed time is acquired via time stamps.
OverflowCount	Out	Int	0	Indicates the number of overflows occurred. Each time, an overflow occurs, the value is increased by 1.
Results	Out	Dictionary	{}	The results are represented in the following syntax: {MeasurementName1:[[Values]],MeasurementName2:[[Values]],"xAxis":[Values] } The xAxis determines the time stamps for the collector.
OfflineOverflowCount	In	Int	0	Lets you specify the value to be used in offline operation mode.
OfflineResults	In	Dictionary	{}	Lets you specify the value to be used in offline operation mode.

Related topics

Basics

Executing Sequences Using Different Operation Modes (AutomationDesk Basic Practices 📖)

HowTos

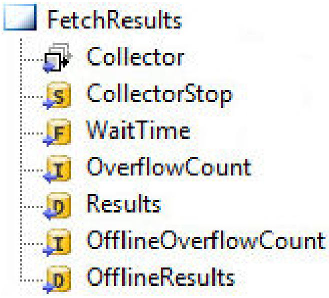
How to Build Measurement Sequences Using FetchMeasurement Blocks.................................. 23

References

FetchResults.. 47
Measurement... 52

FetchResults

Graphical representation



FetchResults
- Collector
- CollectorStop
- WaitTime
- OverflowCount
- Results
- OfflineOverflowCount
- OfflineResults

Purpose

To fetch the available results from the Collector data object.

Description

The FetchResults block must be used to fetch the measurement results if you have started the measurement using a StartCollector block. Stopping the measurement is done by the CollectorStop data object of this block. After fetching the collected results, the result buffer is cleared.

> **Note**
>
> Within a sequence it is not allowed to use this block parallel to other blocks.

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
Collector	In	Extended Data Object	None	The Collector data object is initialized via the StartCollector block. The Collector data object must be referenced.
CollectorStop	In	Boolean	True	Indicates whether the collector shall be stopped or not.
WaitTime	In	Float	10.0	Indicates the maximum time to wait for results.
OverflowCount	Out	Int	0	Indicates the number of overflows occurred at the referenced collector. Each time, an overflow occurs, the value is increased by 1.
Results	Out	Dictionary	{}	The results of the referenced collector are represented in the following syntax: {MeasurementName1:[[Values]],MeasurementName2:[[Values]],"xAxis":[Values] } The xAxis determines the time stamps for the collector
OfflineOverflowCount	In	Int	0	Lets you specify the value to be used in offline operation mode.

Name	In / Out	Type	Default Value	Description
OfflineResults	In	Dictionary	{}	Lets you specifiy the value to be used in offline operation mode.

Related topics

Basics

Executing Sequences Using Different Operation Modes (AutomationDesk Basic Practices 📖)

HowTos

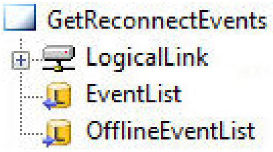
How to Build Measurement Sequences Using FetchResults Blocks... 20

References

AddResultsToReport... 33
Collector... 38
FetchMeasurement.. 45

GetReconnectEvents

Graphical representation



Purpose

To get the list of reconnect-events for the specified LogicalLink data object from ControlDesk.

Description

The dSPACE calibration tool ControlDesk detects when an ECU is unplugged because of a connection failure or because it was switched off. The tool automatically tries to reconnect to the ECU. Its events report changes from the unplugged state to the connected state and vice versa.

The events collected with the LogicalLink data object can be read out via the GetReconnectEvents automation block. The EventList data object contains all the events which occurred in online mode.

Note

The events are only collected as long as the LogicalLink is switched to **Online** (see SwitchOnline on page 59) and while an AutomationDesk execution is running. If a new execution is started when the LogicalLink data object is already **Online**, events are collected again the first time the LogicalLink is accessed. This usually is the case if you use any automation block from the Remote Calibration (COM) library.

An event time stamp indicates when the event arrived in AutomationDesk. This can differ from the actual moment of the ECU connection loss.

Data objects

This automation block provides the following data objects:

Name	In / Out	Data Type	Default Value	Description
LogicalLink	In	LogicalLink	None	Specifies the logical link to the ECU.
EventList	Out	List	[]	Returns a list of tuples containing the time stamp and an event identifier for each ControlDesk event. The time stamp has the standard Python format (seconds since 1970). The identifier can be either **Unplugged** or **Reconnected**.
OfflineEventList	In	List	[]	Lets you specify the value to be used in offline operation mode.

Related topics

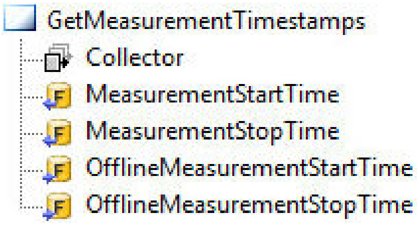
Basics

Executing Sequences Using Different Operation Modes (AutomationDesk Basic Practices 🕮)

References

GetMeasurementTimestamps

Graphical representation



GetMeasurementTimestamps
- Collector
- MeasurementStartTime
- MeasurementStopTime
- OfflineMeasurementStartTime
- OfflineMeasurementStopTime

Purpose

To get the start and stop time of a collector.

