ControlDesk

MCD-3 Automation

For ControlDesk 7.4

Release 2021-A – May 2021

dSPACE

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.

Contents

About This Document

Contents

This document shows you how to automate your calibration, measurement, and diagnostic tasks using ControlDesk's ASAM MCD-3-compatible interface.

Required knowledge

You should be familiar with performing the tasks described above in ControlDesk *without* MCD-3 automation.

Knowledge in handling the host PC and the Microsoft Windows operating system is also assumed.

Symbols

dSPACE user documentation uses the following symbols:

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

Naming conventions

dSPACE user documentation uses the following naming conventions:

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

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

Special folders

Some software products use the following special folders:

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

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

or

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

Documents folder A standard folder for user-specific documents.

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

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

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

Accessing dSPACE Help and PDF Files

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

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

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

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

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

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

Basics and Instructions

Where to go from here

Information in this section

Introduction to ControlDesk MCD-3 Automation

Where to go from here

Information in this section

Basics on MCD Systems

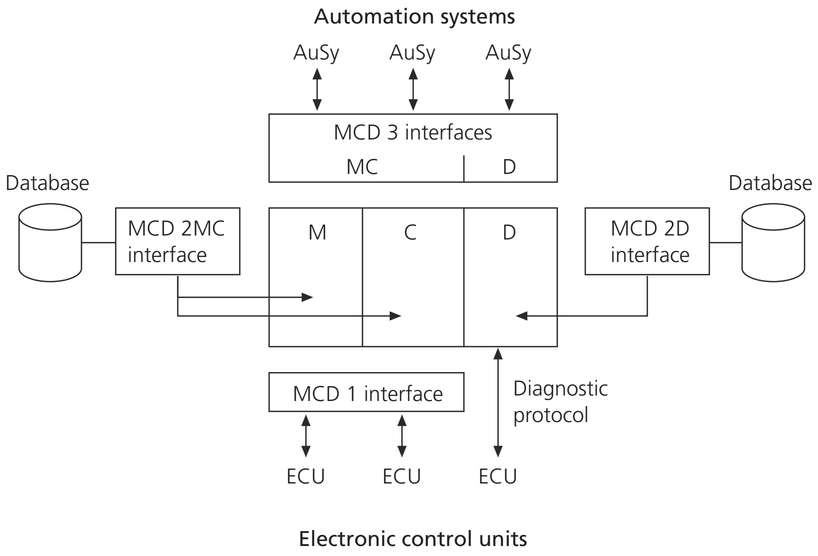
ASAM

ASAM is short for 'Association for Standardization of Automation- and Measuring Systems'. ASAM develops and provides standards for a variety of applications in the automotive field.

The ASAM MCD standard defines interfaces and data exchange formats of MCD systems. MCD systems provide access to electronic control units (ECUs) or measurement devices and are used for measurement, calibration, and diagnostic (MCD) tasks.

MCD systems

Measurement (M), Calibration (C) and Diagnostic (D) are the function blocks of an MCD system. A real system does not have to support all function blocks.



MCD specifications are currently available for the following interfaces:

MCD-1 (protocol interface) Defines the standardized access to an MCD compatible device, for example, an ECU with ROM emulator.

MCD-2 (data description interface) Defines the data exchange format for information on the MCD device. Control unit description files provide information on the parameters, measurement variables and interface parameters of an MCD device.

Diagnostic databases provide information on executable diagnostic communication objects (diagnostic services, diagnostic jobs, and control primitives) and the diagnostic interface.

MCD-3 (functional interface) Allows remote-control of MCD systems. The MCD-3 standard defines an object model that allows you to work with each function block independently of the others. Each function block has its own interfaces. This object model has to be implemented for the specific platforms and programming languages.

MCD-3 object model

The ASAM MCD-3 object model serves as a template for concrete object models that are created to remote-control MCD systems according to the MCD-3 standard. It defines the interfaces and functionality that compatible object models must provide.

The MCD-3 object model contains *database objects* and *run-time objects*.

Database objects Database objects are invariable. They provide access to the basis data of an experiment, for example the properties of variables, ECU Image file data, or information on a device interface. All database classes have a "Db" in their names. The database objects are the basis for creating run-time objects to work with.

Run-time objects The run-time classes have the same names as the database classes, but without "Db". For example, "DbCharacteristic" is the name for a parameter class on the database side. The run-time "Characteristic" object gets its information from the related "DbCharacteristic", for example, its upper and lower limits.

ASAM documentation For detailed information on the MCD-3 object model, refer to the *Application Programming Interface Specification* for ASAM MCD-3. You can download the specifications from http://www.asam.net/.

COM/DCOM object model

The generic MCD-3 object model has been implemented as a (Distributed) Component Object Model (COM/DCOM). DCOM is an extension to COM to support communication between objects on different computers via network connections. Using COM/DCOM makes objects available for different programming and scripting languages and provides mechanisms for remote-control.

The COM/DCOM object model is not a one-to-one translation of the generic MCD-3 object model. Methods in the generic model can be translated to properties in the COM object model. It also depends on the programming language how the methods can be used.

However, basic information on its functionality, as well as background and context information, can be derived from the generic MCD-3 object model.

Related topics

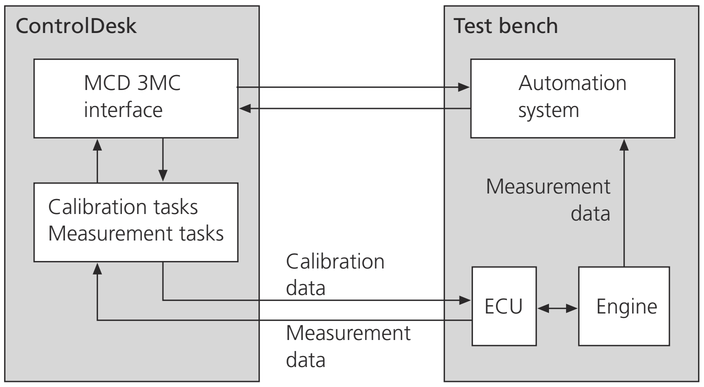
Basics

Basics on Automating Measurement and Calibration via the ControlDesk MCD3 Interface

Introduction

ControlDesk provides an MCD3 interface that is compatible with ASAM MCD-3 MC 1.0.1. You can use the ControlDesk MCD3 interface to automate calibration and measurement tasks in test benches. This enables you to perform various calibration and measurement tasks without changing parameter values and other experiment settings manually.

The automation system sends calibration and measurement commands to ControlDesk and receives data from it. ControlDesk executes the commands on a platform/device.

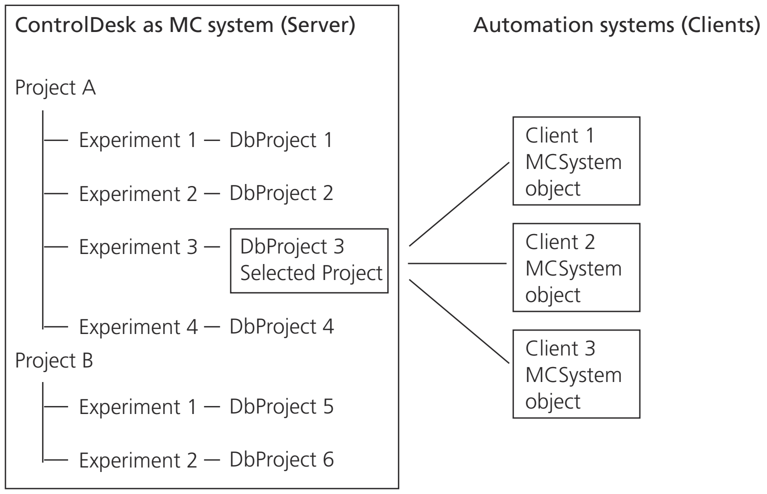


Case-sensitivity of scripts Scripts are case-sensitive. Script text must match the spelling of the API documentation.

MCSystem

To start an automation session and communicate with ControlDesk, a run-time system object (MCSystem) has to be created first.

ControlDesk as the server In an automation session, the MC system (ControlDesk side) acts as the server.


ControlDesk as MC system (Server)
Automation systems (Clients)
Project A
Experiment 1 — DbProject 1
Experiment 2 — DbProject 2
Experiment 3 — DbProject 3 Selected Project
Experiment 4 — DbProject 4
Project B
Experiment 1 — DbProject 5
Experiment 2 — DbProject 6
Client 1 MCSystem object
Client 2 MCSystem object
Client 3 MCSystem object

Client One or more automation systems (application side) can be clients. Each client application connecting to the server creates its own MCSystem object with its own subobjects, but all client applications share the same data. For example, all clients can access only the same project.

⚠ WARNING

All clients have to cooperate. There is currently no mechanism that protects changes or configurations by one client from those made by others.

You can use AutomationDesk as a client application to automate measurement and calibration (MC) tasks on an ECU by using AutomationDesk's *Remote Calibration (COM) library*. For details, refer to AutomationDesk Accessing Remote Calibration COM 🕮.

Project

The DbProjectDescriptions collection lets you select a DbProject to create a run-time Project.

Note

The terminology here is different to that in ControlDesk:
An ASAM MCD-3 project corresponds to a ControlDesk experiment. The DbProjectDescriptions collection contains all the experiments known to ControlDesk, not only those of a single ControlDesk project.

Characteristics and measurements

In ASAM MCD-3 terms, parameters are called Characteristics. You select one DbCharacteristic from the DbCharacteristics collection to create a run-time Characteristic for calibration.

Measurement variables have to be selected from the DbMeasurements collection.

Note

As of dSPACE Release 2021-B, the ControlDesk MCD-3 automation interface will no longer support the access to calculated variables. As a consequence, you will have to adapt MCD-3 automation scripts that involve the access to calculated variables.

LogicalLinks and Modules

LogicalLinks and Modules are objects that represent the connection to a device. Together they provide access to parameters and measurement variables and contain the specific program version.

Collector

To measure variables you must create one or more Collector objects. Measurements are configured and run via a Collector.

Measurements in a multi-client system In a multi-client system, measurement is performed sequentially for all the clients. At first, the Collector of Client 1 receives its measurement data, then the collector of Client 2, and so on. As long as a client receives measurement data, data acquisition is paused for all the other clients.

Database

In contrast to using ASAP3 classic, you do not have to open ControlDesk or rely on notes to get basic information on an experiment. Instead, you can read the database side of the object model for information, for example, to list parameters or measurement variables.

Logging of API commands

You can specify whether the logging of API commands is enabled for automation via ControlDesk's ASAM MCD-3 interface. Refer to MC3 Page on page 85.

Related topics

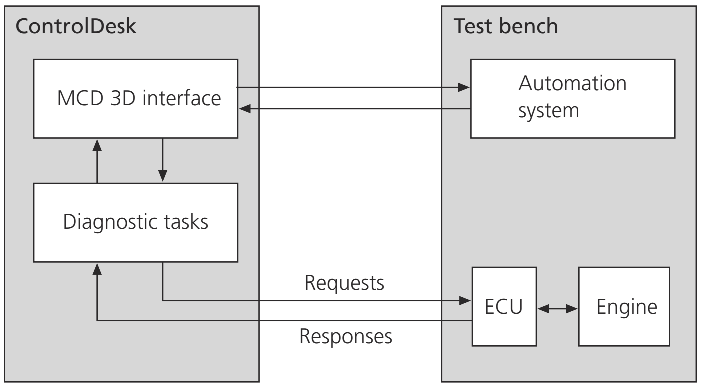
Basics

Basics on Automating ECU Diagnostics via the ControlDesk MCD3 Interface

Introduction

ControlDesk provides an MCD3 interface that is compatible with ASAM MCD-3 D 2.0.2. You can use the ControlDesk MCD3 interface to automate the execution of diagnostic tasks.

To do so, you can access a ControlDesk experiment with an ECU Diagnostics device containing the configuration of a diagnostic database with communication objects (diagnostic services, diagnostic jobs, and control primitives). The database must be compliant with Open Diagnostic Data Exchange (ODX).



Case-sensitivity of scripts Scripts are case-sensitive. Script text must match the spelling of the API documentation.

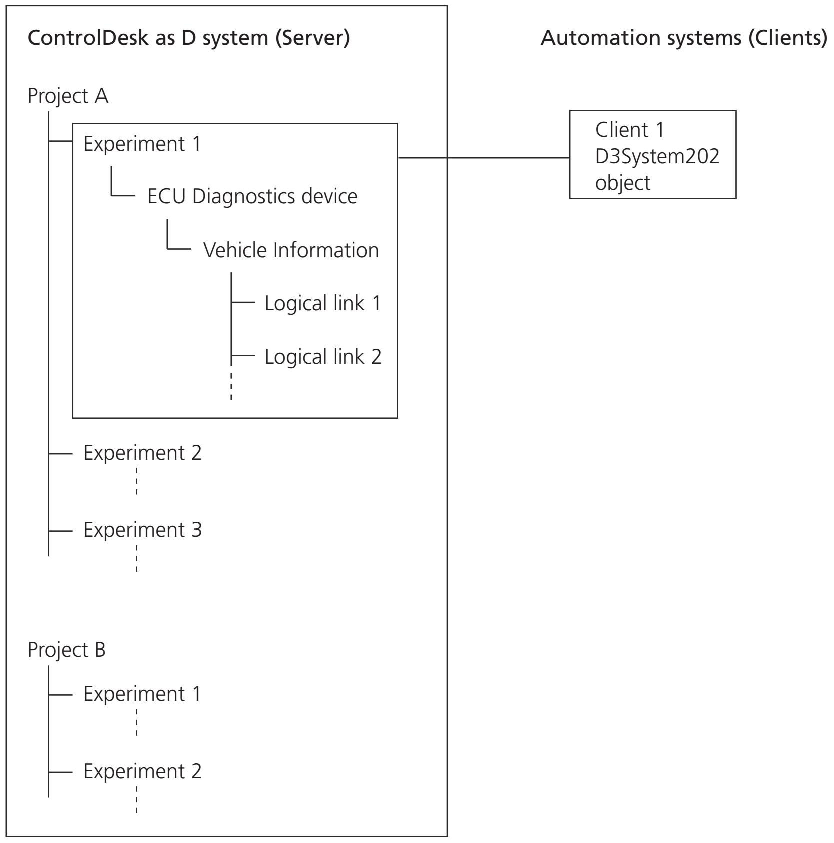
DSystem

To start an automation session and communicate with the ECU via a diagnostic interface, a run-time system object has to be created first. In an automation session, the D system (ControlDesk side) acts as the server and one automation system (application side) is the client. The client application creates a DSystem object when it connects to the server.

ControlDesk as the server In an automation session, the MC system (ControlDesk side) acts as the server.

ControlDesk's ASAM MCD-3 D interface is accessed via the `D3System202` interface.

Using the `D3System202` interface, you use the configuration of an ODX database contained in a ControlDesk experiment.



Client applications You can use AutomationDesk as a client application to automate ECU diagnostics tasks on an ECU by using the AutomationDesk *Remote Diagnostics (COM) library*. For details, refer to AutomationDesk Accessing Remote Diagnostics COM 🕮.

Project

The DbProjectDescriptions collection lets you select a DbProject to create a run-time Project.

Note

The terminology here is different to that in ControlDesk:
When you use the `D3System202` interface, an ASAM MCD-3 D project corresponds to a ControlDesk experiment. The DbProjectDescriptions collection contains all the experiments known to ControlDesk, not only those of a single ControlDesk project. The collection contains even experiments that do not contain an ECU Diagnostics device.

Database

ECU diagnostics with ControlDesk is based on ODX (Open Diagnostic Data Exchange), the ASAM MCD-2 D diagnostics standard. All the information on executable objects or the diagnostic interface can be read from the diagnostic database (ODX database).

ControlDesk supports the following ODX database standards:
- ASAM MCD-2 D V2.0.1
- ASAM MCD-2 D V2.2.0 (ISO 22901-1)

Vehicle information

A vehicle information table contains a list of vehicle configurations. Each vehicle configuration contains LogicalLink objects necessary for connection to the ECUs of a vehicle. The client creates run-time logical links by choosing a vehicle configuration and then selecting logical links from it.

Note

You can access only the vehicle configuration that you selected when you configured the corresponding ECU Diagnostics device in the ControlDesk experiment.

LogicalLinks

A LogicalLink is an object that represents the connection to an ECU.

Note

You can access only the logical links that you selected when you configured the corresponding ECU Diagnostics device in the ControlDesk experiment.

Diagnostic communication objects

The ODX database describes different diagnostic communication objects to communicate with an ECU.

In ASAM-MCD3 D terms these objects are called DiagComPrimitives. DiagComPrimitives are differentiated in DataPrimitives and ControlPrimitives. Diagnostic services and diagnostic jobs are derived from Data Primitives.

Diagnostic services Diagnostic services are implemented on the ECU as basic elements for diagnostic communication. Communication is performed by selecting a service, configuring its parameters, executing it, and receiving the ECU results. When a service is executed, it sends a defined request to the ECU and the ECU answers with a defined response.

Diagnostic jobs Diagnostic jobs (often called Java jobs) are programmed sequences, that are usually built of a sequence of diagnostic services. Diagnostic jobs are either single-ECU jobs or multiple-ECU jobs, depending on whether they communicate with one ECU or multiple ECUs.

Control primitives Control primitives are special communication objects for the changing of communication states or protocol parameters, or for (ECU) variant identification purposes.

Related topics

Basics

Basics on MCD Systems.. 10

Automating Different Versions of the ControlDesk MCD3 Interface

Introduction

You can supply version information when accessing the ControlDesk MCD3 interface.

Accessing ControlDesk's MCD3 interface

To access ControlDesk's MCD3 interface, you can use the `Dispatch` command in your script as shown in the listing below.

```
# Import required modules.
from win32com.client import Dispatch
```

```
# Access ControlDesk's MCD3 interface.
application =  Dispatch("ControlDeskNG.MC3System")
```

Automating a specific version of ControlDesk's MCD3 interface

You can automate a specific version of ControlDesk's MCD3 interface by supplying the related version number. via the `Dispatch` command, as shown in the example listings below.

Examples In Python, you supply version information via the `Dispatch` command.

The following Python listing shows how you can automate a specific version of ControlDesk's MC3 interface.

```
# Access ControlDesk's MC3 interface
application =  Dispatch("ControlDeskNG.MC3System.7.0")
```

The following Python listing shows how you can automate a specific version of ControlDesk's D3 interface.

```
# Access ControlDesk's D3 interface
self.DSystem = Dispatch("ControlDeskNG.D3System202.7.0")
```

Version information The following table lists the relevant version information:

ControlDesk Version	Version Argument for ...	
	... Accessing the `MC3System` Interface	... Accessing the `D3System202` Interface
7.4	`7.4`	`7.4`
7.3	`7.3`	`7.3`
7.2	`7.2`	`7.2`
7.1	`7.1`	`7.1`
7.0	`7.0`	`7.0`
6.4	`6.4`	`6.4`
6.3	`6.3`	`6.3`
6.2	`6.2`	`6.2`
6.1	`6.1`	`6.1`
6.0	`6.0`	`6.0`
5.6	`12`	`10`
5.5	`11`	`9`

Automating the MCD3 interface of the active ControlDesk version

Active ControlDesk version If you have several ControlDesk versions on your PC, one is the *active version*. The latest installed version of ControlDesk is the active version by default regardless of its version number.

- When you remove the currently active version of ControlDesk, there is no active version because ControlDesk does not activate a version automatically. See below for information on activating another ControlDesk version.
- When you remove a currently inactive version of ControlDesk, the currently active version of ControlDesk remains the same.

The MCD3 interface of the *active ControlDesk version* is the version that you access via `ControlDeskNG.MC3System`.

Activating the MCD3 interface of another ControlDesk version ControlDesk's MCD3 interface is registered in the Microsoft Windows Registry automatically during ControlDesk installation.

To access the MCD3 interface of a specific ControlDesk version *without supplying version information*`Dispatch`, activate that version via the dSPACE Installation Manager. For instructions, refer to How to Activate a Single dSPACE Installation (Managing dSPACE Software Installations 🕮).

Related topics

HowTos

How to Activate a Single dSPACE Installation (Managing dSPACE Software Installations 📖)

Running ControlDesk and the Automation System on Different PCs

Where to go from here

Information in this section

Basics on Running ControlDesk and the Automation System on Different PCs

Introduction

To run ControlDesk and the automation system *on different PCs*, the following preconditions must be met:

- The PC on which the automation system is installed (*client PC*) and the PC on which ControlDesk is installed (*server PC*) must be connected either peer-to-peer or via LAN or WAN connection.

  Note

  Only use 64-bit client applications. 32-bit automation client applications are not supported.

- The DCOM settings must be configured on the server PC. Refer to How to Configure DCOM Settings on the Server PC on page 22.
- Both PCs must be started and the same user must be logged on. If the user is not defined via a domain, create an account with the same name and password on both PCs. The password must not be empty.

Required software (server PC)

On the server PC, the *ControlDesk* product set must be installed. The related license(s) must be available.

Required software (client PC)

On the client PC, the *ControlDesk* product set must be installed.

Related topics

HowTos

How to Configure DCOM Settings on the Server PC

Objective

To run ControlDesk and the system for automating ControlDesk via ASAM MCD-3 *on different PCs*, you have to configure the DCOM settings on the server PC.

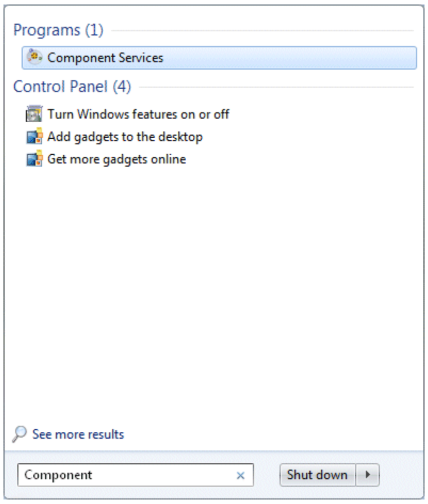
Precondition

You need administrator rights on the server PC. To check whether you have the required rights, refer to Required User Rights (Installing dSPACE Software 🕮).

