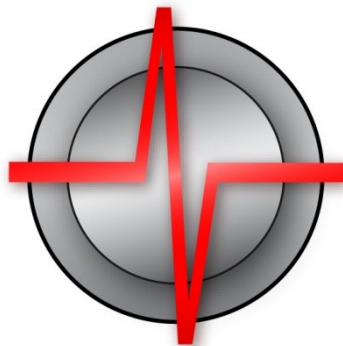


# **ROTAS**

## Noise Analysis System



## Manual and Introduction

© 2013 Discom Industrial Measurement and Test Systems Inc.

Neustadt 10-12, 37073 Göttingen, Germany

Tel.: +(49) 551 548 33 10

Fax: +(49) 551 548 33 43

Email : [info@discom.de](mailto:info@discom.de)

[www.discom.de](http://www.discom.de)

# ROTAS

## *Noise Analysis System*

---

### Contents

<b>Introduction .....</b>	<b>6</b>
About This Manual .....	6
Components of the Analysis System .....	9
The Measurement Computer .....	11
The TAS Box .....	12
<b>Concepts and Basics .....</b>	<b>16</b>
Important Terms .....	16
Limit Values .....	22
Noise Analysis Theory .....	27
<b>The TasAlyser Program .....</b>	<b>37</b>
The Project Directory .....	37
Top View .....	38
Operating the Windows .....	42
Test Bench Connection .....	47
Manual Control .....	48
User Rights and Authorization Levels .....	52
<b>Parameter Administration with TasForms .....</b>	<b>54</b>
The Database in the Overall System .....	54
Security and Maintenance Measures .....	55
Creating and Erasing a Type .....	57
General Form Functions .....	63

Test Setup .....	66
Setting Limits .....	67
Parameterizing Measured Values .....	70
Learn Parameters .....	89
Defining Error Codes.....	92
<b>The Talimer.....</b>	<b>94</b>
Overview .....	95
Navigation .....	96
Editing a polygon.....	96
Polygon Management.....	99
Reference Measurements.....	101
Individual Polygons.....	107
Limit Table .....	109
<b>Further TasAlyser Functions .....</b>	<b>110</b>
Configuration, Favorites and Windows.....	110
Speed and other Command Variables.....	113
Trigger .....	119
Wave Audio Recording and Playback.....	124
TasWavEditor.....	129
Configuring the TAS Box.....	134
Log File .....	138
<b>Signal Monitoring, Calibration, Filter.....</b>	<b>139</b>
Audio Signal Monitoring and Signal Level.....	139
Calibration .....	140
Using the Calibration Function.....	141
Extended Functions .....	145
Using Filters .....	148
<b>Measurement Archives and Evaluation.....</b>	<b>153</b>
Archiving in the TasAlyser.....	153
The Presentation Program .....	156

<b>Help from Discom.....</b>	<b>159</b>
Transmitting Files.....	159
When the TasAlyser Does Not Work.....	160
Strange Noises.....	161
Unwanted Test Results.....	162
<b>Appendix A: Rotas Mobile .....</b>	<b>163</b>
Setting Up the Hardware.....	163
The TasAlyser Mobile Project.....	166
Driving Measurements.....	168
Storing the Results.....	169
Block Diagram.....	170
<b>Appendix B: Signal Processing.....</b>	<b>171</b>
RMS, Crest, Kurtosis.....	171
Exponential Averaging.....	172
Filter Definition.....	173
<b>Appendix D: Shift Force Evaluation.....</b>	<b>178</b>
Overview.....	178
Setup.....	179
Evaluation methods.....	180
Limit setup.....	182



# Introduction

---

## About This Manual

This manual describes the Rotas Noise Analysis System, focusing on the measurement program and the parameter database. The goal is to enable you, as noise analysis users, to operate the system in everyday situations and to deal with the tasks which typically occur.

The noise analysis system consists of several components (see the next section). Each of these components is very efficient and offers numerous possibilities for widely diverse applications and tasks. Therefore this manual can only serve as an introduction and is not able to cover each detail – this task is reserved for the specialized manuals on the individual components.

This manual assumes a “typical” application situation in noise analysis, the routine testing of aggregates (e.g. gearboxes) on an end-of-line test bench. On the test bench, different types of aggregates (like gearboxes with different gear ratios) are tested. Noise analysis helps to find loud aggregates and thereby identify different kind of defects. One of the main tasks for the operator is to take care of the limit values which means drawing the line between good (O.K.) and bad (not O.K.).

The Rotas system can also be used for mobile noise measuring, for instance in car test drives, or in the continuous operation testing of individual aggregates at test benches. In principle, the routine test and the mobile measurement are very similar. Where differences between mobile and continuous operation testing are of particular importance, these will be dealt with at the appropriate place in the manual.

## Contents Overview

At the beginning, you probably don't have the time to read the entire manual. Many aspects will become clear when you have gained some experience in operating the noise analysis system or when faced with specific problems. The following overview offers a brief description of the content of each chapter in the manual. If you have to get going straightaway, you will get tips which of the chapters you should read first.