Description

The start time and the stop time can be used to calculate the duration of a measurement. You can also use the time stamps to synchronize the measurements with the ControlDesk events available via the GetReconnectEvents automation block (see GetReconnectEvents on page 48).

Data objects

This automation block provides the following data objects:

Name	In / Out	Data Type	Default Value	Description
Collector	In	Collector	None	Lets you specify the Collector data object.
MeasurementStartTime	Out	Float	0.0	Contains the start time of the latest measurement of the specified collector.
MeasurementStopTime	Out	Float	0.0	Contains the stop time of the latest measurement of the specified collector.
OfflineMeasurementStartTime	In	Float	0.0	Lets you specify the value to be used in offline operation mode.
OfflineMeasurementStopTime	In	Float	0.0	Lets you specify the value to be used in offline operation mode.

Related topics

Basics

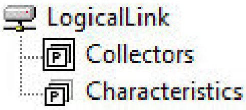
Executing Sequences Using Different Operation Modes (AutomationDesk Basic Practices 🕮)

References

Collector 38

LogicalLink

Graphical representation



Purpose

To set up the logical link to the ECU or other devices.

Description

The LogicalLink data object can be considered as a grouping layer for the Collectors and Characteristics data object. For each logical link and binary file, the variables, variable type and an optional description are displayed. The binary file is the ECU image file that holds the data of the ECU application. You can add characteristics and measurements. You can use any platform and device which is supported by the calibration tool. For example, with ControlDesk as calibration tool, you can use PHS-bus based dSPACE hardware like a DS1005, but also SCALEXIO, or VEOS for offline simulation. For further information, refer to ControlDesk Platform Management 📖.

The following parameters are specified:

Name	In / Out	Type	Default Value	Description
LogicalLinkName	In	String	" "	To select the type of connection (e.g. XCP, CAN) or any arbitrarily string. The parameterization can be performed without a connected MC system, too. All available logical links are displayed if a project was selected.
BinaryName	In	String	" "	To select the binary file for the logical link (e.g. XCP_IMAGE). If you select an empty binary (an empty string), the default binary file is set for the MC system. The parameterization can be performed without an MC system, too.

Access via Exec block

This data object provides some methods that you can access via Python script in an Exec block. The following methods are available:

Method	Description
GetLogicalLinkName()	To get the specified logical link.
SetLogicalLinkName(LogicalLinkName)	To set the logical link. The value that you want to set, must be available in the list of logical links (see GetAvailableLogicalLinkNames() method).
GetAvailableLogicalLinkNames()	To get a list of strings providing the logical links from which you can choose one. The project must be selected beforehand.
GetBinaryName()	To get the name of the specified binary file.
SetBinaryName(BinaryName)	To set the name of the binary file. The value that you want to set, must be available in the list of binary names (see GetAvailableBinaryNames() method).
GetAvailableBinaryNames()	To get a list of strings providing the binary names from which you can choose one. The project must be selected and the LogicalLinkName must be set beforehand.
DynamicSetCharacteristics(Characteristics)	To clean up the list of Characteristics objects during run time. Characteristics must be a plain list optional filled with strings. Precondition: If the logical link does not exist in the selected project it will be created.

Example

```
_AD_.LogicalLink.SetLogicalLinkName('MyLogicalLink')
_AD_.ListOfBinaryNames = _AD_.LogicalLink.GetAvailableBinaryNames()
_AD_.LogicalLink.SetBinaryName(_AD_.ListOfBinaryNames[1])
```

Related topics

HowTos

How to Create The Required Data Objects For a Calibration Task... 13

References

Project.. 53
SwitchMemoryPage... 57
SwitchOffline.. 58
SwitchOnline... 59
System... 60

Measurement

Graphical representation

Measurement

Purpose

To create a new measurement.

Description

New measurements can be added to your collector. A measurement is determined by its name. A measurement is subordinated to one or more Collector data objects.

Access via Exec block

This data object provides some methods that you can access via Python script in an Exec block. The following methods are available:

Method	Description
GetMeasurementName()	To get the name of the specified measurement variable.
SetMeasurementName(MeasurementName)	To set the name of the measurement variable. The collector must be stopped beforehand.

Example

```
_AD_.Measurement.SetMeasurementName('K_y')
```

Related topics

HowTos

How to Create The Required Data Objects For a Calibration Task.. 13

References

FetchMeasurement.. 45

Project

Graphical representation

Project

Purpose

To add a new or existing project.

Description

You can add a new project to a System data object. You can add multiple projects subordinated to a System data object, but only one project can be selected during run time of a calibration system. If an error occurs during the selection of a project, the system is disconnected. The following parameter is specified:

Name	In / Out	Type	Default Value	Description
ProjectName	In	String	" "	States the name of the project

Access via Exec block

This data object provides some methods that you can access via Python script in an Exec block. The following methods are available:

Method	Description
GetProjectName()	To get the specified MC project.
SetProjectName(Project)	To set the MC project. The value that you want to set, must be available in the list of project names (see GetAvailableProjectNames() method). The project must be deselected when you set the parameter.
GetAvailableProjectNames()	To get a list of strings providing the project names from which you can choose one. The system must be connected beforehand.
IsMC3ProjectSelected()	To check whether the MC project is selected. The method returns a boolean value.