Method

To configure DCOM settings on the server PC

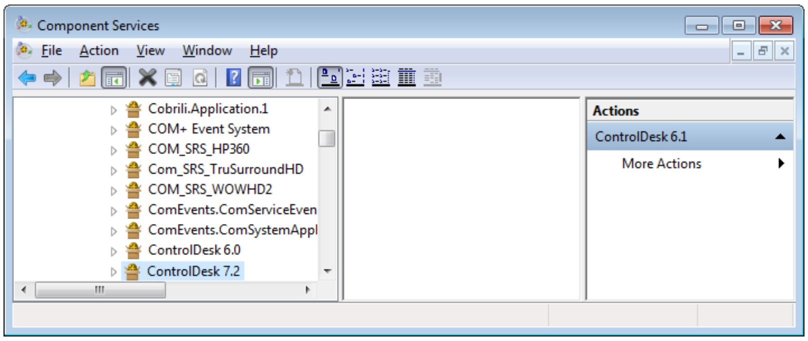
1 Search for Component Services, then click Component Services.



This opens the Component Services dialog.

2 In the Component Services dialog, navigate to the ControlDesk 7.4 entry.

See the following illustration as an example.



3 From the context menu of the ControlDesk 7.4 entry, select Properties.

This opens the ControlDesk 7.4 Properties dialog.

4 On the General page of the ControlDesk 7.4 Properties dialog, select "Default" as the Authentication Level.

5 On the Location page of the ControlDesk 7.4 Properties dialog, select the following checkboxes:

- Run application on the computer where the data is located.
- Run application on this computer.

Note

If the checkbox is grayed, select the Run application on the following computer checkbox, and specify the current PC.

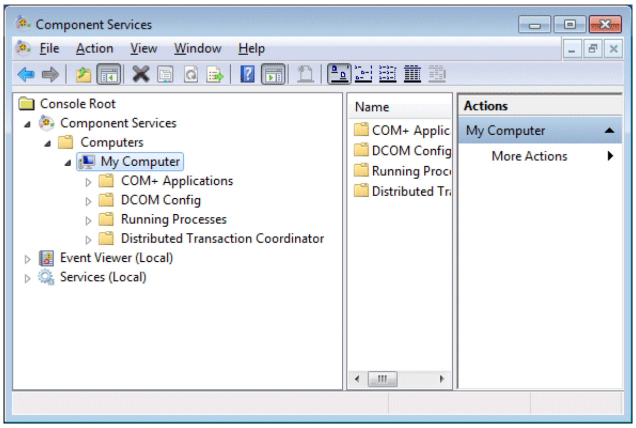
6 On the Security page of the ControlDesk 7.4 Properties dialog, select "Use Default" for each permission category.

7 On the Endpoints page of the ControlDesk 7.4 Properties dialog, keep the default settings.

8 On the Identity page of the ControlDesk 7.4 Properties dialog, select "The interactive user."

9 In the ControlDesk 7.4 Properties dialog, click OK to close the dialog.

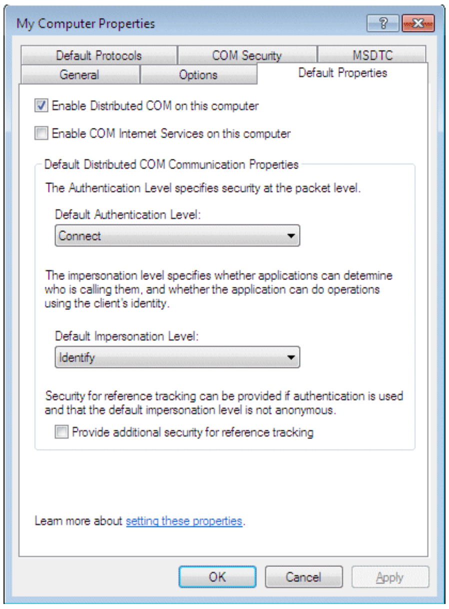
10 In the Component Services dialog, navigate to the My Computer entry.



11 From the context menu of the My Computer entry, select Properties.

This opens the My Computer Properties dialog.

12 On the Default Properties page of the My Computer Properties dialog, specify the settings as shown in the illustration below:



☑ Enable Distributed COM on this computer

☐ Enable COM Internet Services on this computer

Default Distributed COM Communication Properties

The Authentication Level specifies security at the packet level.

Default Authentication Level: Connect

The impersonation level specifies whether applications can determine who is calling them, and whether the application can do operations using the client's identity.

Default Impersonation Level: Identify

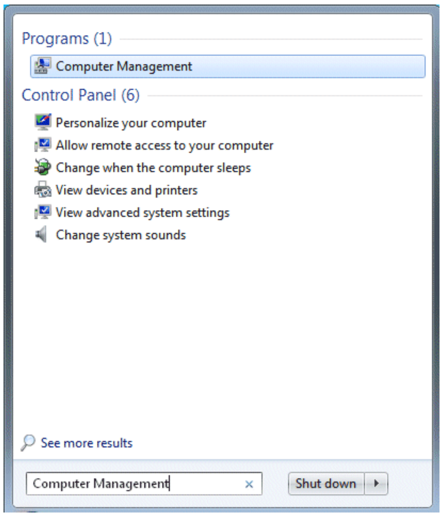
Security for reference tracking can be provided if authentication is used and that the default impersonation level is not anonymous.

☐ Provide additional security for reference tracking

Learn more about setting these properties.

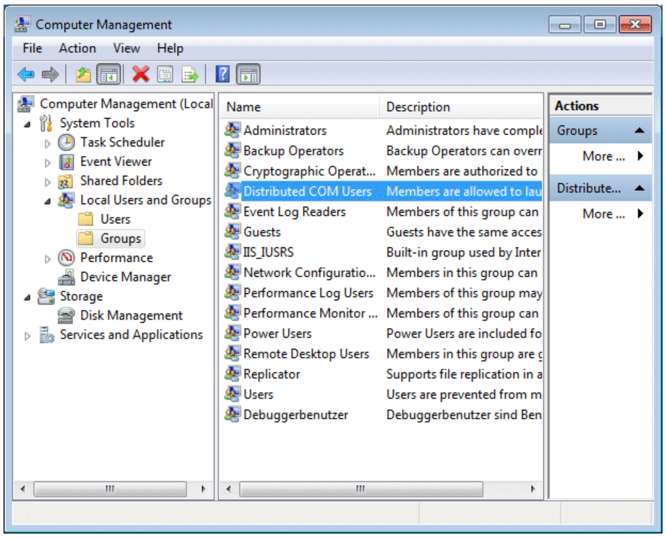
13 In the My Computer Properties dialog, click OK to close the dialog.

14 Search for Computer Management, then click Computer Management.


Programs (1)
Computer Management
Control Panel (6)
Personalize your computer
Allow remote access to your computer
Change when the computer sleeps
View devices and printers
View advanced system settings
Change system sounds
See more results
Computer Management
Shut down

This opens the Computer Management dialog.

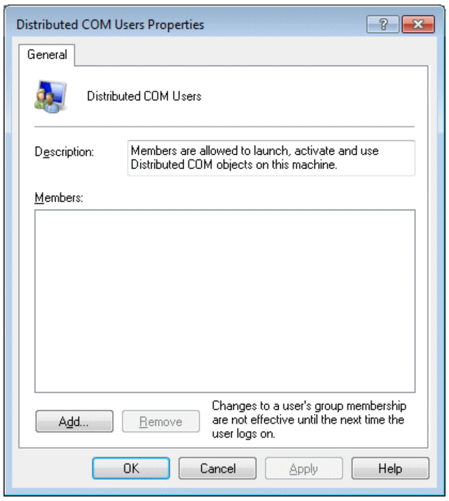
15 In the Computer Management dialog, navigate to the Distributed COM Users group below Local Users and Groups.



Name	Description
Administrators	Administrators have comple
Backup Operators	Backup Operators can overr
Cryptographic Operat...	Members are authorized to
Distributed COM Users	Members are allowed to lau
Event Log Readers	Members of this group can
Guests	Guests have the same acces
IIS_IUSRS	Built-in group used by Inter
Network Configuratio...	Members in this group can
Performance Log Users	Members of this group may
Performance Monitor ...	Members of this group can
Power Users	Power Users are included fo
Remote Desktop Users	Members in this group are g
Replicator	Supports file replication in a
Users	Users are prevented from m
Debuggerbenutzer	Debuggerbenutzer sind Ben

16 From the context menu of the Distributed COM Users group, select Properties.

This opens the Distributed COM Users Properties dialog.



Description: Members are allowed to launch, activate and use Distributed COM objects on this machine.

Members:

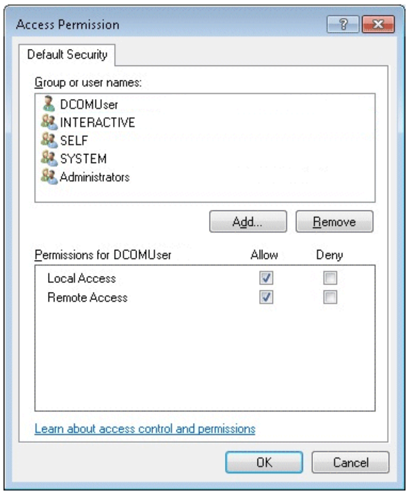
Changes to a user's group membership are not effective until the next time the user logs on.

17 In the Distributed COM Users Properties dialog, add all the users who should get access permissions, then click OK to close the Distributed COM Users Properties dialog.

18 On the COM Security page of the My Computer Properties dialog, click Edit Default below Access Permissions.

This opens the Access Permission dialog.

19 In the Access Permission dialog, add all the users who should get access permissions. Allow local access and remote access permissions for each user as shown in the illustration below:



Permissions for DCOMUser:

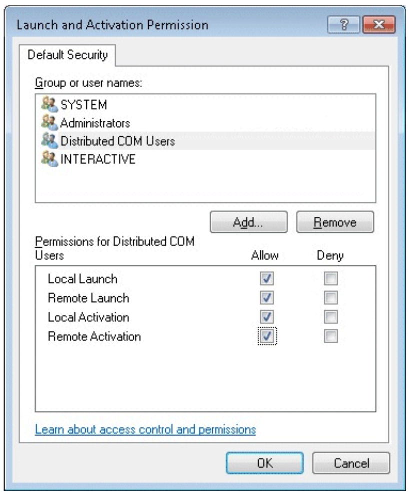
	Allow	Deny
Local Access	✓	
Remote Access	✓	

20 In the Access Permission dialog, click OK to close the dialog.

21 On the COM Security page of the My Computer Properties dialog, click Edit Default below Launch and Activation Permissions.

This opens the Launch and Activation Permission dialog.

22 In the Launch and Activation Permission dialog, add the Distributed COM Users group. Allow all the launch permissions as shown in the illustration below:



Permissions for Distributed COM Users:

	Allow	Deny
Local Launch	✓	
Remote Launch	✓	
Local Activation	✓	
Remote Activation	✓	

23 In the Launch and Activation Permission dialog, click OK to close the dialog.

24 In the My Computer Properties dialog, click OK to close the dialog.

Result

You have configured the DCOM settings on the server PC.

Next steps

- On the client PC(s), the *ControlDesk* product set must be installed.
- You should test whether the DCOM configuration on the server PC was successful. Refer to Testing the DCOM Configuration on page 27.

Related topics

Basics

Testing the DCOM Configuration.. 27

Testing the DCOM Configuration

Introduction

There are two ways to test whether the DCOM configuration on the server PC was successful.

Testing via AutomationDesk

To test the DCOM configuration via AutomationDesk, specify the IP address of the server PC via the System object of AutomationDesk's Remote Calibration (COM) library. Then try to connect to the server PC.

If the following error message does NOT occur, the remote access works correctly:

```
com_error: (-2147024891, 'Access is denied.', None, None)
```

Testing via MC3 demos

You can test the DCOM configuration via the MC3 Demos on page 28. To test whether the client PC can receive measurement data from the server PC, use one of the measurement demos.

To use the demos, replace the default IP address "127.0.0.1" (local host) by the IP address of the server PC.

Related topics

HowTos

How to Configure DCOM Settings on the Server PC.. 22

Demos for ControlDesk MCD-3 Automation

Where to go from here

Information in this section

MC3 Demos

Description of the demos

Several Python demo scripts show how to use the commands of the `asammc3` Python library to remote-control ControlDesk.

Remote-controlling the CalDemo project The demos are configured to automate measurement and calibration tasks with the *XCP on CAN* experiment of the CalDemo project. To automate measurement tasks, you have to start the CalDemo ECU.

Demo scripts for calibration and measurement The table below lists the calibration and measurement demo scripts, and the actions the scripts automate:

Demo Script	Description
`CalibrationDemoScalar.py`[1)]	Shows how to read and write scalar characteristics.
`CalibrationDemoCurve.py`[1)]	Shows how to read and write CurveCharacteristics.
`CalibrationDemoMap.py`[1)]	Shows how to read and write MapCharacteristics.
`MeasurementDemo.py`[1)]	Shows how to measure values via collector events.
`MeasurementDemoPolling.py`[1)]	Shows how to measure values by polling the collector.
`MeasurementDemoRecording.py`[1)]	Shows how to record values.

1) The script contains named objects corresponding to the CalDemo project and the CalDemo ECU.

Demo scripts accessing the database side of the ASAM MCD-3 interface The table below lists the demos accessing the database side of the ASAM MCD-3 interface, and the actions the scripts automate:

Demo Script	Description
`ListDbObjectsDemo.py`	▪ The `ShowDbLogicalLink` function shows how to iterate through the projects, logical links and binaries available in the MCSystem. ▪ The `ShowDbObjects` function loads the first project available in the system and shows its contents, which is basically the contents of one A2L file.
`ListDbObjectsDemo_Devices.py`[1]	The `PrintDeviceInfo` function shows how to list devices and variable descriptions of an experiment.
`ListDbObjectsDemo_VariableInfo.py`[1]	The `PrintDbLocationInfo` function shows how to list names and properties of variables in an experiment.

[1] The script contains named objects corresponding to the CalDemo project and the CalDemo ECU.

Tip

You can translate the Python demo scripts into other programming languages, such as C#, Visual Basic, and MATLAB M-files. As a starting point, you can use the demo source files in `.\Demos\MC3\<ProgrammingLanguage>` (available for C#, MATLAB and VB). With these demo source files, you should be able to translate the Python demo scripts into the language of your choice.

For further information, refer to Translating Python Code into Different Programming Languages on page 66.

Location of the demos

All the Python demo scripts are located in the `.\Demos\MC3\Python` folder of your ControlDesk installation.

Related topics

Basics

Basics on Automating Measurement and Calibration via the ControlDesk MCD3 Interface..................... 12

D3 Demo

Description of the demo

The `MCD3D_v2_0_2_DiagDemo.py` demo script shows how to use the commands of the `asamd3` Python library to remote-control ControlDesk.

The demo script accesses ControlDesk's ASAM MCD-3 D interface via the `D3System202` interface. Using the `D3System202` interface, you can access the configuration of an ECU Diagnostics device in a ControlDesk experiment. The ControlDesk experiment represents the D3 project.

Remote-controlling the DiagDemo project The demo script is configured to automate the DiagDemo project (refer to ECU Diagnostics Demo (ControlDesk Introduction and Overview 🕮)). It uses the ECU Diagnostics device configuration in the *ECU Diagnostics (MCD-3D v2.0.2)* experiment (`MCD3D_v2_0_2_DiagDemo.py`) of the DiagDemo project.

The demo script automates the following actions:

1. Creating the DSystem object
2. Selecting the D3 project (ControlDesk experiment)
3. Selecting vehicle information
4. Printing ODX database content for the selected vehicle
5. Creating and opening the logical link
6. Configuring the logical link
7. Executing some diagnostic services
8. Executing a single-ECU job
9. Deinitializing the DSystem object

Location of the demo

The demo script is located in the `.\Demos\D3\Python\DiagDemo` folder of your ControlDesk installation.

Related topics

Basics

Basics on Automating ECU Diagnostics via the ControlDesk MCD3 Interface........................ 15

HowTos

How to Perform Diagnostic Tasks.. 65

Programming ControlDesk MCD-3 Automation

Where to go from here

Information in this section

Automating ControlDesk's Project Management

Where to go from here

Information in this section

Basics of Automating ControlDesk's Project Management

ControlDesk's project management

In ControlDesk, you use projects and experiments to structure calibration tasks. The main tasks of ControlDesk's project management are to define projects and experiments, and to open and activate them.

Related ControlDesk documentation For more information on using projects and experiments in ControlDesk, refer to Basics on Projects and Experiments (ControlDesk Project and Experiment Management 🕮).

Automating ControlDesk's project management

Automating ControlDesk's project management means activating existing experiments. The basis of the automation process are projects and experiments that were created in ControlDesk.

Defining projects and experiments

Projects and experiments must be defined in ControlDesk. These steps in ControlDesk's project management are currently not intended to be automated.

ControlDesk project

In ControlDesk a project manages different experiments belonging together, such as the different tasks for calibrating a specific engine variant. It holds the experiments related to these tasks, and documents relevant to the entire project.

MCD 3 Project

In ASAM MCD-3, a Project is defined as "logical grouping of defined test installations selected by the user". This corresponds to a single experiment in ControlDesk.

The DbProjectDescriptions collection contains all experiments known to ControlDesk, not only those of a single ControlDesk project.

Terminology table

The terminology in the ASAM MCD-3 object model is different from that in ControlDesk. The following tables list similar objects and terms. Bear in mind that there is not always a one-to-one relationship between the terms. If you need more information on a term, refer to the MCD-3 Automation Glossary on page 76.

ControlDesk -> MCD 3 ASAM MCD-3 terms that correspond to ControlDesk terms:

ControlDesk Term(s)	ASAM MCD-3 Term(s)
Experiment	Project
Project	-

MCD 3 -> ControlDesk ControlDesk terms that correspond to ASAM MCD-3 terms:

ASAM MCD-3 Term(s)	ControlDesk Term(s)
Project	Experiment

Related topics

Basics

Basics on Projects and Experiments (ControlDesk Project and Experiment Management 📖)

HowTos

How to Activate an Experiment

Objective

A ControlDesk project usually contains several experiments. You have to activate the one you want to work with.

MCD 3 project

In the ASAM MCD-3 object model, an experiment belonging to an MCSystem is called an *MCProject*. An active experiment is an MCProject that was selected from the MCDbProjectDescriptions collection.

Restrictions

- In a multiclient system, the experiment activated first has to be used by every client. You cannot activate another experiment (MCProject) until the active experiment has been closed by all clients connected to this experiment.
- The DbProjectDescriptions collection contains all experiments known to ControlDesk. The name of the experiment you want to activate therefore must be unique over all ControlDesk experiments, independently of ControlDesk projects.

Method

To activate an experiment

1. Import modules for ControlDesk's ASAM MCD-3 MC interface.
2. Create an MCSystem object.
3. You can activate the MCProject (ControlDesk experiment) in two ways:
   - Select the MCProject via its ShortName (`SelectProjectByName`).
   - Select the MCProject by selecting an item from the DbProjectDecriptions by index number (`SelectProject`).

   Now you can select a platform/device and an ECU Image file and then run your calibration or measurement task.
4. Deselect and delete run-time objects (MCProject, MCSystem object).

Result

You activated an experiment.

Examples

Activation by experiment name The following example shows how to activate an experiment named "ExpA".

```
Project = System.SelectProjectByName("ExpA")
```

Activation by experiment index The following example shows how to activate an experiment via its index number.

```
Project= System.SelectProject(System.DbProjectDescriptions[0])
```

Script examples

In the `.\Demos\MC3\` folder of your ControlDesk installation, you can find various demo scripts that show how to use the commands of the `asammc3` Python library to remote-control ControlDesk. Refer to MC3 Demos on page 28.

Next steps

You can select a platform/device and a corresponding ECU Image file. Refer to How to Select a Platform/Device on page 36.

Related topics

Basics

Automating ControlDesk's Platform Management

Where to go from here

Information in this section

Basics of Automating ControlDesk's Platform Management

ControlDesk's platform management

In ControlDesk, platforms/devices are used for carrying out calibration and/or measurement tasks. They represent hardware components such as ECUs.

ControlDesk's platform management lets you add plaforms/devices to an experiment, configure the platforms/devices.

Related ControlDesk documentation For more information on managing platforms/devices in ControlDesk, refer to Basics of Platforms/Devices (ControlDesk Platform Management).

DbLogicalLink and DbBinary

To connect to a platform/device and go online via ControlDesk's ASAM MCD-3 MC interface, you have to use the ASAM MCD-3 interfaces DbLogicalLink and DbBinary.

LogicalLink

In the ASAM MCD-3 object model, a *LogicalLink* object handles "the logical and physical connection to an ECU (...) as well as the protocol specific tasks (...) to measure with Collectors and to adjust Characteristics."

Since no default LogicalLink is created, you always have to select one explicitly.

Binary

The Binary is a program version that is assigned to a LogicalLink. It represents a specific program version of an ECU, contained in an ECU Image file.

A Binary can be changed only by instantiating a new LogicalLink.

Terminology table

The terminology in the ASAM MCD-3 object model is different from that in ControlDesk. The following tables list similar objects and terms. Bear in mind that there is not always a one-to-one relationship between the terms. If you need more information on a term, refer to the MCD-3 Automation Glossary on page 76.

ControlDesk -> MCD 3 ASAM MCD-3 terms that correspond to ControlDesk terms:

ControlDesk Term(s)	ASAM MCD-3 Term(s)
Platform/Device	Logical Link
	Module
ECU Image file	Binary

MCD 3 -> ControlDesk ControlDesk terms that correspond to ASAM MCD-3 terms:

ASAM MCD-3 Term(s)	ControlDesk Term(s)
Binary	ECU Image file
Logical Link	Platform/Device

Related topics

Basics

Basics of Platforms/Devices (ControlDesk Platform Management 📖)

HowTos

How to Select a Platform/Device

Objective

To use a platform/device for calibration or measurement, you have to select it. In ASAM MCD-3 terms, you have to create a LogicalLink by selecting a DbLogicalLink and a corresponding DbBinary.

Logical Link and Binary

According to the ASAM MCD-3 MC object model, you use a Logical Link object to connect to a platform/device and a Binary to get data for variable values. For more information, refer to Basics of Automating ControlDesk's Platform Management on page 35.