### Introduction

The rest of this chapter gives an overview of the system and its components. Furthermore, it explains the measurement computer and its connections to the environment. This chapter is not long, but it helps you greatly to get orientation and therefore you should read it first.



## Concepts and Basics

In this chapter, some basic terms are introduced first. Then you get information about how evaluation limits are produced. These things are important for understanding how the noise analysis system operates and therefore essential for you to read about them. In addition, the chapter describes the procedure of rotationally synchronous analysis and the identification of production errors based on the noise pattern. If you are just starting with noise analysis, you may skip the “theory part” for now, but it does help in gaining a deeper understanding of the interconnections.

## The TasAlyser Program

The most important control elements, displays and windows of the “TasAlyser” measurement program are introduced in this chapter. Browse through it, look at the headlines and illustrations, and check whether there is anything you need to know straightaway.

## The Parameter Database TasForms

This chapter tells you how to operate the parameter database. You will see how to create new aggregate types and how to manage existing types and their limits. The advanced possibilities of the parameter database, e.g. the creation of measurement procedures and trigger profiles, are described detailedly in a separate manual on the parameter database.

## The Limit Curve Editor Talimer

This is a graphical tool to edit limit curves. It operates on the parameter database but connects the limits with measured data. Where the abilities of the data base interface end, Talimer continues.

## More TasAlyser Functions

In this chapter we describe some other functions of the TasAlyser program which are occasionally necessary in normal operation, e.g. the recording and playback of wave-audio data. Search here or in the index if you are looking for something specific.

## Signal Monitoring and Calibration

This section explains the calibration functionality, which is integrated into the TasAlyser program.

## Measurement Data Archive and Evaluation with Marvis

Here you can read what happens to the measurement data after they have been stored to disk. Furthermore, you get a short introduction to the Marvis evaluation program (formerly referred to as “Presentation”). Please refer to



the detailed Marvis manual if you frequently work with the evaluation program.

### **Help from Discom**

Discom will, of course, help you with any problem in noise analysis; not only with operation but also with the analysis of noise phenomena. This chapter describes how you can provide us with the information we need so that we can help you as efficiently as possible.

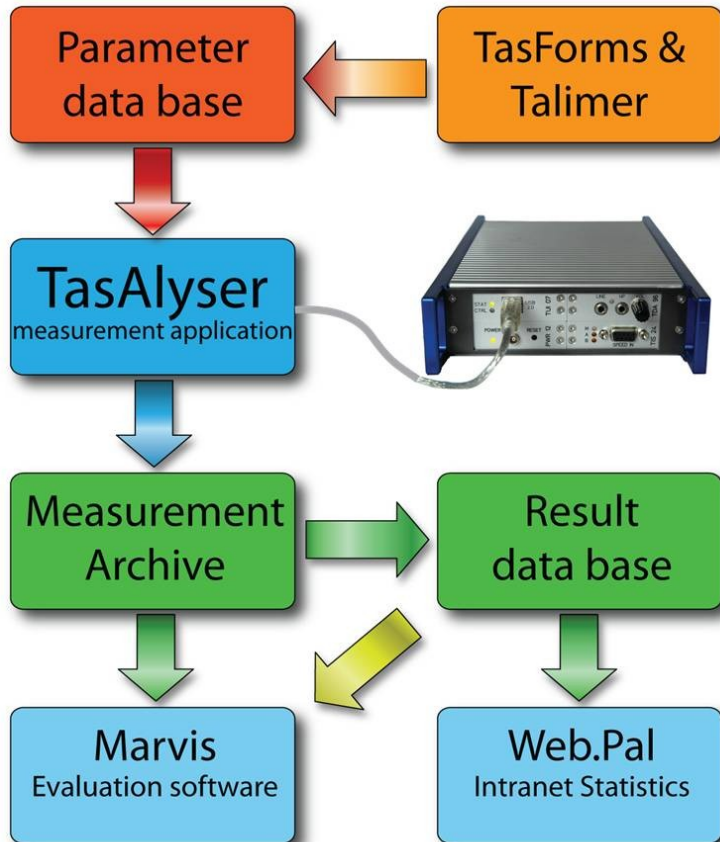
### **Appendix: Rotas Mobile**

This chapter deals with the use of the measurement program and the TAS Box for mobile measurements, e.g. when driving a car, and describes the “Mobile” measurement project.



## Components of the Analysis System

This manual describes the Rotas noise analysis system and its central components. These components are shown in the picture and explained below:



- The “TasAlyser“ measurement program: The TasAlyser runs on a PC which is connected to a “TAS Box” data collector. The TasAlyser program operates in real time, generates measured values and variables, compares these with limit values, evaluates them accordingly and stores the results in a *measurement data archive*.
- The parameter database *TasForms* and *Talimer*: The user interface of the parameter database *TasForms* is used to administer the design data of the candidates and types, so that the *TasAlyser* program can calculate order position and gear ratios. Furthermore, the parameter database sets which measurement procedure shall be applied and which measured values shall be generated. Finally, the parameter



database contains the settings for the generation of limit values. For editing limit curves, you can use the tool Talimer (=TAs LIMit Editor) which is especially designed for this task.