Example

```
_AD_.Project.SetProjectName('MyControlDeskProject')
```

Related topics

HowTos

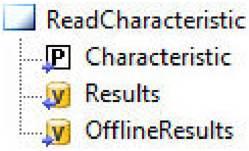
How to Create The Required Data Objects For a Calibration Task.. 13

References

DeselectProject.. 44
SelectProject.. 55
System.. 60

ReadCharacteristic

Graphical representation



ReadCharacteristic
- Characteristic
- Results
- OfflineResults

Purpose

To read the characteristics from the remote MC system.

Description

The following characteristics can be read:

- Scalar
- Curve
- Map

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
Characteristic	In	Extended Data Object	None	The Characteristic data object must be referenced.
Results	Out	Variant	None	Depends on the selected type. The following formats are predefined: ▪ Scalar [value] ▪ Curve [[XAxisValue], [Value]] ▪ Map [[XAxisValue], [YAxisValue], [Value]]
OfflineResults	In	Variant	None	Lets you specify the value to be used in offline operation mode.

Related topics

Basics

Executing Sequences Using Different Operation Modes (AutomationDesk Basic Practices 📖)

HowTos

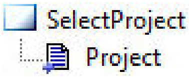
How to Build Sequences to Read Characteristics........ 25

References

Characteristic........ 34
Characteristics........ 37
WriteCharacteristic........ 62

SelectProject

Graphical representation



Purpose

To select a project and set up the connection to the MC system.

Description

After connecting to the MC system, the currently selected project is loaded. If the project was already loaded, no further operation is performed.

> **Note**
>
> Within a sequence it is not allowed to use this block parallel to other blocks.

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
Project	In	Project	None	The project is loaded in the MC system. The parameterization of this data object can be performed without connected MC system.

Related topics

HowTos

How to Build Basic Sequences for Calibration and Measurement Tasks.. 18

References

DeselectProject.. 44
Project.. 53
System.. 60

StartCollector

Graphical representation

StartCollector
Collector

Purpose

To start the event-based measurement.

Description

With this automation block, you can start the measurement of measurement values. The measurement is always used in time stamping mode. The execution of the block must have finished before the results can be fetched. No parameters are measured when executing the StartCollector block. The ECU must be in online mode. After the execution of the StartCollector block, you can fetch the results via the FetchResults block.

Note

Using a **StartCollector** block in parallel to other blocks within a sequence is not supported.

- If you want to start multiple measurements simultaneously, you must add the related **Collector** data objects to a single **StartCollector** block.
- If a sequence contains a high number of **StartCollector** blocks before a **FetchResults** block, only the first **StartCollector** block is executed.

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
Collector	In	Extended Data Object	None	The Collector data object is initialized via the StartCollector block. The Collector data object must be referenced.

Name	In / Out	Type	Default Value	Description
				You can add further Collector data objects to the block by dragging a Collector data object from the Library Browser to the StartCollector block in your sequence.

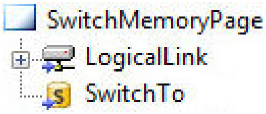
Related topics

References

Collector 38
Collectors 43
FetchResults 47
System 60

SwitchMemoryPage

Graphical representation



SwitchMemoryPage
LogicalLink
SwitchTo

Purpose

To switch between the memory pages of the MC system.

Description

Lets you activate the desired memory page of the ECU. Each MC system provides two different memory pages. It is possible to switch between the reference page and working page of the corresponding ECU. The working page contains a complete data set with all ECU parameters. The working page allows you to perform parameter calibration.

The reference page also contains a complete data set of all ECU parameters, but you cannot change the values of these parameters. The page is a read-only reference, which allows you to run the ECU with a proven set of ECU parameters.

> **Note**
>
> - Within a sequence it is not allowed to use this block parallel to other blocks.
> - Platform devices specified in ControlDesk, for example, DS1005 or SCALEXIO, do not support memory page handling. An unspecific memory of the dSPACE real-time hardware is used for the working data set. For further information, refer to Data Sets and their Relation to Memory Pages (ControlDesk Calibration and Data Set Management 📖).
>
>   Do not use the SwitchMemoryPage block in this context, otherwise an exception is raised.
> - If the page concept of your device provides only one memory page, the SwitchMemoryPage block raises an exception.

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
LogicalLink	In	LogicalLink	None	To select the type of connection (e.g. XCP, CAN) or any arbitrarily string. The parameterization can be performed without an MC system, too.
SwitchTo	In	String	"WorkingPage"	Allows to switch between working page and reference page on the currently active calibration device.

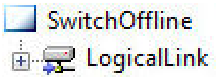
Related topics

References

SwitchOffline 58
SwitchOnline 59
System 60

SwitchOffline

Graphical representation



SwitchOffline
LogicalLink

Purpose

To switch the logical link to offline mode.