Method

To select a platform/device

1. Import modules for ControlDesk's ASAM MCD-3 MC interface.
2. Create an MCSystem object.
3. Select an MCProject (ControlDesk experiment) via its ShortName or index number.
4. Select a Logical Link (ControlDesk platform/device) via its ShortName or index number.
5. Select a Binary via its ShortName or index number.
6. Create a run-time LogicalLink.
   Now you can go online and run your calibration or measurement task.
7. Delete or deselect run-time objects (MCProject, LogicalLink, MCSystem object).

Result

The platform/device is active and can be used for calibration tasks.

Examples

Selection by name The following example shows how a Logical Link and a Binary are selected via their names. The example refers to the CalDemo "XCP on CAN" experiment.

```
UsedDbLogicalLink = Project.DbProject.DbVehicleInformations[0].DbLogicalLinks.GetItemByName("XCP")
UsedDbBinary = UsedDbLogicalLink.DbLocation.GetItemByName("CalDemo")
SelectedLogicalLink = Project.LogicalLinks.AddByNames(UsedDbLogicalLink.ShortName,UsedDbBinary.ShortName)
```

Selection by index The following example shows how a Logical Link is selected via its index number. Data is taken from the first Binary in the DbBinary collection.

```
UsedDbLogicalLink = Project.DbProject.DbVehicleInformations[0].DbLogicalLinks[0]
UsedDbBinary = UsedDbLogicalLink.DbLocation.DbBinaries[0]
SelectedLogicalLink = Project.LogicalLinks.AddByNames(UsedDbLogicalLink.ShortName,UsedDbBinary.ShortName)
```

Script examples

In the `.\Demos\MC3\` folder of your ControlDesk installation, you can find various demo scripts that show how to use the commands of the `asammc3` Python library to remote-control ControlDesk. Refer to MC3 Demos on page 28.

Next steps

You can start online calibration to change parameter values or start measuring.

- For information on starting online calibration, refer to How to Start Online Calibration on page 37.
- For information on calibrating parameters, refer to Automating ControlDesk's Calibration Features on page 42.
- For information on measuring, refer to Automating ControlDesk's Measurement and Recording Features on page 48.

Related topics

Basics

Basics of Automating ControlDesk's Platform Management...35

How to Start Online Calibration

Objective

After selecting a platform/device, you can calibrate parameters offline. If you want to calibrate parameters online or perform a measurement, online calibration must be started.

Online calibration

In ASAM MCD-3 terms, starting online calibration means "connecting to the Module". Stopping online calibration means disconnecting from it.

Starting and stopping online calibration is done via the LogicalLink object.

Resolving inconsistencies

The data loaded to the MC system via the Binary object might not be consistent with the data on the platform/device memory. When you start online calibration, you can resolve data inconsistencies in two ways.

Resolving inconsistencies by uploading data To resolve data inconsistencies, you can upload data from the platform/device memory to ControlDesk with the `LogicalLink.ConnectToModule(eLT_UPLOAD)` option.

Resolving inconsistencies by downloading data To resolve data inconsistencies, you can download data from ControlDesk to the platform/device memory with the `LogicalLink.ConnectToModule(eLT_DOWNLOAD)` option.

Preconditions

To start online calibration, a platform/device must be selected. You must have performed all the steps to create a run-time LogicalLink. Refer to How to Select a Platform/Device on page 36.

Method

To start online calibration

1 Connect to the Module.

Result

ControlDesk is online and can be used for online calibration and measurement tasks.

What happens if ControlDesk cannot start online calibration because the platform/device is not connected, is platform-/device-specific. Some platforms/devices interrupt the automation process by showing an error message, others do not interrupt it but send data when they are started.

Example

The following example shows how to connect to a Module.

```
SelectedLogicalLink.ConnectToModule(asammc3.constants.eLT_DOWNLOAD)
```

Script examples

In the `.\Demos\MC3\` folder of your ControlDesk installation, you can find various demo scripts that show how to use the commands of the `asammc3` Python library to remote-control ControlDesk. Refer to MC3 Demos on page 28.

Next steps

You can now start online calibration or measurement.

- For information on calibrating a parameter, refer to Automating ControlDesk's Calibration Features on page 42.
- For information on measuring, refer to Automating ControlDesk's Measurement and Recording Features on page 48.

Related topics

Basics

Automating the Handling of Variables

Where to go from here

Information in this section

Basics of Automating the Handling of Variables

ControlDesk's variable handling

In ControlDesk, all variables are listed in the **Variables** controlbar. You can sort and filter them and open a dialog to show their properties.

Related ControlDesk documentation For more information on ControlDesk's variable management, refer to Handling Variable Descriptions (ControlDesk Variable Management 📖).

Database project

The information on parameters and measurement variables is included in the database part of the ASAM MCD-3 object model.

To get information on the variables, you do not have to create run-time objects. You can load a database project (DbProject) and browse the contents of the variable description file.

Location

The ASAM MCD-3 object model defines the DbLocation object as "the entry point for available (...) Measurements, Characteristics, CompuMethods and so on. (...) To each Logical Link belongs exactly one Location (...) and one Interface (MC) according to the entry in the Logical Link Table."

For the measurement and calibration part, the DbLocation mainly represents the information contained in the variable description file (A2L).

Module

There are a number of Location classes for the diagnostics part and one for the measurement and calibration part, which is called a Module.

Accessing variables

To access parameters, measurement variables, axis points and other variable-specific objects, you must specify a DbLocation object. You can access variables in two ways.

Accessing variables by selecting a LogicalLink You can access variables implicitly by selecting a LogicalLink. As exactly one DbLocation belongs to each DbLogicalLink, the DbLocation is specified implicitly by selecting a LogicalLink.

Accessing variables by selecting a DbLocation You can access variables explicitly by selecting a DbLocation. If you have loaded a database project (DbProject), you can select a DbLocation from the DbModuleLocations collection.

Terminology table

The terminology in the ASAM MCD-3 object model is different from that in ControlDesk. The following tables list similar objects and terms. Bear in mind that there is not always a one-to-one relationship between the terms. If you need more information on a term, refer to the MCD-3 Automation Glossary on page 76.

ControlDesk -> MCD-3 ASAM MCD-3 terms that correspond to ControlDesk terms:

ControlDesk Term(s)	ASAM MCD-3 Term(s)
Axis	Axis
Computation method	CompuMethod
Conversion table	CompuTab
Curve	CurveCharacteristic
Map	MapCharacteristic
Measurement variable	Measurement
Parameter	Characteristic
Raster	Rate
Variable description	DbLocation Module
Value	ScalarCharacteristic
Verbal conversion	CompuVTab
Verbal conversion range	CompuVTabRange

MCD-3 -> ControlDesk ControlDesk terms that correspond to ASAM MCD-3 terms:

ASAM MCD-3 Term(s)	ControlDesk Term(s)
Axis	Axis
Characteristic	Parameter
CompuMethod	Computation method
CompuTab	Conversion table

ASAM MCD-3 Term(s)	ControlDesk Term(s)
CompuVTab	Verbal conversion
CompuVTabRange	Verbal conversion range
CurveCharacteristic	Curve
DbLocation	Variable description
MapCharacteristic	Map
Measurement	Measurement variable
Module	Variable description
Rate	Raster
ScalarCharacteristic	Value

Related topics

Basics

Handling Variable Descriptions (ControlDesk Variable Management 🕮)

HowTos

How to List Properties of Variables

Objective

If you need information on variables in the active variable description, for example, their upper and lower limits, you can list variables and show their properties.

Database

You can query various database objects via the DbLocation object. For more information, refer to Basics of Automating the Handling of Variables on page 39.

Method

To list properties of variables

1. Import modules for ControlDesk's ASAM MCD-3 MC interface.
2. Create an MCSystem object.
3. Load a DbProject (ControlDesk experiment).
4. Select a DbLocation (includes variable description) from the DbModuleLocations collection.
5. Show the properties of variable objects of the DbLocation by querying their properties.
6. Close the DbProject.

Result

You listed the properties of variables of the selected platform/device.

Script examples

In the `.\Demos\MC3\` folder of your ControlDesk installation, you can find various demo scripts that show how to use the commands of the `asammc3` Python library to remote-control ControlDesk. Refer to MC3 Demos on page 28.

Related topics

Basics

Automating ControlDesk's Calibration Features

Where to go from here

Information in this section

Basics of Automating ControlDesk's Calibration Features

ControlDesk's calibration features

In ControlDesk, you calibrate parameters by changing their values in instruments on a layout. ControlDesk offers specialized instruments for editing variable types, for example, the Variable Array for scalar parameters and the Table Editor for nonscalar parameters.

Related ControlDesk documentation For more information on the calibration of parameters in ControlDesk, refer to Variable Types That can be Calibrated (ControlDesk Calibration and Data Set Management 📖).

Run-time Characteristic

In the ASAM MCD-3 MC object model, *Characteristic* is the generic term for parameters.

The term additionally includes the variable type of the parameter:
- ScalarCharacteristic
- CurveCharacteristic
- MapCharacteristic

Adding a Characteristic The database part of the MCProject contains the information on a Characteristic. After you create a run-time LogicalLink, you have to select a DbCharacteristic which serves as a template for the run-time Characteristic. You create a run-time Characteristic by adding the DbCharacteristic to the Characteristics collection of the run-time LogicalLink.

You can add a Characteristic only once. Any further attempt to add the same DbCharacteristic to the Characteristics collection will result in an error. In this case, use `Characteristics.GetItemByName` to retrieve the Characteristic object.

Variable types

You can calibrate different elements depending on the variable type of the parameter.

Parameter Type	Characteristic	Elements
Scalar	ScalarCharacteristic	Value
Curve	CurveCharacteristic	X-axis points Function values
Map	MapCharacteristic	X-axis points Y-axis points Function values

Changing values

The general steps for automating parameter calibration are the same for each variable type until it actually comes to changing the value.

You can change the value of a parameter with the `Write` method of the Characteristic object. The current value of a parameter can be read with the `Read` method.

Note

In contrast to the specifications for the ASAM MCD-3 object model, the `Read` method of CurveCharacteristics and MapCharacteristics does not return MCDValue objects.
ScalarCharacteristics provide two `Read` methods:
- `Read`: To return an MCDValue object.
- `ReadVariant`: To return a variant value.

The methods `Write` and `WriteVariant` work accordingly.

The writing action is specified by the value type that is one input parameter of the Write method.

Writing Action	Value Type
Writing absolute values	`asammc3.constants.eVT_VAL`
Writing a constant value to a range of a map or curve	`asammc3.constants.eVT_CONST`
Adding a constant value to a range of a map or curve	`asammc3.constants.eVT_OFFSET_POS`
Subtracting a constant value from a range in a map or curve	`asammc3.constants.eVT_OFFSET_NEG`

To write a list of values, the value type must be `asammc3.constants.eVT_VAL`.

Representation type You can specify the conversion mode for each value you read or write by selecting a representation type.

Conversion Mode	Description	Representation Type (`RepType`)
Source	Value in its original format on the hardware	`asammc3.constants.eRT_ECU`
Converted	Value in a converted form (Default)	`asammc3.constants.eRT_PHYSICAL`

If no representation type is specified, values are processed as *converted values*.

Start and stop index When writing 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 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. See the code examples in the table below.

Input parameters The following table shows the input parameters of the Write method for the various parameter types.

Parameter Type	Input Parameters	Code Examples

ScalarCharacteristic	Value
	ValueType
	RepType (default = eRT_PHYSICAL)
		To change a parameter value to 0.9 (representation type = eRT_PHYSICAL):
		`Characteristic.Write(0.9, asammc3.constants.eVT_Val)`
		To change a parameter value to 8 (representation type = eRT_ECU):
		`Characteristic.Write(8, asammc3.constants.eVT_Val, asammc3.constants.eRT_ECU)`
		To change a value of a parameter with verbal conversion:
		`Characteristic.Write(“high", asammc3.constants.eVT_Val)`

CurveCharacteristic	Value
	StartIndex
	StopIndex
	ValueType
	RepType
		To add a constant value (0.5) to the axis points at the index positions 3 to 8:
		`Characteristic.Axis.Write(0.5,3,9, asammc3.constants.eVT_CONST)`
		To write 3 values (4,5,6) to the function values at the index positions 2 to 4:
		`Characteristic.Value.Write([4,5,6],2,5, asammc3.constants.eVT_VAL)`

Axis of a MapCharacteristic	Value
	StartIndex
	StopIndex
	ValueType
	RepType
		To subtract 4 from each x-axis point (0,-1):
		`Characteristic.XAxis.Write(4,0,-1, asammc3.constants.eVT_OFFSET_NEG)`

Parameter Type	Input Parameters	Code Examples

Function values of a MapCharacteristic	Value
	X-StartIndex
	X-StopIndex
	Y-StartIndex
	Y-StopIndex
	ValueType
	RepType
		To write 6 new values to function values (table cells 1 to 3 in the first and second table rows):
		`Characteristic.Write([[3,3,3],[2,3,4]],`
		`0,3,0,2, asammc3.constants.eVT_VAL)`

Parameters with verbal conversion If you want to calibrate a parameter that uses a verbal conversion table as computation method, you can use the defined strings to write a value. The following example shows how to calibrate a parameter with the following verbal conversion table: 1 = low, 2 = medium, 3 = high.

Example

```
DbCompuMethod = Characteristic.DbObject.RefDbCompuMethod
if DbCompuMethod:
  # Checking if a verbal conversion table is used
  if asammc3.constants.eCT_TAB_VERB == DbCompuMethod.\
     ConversionType:
  print("Value of %-10s is : %s" % (Characteristic.DbObject.\
     ShortName, Characteristic.ReadVariant()))
  Characteristic.Write("low", asammc3.constants.eVT_VAL)
  print("New Value of %-10s is : %s" % (Characteristic.\
     DbObject.ShortName, Characteristic.ReadVariant()))
```

Terminology table

The terminology in the ASAM MCD-3 object model is different from that in ControlDesk. The following tables list similar objects and terms. Bear in mind that there is not always a one-to-one relationship between the terms. If you need more information on a term, refer to the MCD-3 Automation Glossary on page 76.

ControlDesk -> MCD 3 ASAM MCD-3 terms that correspond to ControlDesk terms:

ControlDesk Term(s)	ASAM MCD-3 Term(s)
Axis	Axis
Curve	CurveCharacteristic
Computation method	CompuMethod
Conversion mode	Representation type
Function value	Value
Map	MapCharacteristic
Parameter	Characteristic
Value	ScalarCharacteristic

MCD 3 -> ControlDesk terms: ControlDesk terms that correspond to ASAM MCD-3

ASAM MCD-3 Term(s)	ControlDesk Term(s)
Axis	Axis
Characteristic	Parameter
CompuMethod	Computation method
CurveCharacteristic	Curve
MapCharacteristic	Map
Representation type	Conversion mode
ScalarCharacteristic	Value

Related topics

Basics

Variable Types That can be Calibrated (ControlDesk Calibration and Data Set Management 📖)

HowTos

How to Calibrate a Parameter

Objective

The general steps you have to perform to calibrate a parameter are the same for each parameter type.

Differences for parameter types

You can change a parameter value with the Write method of the Characteristic object. The input parameters of the Write method are different for the various parameter types. Refer to Input parameters on page 44.

Restrictions

The following restrictions apply to the automated calibration of multidimensional parameters (curves and maps).

No monotony violation If you change axis points in ControlDesk's Table Editor and the new values violate the monotony of axis points, the Monotony Violation dialog opens. The dialog lets you confirm monotony violation and write the values. However, such an intentional violation of axis monotony is not possible in an automated calibration task.

Interpolation of function values If you change axis points in ControlDesk's Table Editor, you have to choose between interpolation and no interpolation of the function values. However, in an automated calibration task, the function values remain the same. The interpolation of function values according to the changed axis points cannot be automated.

No common axis warning If you change the axis points of a common axis, you get no warning that referring variables are affected.

Source format and conversion tables If a variable uses a conversion table as computation method, the ranges of the table will be ignored when writing values in source format. No warning is generated if a value is outside the defined ranges.

When writing values in source format, no warning is generated if values written are not in range of a conversion table.

Method

To calibrate a parameter

1. Import modules for ControlDesk's ASAM MCD-3 MC interface.
2. Create an MCSystem object.
3. Select an MCProject (ControlDesk experiment) via its ShortName or index number.
4. Select a DbLogical Link (ControlDesk platform/device) via its ShortName or index number.
5. Select a DbBinary (ControlDesk ECU Image file) via its ShortName or index number.
6. Create a run-time Logical Link (ControlDesk platform/device).
7. Prepare online calibration.
8. If your ECU or ECU interface provides two memory pages, activate the working data set.

   Note

   An exception is thrown if you try to access a nonexistent memory page.
   - If you are accessing a single-page ECU or a platform, skip this step.
   - To avoid an exception being thrown, encapsulate the memory page access in a `try … except` clause.

9. Create one or more run-time Characteristics (ControlDesk parameters).
10. Write a new value for the Characteristic.
11. Deselect and delete run-time objects (MCProject, LogicalLink, MCSystem object).

Result

The value of the selected parameter was changed.

Script examples

In the `.\Demos\MC3\` folder of your ControlDesk installation, you can find various demo scripts that show how to use the commands of the `asammc3` Python library to remote-control ControlDesk. Refer to MC3 Demos on page 28.

Next steps

You can measure the results of your calibrations. Refer to How to Measure Variables on page 55.

Related topics

Basics

Automating ControlDesk's Measurement and Recording Features

Where to go from here

Information in this section

Basics on Automating ControlDesk's Measurement and Recording Features

ControlDesk's measurement and recording features

In ControlDesk, you select the variables to be measured and recorded by adding them to the measurement signal list. The selected variables are displayed in the ControlDesk **Measurement Configuration** controlbar to specify measurement and recording.

Related ControlDesk documentation For more information on performing measurements and recordings in ControlDesk, refer to Measuring Data (ControlDesk Measurement and Recording) and Recording Data (ControlDesk Measurement and Recording).

Collector

To configure and run measurements or recordings via the ASAM MCD-3 interface, you must create Collector objects.

CollectedObjects list

In ControlDesk, you first have to select the variables to be measured or recorded by adding them to the measurement signal list. According to the ASAM MCD-3 object model, you have to add these variables to the CollectedObjects list of a Collector. Measurements (including measurement arrays) and scalar parameters can occur only once per CollectedObjects list.

Note

Multidimensional parameters can occur multiple times, if each occurrence defines a different range to be measured (the measurement and recording of multidimensional parameters is currently not supported).

Platform-/device- and raster-specific Collectors

In ControlDesk you have only one measurement signal list for all platforms/devices and measurement rasters. In the ASAM MCD-3 object model, you have to create a new Collector with its own CollectedObjects list for each platform/device and each raster that you want to include in measurement and recording.

One Collector for each platform/device To measure signals on a particular platform/device, you have to create at least one Collector on the LogicalLink representing the platform/device. The DbLocation that goes with the LogicalLink determines the variables that are generally available for measurement on the selected platform/device.

One Collector for each raster All the variables in a CollectedObjects list must have the same raster setting. You must create a new Collector for each new measurement raster.

You can therefore have multiple Collectors for a single LogicalLink.

For example, if you want to use two different interfaces to the ECU and in one case want to measure with two different rasters, you must create three Collectors as shown in the following table:

	Collector A	**Collector B**	**Collector C**
ECU	ECU 1	ECU 1	ECU 1
Device	XCP on CAN	DCI-GSI2	DCI-GSI2
Measurement raster	5 ms	5 ms	10 ms
CollectedObjects list	▪ Variable_A ▪ Variable_B ▪ Variable_C	▪ Variable_A ▪ Variable_E ▪ Variable_F	▪ Variable_A ▪ Variable_E ▪ Variable_F

	Collector A	Collector B	Collector C
			▪ Variable_G

States of a Collector

Each Collector runs through the following states during a measurement or recording task.

State	Description
e_CREATED	When a new Collector is created, its CollectedObjects list is empty. You can add simple measurement variables, measurement arrays and parameters (ControlDesk: scalar parameters only) to the CollectedObjects list. All configuration steps have to be done in this state.
e_CONFIGURED	The Collector configuration has been checked for valid settings, for example, that a valid rate has been set and the CollectedObjects list is not empty.
e_ACTIVATED	The platform/device the Collector belongs to is set online. The CollectedObjects prepare to receive data from ControlDesk.
e_STARTED	The platform/device is measuring and transferring data to the Collector.

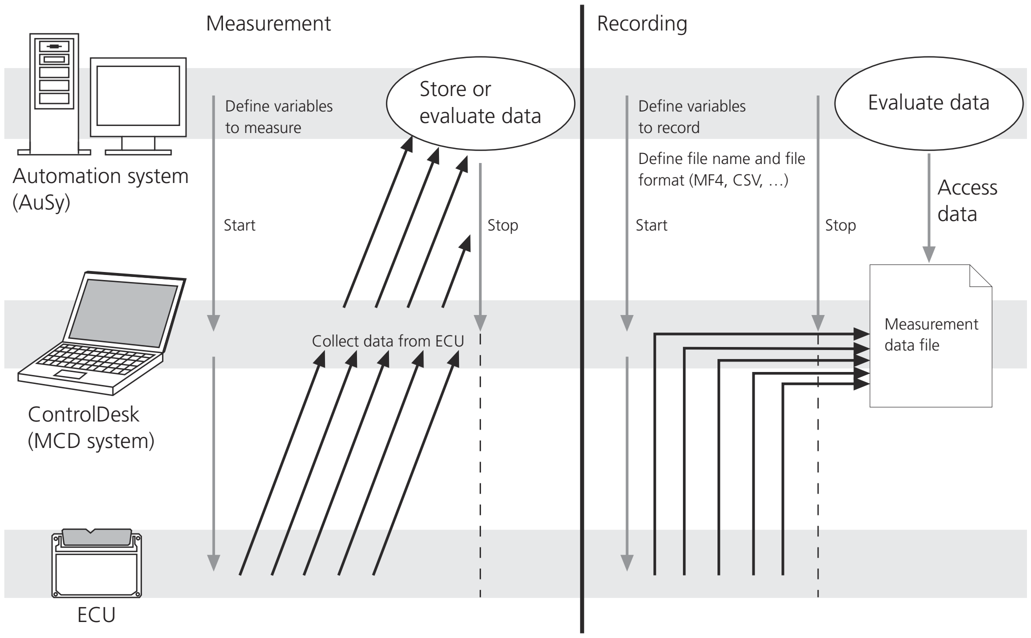
State transition events

All state transitions are reported to the automation system (AuSy) by events, if the client application has subscribed to receive these events. In the Collector states e_ACTIVATED and e_STARTED, the platform/device (LogicalLink) must be online. Any attempt to switch offline while a Collector is running is ignored.