- The result database: The TasAlyser stores the results and data of each measurement in a file, the measuring data archive. An auxiliary program, the Collector, sorts these files into a central database. This database serves as the basis for statistic analyses to set limit values, as well as for answering questions on the characteristics of individual aggregates, which were, in some cases, measured some time ago.
- The production statistics tool *Web.Pal*: This Intranet-based tool uses the result database to make production and error statistics available. In the analysis of the distribution and the time series of measured values, Web.Pal also offers an early warning function, which recognizes possible failure issues, before there are real failures at the test stand.
- The evaluation program *Marvis* / Presentation: The information stored in the archives and measurement database can be made visible and evaluated using Marvis (Measurement ARchive VISualisation). The program also allows the automatic generation of reports, statistical analyses as well as detailed analyses of noise phenomena.

In addition to these major components, there are other elements, such as the Collector mentioned above or the TasWavEditor. These auxiliary programs are also explained in this manual.

The TasAlyser program runs on the measurement computer that is connected to the test bench. Both the parameter administration TasForms and the measured value database can be installed independently of each other, either on the measurement computer or on another computer (server). If several measurement computers (lines) are used in parallel, then installation on a server is preferable since all the test benches can be administered by one parameter database and the results can be filed in a common measured value database.

The evaluation programs (WebPal, Marvis) access the data via network. Both can therefore be run locally on the measurement computer or on the server, as well as on any other workstation which can access the database. Similarly, TasForms and Talimer, the user interfaces of the parameter database, can be run on another computer connected by network.



## “Getting started”

Usually, there are links on the measurement computer desktop to start the measurement program, the parameter administration and the evaluation:



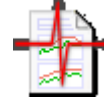
*TasAlyser*



*TasForms*



*Talimer*



*Presentation*

Usually, the desktop shows also a folder “Rotas for Experts” containing the links above (which are the most important tools on the measurement computer) and more.

Normally, a reboot of Windows will automatically start the measurement program.

---

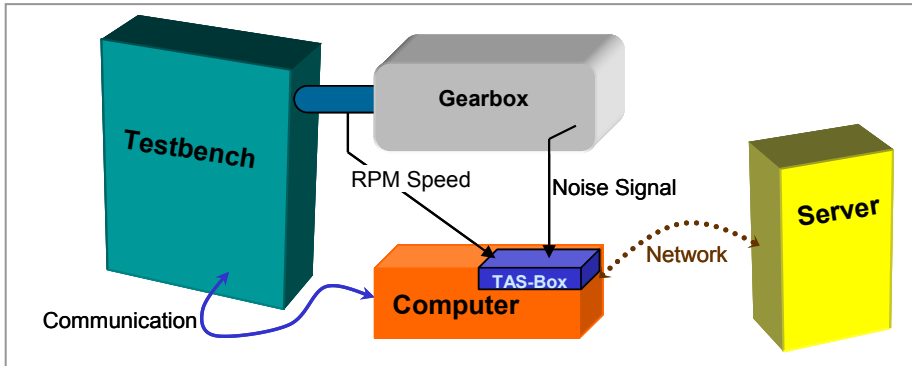
## The Measurement Computer

The measurement computer is a Windows PC which is equipped with the TAS hardware for data collection. The TAS Box is of modular design and equipped according to the requirements of the test. More details on the TAS Box can be found in the section “The TAS Box” below.

The TAS Box is connected to the measurement computer via USB. If the TAS hardware is built into the measurement computer, the USB wire can be reached from the outside. With mobile systems the TAS Box is run as a separate piece of equipment. For demanding applications several TAS Boxes can be used at one computer.

With the Tas Box, the TasAlyser program records sensor data like noise, rotational speeds, sometimes torque, temperature etc. In order to carry out the test the TasAlyser also requires information about the test cycle, especially type and serial number of the aggregate or the current test step (gear). This information is transmitted to the measurement computer by the test bench control. The test bench control on the other hand can enquire evaluation results, as well as further information, such as error reports.

Usually, the measurement computer is connected with a server over a network. The measurement data archives are sent to this server for sorting into the database.



The figure shows the measurement computer in its environment:

The network connection to the server is optional and can only be available intermittently. A permanent network connection, however, opens up the possibility of remote maintenance of the measurement computer.

In case of a mobile system, there is no test bench. Communication occurs between the measurement program and the driver. If desired, the server can be connected before and after mobile measurements.

## Communication with the Test Bench

The connection with the test bench can be done in many different ways, such as, for example, an “antiquated” serial line, Profibus or UDP network protocol. In most cases, the TasAlyser program and the test bench software communicate over a command-oriented protocol with plaintext instructions. A window in the TasAlyser program allows you to monitor communication.

The communication volume depends on the requirements of the task and also on test bench capabilities. Usually, at the begin of a test cycle, the test bench transmits the aggregate type and a serial number, during the test the name of the next test step (e.g. gear when testing gearboxes), and, at the end, the information that the test cycle is completed. The test bench then queries the evaluation result.