Description

The currently active logical link to the MC system is switched to offline mode.

> **Note**
>
> Within a sequence it is not allowed to use this block parallel to other blocks.

Data objects

This automation block provides the following data object:

Name	In / Out	Type	Default Value	Description
LogicalLink	In	LogicalLink Extended Data Object	None	To select the type of connection (e.g. XCP, CAN) or any arbitrarily string. The parameterization can be performed without an MC system, too.

Related topics

HowTos

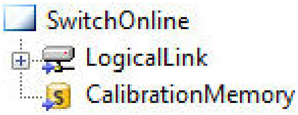
How to Build Basic Sequences for Calibration and Measurement Tasks.. 18

References

SwitchMemoryPage.. 57
SwitchOnline.. 59
System... 60

SwitchOnline

Graphical representation



Purpose

To switch the logical link to online mode.

Description

The currently active logical link between the MC system and the ECU is switched to online mode. If the logical link is already in online mode, no further operation is performed. The calibration memory is updated if you switch from offline to online mode. Furthermore you can select, whether you want to upload from the ECU to the MC system, or download from the MC system to the ECU.

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
LogicalLink	In	LogicalLink Extended Data Object	None	To select the type of connection (e.g. XCP, CAL) or any arbitrarily string. The parameterization can be performed without an MC system, too.
CalibrationMemory	In	String	"Upload"	Determines, whether a upload or download shall be performed if you switch from offline to online mode.

Related topics

HowTos

How to Build Basic Sequences for Calibration and Measurement Tasks.. 18

References

SwitchMemoryPage.. 57
SwitchOffline.. 58
System.. 60

System

Graphical representation

System

Purpose

To set the interface and host of the MC system.

Description

You can state the interface name and host (IP address or computer name) of the MC system. The interface name and host (IP address or computer) are valid as long as no different parameters are determined. You can connect to and disconnect from the selected MC system. If a connection to the MC system was performed successfully, all available projects are displayed. The following parameters are specified:

Name	In / Out	Type	Default Value	Description
Interface	In	String	"ControlDesk"	Connects to the selected MC system via COM/DCOM connection. You can select: ▪ ControlDesk ▪ INCA ▪ CANape

Name	In / Out	Type	Default Value	Description
Host	In	String	"127.0.0.1"	States the IP address or computer name of the PC on which the MC system is working.

Note

To avoid incompatibilities between 32-bit versions and 64-bit versions of AutomationDesk and ControlDesk, use the products of the same dSPACE Release.

Access via Exec block

This data object provides some methods that you can access via Python script in an Exec block. The following methods are available:

Method	Description
GetInterface()	To get the specified MC system.
SetInterface(Interface)	To set the MC system. The value that you want to set, must be available in the list of interfaces (see GetAvailableInterfaceNames() method). The system must be disconnected when you set the parameter.
GetHost()	To get the specified host.
SetHost(Host)	To set the host by specifying an IP address or a computer name. The system must be disconnected when you set the parameter.
GetAvailableInterfaceNames()	To get a list of strings providing the interfaces from which you can choose one.
IsMC3SystemConnected()	To check whether the MC system is connected. The method returns a boolean value.
Connect()	To connect to the specified MC system. See also Connect on page 69.
Disconnect()	To disconnect from the specified MC system. See also Disconnect on page 72.

Example

```
_AD_.System.SetInterface('ControlDesk')
_AD_.System.SetHost('127.0.0.1')
if not _AD_.System.IsMC3SystemConnected():
    AD_System.Connect()
...
```

Related topics

HowTos

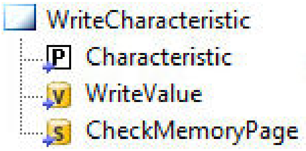
How to Create The Required Data Objects For a Calibration Task.. 13

References

LogicalLink.. 50
Project... 53

WriteCharacteristic

Graphical representation



WriteCharacteristic
Characteristic
WriteValue
CheckMemoryPage

Purpose

To write values of characteristic variables to the remote-controlled MC system.

Description

The following remote calibration characteristics can be written:

- Scalar
- Curve
- Map

The working memory page of the MC system must be activated, before you can use the WriteCharacteristic block. You can write in online and offline mode.

> **Note**
>
> **Using ControlDesk with a platform device specified**
>
> Platform devices specified in ControlDesk, for example, DS1005 or SCALEXIO, do not support memory page handling. An unspecific memory of the dSPACE real-time hardware is used for the working data set. For further information, refer to Data Sets and their Relation to Memory Pages (ControlDesk Calibration and Data Set Management 📖).
>
> If you use a dSPACE platform specified in ControlDesk, you must parameterize the CheckMemoryPage data object to *False*. Otherwise, the WriteCharacteristics block raises an exception.