File storing

Usually the data of automated measurements is transferred back from the MC system to the AuSy, where it can be evaluated and stored. You can omit this retransfer by using ControlDesk's recording function.

Automated ControlDesk recordings are stored via the MC system (ControlDesk side), not via the AuSy (application side). However, you can store the recording on a network drive. This can be a hard disk of the AuSy, if it is accessible from the MC system with write permission.



Note

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.

Collector properties

The following table shows various Collector properties.

Collector Property	Description	Code Example

CollectedObjects list	Defines the variables to be measured	To add the variable SignalGenOuput:
		`DbMeasurement =`
		`  DbLocation.DbMeasurements.\`
		`  GetItemByName("SignalGenOutput")`
		`Collector.CollectedObjects.\`
		`  AddScalar(DbMeasurement,\`
		`  asammc3.constants.eCDT_SCALAR_\`
		`  MEASUREMENT)`

TimeStamping	If enabled, time information is given at the beginning of each new sample of acquisition data	To activate TimeStamping: `Collector.Buffer.TimeStamping = True`

Collector Property	Description	Code Example

Rate	Measurement raster setting. Defines the time interval between single measurements of a value	To set the measurement raster according to the first defined one for the LogicalLink:
		`Collector.Buffer.Rate =`
		`SelectedLogicalLink.RateInfos[0]`

DownSampling	If you want to reduce the amount of measurement data, you can specify a downsampling factor. A downsampling factor of 2 means that only every second data sample according to the data acquisition rate is saved to the measurement buffer. All data samples in between are ignored (no interpolation).
	In an automated recording via ControlDesk the downsampling factor is ignored. Refer to File storing on page 50.
		To store each data sample:
		`Collector.Buffer.DownSampling = 1`
		To store every third data sample:
		`Collector.Buffer.DownSampling = 3`

Buffer size	Specifies the number of Result lines to be stored 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. Default setting: 1 sample line.	To set the buffer size to 60 Result objects: `Collector.Buffer.Size = 60`

Collector methods

The following table shows various methods provided by the Collector.

Collector Method	Description	Code Example
CreateClientID\ForPolling	You need a ClientID to poll measurement values.	`Collector.CreateClientIDForPolling()`
ConfigureStorage	You can select between *transferring the measured data to the AuSy* and *storing measured data in a file on the MC system*.	

	(a) Transferring the measured data to the AuSy.
	The second parameter (`FormatIdOrNoOfSamples`) specifies the number of samples used for the OnCollectorResultReady event (see Collector events on page 53). In polling mode, this parameter is ignored.
		To transfer the data to the AuSy:
		`Collector.ConfigureStorage\`
		`(asammc3.constants.eST_AUSY,\`
		`10, "")`

	(b) Storing the measured data in a measurement data file via the MC system (ControlDesk side).
	The second parameter (`FormatIdOrNoOfSamples`) specifies the user-defined storage format.
	For more information and limitations, refer to File storing on page 50.
		To store the measured data in an MF4 file on the MC system:
		`Collector.ConfigureStorage\`
		`(asammc3.constants.eST_File,\`
		` asammc3.constants.eFID_USER,\`
		`FileName)`
		In this example, the format ID (`asammc3.constants.eFID_USER`) is set. For ControlDesk, this specifies to store the data in a measurement data file. Valid are all formats, that are supported by ControlDesk, such as MF4, CSV, or MAT.

Collector Method	**Description**	**Code Example**
		The file name extension lets you specify the concrete file format: `MyMC3System.ConfigureCollector(FileName = \` `os.getcwd() + "\\recording.mf4")`
GetFillingLevel	Returns the current number of 'result lines' in the measurement buffer for a particular client identified by its ClientID.	To get the filling level for the client: `Collector.GetFillingLevel(ClientID)`
Check	To check the Collector configuration. Sets the Collector to the eConfigured state.	`Collector.Check()`
Activate	Sets the Collector to the state eActivated.	`Collector.Activate()`
Start	To start transferring data to the AuSy or writing measurement data to a file, if the start condition is reached.	`Collector.Start()`
Suspend	To stop the data transfer to the AuSy.	`Collector.Suspend()`
Change	To set the Collector to eCreated mode. In this state you can configure the settings.	`Collector.Change()`
FetchResults	To transfer results from the buffer to the client. All transferred result lines are no longer available in the buffer for this client.	`Results = Collector.FetchResults(0, ClientID)`

Collector events

The Collector object provides various events. After registering an EventHandler, you can use events for various purposes:

- To get status information on a Collector
- To be notified of the existence of a new set of data to fetch
- To be notified of a buffer turnaround and possible loss of data

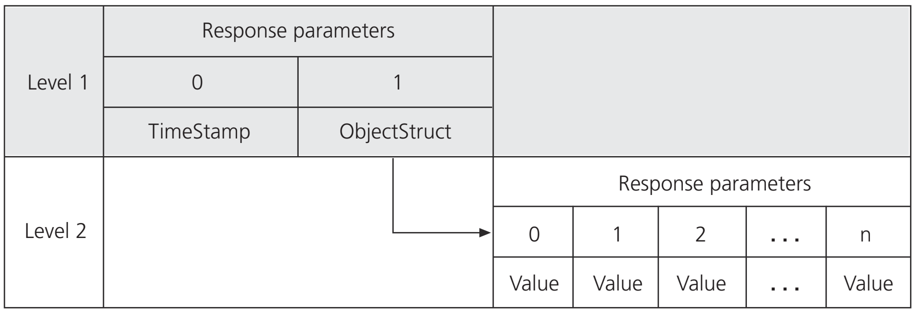
The following table shows the available events:

Event	**Description**
OnCollectorResultReady	One or more new results are available for the client. You can specify the number of the results needed to generate this event via the 'FormatIdOrNoOfSamples' parameter of the Collector method ConfigureStorage.
OnGlobalObjStarted	The Collector has changed its state from eActivated to eSTARTED.
OnGlobalObjActivated	The Collector has changed its state from eConfigured to eACTIVATED.
OnGlobalObjConfigured	The Collector has changed its state from eCreated to eCONFIGURED.
OnGlobalObjStopped	The Collector has changed its state from eSTARTED to eACTIVATED.
OnGlobalObjDeactivated	The Collector has changed its state from eStARTED or eACTIVATED to eCREATED.
OnGlobalObjChanged	The Collector has changed its state from eCONFIGURED to eCREATED.
OnCollectorError	The Collector reports a possible loss of data for a connected client. This means that the buffer is about to overwrite values for the client, because it has reached the position in the buffer that the client's last call to FetchResults ended with. If the client is now fast enough to call FetchResults, no data will be lost. Another scenario for an OnCollectorError event is this: A client connects to the Collector after another client has started it. If the buffer already turned around, the newly connected client receives the OnCollectorError event to notify it that it has missed an entire buffer cycle.

Result objects

A Result object is a single line of a Collector's buffer. As Collectors are platform-/device-specific, the contents of a line consist of the values of all the CollectedObjects measured on the platform/device that the Collector is associated with at one time. As this is also the definition of a Response, in this case a Result object contains exactly one Response.

One single item in the line is called a Response parameter. If time-stamping is enabled, the first Response parameter is the time information (TimeStamp). The second Response parameter is structure information (ObjectStruct) as the entry point of the second Response parameter level.



In the case of measurement variables and scalar parameters, level 2 contains the measured values. Additionally, the ShortNames of the collected objects are provided. Clients can therefore call GetItemByName to retrieve the Response parameters they are interested in or identify single Response parameters in the Collection.

For a measurement array variable, a "Value" contains an array of data. This array is transported as a COM safearray and is either one-dimensional or contains sub-arrays (if the MATRIX_DIM keyword is assigned in the variable description).

In the case of multidimensional parameters, level 2 contains different structure information, leading to the 3rd level. For curves and maps the measured values are placed on level 4.

As ControlDesk does not support the measuring of multidimensional parameters, you only have to scan level 1 and level 2 for values. On level 1 you get the time information (if enabled) and on level 2 to you get the measured values.

Adding time information

If you need information on the absolute time a value has been measured at, for example to synchronize the measured data with data from other sources such as video cameras, you can capture the PC time of the MC system at the start of a measurement.

Terminology table

The terminology in the ASAM MCD-3 object model is different from that in ControlDesk. The following tables list similar objects and terms. Bear in mind that there is not always a one-to-one relationship between the terms. If you need more information on a term, refer to the MCD-3 Automation Glossary on page 76.

ControlDesk -> MCD-3 terms: ASAM MCD-3 terms that correspond to ControlDesk

ControlDesk Term(s)	ASAM MCD-3 Term(s)
Measurement	Collector
Measurement buffer	Buffer
Measurement signal list	CollectedObjects list
Recording	Collector
Raster	Rate
Time stamp	Time stamp

MCD-3 -> ControlDesk terms: ControlDesk terms that correspond to ASAM MCD-3

ASAM MCD-3 Term(s)	ControlDesk Term(s)
CollectedObjects list	Measurement signal list
Collector	Measurement Recording
Buffer	Measurement buffer
Downsampling	Downsampling
Rate	Raster
Response	Measurement buffer (content)
Result	Measurement buffer (content)
Time stamp	Time stamp

Related topics

Basics

HowTos

How to Measure Variables

Objective

You can automate the measurement of measurement variables and scalar parameters.

Collector

The ASAM MCD-3 object model provides the Collector object to configure and perform measurements and recordings.

For details, refer to Basics on Automating ControlDesk's Measurement and Recording Features on page 48.

Polling vs. event handling

You can specify the method to fetch measured data in two ways:

Polling The client controls the frequency of result access. You have to evaluate the optimal polling frequency.

- If the Collector is polled too often, an empty results structure is delivered.
- If the client polls too rarely, loss of data is inevitable.

You can control this circumstance to some extent by additionally using the GetFillingLevel method to verify if results are currently available.

Event handling Collector events are used to control the fetching of results. The client fetches data from the buffer only when it receives an OnCollectorResultReady event. This method is best suited for preventing loss of data.

For details on events, refer to Collector events on page 53.

Start sequence for multiple Collectors

If you use more than one Collector, you must first check and activate all the Collectors, before you can start them. See the following example for two Collectors.

1. `Collector1.Check`
   `Collector1.Activate`
2. `Collector2.Check`
   `Collector2.Activate`
3. `Collector1.Start`
   `Collector2.Start`

Note

If you have two separate client applications with one Collector each, you cannot apply this rule for technical reasons.

If both Collectors are connected to the same platform/device, but they do not measure the same variables, the second Collector that starts will receive data only if the following two conditions are met:

- The variables of the second Collector have been added to ControlDesk's measurement signal list.
- The raster setting of the Collector and the raster settings of its variables in the measurement signal list must be equal.

Otherwise the second Collector will receive no data at all.

Restrictions

- Multidimensional parameters like maps and curves cannot be measured.
- Measurements and ScalarCharacteristics can occur only once per CollectedObjects list, complex Characteristics can occur multiple times, if each occurrence defines a different range to be measured (measurement of complex Characteristics is currently not supported).

Method

To measure variables

1. Import modules for ControlDesk's ASAM MCD-3 MC interface.
2. Create an MCSystem object.
3. Select an MCProject (ControlDesk experiment) via its ShortName or index number.
4. Select a Logical Link (ControlDesk platform/device) via its ShortName or index number.
5. Select a DbBinary (ControlDesk ECU Image file) via its ShortName or index number.
6. Create a run-time Logical Link.
7. Start online calibration mode.
8. Create one or more Collectors (ControlDesk measurements).
9. Configure the measurement settings.
10. Start measuring: Check, activate and start the Collector.

    If you use more than one Collector, first check and activate all the Collectors and then start them.
11. Fetch measurement results to display them.
12. Stop measuring.
13. Destroy the Collector.
14. Stop online calibration mode.
15. Delete or deselect run-time objects (MCProject, LogicalLink, MCSystem object).

Result

The measured values are available at the client.

Script examples

In the `.\Demos\MC3\` folder of your ControlDesk installation, you can find various demo scripts that show how to use the commands of the `asammc3` Python library to remote-control ControlDesk. Refer to MC3 Demos on page 28.

Related topics

Basics

Basics on Automating ControlDesk's Measurement and Recording Features............................ 48

How to Record Variables

Objective

You can automate the recording of measurement variables and scalar parameters.

Collector

The ASAM MCD-3 object model provides the Collector object for configuring and performing measurements and recordings.

For details, refer to Basics on Automating ControlDesk's Measurement and Recording Features on page 48.

File storing

Automated ControlDesk recordings are stored via the MC system (ControlDesk side), not via the AuSy (application side). However, you can store the recording on a network drive. This can be a hard disk of the AuSy, if it is accessible from the MC system with write permission.

It is not possible to use a single collector for recording *and* data transfer to the client (polling or event handling) simultaneously. A recording is always stored in a single measurement file, even if you use multiple Collectors for it.

Start sequence for multiple Collectors

If you use more than one Collector, you must first check and activate all the Collectors, before you can start them. See the following example for two Collectors.

1. `Collector1.Check`
   `Collector1.Activate`
2. `Collector2.Check`
   `Collector2.Activate`
3. `Collector1.Start`
   `Collector2.Start`

Note

If you have two separate client applications with one Collector each, you cannot apply this rule for technical reasons.
If both Collectors are connected to the same platform/device, but they do not measure the same variables, the second Collector that starts will receive data only if the following two conditions are met:
- The variables of the second Collector have been added to ControlDesk's measurement signal list.
- The raster setting of the Collector and the raster settings of its variables in the measurement signal list must be equal.

Otherwise the second Collector will receive no data at all.

Restrictions

- Multidimensional parameters like maps and curves cannot be recorded.
- Automated ControlDesk recordings are stored via the MC system (ControlDesk side), not via the AuSy (application side).
- Measurements and ScalarCharacteristics can occur only once per CollectedObjects list, complex Characteristics can occur multiple times, if each occurrence defines a different range to be measured (measurement of complex Characteristics is currently not supported).

Method

To record variables

1. Import modules for ControlDesk's ASAM MCD-3 interface.
2. Create an MCSystem object.
3. Select an MCProject (ControlDesk experiment) via its ShortName or index number.
4. Select a Logical Link (ControlDesk platform/device) via its ShortName or index number.
5. Select a DbBinary (ControlDesk ECU Image file) via its ShortName or index number.
6. Create a run-time Logical Link.
7. Start online calibration mode.
8. Create one or more Collectors (ControlDesk measurements).
9. Configure the measurement settings. The file name extension specifies the file format.
10. Start recording: Check, activate and start the Collector.

    If you use more than one Collector, first check and activate all the Collectors and then start them.
11. Fetch measurement results to display them. Fetching measurement results simultaneously to recording is only possible if you use another Collector. A Collector object configured for recording is not able to send data to the client application.
12. Stop recording.
13. Destroy the Collector.
14. Stop online calibration mode.
15. Delete or deselect run-time objects (MCProject, LogicalLink, MCSystem object).

Result

The recorded data is saved to a measurement data file on the MC system.

Script examples

In the `.\Demos\MC3\` folder of your ControlDesk installation, you can find various demo scripts that show how to use the commands of the `asammc3` Python library to remote-control ControlDesk. Refer to MC3 Demos on page 28.

Related topics

Basics

Adding Absolute Time Information

Introduction

You can add PC time information to the measured variables.

Absolute time information

If you need information on the absolute time a value has been measured, for example to synchronize the measured data with data from other sources not processed by ControlDesk, you can capture the PC time of the MC system at the start of a measurement. The absolute time information is transferred to the automation system (and for a recording, stored in the measurement data file).

This allows you to compute the absolute time for each measured value on the automation system. Time differences between the MC system, the automation system and other PCs in your network can be eliminated by synchronizing their PC times (this is usually already implemented by your network administrator).

Code example

The following example shows which parts of the `MeasurementDemoPolling.py` demo script must be changed to add absolute time information to each measurement value. The output then contains the following time information:

```
Current time: Thu Jan 24 07:18:37 2008, AbsoluteMeasurementStartTime: Thu Jan 24 07:18:25 2008
10 [ms]control_out: -0.035156,
...
```

All parts of changed code begin and end with one or more lines of code that is not changed. You can search for these lines in the script to get the exact positions you have to edit.

```
#-------------------------------------------------------------------------------
# 1 - Import modules
#-------------------------------------------------------------------------------
import asammc3
import win32api
import MC3DemoUtilities
import time
from win32com.client import Dispatch
import time
import pythoncom
#-------------------------------------------------------------------------------
# Global function declarations
#-------------------------------------------------------------------------------
...
    return time.time()
#--------------------------------------------------------------------------------
#  11 (b) Fetch measurement results to display them
#--------------------------------------------------------------------------------
...
    if EnableTimings:
        Start = GetPreciseTickCount()
    if Collector.GetFillingLevel(ClientID) > 0:
        # Get all available lines (indicated by parameter NumReq = 0, use -1 to get
        # the latest line only)
        Results = Collector.FetchResults(0, ClientID)
        if EnableTimings:
            AfterFetch = GetPreciseTickCount()
        AbsMeasStartTime = Dispatch(Collector._oleobj_).AbsoluteMeasurementStartTime
        CurTime = time.time()
        print "Current time: %s, AbsoluteMeasurementStartTime: %s" %(time.ctime(CurTime), \
                             time.ctime(AbsMeasStartTime))
        # Iterate over all results that are available
        for Result in Results:
...
            if Collector.Buffer.TimeStamping:
                if EnableTimings:
                    if EnableOutput:
                        CurTimestamp = Response.ResponseParameters.GetItemByIndex(0).ValueVariant
                        Line += "%f [ms]" % (CurTimestamp)
                Index += 1
...
            for ResponseParameterValue in ResponseParameter.ResponseParameters:
                if EnableOutput:
                    LatestValue = ResponseParameterValue.ValueVariant
                    Name = ResponseParameterValue.ShortName
                    try:
                        Line += "%s: %f, " % (Name, LatestValue)
                    except:
...
```

Related topics

HowTos

Automating ControlDesk's Diagnostics Features

Where to go from here

Information in this section

Basics of Automating ControlDesk's Diagnostics Features

ControlDesk's diagnostics features

In ControlDesk, you add an ECU Diagnostics device to an experiment to access an ECU via a Diagnostic interface. Then you place diagnostics instruments on a layout:

- Fault Memory Instrument
- Diagnostics Instrument

Related ControlDesk documentation For more information on performing diagnostic tasks in ControlDesk, refer to Basics of ECU Diagnostics with ControlDesk (ControlDesk ECU Diagnostics).

DSystem

To perform diagnostic tasks via the ASAM MCD-3 D interface using a ControlDesk experiment, you must create a DSystem and select a ControlDesk experiment that contains the configuration of the ODX database. You can then execute diagnostic communication objects via a logical link selected from the vehicle configuration.

This topic describes the basics of accessing ControlDesk's ASAM MCD-3 D interface via the `D3System202` interface. Using the `D3System202` interface, you access the configuration of an ECU Diagnostics device in a ControlDesk experiment.

DProject

In the ASAM MCD-3 D object model, an ODX database loaded by the DSystem is called a DProject.

Note

To use an ODX database stored in a ControlDesk experiment, you have to note some specifics:

- The DProject is named after the ControlDesk experiment that contains the ECU Diagnostics device with the ODX database. If you select an experiment/DProject with a name that has been used more than one time, the first experiment/DProject with this name is used.
- All experiments within a ControlDesk project can be selected. It is not checked whether an experiment contains an ECU Diagnostics device and therefore is suitable for automating diagnostics tasks.
- The ControlDesk experiment is the basis for the DProject. If the experiment does not contain a properly configured ECU Diagnostics device, the experiment cannot be used as DProject. You will not get an error message when selecting such an experiment, but automation will fail on run-time, for example during the selection of a vehicle.

Restrictions You cannot change the configuration of the DProject/ControlDesk experiment via the ASAM MCD-3 D interface:

- You cannot select another vehicle than the vehicle used in the experiment, even if the ODX database contains more than one vehicle. To use another vehicle, you must change the ODX database configuration in the experiment. The vehicle collection contains only the selected vehicle. If no vehicle was selected in the configuration of the ODX database, the vehicle collection is empty.
- Only those logical links are available in the DBLogicalLinks collection, that have been configured and selected in the ControlDesk experiment. If no logical link was configured in the experiment, the collection is empty.
- The selection of the physical interfaces for the logical links is also adopted from the ControlDesk experiment.
- You cannot automate adding and configuring an ECU Diagnostics device using ControlDesk's ASAM MCD-3 D compatible interface.

Requirements to the diagnostic database

The information on the executable communication objects (diagnostic services, diagnostic jobs, and control primitives) is read from a diagnostic database. This database must be compliant with Open Diagnostic Data Exchange (ODX), the ASAM MCD-2 D standard.

ControlDesk supports the following ODX database standards:

- ASAM MCD-2 D V2.0.1
- ASAM MCD-2 D V2.2.0 (ISO 22901-1)

For further information and conventions in connection with the diagnostic database, refer to Conventions in Connection with ODX Databases (ControlDesk ECU Diagnostics 🕮).

Object names

Objects in the ODX database (for example, a logical link or a diagnostic service) can have a short and a long name.

Option	Description
Short name	Short identifier. Unique for items within one object collection.
Long name	Long, more detailed identifier. Not necessarily unique within one object collection.

To select an object, you must use its *short name*.

Terminology table

The terminology in the ASAM MCD-3 object model is different from that in ControlDesk. The following tables list similar objects and terms. Bear in mind that there is not always a one-to-one relationship between the terms. If you need more information on a term, refer to the MCD-3 Automation Glossary on page 76.

ControlDesk -> MCD-3 D ASAM MCD-3 D terms that correspond to ControlDesk terms:

ControlDesk Term(s)	ASAM MCD-3 D Term(s)
Diagnostic communication objects	DiagComPrimitives
Diagnostic Job	Job
Diagnostic Service	Service (DiagService)
ODX database	Project

MCD-3 D -> ControlDesk ControlDesk terms that correspond to ASAM MCD-3 D terms:

ASAM MCD-3 D Term(s)	ControlDesk Term(s)
DiagComPrimitives	Diagnostic communication objects
Job	Diagnostic Job
Service (DiagService)	Diagnostic Service
Project	ODX database

Related topics

HowTos

How to Perform Diagnostic Tasks.. 65

How to Perform Diagnostic Tasks

Objective

You can automate diagnostic tasks by remote-controlling the execution of diagnostic services, diagnostic jobs, or control primitives specified in the ODX database.