The test bench can also query intermediate data, detailed error reports and even measured values. You can find further details on the communication protocol in another section of this manual.

---

## The TAS Box

The special data collection hardware of the TAS System consists of individual modules which are built into a 5¼ inch frame (the same size as a



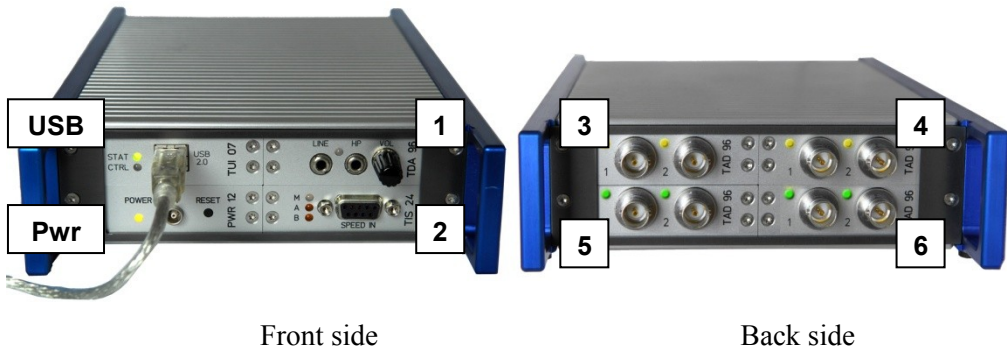
DVD-drive, for example). This TAS Box can take 8 modules, whereby two of these places are required by the USB connection module and the power module which supplies the other cards with the necessary voltage. The other six places can be used according to need and requirement.

The following modules are available:

- USB connection module. As already mentioned, every TAS Box must contain one such module.
- Power Module: supplies the other cards with stable voltage, also necessary 1 x per Box.
- A/D Converter Module: This module possesses two input channels with a maximal scanning rate of 100 kHz and a resolution of 32 Bit. ICP supply for appropriate sensors (accelerometer, microphones) can be switched on. It is also possible to capture rotational speeds or DC voltage signals (e.g. torque). The maximal input voltage for signals is  $\pm 30V$ ; different sensitivity ranges and amplification levels can be selected.
- TIS Rotational Speed Module: This module is optimized for the capture and pre-processing of speed sensor signals and permits pulse rates of up to 10 MHz. A TIS module can capture four rotational speeds simultaneously.
- D/A Converter Module: As well as being able to listen to sensor signals using the PC sound card (a possibility built into the TasAnalyser), signals can also be emitted over the D/A converter module.
- Empty module: unused places are filled with empty modules.

The TAS Box assembly results from project requirements: for example, with a TIS card and 5 A/D cards, 4 rotational speeds and further 10 sensor signals can be captured.

The illustration below shows the front and the back of a TAS Box which has not been built into a measurement computer, as deployed, for example, with a mobile system (see Appendix A: Rotas Mobile). Four of the eight modules in the TAS Box can be reached from the front side, the other four from the back. They are enumerated as follows:



Front side

Back side

The USB connection module is located upper left on the front side, with the power module underneath. The power module has an input for an additional power supply (see below) as well as the RESET button. A D/A module can be seen above right (Slot 1) and a TIS module on the lower right (Slot 2). USB and power module are always on the front side of the Tas Box in these two slots.

On the back side of the Tas box shown above, you see four A/D cards. It depends on the project which of the possible cards are used in the slots 1-6. For details, please refer to chapter Configuring the TAS Box on page 134.

The A/D converter module and the TIS module are designed for extremely low power consumption. This allows that a TAS Box with up to four A/D modules and TIS modules can be supplied with USB power only. However, you can use ICP supply for only 5 A/D channels at the most when you use USB power. This is just enough for a mobile system with three A/D converter modules and a TIS (five microphones or structure borne noise sensors). It can be supplied with USB power only and requires no additional power supply.

If the TAS Box either contains more modules, or more sensors must be supplied with ICP voltage, then the power module must be attached to a 12 V supply.

When the TAS Box is built into a measurement computer, then you will only see the four modules on the front (similar to the illustration above). The connectors of the modules at the rear are wired to connectors at the back of the computer.

When you use a complete measurement computer, you will usually not have anything to do with the TAS Box directly, since all settings are controlled by the TasAlyser software. With a mobile system, the situation is similar. You only have the TAS Box as a separate unit. All you have to do is to attach the necessary cables to the TAS Box.



## Frequent fittings of the TAS Box

As already mentioned above, the fitting of the TAS Box depends on the current test situation and can vary extremely between different projects. Nevertheless, you find some standard fittings which are equal between most projects.

### Tas Box internally

When a Tas Box is built into a PC, you expect to connect the necessary signals at the back of the PC like the other connectors (display, keyboard, etc.). This has the consequence that you have to use the slots at the back of the Tas Box. The front side holds only USB and power module. Since you need at least one noise and one speed signal for noise analysis, this leads to one TAD96 card in slot 3. In case you want to use digital speed signals, you find the TIS card in slot 4. Further TAD96 can eventually be found in slots 5 and 6.