Data objects

This automation block provides the following data objects:

Name	In / Out	Type	Default Value	Description
Characteristic	In	Extended Data Object	None	The Characteristic data object must be referenced.
WriteValue	In	Variant	None	The following formats are predefined: ▪ Scalar: [Value] ▪ Curve: [[XAxisValue], [Value]] ▪ Map: [[XAxisValue], [YAxisValue], [Value]]
CheckMemoryPage	In	String	"True"	▪ True: The characteristic value is only written if the specified memory page is the working page. If no page or

Name	In / Out	Type	Default Value	Description
				the reference page is detected, the calibration is stopped with an exception. (This is the same behavior as with AutomationDesk versions < 3.3.) ▪ False: There is no page check, the block writes the characteristic value ignoring the specified memory page.

Note

- A Characteristic data object is also called a parameter.
- To write characteristics to only one axis, you must set the other axis to `"None"`. For more information, you can import the demo project `Remote Calibration (COM) ControlDesk.zip`.
  It is located at `<DocumentsFolder>\Remote Calibration (COM)`. Furthermore, look at the `WriteAxisOrValueOnly` sequence stored in `Examples\Characteristics\Write`.
- If the page concept of your device provides only one memory page, you must set CheckMemoryPage to `False`. Otherwise the block raises an exception.

Related topics

HowTos

How to Build Sequences to Write Characteristics.. 26

References

Characteristic.. 34
Characteristics... 37
ReadCharacteristic.. 54

Commands And Dialogs

Where to go from here

Information in this section

Add Characteristic

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ Characteristics data object in the Project Manager ▪ Characteristics data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To add a characteristic ECU variable to be calibrated to an ECU device.

Result

A **Characteristic** data object is added to the **Characteristics** data container of the **LogicalLink** data object.

Description

The **Characteristic** data object represents an ECU variable to be calibrated from the A2L file. You can add many **Characteristic** data objects to a **Characteristics** data container. The term **Characteristic** is independent of the variable type's dimension. The following variable types are Characteristics:

- ScalarCharacteristic
- CurveCharacteristic
- MapCharacteristic

Using the Characteristic Configuration dialog, you can parameterize the added data object, refer to Edit (Characteristic) on page 74.

Tip

A **Characteristic** data object is also called parameter.

Related topics

References

Characteristic.. 34
Edit (Characteristic).. 74

Add Collector

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ Collectors data object in the Project Manager ▪ Collectors data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To add a measurement object to an ECU device in AutomationDesk.

Result

A **Collector** data object is added to the **Collectors** data container of the **LogicalLink** data object.

Description

A **Collector** data object is used to configure, perform and manage measurements. The variables to be measured must be added to the Collector using the Add Measurement Variable command.

A new Collector can be created for each supported acquisition rate. A **Collectors** data object can contain multiple **Collector** data objects. A **Collector** data object must be a subelement of a **Collectors** data object.

Related topics

References

Add Measurement Variable.. 67
Collectors.. 43
Edit (Collector)... 77
LogicalLink.. 50

Add LogicalLink

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor

	▪ Project data object in the Project Manager ▪ Project data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To add a representation for the logical and physical ECU connection to a calibration project.

Result

A new LogicalLink data object is added to a Project data object.

Description

A LogicalLink data object, which describes the ECU device, must be a subelement of a Project data object. A Project data object can contain multiple LogicalLink data objects. The settings of a LogicalLink are automatically loaded from the connected calibration tool. A LogicalLink data object contains a Collectors data container and a Characteristics data container.

Note

You can create and use a single LogicalLink data object at any hierarchy of your AutomationDesk project, but it must be referenced to a LogicalLink data object contained in an entire calibration project definition.This applies to all data objects except for system data objects.

Related topics

References

Characteristics	37
Collectors	43
Edit (LogicalLink)	78
LogicalLink	50
Project	53

Add Measurement Variable

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor

	▪ Collector data object in the Project Manager ▪ Collector data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To add a measurement variable to the Collector.

Result

A **Measurement** data object is added to the corresponding **Collector** data object.

Description

The variable to be measured can be specified by dragging a variable from the **LogicalLink Configuration** dialog's variable browser to the Measurement data object or by entering the variable name in the Measurement Configuration dialog.

Note

You can create and use a MeasurementVariable data object at any hierarchy of your AutomationDesk project, but it must be referenced to a MeasurementVariable data object contained in an entire calibration project definition.

Related topics

References

Add LogicalLink 66
Edit (LogicalLink) 78
Edit (Measurement) 79

Add Project

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ System data object in the Project Manager ▪ System data object in the Sequence Hierarchy Browser

Shortcut key	None
Icon	None

Purpose

To add a project to the calibration system.

Result

The System data object contains a project element, which represents a project of the calibration tool.

Description

A **Project** data object must be a subelement of a **System** data object. A system can contain multiple projects. The project settings can be loaded from the connected calibration tool using the Select command.

> **Note**
>
> You can create and use a single Project data object at any hierarchy of your AutomationDesk project, but it must be referenced to a Project data object contained in an entire calibration project definition.

Related topics

References

Edit (Project) 80
Project 53
Select 82
System 60

Connect

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ System data object in the Project Manager ▪ System data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To connect the calibration tool with AutomationDesk.

Result

AutomationDesk is connected to the calibration tool server.

Description

The calibration tool and AutomationDesk exchange calibration data via COM/DCOM interface. If the system is connected, all project information from the calibration tool database can be loaded to AutomationDesk.