Demo script

In the `.\Demos\D3\` folder of your ControlDesk installation, you can find a demo script that shows how to use the commands of the `asamd3` Python library to remote-control ControlDesk. Refer to D3 Demo on page 29.

Restrictions

ControlDesk's ASAM MCD-3 D automation interface (diagnostics part) has the following limitations:

- Only one client (automation system) can access the ASAM MCD-3 D automation interface at the same time. The automation interface cannot handle multiple clients.
- Only one run-time system object (DSystem) can be created by the client.

Method

To perform diagnostic tasks

1. Import modules for ControlDesk's ASAM MCD-3 D interface (diagnostics part).
2. Create a DSystem object.
3. Select a DProject (ControlDesk experiment).
4. Select Vehicle Information via its ShortName or index number.
5. Create and open a Logical Link via its ShortName or index number.
6. Configure the Logical Link.
7. Execute diagnostic tasks via control primitives, diagnostic services, or diagnostic jobs.
8. Delete or deselect run-time objects (LogicalLink, VehicleInformation, DProject, DSystem).

Result

You have read the ODX database content and performed diagnostic tasks by executing diagnostic communication objects.

Related topics

Basics

Basics of Automating ControlDesk's Diagnostics Features........ 62
Basics on Automating ECU Diagnostics via the ControlDesk MCD3 Interface........ 15
D3 Demo........ 29

Translating Python Code into Different Programming Languages

Introduction

You can use differenttranslate code examples into other programming languages, such as C#, Visual Basic, and MATLAB M-files.

Where to go from here

Information in this section

Code Sequence Examples in Different Programming Languages

Introduction

The following examples show the same working steps in different programming languages. The examples enable you to translate the Python examples in this documentation into the language of your choice.

Where to go from here

Information in this section

Information in other sections

Control Structures

Python

```
if Item.EnumProperty == ENUM_VALUE and\
   Item.StringProperty == "String":
      ...
```

Visual Basic	`If Item.EnumProperty = ServerLib.ENUM_VALUE`
	`    If Item.StringProperty = "String" Then`
	`        ...`
	`    End If`

M-Code	`if Item.EnumProperty == ENUM_VALUE`
	`    if strcmp(Item.StringProperty, 'String')`
	`        ...;`
	`    end;`
	`end;`

C#	`if  (  item.EnumProperty  == EnumClass.ENUM_VALUE`
	`    && item.StringProperty.Equals("String")`
	`{`
	`    ...`
	`}`

Line Continuation

Python	`CallMethodWithParameter(\`
	`Parameter ...`

Visual Basic	`CallMethodWithParameter ( _`
	`Parameter ...`

M-Code	`CallMethodWithParameter ( ...`
	`Parameter);`

C#	`CallMethodWithParameter (`
	`    Parameter);`

Creation

Python	`Server = Dispatch('Server.Object.1')`
Visual Basic	`Set Server = CreateObject("Server.Object.1")`

M-Code

```
Server = actxserver('Server.Object.1');
```

C#

```
Type remoteType = Type.GetTypeFromProgID(
   "Server.Object.1", "127.0.0.1", true);
object remoteObj = Activator.CreateInstance(remoteType);
IServerInterface server = remoteObj as IServerInterface;
```

Destruction

Python

```
del Server
```

Visual Basic

```
Set Server = Nothing
```

M-Code

```
clear Server;
```

C#

```
Marshal.ReleaseComObject(server);
this.server = null;
```

Calling Methods without Parameters

Python

With parentheses:

```
Server.CallMethod()
```

Visual Basic

Without parentheses:

```
Server.CallMethod
```

M-Code

Without parentheses:

```
Server.CallMethod;
```

C#

With parentheses:

```
Server.CallMethod ();
```

Collections

Python

Indexing possible:

```
Server.Collection[0]
for Item in Server.Collection:
```

Visual Basic

Indexing possible:

```
Server.Collection(0)
For Each Item In Server.Collection
```

M-Code

Indexing not possible:

```
Server.Collection.Item(int32(0))
for Index = 0:( Server.Collection.Count - 1)
    Item = Server.Collection.Item(int32(Index));
end;
```

Note

By default, MATLAB uses the `double` data type when you call a method with a parameter, for example, like this:

```
 Item(0)
```

However, since parameters of the `double` data type cannot be handled by the automation interface of ControlDesk, you have to cast the data type of such a parameter to `Int32` in your M-code.

C#

```
Server.Collection[0]
foreach (string Item in Server.Collection)
{
   ...
}
```

Constants

Python

Access via global variables:

```
ENUM_VALUE = 1
```

Visual Basic

Access via type information, add reference library to your VB project:

```
ServerLib.ENUM_VALUE
```

M-Code

Not accessible

C#

```
EnumClass.ENUM_VALUE
```

Array Handling

Python

Simple brackets:

```
Server.ArrayProperty = [0,0]
```

Visual Basic

Declaration and setting of each value:

```
Dim Values(2) As Variant
Values(0) = 0
Values(1) = 0
Server.ArrayProperty = Values
```

M-Code

Simple brackets:

```
Server.ArrayProperty = [0,0];
```

C#

```
object[] values = new object[2] { 0, 0 };
Server.ArrayProperty = values;
```

Code Examples in Different Programming Languages

Introduction

Code examples in different programming languages show how to create an object, access properties and methods, and select items from a collection.

Where to go from here

Information in this section

Information in other sections

Code Example in Python

Example

The following example shows how to create an object, access properties and methods, and select items from a collection.

Note

The code examples just show the ways the programming constructs should be translated. They do not work on real servers.

```
from win32com.client import Dispatch
ENUM_VALUE = 1
Server = Dispatch('Server.Object.1')
StringValue = Server.StringProperty
print(StringValue)
Item = Server.Collection[0]
Collection = Server.Collection
for Item in Collection:
    if Item.EnumProperty == ENUM_VALUE and\
       Item.StringProperty == "String":
       Item.CallMethod()
       Item.IntegerProperty = 42
ArrayValue = Server.ArrayProperty
print(ArrayValue)
del StringValue
del ArrayValue
del Item
del Collection
Server.Quit()
del Server
```

Tip

You can find more demo scripts written in Python in the following folders:
- `Toolautomation`
- `MC3`
- `D3`

The folders are located in the `.\Demos\` folder of your ControlDesk installation.

Code Example in Visual Basic

Example

The following example shows how to create an object, access properties and methods, and select items from a collection.

Note

The code examples just show the ways the programming constructs should be translated. They do not work on real servers.

```
...
Dim ArrayValue As Variant
Set Server = CreateObject("Server.Object.1")
StringValue = Server.StringProperty
Debug.Print StringValue
Set Item = Server.Collection(0)
Set Collection = Server.Collection
For Each Item In Collection:
    If Item.EnumProperty = ServerLib.ENUM_VALUE Then
        If Item.StringProperty = "String" Then
            Item.CallMethod
            Item.IntegerProperty = 42
        End If
    End If
Next
ArrayValue = Server.ArrayProperty
For Each ArrayItem In ArrayValue
    Debug.Print ArrayItem
Next
Set StringValue = Nothing
Set ArrayValue = Nothing
Set Item = Nothing
Set Collection = Nothing
Server.Quit
Set Server = Nothing
...
```

Tip

You can find more demo scripts written in Visual Basic in the following folders:

- `Toolautomation`
- `MC3`

The folders are located in the `.\Demos\` folder of your ControlDesk installation.

Code Example in M-Code

Example

The following example shows how to create an object, access properties and methods, and select items from a collection.

Note

The code examples just show the ways the programming constructs should be translated. They do not work on real servers.

```
ENUM_VALUE = 1;
Server = actxserver('Server.Object.1');
StringValue = Server.StringProperty;
disp(sprintf('%s', StringValue));
Item = Server.Collection.Item(int32(0));
Collection = Server.Collection;
for Index = 0:(Collection.Count - 1)
  Item = Collection.Item(int32(Index));
  if Item.EnumProperty == ENUM_VALUE
    if strcmp(Item.StringProperty, 'String')
      Item.CallMethod();
      Item.IntegerProperty = 42;
    End;
  end;
end;
ArrayValue = Server.ArrayProperty;
CellArray = cell2mat(ArrayValue);
for Index = 1:(length(CellArray))
    disp(sprintf('%d', CellArray(Index)));
end;
clear StringValue;
clear CellArray;
clear ArrayValue;
clear Item;
clear Collection;
Server.Quit();
clear Server;
```

Tip

In the `.\Demos\MC3\Matlab\` folder of your ControlDesk installation, you can find more demo scripts written in M-code.

Code Example in C#

Example

The following example shows how to create an object, access properties and methods, and select items from a collection.

Note

The code examples just show the ways the programming constructs should be translated. They do not work on real servers.

```
using Company.Program.Interfaces;
...
Type serverType = Type.GetTypeFromProgID("Server.Object.1");
IServerInterface server = Activator.CreateInstance(
   serverType) as IServerInterface;
string stringValue = server.StringProperty;
Console.WriteLine(stringValue);
IItemInterface item = server.Collection[0];
ICollectionInterface collection = server.Collection;
foreach (item in collection)
{
    if (   item.EnumProperty  == EnumClass.ENUM_VALUE
        && item.StringProperty.Equals("String"))
    {
        item.CallMethod();
        item.IntegerProperty = 42;
    }
}
object[] arrayValue = (object[])server.ArrayProperty;
for (object arrayItem in arrayValue)
{
    Console.Write(String.Format("{0:d} ", (int)arrayItem));
}
...
```

Tip

You can find more demo scripts written in C# in the following folders:

- `Toolautomation`
- `MC3`

The folders are located in the `.\Demos\` folder of your ControlDesk installation.

MCD-3 Automation Terminology

Where to go from here

Information in this section

MCD-3 Automation Glossary

A

Axis Multidimensional parameters such as maps and curves let you control an output value by changing axis points and function values. In ControlDesk, you use the Table Editor to change axis points and function values.

The ASAM MCD-3 object model provides appropriate Axis objects for multidimensional parameters.

B

Binary In the ASAM MCD-3 object model, a Binary is a data version that is assigned to a LogicalLink. It represents a specific program version (data) of an ECU, originally coming from the ECU Image file.

A Binary can be changed only by instancing a new LogicalLink.

Buffer A buffer is used to cache measured data. Its size is limited.

In ControlDesk, this buffer is called a measurement buffer. It is global for all the measured values on all the platforms/devices in an experiment.

In the ASAM MCD-3 object model, each Collector has its own buffer for measured values.

As the buffers are ring buffers in both cases, earlier values are overwritten by later values when the buffer capacity is exceeded.

C

Characteristic In the ASAM MCD-3 object model, Characteristic is the generic term for all ECU variable types that can be calibrated. The term Characteristic is independent of the variable type's dimension. The following variable types are Characteristics:

- ScalarCharacteristic
- CurveCharacteristic
- MapCharacteristic

- VectorCharacteristic (represents one-dimensional structures like axes and curve function values)
- MatrixCharacteristic (represents two-dimensional structures like map function values).

In ControlDesk, Characteristics are called parameters.

CollectedObjects list The CollectedObjects list contains all the variables to be measured. Each Collector has its own CollectedObjects list.

ControlDesk uses the term *measurement signal list*. In contrast to the CollectedObjects list, it is global for all platforms/devices and measurement rasters.

Measurements and scalar characteristics can occur only once per CollectedObjects list, complex characteristic can occur multiple times, if each occurrence defines a different range to be measured (measurement of complex characteristics is currently not supported).

Collector In the ASAM MCD-3 object model, a Collector object is used to configure and perform measurements and recordings (recording is currently not supported). A new Collector must be created for each logical link and each supported Rate.

Computation method A formula or a table that defines the transformation of a source value into a converted value (and vice versa). In addition to the computation methods defined in the variable description file, ControlDesk provides the __Identity computation method, which means the converted and the source value are equal.

CompuMethod ASAM MCD-3 term for computation method.

CompuTab In the ASAM MCD-3 object model, a CompuTab object defines a conversion table that specifies the computation of numerical values. The following table shows an example.

Source Value	Converted Value
1	5
4.3	16.8
2	6
4.7	17.2

CompuVTab In the ASAM MCD-3 object model, a CompuVTab object defines a computation table that specifies the computation of numerical source values into strings. The following table shows an example.

Source Value	Converted Value
0	off
1	idle mode

Source Value	Converted Value
2	partial load
3	full load

CompuVTabRange In the ASAM MCD-3 object model, a CompuVTab object defines a Conversion table that specifies the computation of ranges of numerical values to strings. The following table shows an example.

Source Value	Converted Value
0..2	low
3..6	medium
7..9	high

Conversion table A conversion table specifies the computation of a source value into a converted value in the form of a table. In the case of verbal conversion, the converted value is a string that represents one CompuVTab or a CompuVTabRange.

Conversion mode In ControlDesk a numerical value can be displayed in its 'original format' on the hardware (source mode) or in a converted form (converted mode). The computaion method in the variable description or computation formulas in the instrument properties specify the transformation of a value.

The ASAM MCD-3 term for conversion mode is representation type.

Curve A curve is a parameter that consists of:

- One vector (1-dimensional array) containing the axis points for the axis.
- Another vector containing function values. The curve assigns one function value to each axis point.

In the ASAM MCD-3 object model, this parameter is called CurveCharacteristic.

CurveCharacteristic ASAM MCD-3 term for a curve.

D

DbLocation The ASAM MCD-3 object model defines the DbLocation object as "the entry point for available Services, DiagComPrimitives, Measurements, Characteristics, CompuMethods and so on. (...) To each Logical Link belongs exactly one Location (...) and one Interface (MC) according to the entry in the Logical Link Table."

For the measurement and calibration part, the DbLocation mainly represents the information contained in the variable description file (A2L), for the diagnostics part the information contained in the ODX database.

There are a number of Location classes for the diagnostics part (Protocol, ECU Variant, ...) and one for the measurement and calibration part, which is called Module.

Downsampling The amount of cached and transmitted measurement data is decreased by downsampling the data acquisition. You can define that only every n-th sample is transmitted from the device to your measurement and calibration system.

E

ECU Image file A binary file that is part of the ECU application. It usually contains the code of an ECU application and the data of the parameters within the application. It can be stored as an Intel Hex (HEX) or Motorola S-Record (MOT or S19) file.

In ASAM MCD-3 terms, the ECU Image file is called a Binary.

Experiment In ControlDesk, an experiment is the basis for carrying out a specific calibration task, for example, the calibration of an idle speed control. An experiment allows you to manage all the documents related to the task, such as:

- Layouts for visualizing variables
- Variable descriptions specifying the variables of a platform/device
- Data sets containing a set of ECU parameters
- Measurement data files containing the time traces of recorded variables

Automating calibration tasks via the MCD-3 interface involves

- Getting remote access to the variables in a ControlDesk experiment
- Running a script that performs actions on the variables

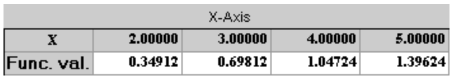
The layouts, data sets and measurement data files of an experiment are not involved in the automating process.

F

Function value Multidimensional parameters such as maps and curves let you control an output value by changing axis points and function values.

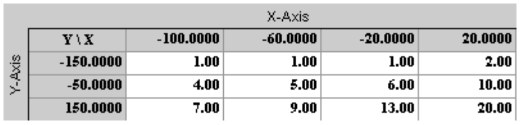
Axis points represent input values. Function values represent output values that are assigned to one or two input values (axis points).

If represented as a table, a curve has x-axis values (input values) in the first row and function values (z-values) in the second row.



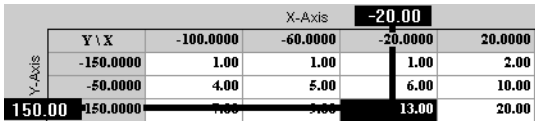
X	2.00000	3.00000	4.00000	5.00000
Func. val.	0.34912	0.69812	1.04724	1.39624

As a map has 2 input values (x and y), it has an x-axis and an y-axis, and uses the first row and the first column of the table for them. In this case the function values (z-values) are on the intersection points.


X-Axis
Y-Axis

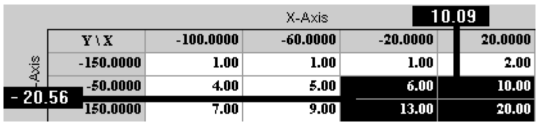
Y \ X	-100.0000	-60.0000	-20.0000	20.0000
-150.0000	1.00	1.00	1.00	2.00
-50.0000	4.00	5.00	6.00	10.00
150.0000	7.00	9.00	13.00	20.00

The actual output value corresponds to the function value if the measured input values are exactly on the axis points.



Y \ X	-100.0000	-60.0000	-20.0000	20.0000
-150.0000	1.00	1.00	1.00	2.00
-50.0000	4.00	5.00	6.00	10.00
150.0000	7.00	9.00	13.00	20.00

Otherwise, it is an interpolation between a number of function values.



Y \ X	-100.0000	-60.0000	-20.0000	20.0000
-150.0000	1.00	1.00	1.00	2.00
-50.0000	4.00	5.00	6.00	10.00
150.0000	7.00	9.00	13.00	20.00

In ControlDesk, you can use the Table Editor to change axis points and function values.

The ASAM MCD-3 object model provides appropriate Axis and Value objects for multidimensional parameters.

L

Location DbLocation

Logical Link In the ASAM MCD-3 object model, a LogicalLink object handles "the logical and physical connection to an ECU (...) as well as the protocol-specific tasks (...) to measure with Collectors and to adjust Characteristics."

A Collector is the object that handles the configuration and execution of measurements and recordings. A Characteristic is the ASAM MCD-3 term for a parameter.

In ControlDesk terms, activating a Logical Link means activating a platform/device.

M

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 (refer to Axis on page 76).
- A matrix containing function values. The map assigns one function value of the matrix to each pair of x-axis and y-axis points.

In the ASAM MCD-3 object model, this parameter is called MapCharacteristic.

MapCharacteristic ASAM MCD-3 term for a map.

Measurement Measuring means viewing and analyzing the time traces of variables. Performing a measurement is important, for example, to observe the effects of ECU parameter changes.

In the ASAM MCD-3 object model a Collector object is used to configure and perform measurements and recordings.

Measurement acquisition list CollectedObjects list

Measurement buffer ControlDesk term for the buffer that collects measured values temporarily.

Measurement signal list In ControlDesk, the measurement signal list contains all variables to be measured. It is global for all platforms/devices and measurement rasters.

In the ASAM MCD-3 object model, each Collector has its own CollectedObjects list.

Measurement variable In ControlDesk, the variables that are intended for measuring are called measurement variables.

In ASAM MCD-3 terms, these variables are called measurements.

Module In the ASAM MCD-3 object model, a DbLocation object in the measurement and calibration part is of the (e)Module type.

P

Parameter In ControlDesk, parameter is the generic term for all ECU variable types that can be calibrated. The term parameter is independent of the variable type's dimension. The following variable types are parameters:

values, value blocks, axis points, curves, and maps.

In the ASAM MCD-3 object model parameters are called parameters.

Platform/Device In ControlDesk platforms/devices are used for carrying out calibration and/or measurement tasks.

In the ASAM MCD-3 object model, the connection to an ECU is handled via Logical Link objects.

Project In ControlDesk, a project manages different experiments belonging together, such as the different tasks for calibrating a specific engine variant. It holds the experiments related to these tasks, and documents relevant to the entire project.

In ASAM MCD-3, a project is defined as "a logical grouping of defined test installations selected by the user". An ASAM MCD-3 project corresponds to an experiment in ControlDesk. The DbProjectDescriptions collection contains all experiments known to ControlDesk, not only those of a single ControlDesk project.

R

Raster In ControlDesk, the measurement raster defines a time interval between single measurements of a value.

In ASAM MCD-3 terms this is called the (acquisition) rate.

Rate ASAM MCD-3 term for measurement raster.

Recording Recording means saving the time traces of variables to a file.

In the ASAM MCD-3 object model, recording is enabled by configuring the Collector's storage type. You can select between transferring the measured data to the AUSY and storing it in a file.

Recording is currently not supported.

Representation type ASAM MCD-3 term for conversion mode. Can be set to eRT_ECU (source mode) and eRT_PHYSICAL (converted mode).

Response Result

Result In the ASAM MCD-3 object model, a Result is a single line of a Collector's buffer. As Collectors are platform-/device - specific, the contents of a line consist of the measured values of one platform/device at one time. As this is also the definition of a Response, in this case a result contains exactly one Response.

One single item in the line is called a Response Parameter. If time-stamping is enabled, the first Response Parameter is the time information. The second Response Parameter is the object structure as the entry point of the second Response Parameter level.

S

ScalarCharacteristic ASAM MCD-3 term for a scalar parameter.

In ControlDesk, this parameter type is called value.

T

Time stamp A time stamp means that a time information is given at the beginning of each new sample of acquisition data.

V

Variable description In ControlDesk, a variable description specifies the variables of a platform/device. It also specifies the interface used to connect the platform/device.

- For calibration devices such as the DCI-GSI2, the variable description specifies the parameters and measurement variables of each device. It also contains information on the device's memory segments.

  The variable description is stored as an ASAM MCD-2 MC (A2L) file.
- For measurement and CAN Bus Monitoring devices, the variable description specifies the measurement variables of each device.

  The variable description is stored as a CAN database (DBC) file.

The ASAM MCD-3 object model defines the DbLocation object as "the entry point for available (...) Measurements, Characteristics, CompuMethods and so on."

Value ControlDesk term for a scalar parameter.

In ASAM MCD-3 terms, this parameter type is called ScalarCharacteristic.

In the ASAM MCD-3 object model, a function value is called a Value.

Related topics

Basics

Terminology Mapping.. 82

Terminology Mapping

Introduction

The terminology in the ASAM MCD-3 object model is different from that in ControlDesk. The following tables list similar objects and terms. Bear in mind that there is not always a one-to-one relationship between the terms. If you need more information on a term, refer to the MCD-3 Automation Glossary on page 76.