### Tas Box externally

If you use a Tas Box externally, you will put the TAD96 and a possible TIS card in the two slots on the front because then all connections are on the same side of the Box. Sometimes, you find a “short” Tas Box which has only the connections at the front. For both situations you find the TAD96 card in slot 1, another TAD96 or a TIS card in slot 2. The mobile system is an exception from this rule. It also uses the connectors on the back, because you don't have the possibility to connect 6 signals (4 microphones, 1 structure bourne noise sensor, 1 speed signal) on the front side only. If you connect them at the back, you have all sensor signals on the back and only USB and power supply at the front.

### Mobile system as test stand spare

In special cases, you have mobile systems with a Tas Box which is equipped like an internal Box. This means, TAD96 in slot 3, a TIS card in slot 4 and two further TAD96 cards in slot 5 and 6. The reason for this fitting is that you want the external system to be compatible to the internal systems at the test stands. Such a box can, if need be, connected to a test bench PC and temporarily replace the internal box (if defect) without having the need to change anything in the configuration of the measurement program.



# Concepts and Basics

---

## Important Terms

When you work with the noise analysis system, you will keep coming across some terms again and again. Some are from the data base background, others from acoustics or from gearbox design. These terms are described here briefly.

### Key and data set

Each entry in the parameter and result data base needs a unique address, its *key*. Each completely specified key designates a unique data set (parameter set, measurement result), and there cannot be two data sets with the same key. Therefore, a data set is a collection of different data which belong to the “object” specified by the key. For example, the object “Human Being” has first name, name, address, etc.

### Type and Base Type

The noise analysis system is designed to deal with several different *types* of aggregates, e.g. gearbox types which differ in their gear ratios, or motor types, which differ in their additional components.

Not all differences between types are relevant for noise analysis. If a manufacturer has two identical types with different names because they are designed for different cars to be built into, this usually is not a difference for noise analysis. Relevant differences are when rotational frequencies differ or when additional noise sources are present. Because you want as little work as possible to set parameters for these types, types which do not differ for aspects of noise analysis find their parameters in the same dataset.

To be more exact, the *types* are the names which the test stand sends to the measurement program for the different analysis objects (e.g. gearboxes). For each type name, there is a *base type* which refers to the dataset. In other words, the base type is the key to the datasets. Usually, a base type has more than one type name for the test stand to “call” it.

### Family or model

Sometimes you have the situation that you want to test aggregates at a test stand which do not only differ in transmission ratios. One such difference is that gear boxes sometimes have 5 and 6 speeds variants. Such a difference is covered by the concept of families or models.

Each aggregate type belongs to a defined family from the beginning. This gives the measurement program the opportunity to notice the differences





between families. A 6 speed gear box will have parameters for the 6<sup>th</sup> gear, a 5 speed transmission will have none.

If you want, you can even have completely different objects in one database. They then belong to different families. The consequence is, that you have to take special care when you make settings. The more the different objects differ, the easier it is to enter useless entries into the database.

### **Test bench, test bench group**

The idea to have as less datasets as possible does also have effect on test benches. Each *test bench group* represents one data set which can be used for different *test benches*. This has the same effect as with types and base types: Each test bench of a test bench group uses the same dataset.

### **Test Step (Mode)**

A complete test, a *test cycle*, consists of a sequence of sections. A possible section in gear box testing, for example, is “3rd gear, rotational speed increasing”. These sections are called *test steps* or *modes*. In each test step the defined measure values are recorded and evaluated individually. Each test step has its own limits. Also for a variety of other settings, there exists one dataset for each test step.

When errors are found, the error message contains information during which test step an error arose. Results are also classified in the measurement data archives according to the test step.

### **Location, Rotor and Order Source**

The noise analysis system tries to specify a noise source as exact as possible. For this, the test candidate is “separated into its parts” for the analysis. These parts of a gearbox for example appear as “Locations” in the system. Most test candidates have turning parts like shafts as well as other parts which make noise such as meshing gears. In the Rotas world, a shaft is a *Rotor* and a meshing gear is an *order source*. The term “order source” shows that this part produces a characteristic frequency (the “base order”) which is expected to show up in the spectrum. On the other hand, the characteristic frequency of a “rotor” is its rotational speed: Everything which turns with the same rotational speed belongs to the same rotor.

The Rotas noise analysis system can be used to test a wide variety of aggregates. The number and characteristics of order sources and rotors in these aggregates is as different as the aggregates themselves. Sometimes, a part is a rotor and an order source at the same time, for example if you test a single gear wheel.



## Channels: Synchronous and “Mix”

A central step in noise analysis is the rotationally synchronous analysis (for details see in the relevant section starting from page 27). The rotationally synchronous processing allows to separate the noise contribution of different rotors. The measured values obtained here are known as *synchronous* values (e.g. synchronous spectra), abbreviated as *Sync*. However, since the noises in an aggregate are not all necessarily linked to a rotor, non rotor synchronous measured values are also generated, which are known as *mix* measured values (e.g. mix spectrum), because they are based on a mixture of all sources of noise.