> **Note**
>
> The DCOM settings depend on your calibration tool. An administrator must configure them.

System Configuration dialog

Interface Lets you select the calibration tool to be used.

Host Lets you enter the IP address of the calibration tool server. The default value 127.0.0.1 (local host) is used, if the calibration tool and AutomationDesk are installed on the same PC.

> **Note**
>
> The System data object represents all parameters and objects used for automating the calibration. It loads the A2L file into the PC RAM. To avoid poor performance at run time, you should only create one System data object for one calibration system in your AutomationDesk session.

Connect/Disconnect – (Available only if the library is set to online operation mode) Lets you connect/disconnect AutomationDesk to/from the calibration tool.

Related topics

References

Disconnect...72
System...60

Deselect

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ Project data object in the Project Manager ▪ Project data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None
Others	Double-click

Purpose

To deselect a previously selected calibration project.

Result

The selected calibration project is deselected and the connection between AutomationDesk and the calibration tool server is closed.

Description

If the calibration tool is connected to AutomationDesk, you can choose one of the available calibration projects as the project to be tested. If you have selected a project, you can deselect it again using this command.

Project Configuration dialog

Project Displays the current selected calibration project which you can deselect

Deselect – (Available only if the library is set to online operation mode) Lets you deselect the Project data object. It is reset to default values.

Related topics

References

Project..53
Select..82

Disconnect

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ System data object in the Project Manager ▪ System data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To disconnect the calibration tool server from AutomationDesk.

Result

The connection between the calibration tool server and AutomationDesk is closed.

Description

The calibration tool and AutomationDesk do not exchange calibration data via COM/DCOM interface any longer. AutomationDesk has no access to project data anymore.

System Configuration dialog

Interface Lets you select the calibration tool to be used.

Host Lets you enter the IP address of the calibration tool server. The default value 127.0.0.1 (local host) is used, if the calibration tool and AutomationDesk are installed on the same PC.

Note

The System data object represents all parameters and objects used for automating the calibration. It loads the A2L file into the PC RAM. To avoid poor performance at run time, you should only create one System data object for one calibration system in your AutomationDesk session.

Connect/Disconnect – (Available only if the library is set to online operation mode) Lets you connect/disconnect AutomationDesk to/from the calibration tool.

Related topics

References

Connect........ 69
System........ 60

Edit (AddResultsToReport)

Access

You can access this command via:

Ribbon	None
Context menu of	None
Shortcut key	None
Icon	None
Others	Double-click AddResultsToReport block in the Sequence Builder

Purpose

To specify subplots, signals and axes of a plot.

Result

The specified properties are assigned to the plot.

Description

A plot can consist of a number of subplots. A subplot can contain various variables of the result.

Adaptor dialog

The dialog opens when you double-click an AddResultsToReport block in the Sequence Builder.

> **Note**
>
> The Adaptor dialog opens only when the AddResultsToReport block is referenced to an appropriate Results data object.

All signals Displays all available signals of the collector and lets you select them. A signal can be assigned to different subplots. It can, however, be assigned only once to a particular subplot.

> Lets you move the selected entry in the All signals list to the Signals list.

>> Lets you move all entries in the **All signals** list to the **Signals** list.

Subplots Displays and lets you select existing subplots.

New Lets you add a new subplot.

Delete Lets you delete the selected subplot.

Signals Displays and lets you select the variables of a selected subplot.

Up Lets you move the selected variable upwards.

Down Lets you move the selected variable downwards.

Delete Lets you delete the selected variable.

X-Axis Lets you specify the X-axis of a subplot. The X-values of signals are merged for a subset, if the signals are coming from different hosts, for example, signals of a multi-board configuration.

Y-Axis Lets you specify the Y-axis of a subplot.

Related topics

References

AddResultsToReport.. 33

Edit (Characteristic)

Access

You can access this command via:

Ribbon	None
Context menu of	None
Shortcut key	None
Icon	None
Others	▪ Button in the Value column of the Data Object Editor ▪ Double-click

Purpose

To parameterize an ECU variable to be calibrated.

Result

The Characteristic Configuration dialog opens.

Description

The Characteristic data object represents an ECU variable to be calibrated from the A2L file. The term Characteristic is independent of the variable type's dimension. The following variable types are Characteristics:

- ScalarCharacteristic
- CurveCharacteristic
- MapCharacteristic

Using the Characteristic Configuration dialog, you can parameterize the added data object.

Characteristic Configuration dialog

If you double-click the Characteristic data object, the dialog opens.

Characteristic type Lets you select the type of the variable to be calibrated:

- Scalar

  A scalar is a parameter that consists of a single function value.
- Curve

  A curve is a parameter that consists of:
  - One vector (1-dimensional array) containing the axis points for the x-axis.
  - Another vector containing function values. The curve assigns one function value to each axis point.
- Map

  A map is a parameter that consist of:
  - Two vectors (1-dimensional arrays) containing the axis points for the x-axis and y-axis.
  - A matrix containing function values. The map assigns one function value of the matrix to each pair of x-axis and y-axis points.