ControlDesk → MCD-3

Correspondences between ControlDesk terms and ASAM MCD-3 terms:

ControlDesk Term(s)	ASAM MCD-3 Term(s)
Axis	Axis
Computation method	CompuMethod
Conversion table	CompuTab
Curve	CurveCharacteristic
Platform/Device	Logical Link(s)
ECU Image file	Binary
Experiment	Project
Function value	Value
Map	MapCharacteristic
Measurement	Collector
Measurement buffer	Buffer
Measurement signal list	CollectedObjects list
Measurement variable	Measurement
Parameter	Characteristic
Project	-
Raster	Rate
Time stamp	Time stamp
Variable description	DbLocation Module
Value	ScalarCharacteristic
Verbal conversion	CompuVTab
Verbal conversion range	CompuVTabRange

MCD-3 → ControlDesk

Correspondences between ASAM MCD-3 terms and ControlDesk terms:

ASAM MCD-3 Term(s)	ControlDesk Term(s)
Axis	Axis
Binary	ECU Image file
Buffer	Measurement buffer
Characteristic	Parameter
CollectedObjects list	Measurement signal list
Collector	Measurement
CompuMethod	Computation method
CompuTab	Conversion table
CompuVTab	Verbal conversion
CompuVTabRange	Verbal conversion range
CurveCharacteristic	Curve
DbLocation	Variable description
Downsampling	Downsampling

ASAM MCD-3 Term(s)	ControlDesk Term(s)
Logical Link	Platform/Device, Logical Link
Module	Variable description
MapCharacteristic	Map
Measurement	Measurement variable
Project	Experiment
Rate	Raster
Response	Measurement buffer (content)
Result	Measurement buffer (content)
ScalarCharacteristic	Value
Time stamp	Time stamp

Related topics

Basics

MCD-3 Automation Glossary........ 76

Reference Information

MC3 Page

Access

This page is part of the ControlDesk Options dialog.

The dialog can be opened via the **Options Command** (ControlDesk User Interface Handling 🕮).

Purpose

To specify general settings for automation via ControlDesk's ASAM-MCD3 MC interface.

Dialog settings

Enable API logging Lets you specify whether the logging of API commands is enabled for automation via ControlDesk's ASAM-MCD3 MC interface. If activated, the API commands are logged to the `DSMC3Server.log` file in the Local Program Data folder.

Use long project identifier (ProjectName/ExperimentName) Lets you specify to use the project and experiment name to identify a project. This makes it easier to differentiate between projects that contain experiments with the same name, for example, *Project001/Experiment_001* and *Project002/Experiment001*. If activated, the long project identifiers are also shown in the list of available projects.

Related topics

References

Options Command (ControlDesk User Interface Handling 🕮)

Limitations

Where to go from here

Information in this section

General Limitation

32-bit automation client applications not supported	Only use 64-bit client applications. 32-bit automation client applications are not supported.

Limitations for ECU Access via ControlDesk's MCD3 Interface

Limitation for accessing ECUs without memory segments via CCP and XCP	An ECU without memory segments cannot be accessed via CCP or XCP by using ControlDesk's ASAM MCD-3 MC automation interface.

Limitations for Calibration via ControlDesk's MCD3 Interface

No monotony violation	If you change axis points in ControlDesk's Table Editor and the new values violate the monotony of axis points, the **Monotony Violation** dialog opens. The dialog lets you confirm monotony violation and write the values. However, such an intentional violation of axis monotony is not possible in an automated calibration task.
Interpolation of function values	If you change axis points in ControlDesk's Table Editor, you have to choose between interpolation and no interpolation of the function values. However, in an automated calibration task, the function values remain the same. The interpolation of function values according to the changed axis points cannot be automated.
No common axis warning	If you change axis points of a common axis, you get no warning that referring variables are affected.

Related topics

Basics

Basics on Automating Measurement and Calibration via the ControlDesk MCD3 Interface.. 12

Limitations for Measurement and Recording via ControlDesk's MCD3 Interface

Start sequence for multiple Collectors

If you use more than one Collector, you must first check and activate all the Collectors, before you can start them. See the following example for two Collectors.

1. `Collector1.Check`
   `Collector1.Activate`
2. `Collector2.Check`
   `Collector2.Activate`
3. `Collector1.Start`
   `Collector2.Start`

Note

If you have two separate client applications with one Collector each, you cannot apply this rule for technical reasons.
If both Collectors are connected to the same platform/device, but they do not measure the same variables, the second Collector that starts will receive data only if the following two conditions are met:

- The variables of the second Collector have been added to ControlDesk’s measurement signal list.
- The raster setting of the Collector and the raster settings of its variables in the measurement signal list must be equal.

Otherwise the second Collector will receive no data at all.

Removing signals manually when using multiple Collectors

If you record with *several* Collectors at the same time and if you stop and release one of the Collectors via the client application while another Collector still is active, the signals of the stopped Collector remain in the measurement signal list. Even when you stop measurement, these signals are not removed from the measurement signal list. You have to remove them manually.

Collector settings for automated recordings

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.

ValueBlocks of SDF files

ControlDesk's automation interface supports Vs. 1.0.1 of the ASAM MCD-3 standard. This version of the ASAM-MCD-3 standard does not support the transmission of ValueBlocks. For this reason, ValueBlocks of SDF files cannot be transmitted via ControlDesk's ASAM MCD-3-compatible interface.

Related topics

Basics

Limitations for ECU Diagnostics via ControlDesk's MCD3 Interface

Automation interface: No multi-client support

ControlDesk's ASAM MCD-3D automation interface for diagnostics has the following limitations:

- Only one client (automation system) can access the ASAM MCD-3D automation interface for diagnostics at the same time. The automation interface cannot handle multiple clients.
- Only one run-time system object (D3System202) can be created by the client.
- An ECU Diagnostics device cannot be reconfigured via ControlDesk's ASAM MCD-3 D compatible interface. As a consequence, only the vehicles and logical links configured in the ControlDesk experiment are accessible to the interface.

  There is one exception: *All* the ECU variants that belong to a base variant using ControlDesk's ASAM MCD-3 D compatible interface are available, regardless of the configuration of the related ECU Diagnostics device. However, you cannot use these ECU variants via ASAM MCD-3 D.

General limitations

For general limitations using the ControlDesk ECU Diagnostics interface, refer to Limitations for ECU Diagnostics (ControlDesk ECU Diagnostics 📖).

Conventions in connection with ODX databases

When performing ECU diagnostics with ControlDesk, you have to observe some conventions in connection with the ODX database you use. Refer to Conventions in Connection with ODX Databases (ControlDesk ECU Diagnostics 📖).

Related topics

Basics

Limitations for Error Message Handling in Connection with ControlDesk's MCD3 Interface

Error message stops automation task

ControlDesk’s error messages are not transported to the automation system. The automation sequence is paused until the error message is confirmed by the user. You therefore have to read and confirm the error message on the ControlDesk PC to continue the automation sequence.

Tip

You can specify whether ControlDesk is to display message boxes containing platform-/device-specific status information. If you want to access ControlDesk via its remote-control interfaces, it is advisable to disable the **Display status information** property in the **Properties** controlbar of the platform/device or during platform/device configuration (not available for all platforms/devices). Refer to General Settings Properties (ControlDesk Platform Management 🕮) and Configure Platform/Device (ControlDesk Platform Management 🕮).

No triggering and delaying

No triggering or delaying mechanisms can be applied to automated measurement or recording tasks because they are not included in the current standard.

Storing of results

Results of automated measurements cannot be stored in a ControlDesk-compatible file format.

Limitations for Using M-Files in Connection with ControlDesk's MCD3 Interface

M-files

- If you want to automate ControlDesk via M-files, MATLAB R13 SP1 (or later) is required.
- With M-files you cannot use event handling in automated measurements. You must poll a Collector for results.

Glossary

Introduction

Briefly explains the most important expressions and naming conventions used in the ControlDesk documentation.

Where to go from here

Information in this section

Numerics

3-D Viewer An instrument for displaying items in a 3-D environment.

A

A2L file A file that contains all the relevant information on measurement and calibration variables in an ECU application and the ECU's communication interface(s). This includes information on the variables' memory addresses and conversion methods, the memory layout and data structures in the ECU as well as interface description data (IF_DATA).

Acquisition An object in the Measurement Configuration controlbar that specifies the variables to be measured and their measurement configuration.

Active variable description The variable description that is currently active for a platform/device. Multiple variable descriptions can be assigned to one platform/device, but only one of them can be active at a time.

Additional write variable A scalar parameter or writable measurement variable that can be connected to an instrument in addition to the main variable. When the value of the main variable changes, the changed value is also applied to all the additional write variables connected to the instrument.

Airspeed Indicator An instrument for displaying the airspeed of a simulated aircraft.

Altimeter An instrument for displaying the altitude of a simulated aircraft.

Animated Needle An instrument for displaying the value of a connected variable by a needle deflection.

Application image An image file that contains all the files that are created when the user builds a real-time application. It particularly includes the variable description (SDF) file. To extend a real-time application, ControlDesk lets the user create an updated application image from a data set. The updated application image then contains a real-time application with an additional set of parameter values.

Artificial Horizon An instrument displaying the rotation on both the lateral and the longitudinal axis to indicate the angle of pitch and roll of a simulated aircraft. The Artificial Horizon has a pitch scale and a roll scale.

Automatic Reconnect Feature for automatically reconnecting to platform/device hardware, for example, when the ignition is turned off and on, or when the physical connection between the ControlDesk PC and the ECU is temporarily interrupted.

If the feature is enabled for a platform/device and if the platform/device is in the 'unplugged' state, ControlDesk tries to re-establish the logical connection to the platform/device hardware. After the logical connection is re-established, the platform/device has the same state as before the unplugged state was detected. A measurement started before the unplugged state was detected is resumed.

Automation A communication mechanism that can be used by various programming languages. A client can use it to control a server by calling methods and properties of the server's automation interface.

Automation script A script that uses automation to control an automation server.

Axis point object Common axis

B

Bar An instrument (or a value cell type of the Variable Array) for displaying a numerical value as a bar deflection on a horizontal or vertical scale.

Bitfield A value cell type of the Variable Array for displaying and editing the source value of a parameter as a bit string.

Bookmark A marker for a certain event during a measurement or recording.

Browser An instrument for displaying HTML and TXT files. It also supports Microsoft Internet Explorer© plug-ins that are installed on your system.

Bus communication replay A feature of the Bus Navigator that lets you replay logged bus communication data from a log file. You can add replay nodes to the Bus Navigator tree for this purpose. You can specify filters to replay selected parts of the logged bus communication.

Bus configuration A configuration of all the controllers, communication matrices, and messages/frames/PDUs of a specific communication bus such as CAN. ControlDesk lets you display and experiment with bus configurations in the Bus Navigator.

Bus connection A mode for connecting dSPACE real-time hardware to the host PC via bus. The list below shows the possible bus connections:

- dSPACE real-time hardware installed directly in the host PC
- dSPACE real-time hardware installed in an expansion box connected to the host PC via dSPACE link board

Bus Instrument An instrument available for the Bus Navigator. It can be configured for different purposes, for example, to display information on received messages (RX messages) or to manipulate and transmit messages (TX messages). The instrument is tailor-made and displays only the message- and signal-specific settings which are enabled for display and/or manipulation by ControlDesk during run time.

Bus logging A feature of the Bus Navigator that lets you log raw bus communication data. You can add logger nodes on different hierarchy levels of the Bus Navigator tree for this purpose. You can specify filters to log filtered bus communication. The logged bus communication can be replayed.

Bus monitoring A feature of the Bus Navigator that lets you observe bus communication. You can open monitoring lists and add monitor nodes on different hierarchy levels of the Bus Navigator tree for this purpose. You can specify filters to monitor filtered bus communication.

Bus Navigator A controlbar for handling bus messages, such as CAN messages, LIN frames, and Ethernet packets.

Bus statistics A feature of the Bus Navigator that lets you display and log statistical information on the bus load during bus monitoring.

Bypassing A method for replacing an existing ECU function by running a new function.

C

Calculated variable A scalar variable that can be measured and recorded, and that is derived from one or more *input variables*.

The following input variable types are supported:

- Measurement variables
- Single elements of measurement arrays or value blocks
- Scalar parameters, or existing calculated variables

The value of a calculated variable is calculated via a user-defined *computation formula* that uses one or more input variables.

Calculated variables are represented by the symbol.

CalDemo ECU A demo program that runs on the same PC as ControlDesk. It simulates an ECU on which the Universal Measurement and Calibration (XCP) protocol and the Unified Diagnostic Services (UDS) protocol are implemented.

The CalDemo ECU allows you to perform parameter calibration, variable measurement, and ECU diagnostics with ControlDesk under realistic conditions, but without having to have a real ECU connected to the PC. Communication between the CalDemo ECU and ControlDesk can be established via XCP on CAN or XCP on Ethernet, and UDS on CAN.

Tip

If communication is established via XCP on Ethernet, the CalDemo ECU can also run on a PC different from the PC on which ControlDesk is running.

The memory of the CalDemo ECU consists of two areas called memory page. Each page contains a complete set of parameters, but only one page is accessible by the CalDemo ECU at a time. You can easily switch the memory pages of the CalDemo ECU to change from one parameter to another in a single step.

Two ECU tasks run on the CalDemo ECU:

- ECU task #1 runs at a fixed sample time of 5 ms. In ControlDesk's **Measurement Configuration**, ECU task #1 is related to the time-based **5 ms**, **10 ms**, **50 ms** and **100 ms** measurement rasters of the CalDemo ECU.
- ECU task #2 has a variable sample time. Whenever the CalDemo ECU program is started, the initial sample time is 5 ms. This can then be increased or decreased by using the **dSPACE CalDemo** dialog.

  ECU task #2 is related to the **extEvent** measurement raster of the CalDemo ECU.

The CalDemo ECU can also be used to execute diagnostic services and jobs, handle DTCs and perform measurement and calibration via ECU diagnostics.

The CalDemo ECU program is run by invoking `CalDemo.exe`. The file is located in the `.\Demos\CalDemo` folder of the ControlDesk installation.

Calibration Changing the parameter values of real-time applications or ECU applications.

Calibration memory segment Part of the memory of an ECU containing the calibratable parameters. Memory segments can be defined as `MEMORY_SEGMENT` in the A2L file. ControlDesk can use the segments to evaluate the memory pages of the ECU.

ControlDesk lets you perform the calibration of:

- Parameters inside memory segments
- Parameters outside memory segments
- Parameters even if no memory segments are defined in the A2L file.

CAN Bus Monitoring device A device that monitors the data stream on a CAN bus connected to the ControlDesk PC.

The CAN Bus Monitoring device works, for example, with PC-based CAN interfaces such as the DCI-CAN2 or the DCI-CAN/LIN1.

The device supports the following variable description file types:

- DBC
- FIBEX
- AUTOSAR system description (ARXML)

CANGenerator A demo program that simulates a CAN system, that is, it generates signals that can be measured and recorded with ControlDesk. The program runs on the same PC as ControlDesk.

The CANGenerator allows you to use the CAN Bus Monitoring device under realistic conditions, but without having to have any device hardware connected to the PC.

The CAN (Controller Area Network) protocol is used for communication between the CANGenerator and ControlDesk. However, since the CANGenerator runs on the same PC as ControlDesk, ControlDesk does not communicate with the device via a real CAN channel, but via a *virtual CAN channel* implemented on the host PC.

You can start the CAN generator program by running `CANGenerator.exe`. The file is located in the `.\Demos\CANGenerator` folder of the ControlDesk installation.

Capture A data packet of all the measurement variables assigned to a measurement raster. The packet comprises the data that results from a single triggering of the raster.

CCP Abbreviation of CAN Calibration Protocol. This protocol can be implemented on electronic control units (ECUs) and allows users to access ECUs with measurement and calibration systems (MCS) such as ControlDesk.

The basic features of CCP are:

- Read and write access to the ECU memory, i.e., providing access for calibration
- Synchronous data acquisition
- Flash programming for ECU development purposes

The CCP protocol was developed by ASAM e.V. (Association for Standardization of Automation and Measuring Systems e.V.). For the protocol specification, refer to http://www.asam.net.

The following device supports ECUs with an integrated CCP service:

- CCP device

CCP device A device that provides access to an ECU with CCP connected to the ControlDesk PC via CAN, for example, for measurement and calibration purposes via CCP (CAN Calibration Protocol).

Check Button An instrument (or a cell type of the Variable Array) for displaying whether the value of a connected variable matches predefined values or for writing a predefined value to a connected variable.

cmdloader A command line tool for handling applications without using the user interface of an experiment software.

Common axis A parameter that consists of a 1-dimensional array containing axis points. A common axis can be referenced by one or more curves and/or maps. Calibrating the data points of a common axis affects all the curves and/or maps referencing the axis.

Common axes are represented by the symbol.

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>`

Computation method A formula or a table that defines the transformation of a source value into a converted value (and vice versa). In addition to the computation methods defined in the variable description file, ControlDesk provides the *__Identity* computation method which means the converted and the source value are equal.

Connected A platform/device state defined by the following characteristics:

- A continuous logical connection is established between ControlDesk and the platform/device hardware.
- A platform/device must be in the 'connected' state before it can change to the 'measuring/recording' or 'online calibration started' state.
- Online calibration is impossible. ControlDesk did not yet adjust the memory segments containing calibration data in the platform/device and on the corresponding hardware. Offline calibration is possible.
- Platform/device configuration is not possible. However, you can invoke platform/device configuration for a platform/device that is in the connected state. ControlDesk temporarily sets the platform/device to the disconnected state.

The 'connected' platform/device state is indicated by the icon.

Connection mode dSPACE real-time systems can be installed within the host PC or connected to the host via a bus interface and/or via Ethernet. When the Ethernet is being used, different network clients might exist. The connection type being used and, in the case of Ethernet, the network client being used, determine the dSPACE systems that can be accessed.

Control primitive A special diagnostic communication object for changing communication states or protocol parameters, or for identifying (ECU) variants.

Controlbar A window or pane outside the working area. Can be docked to an edge of the main window or float in front of it. A controlbar can contain a document, such as a layout, or a tool, such as the Bus Navigator. It can be grouped with other controlbars in a window with tabbed pages.

ControlDesk The main version of ControlDesk for creating and running experiments, and for accessing dSPACE real-time hardware and VEOS. The functionality can be extended by optional software modules.

ControlDesk - Operator Version A version of ControlDesk that provides only a subset of functionality for running existing experiments. The functionality can be extended by optional software modules.

ControlDesk Bus Navigator Module An optional software module for ControlDesk for handling bus messages, such as CAN, LIN, and FlexRay messages, frames, and PDUs and Ethernet packets.

ControlDesk ECU Diagnostics Module An optional software module for ControlDesk that facilitates the calibration and validation of ECU diagnostic functions.

ControlDesk ECU Interface Module An optional software module for ControlDesk for calibration and measurement access to electronic control units (ECUs). The module is also required for calibration and measurement access to virtual ECUs (V-ECUs) used in SIL testing scenarios.

ControlDesk Signal Editor Module An optional software module for ControlDesk for the graphical definition and execution of signal generators for stimulating model variables of real-time/offline simulation applications.

Controller board Single-board hardware computing the real-time application. Contains a real-time processor for fast calculation of the model and I/O interfaces for carrying out the control developments.

Conversion table A table that specifies the value conversion of a source value into a converted value. In the case of verbal conversion, the converted value is a string that represents one numerical value or a range of numerical values.

Conversion type The type of a computation method, for example a linear function or a verbal computation method.

Curve A parameter that consists of

- A 1-dimensional array containing the axis points for the x-axis. This array can also be specified by a reference to a common axis.
- Another 1-dimensional array containing data points. The curve assigns one data point to each axis point.

Curves are represented by the ⊞ symbol.

D

DAQ module A hardware module for the acquisition of physical quantities

Data Cursor One or two cursors that are used to display the values of selected chart positions in a Time Plotter or an Index Plotter.

Data logger An object in the Measurement Configuration controlbar that lets you configure a data logging.

Data logger signal list A list that contains the variables to be included in subsequent data loggings on real-time hardware.

Data logging The recording of data on dSPACE real-time hardware that does not require a physical connection between the host PC and the real-time hardware. In contrast to flight recording, data logging is configured in ControlDesk.

Data set A set of the parameters and their values of a platform/device derived from the variable description of the platform/device. There are different types of data sets:
- Reference data set
- Sub data set
- Unassigned data set
- Working data set

DCI-CAN/LIN1 A dSPACE-specific interface between the host PC and the CAN/CAN FD bus and/or LIN bus. The DCI-CAN/LIN1 transfers messages between the CAN-/LIN-based devices and the host PC via the universal serial bus (USB).

DCI-CAN2 A dSPACE-specific interface between the host PC and the CAN bus. The DCI-CAN2 transfers CAN and CAN FD messages between the CAN-based devices and the host PC via the universal serial bus (USB).

DCI-GSI2 Abbreviation of *dSPACE Communication Interface - Generic Serial Interface 2*. A dSPACE-specific interface for ECU calibration, measurement and ECU interfacing.

DCI-GSI2 device A device that provides access to an ECU with DCI-GSI2 connected to the ControlDesk PC for measurement, calibration, and bypassing purposes via the ECU's debug interface.

DCI-KLine1 Abbreviation of *dSPACE Communication Interface - K-Line Interface*. A dSPACE-specific interface between the host PC and the diagnostics bus via K-Line.

Debug interface An ECU interface for diagnostics tasks and flashing.

Default raster A platform-/device-specific measurement raster that is used when a variable of the platform/device is connected to a plotter or a recorder, for example.

Deposition definition A definition specifying the sequence in which the axis point values of a curve or map are deposited in memory.

Device A software component for carrying out calibration and/or measurement, bypassing, ECU flash programming, or ECU diagnostics tasks.

ControlDesk provides the following devices:

- Bus devices:
  - CAN Bus Monitoring device
  - Ethernet Bus Monitoring device
  - LIN Bus Monitoring device
- ECU Diagnostics device
- GNSS device
- Measurement and calibration devices:
  - CCP device
  - DCI-GSI2 device
  - XCP on CAN device
  - XCP on Ethernet device

Each device usually has a variable description that specifies the device's variables to be calibrated and measured.

Diagnostic interface Interface for accessing the fault memory of an ECU.

Diagnostic job (often called Java job) Programmed sequence that is usually built from a sequence of the diagnostic service. A diagnostic job is either a single-ECU job or a multiple-ECU job, depending on whether it communicates with one ECU or multiple ECUs.