Depending upon its type, a production error is found either more in synchronous or more in mix. For example, a gear wheel nick is found in the synchronous measured values of the appropriate rotor, a loud bearing, on the other hand, more in mix.

As an option, there can also be another type of processing channel, a *fixed frequency* or *fix channel*. This channel does not refer to the speed of a rotor, but uses a fixed sampling rate. Fixed channels are used, for example, in analyzing background noises (such as gear shift noises in gearboxes).

Once more, to make this quite clear: all processing channels - synchronous channels, mix channel, fixed channel - are processed copies of *one* sensor signal. If the noise analysis system is equipped with several sensors, then each sensor has its own set of synchronous channels, its own mix channel and, if necessary, its own fixed channel.

## Instruments

The measurement program calculates (in each test step, see above) a wide range of very different measured values and curves. In order to organize these, we use the term *instrument*: each type of measured variable is formed by an appropriate instrument. For example, there is the instrument “order spectrum”, the instrument “Rms”, or the “Crest” instrument (used for nick detection; see “Crest & Co” on page 29). Many instruments have parameters, which specify the desired calculation in detail. You find the instruments in the parameter database, in the result display of the TasAlyser program, and, of course, in the measurement result archives. The instruments are divided into two main categories: single values and curves. As the name says, the result of a single value instrument consists of a single number. So, for example, the Crest instrument can produce “3.49” as result. Single values are very user friendly: the limit value is also only one number, which allows the results to be shown in a table. Furthermore, single values can easily be evaluated statistically (by generating distributions and time series). Curve instruments, however, produce a curve as their result. Examples are Spectra or level tracks over rotational speed (“order track”). The limits for such



instruments are curves as well. The visualization is more complicated for such results, and statistical evaluations are more difficult.

### **Measurement values and instrument parameter**

A Measurement value is a special kind of analysis which a specific instrument does in a specific channel for a specific location (see Instruments and Channels: Synchronous and “Mix” beginning at page 18). Usually, it depends on the test step whether a specific kind of analysis is possible or not. Some locations are useless in some test steps (if a planetary gear set is blocked, for example, it does not make sense to try an analysis of its gear mesh noise). Most instruments can carry out more than one analysis at one time. In this case, these different kinds of analysis are identified by the instrument parameter. Depending on the instrument, the measurement values can be “single values”, “curves” like spectra and tracks, or curve arrays (“spectrograms”).

### **Clavis**

Some key entries are already used when the measurement program queries the database for settings, for example test bench and type. Still, the measurement program has to distinguish test step, location, channel, sensor, instrument and measurement value. Since this 6 parts are explicitly important for the measurement program to distinguish data, the key which consists of these parts has been given a separate name and is called *clavis*. A *clavis* in the measurement program uniquely identifies a measurement value as well as the corresponding result or its limit. (“Clavis” is Latin and means “key.”)

### **Limits and Error Codes**

The real use of noise analysis becomes clear when the different measured values are evaluated. The Rotas system does this by comparing each individual measured value (whether single value or curve) with an individual limit. If the measurement value failed on that limit<sup>1</sup>, then an error is announced and the test aggregate is declared “n.O.K.” (“not O.K.”). (Measured value = limit is only just “O.K.”).

The Rotas system does not use school grades or results such as “nearly n.O.K.”. Either an aggregate is good and can be used (sold), or it is not good and must be repaired.

---

<sup>1</sup> Depending on the kind of analysis, this can mean “limit exceeded” (most limits are evaluated that way), or it can mean “limit not reached” (e.g. for a sensor signal check).



For special applications, classification into several categories is possible. This means, however, a significantly higher parameterization effort for the user (i.e. for you) and should only be used when there are really good reasons for it. In our experience, we have found that an aggregate is either good or bad.

In the parameter database, an error code and limit settings are assigned to every measured value. (Different measure values can, of course, use the same error code if they indicate the same defect). For each error code, there is an error message. If a measure value fails on its limit, you get a message in TasAlyser's output window, consisting of the error code, the error message and further details (such as the test step, the instrument and the location causing that error). Normally, you need relatively few error codes – only as many as you want to have different message texts.

In addition, the error codes can be transmitted to the test bench and then stored on a data medium at the aggregate. In this case, you will probably like to set up more error codes. You can find more about error codes in the chapter on the parameter database.

## Command Variables and Triggers

A *Command variable* is a measured value which is used to control the measurement or as reference. The typical command variable is the rotational speed, and for each application of noise analysis at least one rotational speed is required. It may be that your aggregate has several independent rotational speeds. Another common command variable is the torque. Time is also a reference variable but a special one: It always exists. You do not need a special sensor for it.

A typical test cycle consists of a sequence of command variable ramps. For example, that the rotational speed is increased steadily from an initial 1000 rpm. to 4000 rpm. and then lowered back to 1000 rpm. That way, two ramps (one rising and one falling) have been run. For noise analysis this results in two test steps (see above.).

In order to specify a measurement range within a ramp (for example from 1500 to 3500 rpm) and to record value tracks with reference to a speed within this measuring range, the measurement program has a *trigger* function. The trigger settings are specified in the parameter database. In the measurement program, they are used to control measurements (start/ stop) and to generate tracking curves.