Characteristic name Lets you enter the name of the parameter to be calibrated from the A2L file.

Representation type Lets you select the representation type of the parameterized value:

A numerical value can be displayed in its 'original format' on the hardware (source mode) or in a converted form (converted mode).

- eRT_Physical

  A numerical value is displayed in a converted form (converted mode). This is the default setting.
- eRT_ECU

  A numerical value is displayed in its 'original format' on the hardware (source mode).

X-axis start Lets you enter the start index of the X-axis. It specifies the first value to be changed in a range of values. This setting is enabled for Curve and Map characteristics.

X-axis stop Lets you enter the stop index of the X-axis. It specifies the last value to be changed in a range of values. The stop index is exclusive, that means, it determines the first position that is not changed. A stop index of -1 means that all the values of a range are to be changed. This setting is enabled for Curve and Map characteristics.

Y-axis start Lets you enter the start index of the Y-axis. It specifies the first value to be changed in a range of values. This setting is enabled for Map characteristics.

Y-axis stop Lets you enter the stop index of the Y-axis. It specifies the last value to be changed in a range of values. The stop index is exclusive, that means, it determines the first position that is not changed. A stop index of `-1` means that all the values of a range are to be changed. This setting is enabled for Map characteristics.

Value type Lets you select the value type of the parameter to be calibrated:

- eVT_CONST

  For writing a constant value to a range of a map or curve.
- eVT_OFFSET_NEG

  For subtracting a constant value from a range in a map or curve.
- eVT_OFFSET_POS

  For adding a constant value to a range of a map or curve.
- eVT_VAL

  For writing absolute values. This is the default setting.

Note

- A Characteristic data object is also called a parameter.
- To write characteristics to only one axis, you must set the other axis to `"None"`. For more information, you can import the demo project `Remote Calibration (COM) ControlDesk.zip`.

  It is located at `<DocumentsFolder>\Remote Calibration (COM)`. Furthermore, look at the `WriteAxisOrValueOnly` sequence stored in `Examples\Characteristics\Write`.
- If the page concept of your device provides only one memory page, you must set CheckMemoryPage to `False`. Otherwise the block raises an exception.

Related topics

References

Edit (Collector)

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ Collector data object in the Project Manager ▪ Collector data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To configure the selected Collector data object.

Result

The properties are assigned to the Collector data object.

Collector Configuration dialog

Type Lets you select the storage type of the measurement results.

- eST_AUSY

  The measurement results are stored in the calibration tool.
- eST_FILE

  The measurement results are stored in a file.

File name Lets you open a standard File dialog to select a file from your file system or to enter a file name. This file is used to store the measurement results in, if you have specified eST_FILE as storage type.

- Absolute Path

  Lets you save the path as an absolute path in the project.
- Relative Path

  Lets you save the path as a relative path in the project. It is then a shortened path relating to the AutomationDesk project file. The path will not be changed to a relative path if the project and the specified file are saved to different drives.

Buffer rate Lets you select the result buffer. Only event-based measurements are supported.

Downsampling Lets you enter the downsampling rate. It specifies after how many values a storage to the internal buffer takes place.

Number of samples Lets you enter the number of samples. It determines how many results are needed before the Calibration Server sends them to the client.

Buffer size Lets you enter the size of the specified buffer. As the buffer is a ring buffer, earlier values are overwritten by later values when the buffer capacity is exceeded.

Representation type Lets you select the representation type of the parameterized value:

A numerical value can be displayed in its 'original format' on the hardware (source mode) or in a converted form (converted mode).

- eRT_Physical

  A numerical value is displayed in a converted form (converted mode). This is the default setting.
- eRT_ECU

  A numerical value is displayed in its 'original format' on the hardware (source mode).

Related topics

References

Edit (LogicalLink)

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ LogicalLink data object in the Project Manager ▪ LogicalLink data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To configure the selected LogicalLink data object.

Result

The properties are assigned to the LogicalLink data object.

LogicalLink Configuration dialog

LogicalLink name Lets you select the LogicalLink specified in the loaded calibration project definition. You can also enter a string if there is no calibration tool connected.

Binary name Lets you select a specific program version of an ECU, originally coming from the ECU image file. A binary can be changed only by instancing a new LogicalLink. If you do not specify a binary name, the calibration tool uses the default binary. You can also enter a string if there is no calibration tool connected.

Variable browser Displays information about the variables on the selected device, if a project is selected. The structure of the loaded variable description file is displayed in the hierarchy tree. The variable list shows the variables of the selected node. You can sort the variables in ascending or descending alphabetical order by clicking the **Variable** cell.

Every entry takes immediately effect if you select a new item or close the dialog (by clicking the Close button in the title bar).

Related topics

References

Add LogicalLink.......... 66
Characteristics.......... 37
Collectors.......... 43
LogicalLink.......... 50
Project.......... 53

Edit (Measurement)

Access

You can access this command via:

Ribbon	None
Context menu of	None
Shortcut key	None
Icon	None
Others	▪ Button in the Value column of the Data Object Editor ▪ Double-click

Purpose

To enter the name of the variable to be measured.

Result

The variable to be measured is specified.