Diagnostic protocol A protocol that defines how an ECU communicates with a connected diagnostic tester. The protocol must be implemented on the ECU and on the tester. The diagnostics database specifies the diagnostic protocol(s) supported by a specific ECU.

ControlDesk's ECU Diagnostics device supports CAN and K-Line as the physical layers for communication with an ECU connected to the ControlDesk PC. For information on the supported diagnostic protocols with CAN and K-Line, refer to Basics of ECU Diagnostics with ControlDesk (ControlDesk ECU Diagnostics).

Diagnostic service A service implemented on the ECU as a basic diagnostic communication element. Communication is performed by selecting a service, configuring its parameters, executing it, and receiving the ECU results. When a service is executed, a defined request is sent to the ECU and the ECU answers with a specific response.

Diagnostic trouble code (DTC) A hexadecimal index for the identification of vehicle malfunctions. DTCs are stored in the fault memory of ECUs and can be read by diagnostic testers.

Diagnostics database A database that completely describes one or more ECUs with respect to diagnostics communication. ControlDesk supports the ASAM MCD-2 D ODX database format, which was standardized by ASAM e.V. (Association for Standardisation of Automation and Measuring Systems e.V.). For the format specification, refer to http://www.asam.net.

Proprietary diagnostics database formats are not supported by ControlDesk.

Diagnostics Instrument An instrument for communicating with an ECU via the diagnostic protocol using diagnostic services, diagnostic jobs, and control primitives.

Disabled A platform/device state defined by the following characteristics:

- No logical connection is established between ControlDesk and the platform/device hardware.
- When a platform/device is disabled, ControlDesk does not try to establish the logical connection for that platform/device. Any communication between the platform/device hardware and ControlDesk is rejected.
- Online calibration is impossible. Offline calibration is possible.
- Platform/device configuration is possible.

The 'disabled' platform/device state is indicated by the icon.

Disconnected A platform/device state defined by the following characteristics:

- No logical connection is established between ControlDesk and the platform/device hardware.
- When a platform/device is in the disconnected state, ControlDesk does not try to re-establish the logical connection for that platform/device.
- Online calibration is impossible. Offline calibration is possible.
- Platform/device configuration is possible.

The 'disconnected' platform/device state is indicated by the icon.

Display An instrument (or a value cell type of the Variable Array) for displaying the value of a scalar variable or the text content of an ASCII variable.

Documents folder A standard folder for user-specific documents.

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

DS1006 Processor Board platform A platform that provides access to a DS1006 Processor Board connected to the host PC for HIL simulation and function prototyping purposes.

DS1007 PPC Processor Board platform A platform that provides access to a single multicore DS1007 PPC Processor Board or a DS1007 multiprocessor system consisting of two or more DS1007 PPC Processor Boards, connected to the host PC for HIL simulation and function prototyping purposes.

DS1104 R&D Controller Board platform A platform that provides access to a DS1104 R&D Controller Board installed in the host PC for function prototyping purposes.

DS1202 MicroLabBox platform A platform that provides access to a MicroLabBox connected to the host PC for function prototyping purposes.

DsDAQ service A service in a real-time application or offline simulation application (OSA) that provides measurement data from the application to the host PC. Unlike the host service, the DsDAQ service lets you perform, for example, triggered measurements with complex trigger conditions.

The following platforms support applications that contain the DsDAQ service:

- DS1007 PPC Processor Board platform
- DS1202 MicroLabBox platform
- MicroAutoBox III platform
- SCALEXIO platform
- VEOS platform
- XIL API MAPort platform

dSPACE Calibration and Bypassing Service An ECU service for measurement, calibration, bypassing, and ECU flash programming. The dSPACE Calibration and Bypassing Service can be integrated on the ECU. It provides access to the ECU application and the ECU resources and is used to control communication between an ECU and a calibration and/or bypassing tool.

With the dSPACE Calibration and Bypassing Service, users can run measurement, calibration, bypassing, and flash programming tasks on an ECU via the DCI-GSI2. The service is also designed for bypassing ECU functions using dSPACE prototyping hardware by means of the RTI Bypass Blockset in connection with DPMEM PODs. The dSPACE Calibration and Bypassing Service allows measurement, calibration, and bypassing tasks to be performed in parallel.

dSPACE Internal Bypassing Service An ECU service for on-target prototyping. The dSPACE Internal Bypassing Service can be integrated in the ECU application. It lets you add additional functions to be executed in the context of the ECU application without the need for recompiling the ECU application.

dSPACE Log A collection of errors, warnings, information, questions, and advice issued by all dSPACE products and connected systems over more than one session.

dSPACE system A hardware system such as a MicroAutoBox III or SCALEXIO system on which the real-time application runs.

Duration trigger A trigger that defines a duration. Using a duration trigger, you can, for example, specify the duration of data acquisition for a measurement raster. A duration trigger can be used as a stop trigger.

E

ECU Abbreviation of *electronic control unit*.

ECU application A sequence of operations executed by an ECU. An ECU application is mostly represented by a group of files such as ECU Image files, MAP files, A2L files and/or software module description files.

ECU calibration interface Interface for accessing an ECU by either emulating the ECU's memory or using a communication protocol (for example, XCP on CAN).

ECU diagnostics Functions such as:

- Handling the ECU fault memory: Entries in the ECU´s fault memory can be read, cleared, and saved.
- Executing diagnostic services and jobs: Users can communicate with an ECU via a diagnostic protocol using diagnostic services, diagnostic jobs, and control primitives.

ControlDesk provides the ECU Diagnostics device device to access ECUs for diagnostic tasks. Communication is via diagnostic protocols implemented on the ECUs.

ECU diagnostics with ControlDesk are completely based on Open Diagnostic Data Exchange (ODX), the ASAM MCD-2 D diagnostics standard.

ControlDesk provides the Fault Memory Instrument and the Diagnostics Instrument for ECU diagnostics tasks.

ECU Diagnostics device A device that provides access to ECUs connected to the ControlDesk PC via CAN or K-Line for diagnostics or flash programming purposes.

ControlDesk provides the *ECU Diagnostics v2.0.2* device, which supports the ASAM MCD-3 D V2.0.2 standard.

ControlDesk supports the following ODX database standards:

- ASAM MCD-2 D V2.0.1
- ASAM MCD-2 D V2.2.0 (ISO 22901-1)

ECU flash programming A method by which new code or data is stored in ECU flash memory.

ECU Image file A binary file that is part of the ECU application. It usually contains the code of an ECU application and the data of the parameters within the application. It can be stored as an Intel Hex (HEX) or Motorola S-Record (MOT or S19) file.

EESPort Configurations controlbar A controlbar for configuring error configurations.

Electrical error simulation Simulating electrical errors such as loose contacts, broken cables, and short-circuits, in the wiring of an ECU. Electrical error simulation is performed by the failure simulation hardware of an HIL simulator.

Electrical Error Simulation port (EESPort) An *Electrical Error Simulation port* (EESPort) provides access to a failure simulation hardware for simulating electrical errors in an ECU wiring according to the ASAM AE XIL API standard.

The configuration of the EESPort is described by a hardware-dependent *port configuration* and one or more *error configurations*.

Environment model A model that represents a part or all of the ECU's environment in a simulation scenario.

The environment model is a part of the simulation system.

Environment VPU The executable of an environment model built for the VEOS platform. An environment VPU is part of an offline simulation application (OSA).

Error An electrical error that is specified by:

- An error category
- An error type
- A load type

Error category The error category defines how a signal is disturbed. Which errors you can create for a signal depends on the connected failure simulation hardware.

Error configuration An XML file that describes a sequence of errors you want to switch during electrical error simulation. Each error configuration comprises error sets with one or more errors.

Error set An error set is used to group errors (pin failures).

Error type The error type specifies the way an error category – i.e., an interruption or short circuit of signals – is provided. The error type defines the disturbance itself.

Ethernet Bus Monitoring device A device that monitors the data stream on an Ethernet network connected to the ControlDesk PC.

The device supports the following variable description file type:

- AUTOSAR system description (ARXML)

Ethernet connection A mode for connecting dSPACE real-time hardware to the host PC via Ethernet. The list below shows the possible Ethernet connections:

- dSPACE real-time hardware installed in an expansion box connected to the host PC via Ethernet.
- MicroAutoBox II/III and MicroLabBox connected via Ethernet.

Ethernet decoding A feature of the Bus Navigator that lets you view protocol data and raw data of an Ethernet frame.

Event An event that is triggered by an action performed in ControlDesk.

Event context The scope of validity of event sources and events. There is one event handler code area for each event context.

Event handler Code that is executed when the related event occurs.

Event management Functionality for executing custom code according to actions triggered by ControlDesk.

Event source An object providing and triggering events.
LayoutManagement is an example of an event source.

Event state State of an event. ControlDesk provides the following event states:

- No event handler is defined
- Event handler is defined and enabled
- Event handler is defined and disabled
- Event handler is defined, but no Python code is available
- Event handler is deactivated because a run-time error occurred during the execution of the Python code

Expansion box A box that hosts dSPACE boards. It can be connected to the host PC via bus connection or via network.

Experiment A container for collecting and managing information and files required for a parameter calibration and/or measurement task. A number of experiments can be collected in a project but only one of them can be active.

Extension script A Python script (PY or PYC file) that is executed each time ControlDesk starts up. An extension script can be executed for all users or user-specifically.

F

Failure insertion unit Hardware unit used with dSPACE simulators to simulate failures in the wiring of an ECU, such as broken wire and short circuit to ground.

Fault memory Part of the ECU memory that stores diagnostic trouble code (DTC) entries with status and environment information.

Fault Memory Instrument An instrument for reading, clearing, and saving the content of the ECU's fault memory.

Firmware update An update for the firmware installed in the board's flash memory. Firmware should be updated if it is older than required by the real-time application to be downloaded.

Fixed axis An axis with data points that are not deposited in the ECU memory. Unlike a common axis, a fixed axis is specified within a curve or map. The parameters of a fixed axis cannot be calibrated.

Fixed parameter A parameter that has a fixed value during a running simulation. Changing the value of a fixed parameter does not immediately affect the simulation results. The affect occurs only after you stop the simulation and start it again. A fixed parameter is represented by an added pin in its symbol, for example: .

Flash job A specific diagnostic job for flashing the ECU memory. A flash job implements the process control for flashing the ECU memory, such as initialization, security access, writing data blocks, etc.

Flight recording The recording of data on dSPACE real-time hardware that does not require a physical connection between the host PC and the real-time hardware. In contrast to data logging, flight recording is not configured in ControlDesk but via RTI and RTLib.

Frame An instrument for adding a background frame to a layout, for example, to visualize an instrument group.

G

Gauge An instrument for displaying the value of the connected variable by a needle deflection on a circular scale.

Gigalink module A dSPACE board for connecting several processor boards in a multiprocessor system. The board allows high-speed serial data transmission via fiber-optic cable.

GNSS data Positioning and timing data that is transmitted by a Global Navigation Satellite System (GNSS), such as GPS, GLONASS, or Galileo. GNSS receivers use this data to determine their location.

GNSS device A device that provides positioning data from a GNSS receiver (e.g., a serial GPS mouse) in ControlDesk.

ControlDesk provides the *GNSS (GPS, GLONASS, Galileo, ...)* device that supports various global navigation satellite systems.

GPX file An XML file that contains geodata, such as waypoints, routes, or tracks. In ControlDesk, you can import GPX files to visualize GNSS positioning data in a Map instrument.

Group A collection of variables that are grouped according to a certain criterion.

H

Heading Indicator An instrument displaying the heading direction of a simulated aircraft on a circular scale.

Host service A service in a real-time application that provides measurement data from the application to the host PC.

The following platforms support applications that contain the host service:

- DS1006 Processor Board platform
- DS1104 R&D Controller Board platform
- MicroAutoBox platform
- Multiprocessor System platform

Index Plotter A plotter instrument for displaying signals that are measured in an event-based raster (index plots).

Input quantity A measurement variable that is referenced by a common axis and that provides the input value of that axis.

Instrument An on-screen representation that is designed to monitor and/or control simulator variables interactively and to display data captures. Instruments can be arranged freely on layouts.

The following instruments can be used in ControlDesk:

- 3-D Viewer
- Airspeed Indicator
- Altimeter
- Animated Needle
- Artificial Horizon
- Bar
- Browser
- Bus Instrument
- Check Button
- Diagnostics Instrument
- Display
- Fault Memory Instrument
- Frame
- Gauge
- Heading Indicator
- Index Plotter
- Invisible Switch
- Knob
- Multistate Display
- Multiswitch
- Numeric Input
- On/Off Button

- Push Button
- Radio Button
- Selection Box
- Slider
- Sound Controller
- Static Text
- Steering Controller
- Table Editor
- Time Plotter
- Variable Array
- XY Plotter

Instrument Navigator A controlbar that displays a tree with all the instruments of the active layout and all the variables that are connected to them. The Instrument Navigator's main function is easy selection of instruments in complex layouts.

Instrument script A Python script used to extend the functionality of an instrument.

Instrument Selector A controlbar that provides access to ControlDesk's instruments. The instruments can be placed on a layout via double-click or drag & drop.

Interface description data (IF_DATA) An information structure, mostly provided by an A2L file, describing the type, features and configuration of an implemented ECU interface.

Internal Interpreter ControlDesk's built-in programming interface for editing, running and importing Python scripts. It contains an Interpreter controlbar where the user can enter Python commands interactively and which displays output and error messages of Python commands.

Interpreter controlbar A controlbar that can be used to execute line-based commands. It is used by the Internal Interpreter to print out Python standard error messages and standard output during the execution or import of Python scripts.

Invisible Switch An instrument for defining an area that is sensitive to mouse operations.

IOCNET IOCNET (I/O carrier network) is a dSPACE-specific high-speed serial communication bus that connects all the real-time hardware in a SCALEXIO system. IOCNET can also be used to build a multiprocessor system that consists of multiple SCALEXIO processor hardware components.

K

Knob An instrument for displaying and setting the value of the connected variable by means of a knob on a circular scale.

L

Label list A list of user-defined variables that can be used for saving connected variables, etc.

Layout A window with instruments connected to variables of one or more simulation models.

Layout Navigator A controlbar that displays all opened layouts. It can be used for switching between layouts.

Layout script A Python script used to extend the functionality of a layout.

Leading raster The measurement raster that specifies the trigger settings for the Time Plotter display. The leading raster determines the time range that is visible in the plotter if a start and stop trigger is used for displaying the signals.

LIN Bus Monitoring device A device that monitors the data stream on a LIN bus connected to the ControlDesk PC.

The LIN Bus Monitoring device works, for example, with PC-based LIN interfaces.

The device supports the following variable description file types:

- LDF
- FIBEX
- AUTOSAR system description (ARXML)

Load type The load type specifies the option to disturb a signal with or without load rejection.

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>`

Logical link A representation of an ECU specified in the diagnostics database. A logical link contains information on the ECU itself, and all the information required for accessing it, such as the diagnostic protocol used for communication between the ECU and ControlDesk. Each logical link is represented by a unique short name in the ODX database.

Look-up table A look-up table maps one or more input values to one output value. You have to differentiate between the following look-up table types:

- A 1-D look-up table maps one input value to one output value.
- A 2-D look-up table maps two input values to one output value.
- An n-D look-up table maps multidimensional table data with 3 or more input values to one output value.

Look-up table is a generic term for curves and maps.

M

Main variable A scalar variable that is visualized in an instrument that can be used to change parameter values. In addition to the main variable, additional write variables can also be connected to (but not visualized in) the same instrument. When you change the value of the main variable in an instrument, the changed value is also applied to all the additional write variables connected to that instrument.

Map A parameter that consists of

- A 1-dimensional array containing the axis points for the x-axis. This array can also be specified by a reference to a common axis.
- A 1-dimensional array containing the axis points for the y-axis. This array can also be specified by a reference to a common axis.
- A 2–dimensional array containing data points. The map assigns one data point of the array to each pair of x-axis and y-axis points.

Maps are represented by the symbol.

Map file A file that contains symbols (symbolic names) and their physical addresses. It is generated during a build process of an ECU application.

Map instrument A customized Browser instrument. It uses an instrument script to open a web map and connect positioning data to the map. The Map instrument offers prepared connection nodes to connect variables with GNSS data.

Measurement Viewing and analyzing the time traces of variables, for example, to observe the effects of ECU parameter changes.

ControlDesk provides various instruments for measuring variables.

Measurement (variable type) A scalar variable that can be measured, including individual elements of a measurement array.

Measurement variables are represented by the symbol.

Measurement array A 1-, 2-, or 3-dimensional array of measurement variables. In variable lists, ControlDesk displays entries for the measurement array itself and for each array element.

Measurement arrays are represented by the symbol.

Measurement buffer A ring buffer that buffers measurement data at the start of a measurement. The measurement buffer size determines the amount of data that can be buffered. Earlier values are overwritten by later values when the buffer capacity is exceeded (buffer overflow).

Measurement Configuration A controlbar that allows you to configure measurement, recording and data logging.

Measurement Data API Application programming interface for accessing measurement data. The API lets the user access measurement data without having to use ControlDesk.

Measurement Data Pool A controlbar that provides access to measurement data recorded in measurement data files.

Measurement raster Specification of how often a value of a variable is updated during a measurement. A measurement raster can be derived from a measurement service.

Measurement service The generic term for the following services:

- CCP service
- DsDAQ service
- Host service
- XCP service

Measurement signal list A list containing the variables to be included in subsequent measurements and recording. The list is global for all platforms/devices of the current experiment. The measurement signal list is available in the configuration area of the **Measurement Configuration** controlbar.

Measurement variable Any variable type that can be measured but not calibrated.

Measuring/recording A platform/device state defined by the following characteristics:

- A continuous logical connection is established between ControlDesk and the platform/device hardware.
- Online calibration is possible. Parameter values can be changed directly on the platform/device hardware.
- A measurement (or recording) is running.
- Platform/device configuration is not possible.

The 'measuring' / 'recording' platform/device state is indicated by the icon.

Memory page An area of a calibration memory. Each page contains a complete set of parameters of the platform/device hardware, but only one of the pages is “visible” to the microcontroller of the ECU or the real-time processor (RTP) of the platform hardware at a time.

ControlDesk supports platform/device hardware with up to two memory pages. These are usually the working page and the reference page. The parameter values on the two memory pages usually are different. ControlDesk lets you switch from one page to the other, so that when parameters are changed on one page, the changes can be made available to the ECU or prototyping hardware via a single page switch.

Messages controlbar A controlbar displaying a history of all error and warning messages that occur during work with ControlDesk.

MicroAutoBox III platform A platform that provides access to a MicroAutoBox III connected to the host PC for function prototyping purposes such as Bypassing.

MicroAutoBox platform A platform that provides access to a MicroAutoBox II connected to the host PC for function prototyping purposes such as bypassing.

Mirrored memory A memory area created by ControlDesk on the host PC that mirrors the contents of the available memory pages of calibration and prototyping hardware. For hardware with two memory pages, the mirrored memory is divided into a reference and a working page, each of them containing a complete set of parameters. When a calibration or prototyping platform/device is added to an experiment, ControlDesk initially fills the available memory pages of the mirrored memory with the contents of the ECU Image file (initial filling for calibration devices) or with the contents of the SDF file (initial filling for platforms).

- Mirrored memory for offline calibration

  Parameter values can even be changed offline. Changes to parameter values that are made offline affect only the mirrored memory.
- Offline-to-online transition for online calibration

  For online calibration, an offline-to-online transition must be performed. During the transition, ControlDesk compares the memory pages of the hardware of each platform/device with the corresponding pages of the mirrored memory. If the pages differ, the user has to equalize them by uploading them from the hardware to the host PC, or downloading them from the host PC to the hardware.
- Mirrored memory for online calibration

  When ControlDesk is in the online mode, parameter value changes become effective synchronously on the memory pages of the hardware and in the mirrored memory. In other words, parameter values on the hardware and on the host PC are always the same while you are performing online calibration.

Modular system A dSPACE processor board and one or more I/O boards connected to it.

Multi-capture history The storage of all the captures acquired during a triggered measurement. The amount of stored data depends on the measurement buffer.

Multi-pin error A feature of the SCALEXIO concept for electrical error simulation that lets you simulate a short circuit between three or more signal channels and/or bus channels. The channels can be located on the same or different boards or I/O units. You can simulate a short circuit between:

- Channels of the same signal category (e.g., four signal generation channels)
- Channels of different signal categories (e.g., three signal generation channels and two signal measurement channels)
- Signal channels and bus channels (e.g., two signal generation channels, one signal measurement channel, and one bus channel)

Multiple electrical errors A feature of the SCALEXIO concept for electrical error simulation that lets you switch electrical errors at the same time or in succession. For example, you can simulate an open circuit for one channel and a short circuit for another channel at the same time, without deactivating the first error.

Multiprocessor System platform A platform that provides access to:

- A multicore application running on a multicore DS1006 board
- A multiprocessor application on a multiprocessor system consisting of two or more DS1006 processor boards interconnected via Gigalink.

ControlDesk handles a multiprocessor/multicore system as a unit and uses one system description file (SDF file) to load the applications to all the processor boards/cores in the system.

Multistate Display An instrument for displaying the value of a variable as an LED state and/or as a message text.

Multistate LED A value cell type of the Variable Array for displaying the value of a variable as an LED state.

Multiswitch An instrument for changing variable values by clicking sensitive areas in the instrument and for visualizing different states depending on the current value of the connected variable.

N

Numeric Input An instrument (or a value cell type of the Variable Array) for displaying and setting the value of the connected variable numerically.

O

Observing variables Reading variable values cyclically from the dSPACE real-time hardware and displaying their current values in ControlDesk, even if no measurement is running. Variable observation is performed without using a measurement buffer, and no value history is kept.

For platforms that support variable observation, variable observation is available for parameters and measurement variables that are visualized in single-shot instruments (all instruments except for a plotter). If you visualize a variable in a single-shot instrument, the variable is not added to the measurement signal list. Visualizing a parameter or measurement variable in a plotter automatically adds the variable to the measurement signal list.

ControlDesk starts observing variables if one of the following conditions is true:

- Online Calibration is started for the platform.

  All the parameters and measurement variables that are visualized in single-shot instruments are observed.
- Measurement is started for the platform.

  All the visualized parameters and measurement variables that are not activated for measurement in the measurement signal list are observed. Data of the activated parameters and measurement variables is acquired using measurement rasters.

ODX database Abbreviation of Open Diagnostic Data Exchange, a diagnostics database that is the central ECU description for working with an ECU Diagnostics device in ControlDesk. The ODX database contains all the information required to perform diagnostic communication between ControlDesk and a specific ECU or set of ECUs in a vehicle network. ControlDesk expects the database to be compliant with ASAM MCD-2 D (ODX).

Offline State in which the parameter values of platform/device hardware in the current experiment cannot be changed. This applies regardless of whether or not the host PC is physically connected to the hardware.

The mirrored memory allows parameter values to be changed even offline.

Offline simulation A PC-based simulation in which the simulator is not connected to a physical system and is thus independent of the real time.

Offline simulation application (OSA) An offline simulation application (OSA) file is an executable file for VEOS. After the build process with a tool such as the VEOS Player, the OSA file can be downloaded to VEOS.

An OSA contains one or more VPUs, such as V-ECUs and/or environment VPUs.

On/Off Button An instrument (or a value cell type of the Variable Array) for setting the value of the connected parameter to a predefined value when the button is pressed (On value) and released (Off value).

Online calibration started A platform/device state defined by the following characteristics:

- A continuous logical connection is established between ControlDesk and the platform/device hardware.
- Online calibration is possible. Parameter values can be changed directly on the platform/device hardware.
- Platform/device configuration is not possible.

Before starting online calibration, ControlDesk lets you compare the memory pages on the platform/device hardware with the corresponding pages of the mirrored memory. If the parameter values on the pages differ, they must be equalized by uploading the values from the hardware to ControlDesk, or downloading the values from ControlDesk to the hardware. However, a page cannot be downloaded if it is read-only.

The 'online calibration started' platform/device state is indicated by the symbol.

Operation signal A signal which represents the result of an arithmetical operation (such as addition or multiplication) between two other signals.

Operator mode A working mode of ControlDesk in which only a subset of the ControlDesk functionality is provided. You can work with existing experiments but not modify them, which protects them from unintentional changes.

Output parameter A parameter or writable measurement whose memory address is used to write the computed value of a calculated variable to.

P

Parameter Any variable type that can be calibrated.

Parameter (variable type) A scalar parameter, as well as the individual elements of a value block.

Scalar parameters are represented by the P symbol.

Parameter limits Limits within which parameters can be changed. Parameters have hard and weak limits.

- Hard limits

  Hard limits designate the value range of a parameter that you *cannot* cross during calibration.

  The hard limits of a parameter originate from the corresponding variable description and cannot be edited in ControlDesk.
- Weak limits

  Weak limits designate the value range of a parameter that you *should not* cross during calibration. When you cross the value range defined by the weak limits, ControlDesk warns you.

  In ControlDesk, you can edit the weak limits of a parameter within the value range given by the parameter’s hard limits.

PHS (Peripheral High Speed) bus A dSPACE-specific bus for communication between a processor board and the I/O boards in a modular system. It allows direct I/O operations between the processor board (bus master) and I/O boards (bus slaves).

PHS-bus-based system A modular dSPACE system consisting of a processor board such as the DS1006 Processor Board and I/O boards. They communicate with each other via the PHS (Peripheral High Speed) bus.

Pitch variable A variable connected to the pitch scale of an Artificial Horizon.

Platform A software component representing a simulator where a simulation application is computed in real-time (on dSPACE real-time hardware) or in non-real-time (on VEOS).

ControlDesk provides the following platforms:

- DS1006 Processor Board platform
- DS1007 PPC Processor Board platform
- DS1104 R&D Controller Board platform
- DS1202 MicroLabBox platform
- MicroAutoBox platform
- MicroAutoBox III platform
- Multiprocessor System platform
- SCALEXIO platform
- VEOS platform
- XIL API MAPort platform

Each platform usually has a variable description that specifies its variables.

Platform trigger A trigger that is available for a platform and that is evaluated on the related dSPACE real-time hardware or VEOS.

Platforms/Devices controlbar A controlbar that provides functions to handle devices, platforms, and the applications assigned to the platforms.

Plotter instrument ControlDesk offers three plotter instruments with different main purposes:

- The Index Plotter displays signals in relation to events.
- The Time Plotter displays signals in relation to measurement time.
- The XY Plotter displays signals in relation to other signals.

Port configuration To interface the failure simulation hardware, an EESPort needs the hardware-dependent *port configuration file* (PORTCONFIG file). The file's contents must fit the connected HIL simulator architecture and its failure simulation hardware.

Postprocessing The handling of measured and recorded data by the following actions:

- Displaying measured or recorded data
- Zooming into measured or recorded signals with a plotter
- Displaying the values of measurement variables and parameters as they were at any specific point in time

Processor board A board that computes real-time applications. It has an operating system that controls all calculations and communication to other boards.

Project A container for collecting and managing the information and files required for experiment/calibration/modification tasks in a number of experiments. A project collects the experiments and manages their common data.

Project controlbar A controlbar that provides access to projects and experiments and all the files they contain.

Project root directory The directory on your file system to which ControlDesk saves all the experiments and documents of a project. Every project is associated with a project root directory, and several projects can use the same project root directory. The user can group projects by specifying several project root directories.

ControlDesk uses the Documents folder as the default project root directory unless a different one is specified.

Properties controlbar A controlbar providing access to the properties of, for example, platforms/devices, layouts/instruments, and measurement/recording configurations.

Proposed calibration A calibration mode in which the parameter value changes that the user makes do not become effective on the hardware until they are applied. This allows several parameter changes to be written to the hardware together. Being in proposed calibration mode is like being in the offline calibration mode temporarily.

Push Button An instrument (or a value cell type of the Variable Array) for setting the value of the connected parameter by push buttons.

Python Editor An editor for opening and editing PY files.

Q

Quick start measurement A type of measurement in which all the ECU variables configured for measurement are measured and recorded, starting with the first execution of an ECU task. ControlDesk supports quick start measurements on ECUs with DCI-GSI2, CCP, and XCP (except for XCP on Ethernet with the TCP transmission protocol).

Quick start measurement can be used to perform cold start measurements. Cold start means that the vehicle and/or the engine are cooled down to the temperature of the environment and then started. One reason for performing cold start measurements is to observe the behavior of an engine during the warm-up phase.

R

Radio Button An instrument for displaying and setting the value of the connected parameter by radio buttons.

Real-time application An application that can be executed in real time on dSPACE real-time hardware. A real-time application can be built from a Simulink model containing RTI blocks, for example.

Record layout A record layout is used to specify a data type and define the order of the data in the memory of the target system (ECU, for example). For scalar data types, a record layout allows you to add an address mode (direct or indirect). For structured (aggregated) data types, the record layout specifies all the structure elements and the order they appear in.

The `RECORD_LAYOUT` keyword in an A2L file is used to specify the various record layouts of the data types in the memory. The structural setup of the various data types must be described in such a way that a standard application system will be able to process all data types (reading, writing, operating point display etc.).

Record layout component A component of a record layout. A structured record layout consists of several components according to the ASAP2 specification. For example, the AXIS_PTS_X component specifies the x-axis points, and the FNC_VALUES component describes the function values of a map or a curve.

Recorder An object in the Measurement Configuration controlbar that specifies and executes the recording of variables according to a specific measurement configuration.

Recorder signal list A list that contains the variables to be included in subsequent recordings.

Recording Saving the time traces of variables to a file. Both measurement variables and parameters can be recorded. Recorded data can be postprocessed directly in ControlDesk.

A recording can be started and stopped immediately or via a trigger:

- Immediate recording

  The recording is started and stopped without delay, without having to meet a trigger condition.
- Triggered recording

  The recording is not started or stopped until certain trigger conditions are met. These conditions can be defined and edited in ControlDesk.

Reduction data Additional content in an MF4 file that allows for visualizing the MF4 file data depending on the visualization resolution. Reduction data therefore improves the performance of the visualization and postprocessing of measurement data.

Reference data set A read-only data set assigned to the reference page of a device that has two memory pages. There can be only one reference data set for each device. The reference data set is read-only.

Reference page Memory area containing the parameters of an ECU. The reference page contains the read-only reference data set.

Note

Some platforms/devices provide only a working page. You cannot switch to a reference page in this case.

Resynchronization Mechanism to periodically synchronize the drifting timers of the platform/device hardware ControlDesk is connected to. Resynchronization means adjustment to a common time base.

Roll variable A variable connected to the roll scale of an Artificial Horizon.

S

Sample count trigger A trigger that specifies the number of samples in a data capture.

A sample count trigger can be used as a stop trigger.

SCALEXIO platform A platform that provides access to a single-core, multicore or multiprocessor SCALEXIO system connected to the host PC for HIL simulation and function prototyping purposes.

SCALEXIO system A dSPACE hardware-in-the-loop (HIL) system consisting of at least one processing hardware component, I/O boards, and I/O units. They communicate with each other via the IOCNET. In a SCALEXIO system, two types of processing hardware can be used, a DS6001 Processor Board or a real-time industry PC as the SCALEXIO Processing Unit. The SCALEXIO system simulates the environment to test an ECU. It provides the sensor signals for the ECU, measures the signals of the ECU, and provides the power (battery voltage) for the ECU and a bus interface for restbus simulation.

SDF file The system description file that describes the files to be loaded to the individual processing units of a simulation platform. It also contains the variable description of the relevant simulation application.

The SDF file is generated automatically when the TRC file is built.

Segment The minimum part a segment signal can consist of.

There are different kinds of segments to be used in segment signals:

- Segments to form synthetic signal shapes (sine, sawtooth, ramp, etc.)
- Segments to perform arithmetical operations (addition, multiplication) with other segments
- Segments to represent numerical signal data (measured data)

Segment signal A signal consisting of one or more segments.

Selection Box An instrument for selecting a text-value entry and setting the respective numerical value for the connected variable.

Signal

- Representation of a variable measured in a specific measurement raster.
- Generic term for segment signals and operation signals.

  A signal is part of a signal description set which can be displayed and edited in the working area.

Signal description set A group of one or more signals.

A signal description set and its signals can be edited in the working area by means of the Signal Editor. Each signal description set is stored as an STZ file either in the **Signal Description Sets** folder or in the **Signal Generators** folder.

Signal Editor A software component to create, configure, display, and manage signals in signal description sets.

Signal file A file that contains the wiring information of a simulator and that is part of the standard dSPACE documentation of dSPACE Simulator Full-Size. Normally, dSPACE generates this file when designing the simulator. Before using a failure simulation system, users can adapt the signal file to their needs.

Signal generator An STZ file containing a signal description set and optional information about the signal mapping, the description of variables, and the real-time platform.

The file is located in the **Signal Generators** folder and used to generate, download, and control Real-Time Testing sequences, which are executed on the real-time platform to stimulate model variables in real time.

Signal Mapping A controlbar of the Signal Editor to map model variables to signals and variable aliases of a signal generator.

Signal Selector A controlbar of the Signal Editor. The Signal Selector provides signals and segments for arranging and configuring signal description sets in the working area.

SIL testing Abbreviation of *software-in-the-loop testing*.

Simulation and testing of individual software functions, complete virtual ECUs (V-ECUs), or even V-ECU networks on a local PC or highly parallel in the cloud independently of real-time constraints and real hardware.

Simulation application The generic term for offline simulation application (OSA) and real-time application.

Simulation system A description of the composition of V-ECU models, environment models, real ECUs, and their interconnections required for simulating the behavior of a system. A simulation system is the basis for the generation of a simulation application for a given simulator platform.

Simulation time group Group of platforms/devices in an experiment whose simulation times are synchronized with each other. If resynchronization is enabled, ControlDesk synchronizes a simulation time group as a whole, not the single members of the group individually.

Simulator A system that imitates the characteristics or behaviors of a selected physical or abstract system.

Single-processor system A system that is based on one dSPACE processor or controller board.

Single-shot instrument An instrument that displays an instantaneous value of a connected variable without keeping a value history. In ControlDesk, all instruments except for a plotter are single-shot instruments. For platforms that support the variable observer functionality, you can use single-shot instruments to observe variables.

Slave application An application assigned to the slave DSP of a controller or I/O board. It is usually loaded and started together with the real-time application running on the corresponding main board.

Slave DSP A DSP subsystem installed on a controller or I/O board. Its slave application can be loaded together with the real-time application or separately.

Slider An instrument (or a value cell type of the Variable Array) for displaying and setting the value of the connected variable by means of a slide.

Sound Controller An instrument for generating sounds to be played.

Standard axis An axis with data points that are deposited in the ECU memory. Unlike a common axis, a standard axis is specified within a curve or map. The parameters of a standard axis can be calibrated, which affects only the related curve or map.

Start trigger A trigger that is used, for example, to start a measurement raster. A platform trigger can be used as a start trigger.

Static Text An instrument for displaying explanations or inscriptions on the layout.

Steering Controller An instrument for changing variable values using a game controller device such as a joystick or a steering wheel.

Stimulation Writing signals to variables in real-time models during a simulation run.

Stop trigger A trigger that is used, for example, to stop a measurement raster.

String A text variable in ASCII format.

Strings are represented by the symbol.

Struct A variable with the struct data type. A struct contains a structured list of variables that can have various data types. In ControlDesk, a struct variable can contain either parameters and value blocks or measurement variables and measurement arrays. ControlDesk supports nested structs, i.e., structs that contain further structs and struct arrays as elements.

Structs are represented by the symbol.

Struct array An array of homogeneous struct variables.

Struct arrays are represented by the symbol.

STZ file A ZIP file containing signal descriptions in the STI format. The STZ file can also contain additional MAT files to describe numerical signal data.

Sub data set A data set that does not contain the complete set of the parameters of a platform/device.

Symbol A symbolic name of a physical address in a MAP file. A symbol can be associated to a variable in the Variable Editor, for example, to support an address updates.

System variable A type of variable that represents internal variables of the device or platform hardware and that can be used as measurement signals in ControlDesk to give feedback on the status of the related device or platform hardware. For example, an ECU's power supply status or the simulation state of a dSPACE board can be visualized via system variables.

T

Table Editor An instrument for displaying and setting values of a connected curve, map, value block, or axis in a 2-D, 3-D, and grid view. The Table Editor can also display the values of a measurement array.

The Table Editor can be used for the following variable types:

- Common axis ( )
- Curve ( )
- Map ( )
- Measurement array ( )
- Value block ( )

Time cursor A cursor which is visible at the same time position in the following instruments:

- In all Time Plotters
- In all XY Plotters
- In all bus monitoring lists

You can use the time cursor to view signal values at a specific point in time. If you move the time cursor, all measured signals and the respective parameters are updated. Instruments and bus monitoring lists display the values that are available at the selected time position.

Time Plotter A plotter instrument for displaying signals that are measured in a time-based raster (time plots).

Topology A description of the processor boards belonging to a multiprocessor system and their interconnections via Gigalinks. The topology also contains information on which Gigalink port of each processor board is connected to the Gigalink ports of other processor boards in the multiprocessor system.

Topology information is contained in the real-time application (PPC/x86/RTA) files of the multiprocessor system's processor boards.

TRC file A variable description file with information on the variables available in an environment model running on a dSPACE platform.

Trigger A condition for executing an action such as starting and stopping a measurement raster or a recorder.

The generic term for the following trigger types:

- Duration trigger
- Platform trigger
- Sample count trigger

Trigger condition A formula that specifies the condition of a trigger mathematically.

Triggered measurement The measurement of a measurement raster started by a platform trigger. The data flow between the dSPACE real-time hardware or VEOS and the host PC is not continuous.

U

Unassigned data set A data set that is assigned neither to the working page nor to the reference page of a platform/device. An unassigned data set can be defined as the new working or reference data set. It then replaces the "old" working or reference data set and is written to the corresponding memory page, if one is available on the platform/device.

Unplugged A platform/device state defined by the following characteristics:

- The logical connection between ControlDesk and the hardware was interrupted, for example, because the ignition was turned off or the ControlDesk PC and the hardware were disconnected.
- Before the state of a platform/device changes to 'unplugged', the platform/device was in one of the following states:
  - 'Connected'
  - 'Online calibration started'
  - 'Measuring' / 'Recording'

Tip

A device for which the connection between ControlDesk and the device hardware currently is interrupted is also set to the 'unplugged' state when you start online calibration if both the following conditions are fulfilled:

- The device's **Start unplugged** property is enabled.
- The **Start online calibration behavior** property is set to 'Ignore differences'.

This is possible for CCP and XCP devices. For details on the two properties listed above, refer to General Settings Properties (ControlDesk Platform Management).

- If the Automatic Reconnect feature is enabled for a platform/device and if the platform/device is in the 'unplugged' state, ControlDesk periodically tries to re-establish the logical connection for that platform/device.
- Online calibration is impossible. Offline calibration is possible.
- Platform/device configuration is possible.

The 'unplugged' platform/device state is indicated by the ⚠ icon.

Untriggered measurement The measurement of a measurement raster not started by a platform trigger. The data flow between the dSPACE real-time hardware or VEOS and the host PC is continuous.

User function An external function or program that is added to the ControlDesk user interface for quick and easy access during work with ControlDesk.

User Functions Output A controlbar that provides access to the output of external tools added to the **Automation** ribbon.

V

Value block A parameter that consists of a 1- or 2-dimensional array of scalar parameters.

In variable lists, ControlDesk displays entries for the value block itself and for each array element.

Value blocks are represented by the ▥ symbol.

Value conversion The conversion of the original *source values* of variables of an application running on an ECU or dSPACE real-time hardware into the corresponding scaled *converted values*.

Variable Any parameter or measurement variable defined in a variable description. ControlDesk provides various instruments to visualize variables.

Variable alias An alias name that lets the user control the property of a segment by a model parameter of a real-time application.

Variable Array An instrument for calibrating parameters and displaying measurement variable values.

The Variable Array can be used for the following variable types:

- Measurement ( )
- Measurement array ( )
- String ( )
- Struct ( )
- Struct array ( )
- Value ( )
- Value block ( )

Variable connection The connection of a variable to an instrument. Via the variable connection, data is exchanged between a variable and the instrument used to measure or calibrate the variable. In other words, variable connections are required to visualize variables in instrument.

Variable description A file describing the variables in a simulation application, which are available for measurement, calibration, and stimulation.

Variable Editor A tool for viewing, editing, and creating variable descriptions in the ASAM MCD-2MC (A2L) file format. The Variable Editor allows you to create A2L files from scratch, or to import existing A2L files for modification.

Variable Filter A variable filter contains the filter configuration of a combined filter, which is used to filter the variable list in the **Variables** controlbar using a combination of filter conditions.

Variables controlbar A controlbar that provides access to the variables of the currently open experiment.

V-ECU Abbreviation of *virtual ECU*.

ECU software that can be executed in a software-in-the-loop (SIL) testing environment such as a local PC or highly parallel in the cloud independently of real-time constraints and real ECU hardware.

Vehicle information The ODX database can contain information for one or more vehicles. Vehicle information data is used for vehicle identification purposes and for access to vehicles. It references the access paths (logical links) to the ECUs.

VEOS A simulator which is part of the PC and allows the user to run an offline simulation application (OSA) without relation to real time.

VEOS Player is the graphical user interface for VEOS.

VEOS platform A platform that configures and controls the offline simulation application (OSA) running in VEOS and that also provides access to the application's environment VPU.

VEOS Player An application running on the host PC for editing, configuring and controlling an offline simulation application (OSA) running in VEOS.

Verbal conversion A conversion in which a conversion table is used to specify the computation of numerical values into strings. The verbal conversion table is used when you switch the value representation from source to converted mode and vice versa.

Verbal conversion range A conversion in which a conversion table is used to specify the computation of a range of numerical values into strings. The verbal conversion range table is used when you switch the value representation from source to converted mode and vice versa.

View set A named configuration of the controlbars of ControlDesk. A view set has a default state and a current state that can differ from the default state. The configuration includes the geometry, visibility, and docking or floating state of controlbars.

Visualization The representation of variables in instruments:

- Measurement variables are visualized in instruments to view and analyze their time traces.
- Calibration parameters are visualized in instruments to change their values.

VPU Abbreviation of *virtual processing unit*. A VPU is part of an offline simulation application in VEOS. Each VPU runs in a separate process of the PC.

VPU is also the generic term for:

- V-ECUs
- Environment VPUs
- Controller VPUs
- Bus VPUs

W

Working data set The data set currently residing in the memory of a platform/device hardware. There can be only one working data set for each calibration platform/device. The working data set is read/write.

Working page Memory area containing the parameters of an ECU or prototyping hardware (memory page). The working page contains the read/write working data set.

If the platform/device also provides a reference page, ControlDesk lets you switch between both pages.

Writable measurement A scalar variable that can be measured and calibrated.

X

XCP Abbreviation of *Universal Measurement and Calibration Protocol*. A protocol that is implemented on electronic control units (ECUs) and provides access to ECUs with measurement and calibration systems (MCS) such as ControlDesk.

XCP is based on the *master-slave principle*:

- The ECU is the slave.
- The measurement and calibration system is the master.

The “X“ stands for the physical layers for communication between the ECU and the MCS, such as CAN (Controller Area Network) and Ethernet.

The basic features of XCP are:

- ECU parameter calibration (CAL)
- Synchronous data acquisition (DAQ)
- Synchronous data stimulation (STIM), i.e., for bypassing
- ECU flash programming (PGM)

The XCP protocol was developed by ASAM e.V. (Association for Standardisation of Automation and Measuring Systems e.V.). For the protocol specification, refer to http://www.asam.net.

The following ControlDesk devices support ECUs with an integrated XCP service:

- XCP on CAN device
- XCP on Ethernet device

XCP on CAN device A device that provides access to an ECU with XCP connected to the ControlDesk PC via CAN. Using the XCP on CAN device, you can access the ECU for measurement and calibration purposes via XCP (*Universal Measurement and Calibration Protocol*).

XCP on Ethernet device A device that provides access to an ECU or V-ECU with XCP connected to the ControlDesk PC via Ethernet. The XCP on Ethernet device provides access to the ECU/V-ECU via XCP (*Universal Measurement and Calibration Protocol*) for measurement and calibration purposes.

XIL API EESPort Electrical Error Simulation port (EESPort)

XIL API MAPort platform A platform that provides access to a simulation platform via the ASAM XIL API implementation that is installed on your host PC.

XY Plotter A plotter instrument for displaying signals as functions of other signals.