Measurement Configuration dialog

If you double-click the Measurement data object, the dialog opens.

Measurement name Lets you enter the name of the variable to be measured.

Related topics

References

Add Measurement Variable.. 67

Edit (Project)

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ Project data object in the Project Manager ▪ Project data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To configure the selected Project data object.

Result

The selected calibration project is assigned to the Project data object.

Project Configuration dialog

Project Lets you select a calibration project from the drop-down list. The list contains the available projects on the calibration tool server provided there is a connection to it before you open the dialog.

Select/Deselect – (Available only if the library is set to online operation mode) Lets you select/deselect the calibration project. When you deselect it, it is reset to default values.

Related topics

References

Add Project.. 68
Deselect.. 71
Project.. 53

Select..82
System..60

Edit (System)

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ System data object in the Project Manager ▪ System data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None

Purpose

To configure the selected System data object.

Result

The properties are assigned to the System data object.

System Configuration dialog

Interface Lets you select the calibration tool set to be used.

Host Lets you enter the IP address of the calibration tool server. The default value 127.0.0.1 (local host) is used, if the calibration tool and AutomationDesk are installed on the same PC.

Connect/Disconnect – (Available only if the library is set to online operation mode) Lets you connect/disconnect AutomationDesk to/from the calibration tool.

Related topics

References

System..60

Select

Access

You can access this command via:

Ribbon	None
Context menu of	▪ Data Objects column of the Data Object Editor ▪ Project data object in the Project Manager ▪ Project data object in the Sequence Hierarchy Browser
Shortcut key	None
Icon	None
Others	Double-click

Purpose

To select a calibration project.

Result

The calibration project is selected, that means, that its information is enabled for the use in other configuration dialogs.

Description

The configuration dialogs that contain drop-down lists or browser, for example the variable browser, get their values from the selected calibration project.

Project Configuration dialog

Project Lets you select one of the available calibration projects from a drop-down list.

Select – (Available only if the library is set to online operation mode) Lets you select the calibration project.

Related topics

References

Deselect.......... 71
Project.......... 53

Automation

Basics on Automating the Access to Remote Calibration COM

Introduction

AutomationDesk provides a COM-based API to automate the handling of AutomationDesk.

Related information

The AutomationDesk COM API provides the following objects for configuring the access to a calibration system via the Remote Calibration COM library:

- MC3System (AutomationDesk Automation 📖)
- MC3Project (AutomationDesk Automation 📖)
- MC3LogicalLink (AutomationDesk Automation 📖)
- MC3Characteristics (AutomationDesk Automation 📖)
- MC3Collector (AutomationDesk Automation 📖)
- MC3Measurement (AutomationDesk Automation 📖)

For basic information and instructions, refer to Basics and Instructions on page 7.

Limitations

Limitations When Using the Remote Calibration (COM) Library

Limitations for INCA 5.1.2

The Remote Calibration (COM) library is based on the ASAM-MCD3 MC standard. Calibration tools that support this standard can be automated by AutomationDesk.

There are some limitations, if you use INCA 5.1.2:

- You cannot measure characteristics.
- If you use collectors with eST_AUSY storage type and eST_FILE storage type, the file-storing collector must be stopped before the AUSY-storing collector. Otherwise the file will not be created. You can stop a collector by using a FetchResults or FetchMeasurement block.
- If you use the same variable names for file-storing and AUSY-storing collectors, you must check the file-storing collectors before the AUSY-storing collectors. Otherwise the measured variables are not available in the file. A collector is automatically checked if you use the StartCollector block.
- If you reconfigure a collector, removed measurements are not considered. The rasters will be filled up. To avoid this, use a SwitchOffline block after collector reconfiguration.
- If collectors are destroyed during a measurement, the memory required by AutomationDesk increases.
- A measurement file can contain too many variables if you use several file-storing collectors with different variables. To avoid this, use a SwitchOffline block before starting another file-storing collector.

Labels in TRC file

Variables in the *Labels* group of a TRC file are not supported.

Server execution error

If you initialize and close the access to ControlDesk several times, the COM object in ControlDesk might not be cleared already. Finishing ControlDesk requires some time before its COM object is cleared completely. A server execution error message will be displayed. To avoid this, you can add a Sleep block at the end of your calibration sequences.

Automated recording with ControlDesk

If you store a recording via the MC system, some of the recording settings in your automation script may be ignored. For example, ControlDesk ignores downsampling and representation type settings, because it does not perform DAQ downsampling and it defines the representation type according to the measurement data file type. If the file type supports the storing of conversion formulas (for example, MF4 files), source values are stored. If the file type does not support the storing of conversion formulas (for example, MAT and CSV files), physical values are stored.

Memory pages of platform devices used with ControlDesk

Platform devices specified in ControlDesk, for example, DS1005 or SCALEXIO, do not support memory page handling. An unspecific memory of the dSPACE real-time hardware is used for the working data set. For further information, refer to Data Sets and their Relation to Memory Pages (ControlDesk Calibration and Data Set Management 🕮).

Do not use the **SwitchMemoryPage** block in this context, otherwise an exception is raised.