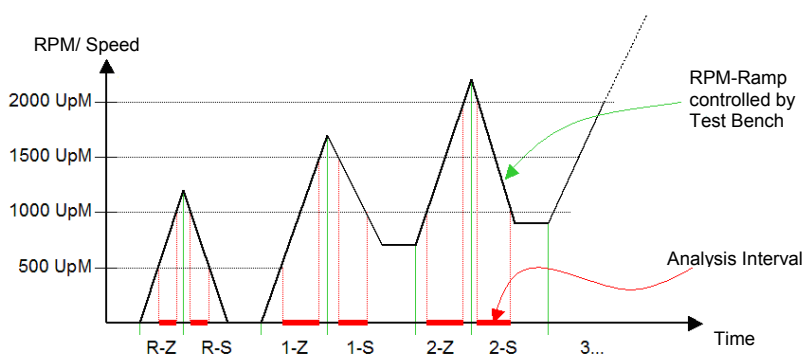
## The Test Cycle

A typical test cycle for an aggregate in end-of-line testing is as follows:



1. The aggregate is clamped onto the test bench. The test bench transmits the aggregate type and serial number to the TasAlyser, whereupon the TasAlyser loads the parameter and limits which are currently valid for this type. This step is called *Insert*. It is the start of the test cycle.
2. The test bench transmits the name of the first test step to the TasAlyser. The TasAlyser begins to monitor the rotational speed (or another reference variable as specified in parameterization).
3. The rotational speed (or other reference variable) reaches the initial value specified in the trigger settings. From now on, measure values are captured. This moment is called *measurement start*.
4. When the reference variable reaches the defined target value, the trigger determines *measurement end*. Measure value capturing is completed now, and the results for this test step are evaluated and displayed.
5. The test bench transmits the name of the next test step. Repeat with step 2.
6. At the end of the test cycle, the test bench transmits the *Remove* instruction to the TasAlyser. At this moment, the final result of the test is evaluated and can then be queried by the test bench. The TasAlyser stores all measurement data in an archive file, which can be sent later to the result database if required.

The following diagram illustrates a typical test cycle for a gearbox: rotational speed is increased (“drive”) and decreased (“coast”) in the ramps. The analysis intervals within the ramps are specified by the trigger settings. The duration of these intervals (in seconds) depends on the steepness of the ramps.



The sequence of the test steps is random, at least as far as the TasAlyser is concerned. A test step may also be repeated (immediately or later), whereby



all the results and error messages from the first measurement<sup>2</sup> are discarded and new ones are recorded. You do not have to use all of the test steps which are contained in the database in a test cycle<sup>3</sup>.

If the test bench announces a new test step (step 5), before the measurement end condition of the previous test step has been reached (step 4), then the results of this test step are discarded and it is regarded as being not measured.

Beside the regular measurements within the test steps, other measurements can be done, which are not connected to the normal test steps. An example is gear shift noises with gearboxes, which typically occur during the *transition* between test steps. Another example is the gear ratio test, with which the TasAlyser tests the correct gear ratio of a gearbox on the basis of two rotational speeds. The gear ratio test is started and terminated by the test bench with separate commands.

The test cycle can also be controlled manually, which is necessary when you use the mobile system. The TasAlyser has a corresponding control window for this purpose (see the following chapter).

It is also possible to cancel a test cycle either by a test bench command or manually. In this case no evaluation result will be generated, all the measured values are discarded and no result data archive is produced.

---

## Limit Values

As already described in the previous section, the noise test uses limit values to separate good and bad. Every measured value has a limit value (or limit curve), which can be influenced individually.

The following comments refer to *upper limits*, i.e. limit values which, when exceeded, result in an n.O.K. evaluation. This is, by far, the most common case. However, the basic principles are the same for measured values which are tested against a lower limit or for deviation from a target value.

### How Limits Are Generated

Every limit value is generated by combining learned values with fixed settings.

---

<sup>2</sup> It is possible to generate the mean or the maximum of repeated measurements.

<sup>3</sup> However, it is possible to set up the TasAlyser in a way that an error is reported if one of the essential test steps have not been measured or if measurement values are missing.



“Learning” consists of calculating the mean and the standard deviation (variance) of the measured value – refer also to the following section "How Limits Are Learned". Using mean and standard deviation, the learned limit is calculated as follows:

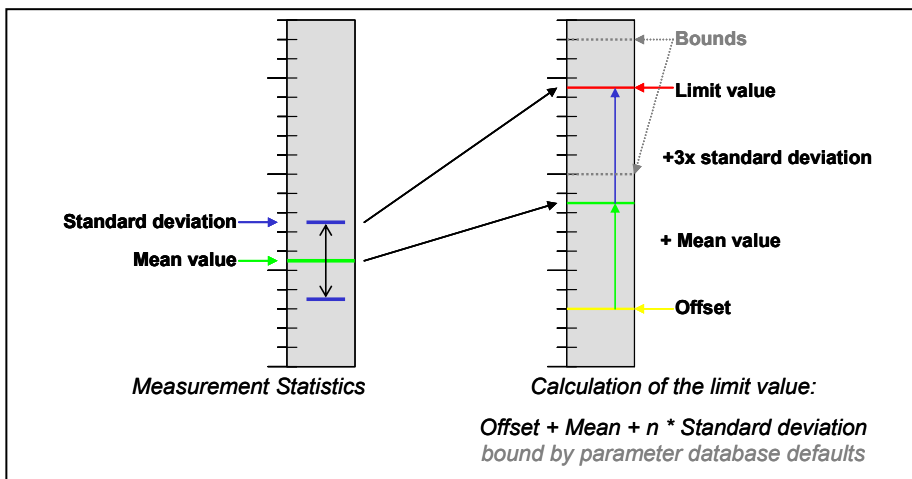
$$\text{Limit value} = \text{Base value ("Offset")} + \text{Mean} + \text{Factor} \times \text{Standard deviation}$$

The numbers for *Offset* and *Factor* are set in the parameter database. An example: A mean of 77.5 and a standard deviation of 2.8 with a frequently used offset = 5 and factor = 3 results in:

$$\text{Limit value} = 5 + 77.5 + 3 \times 2.8 = 90.9$$

In addition to *Offset* and *Factor* the parameter database also contains a lower and an upper barrier for each limit value. The actual limit value is not permitted to lie outside these bounds. If, in the above example, a lower bound of 95 and an upper one of 110 are entered in the parameter database, then the limit value used is 95 and not just 90.9.

The following diagram illustrates the generation of the limit value:

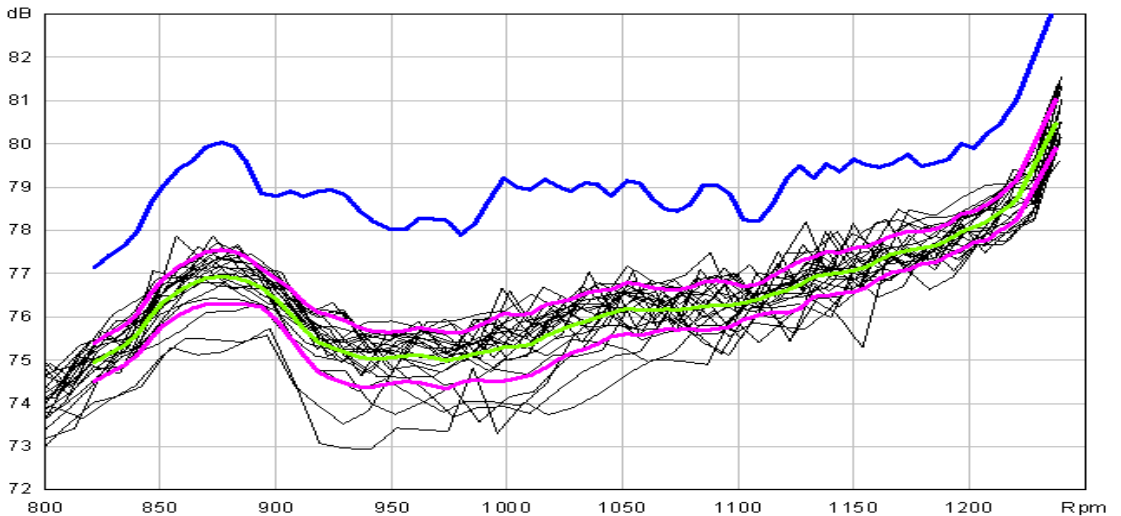


If the lower and the upper bound are set equal in the parameter database, then learning is completely overdriven: the bound value will always be used as the (fixed) limit value.

The description above refers to a single value. For spectra and curves each point of the curve is learned separately. (This means: mean and standard deviation are calculated for each point). *Offset* and *factor* are values which are valid for the whole curve. The lower and upper bounds are again curves, which are defined in the database as polygons. They are known as minimum and maximum polygons. It is also true here that if minimum polygon = maximum polygon (also possible in sections), then the polygons determine a fixed limit curve in this range.



The figure below shows a number of measurements (black), their mean value (green), the band  $\pm 1$  x standard deviation (magenta) and a possible learned limit from mean  $+ 5$  x standard deviation (blue).



### Spectral Values: “Hats” in Spectral Limits

The Rotas noise analysis system has an extra feature for spectral limit curves: the *spectral values*, also called *hats*. It can be specified in the database that for the characteristic frequencies (orders, see also “Frequency, Order, Harmonics” on pages 30ff.) of order sources separate single values (such as “meshing order of gear wheel A”) are extracted, evaluated and learned. The limits of these single values can also be set as fixed values (by means of minimum bound = maximum bound), although the rest of the spectrum is learned normally. The learned limit curve is turned off at the positions of these “hats”.

The limits of these spectral single values are inserted into the (otherwise) learned limit curve, which then shows these “hats” at the corresponding positions (that’s how they got their name).

The reason for these spectral values is that the behaviour of order sources (concerning standard deviation) is different to the rest of the spectrum. This is the reason why you want to specify independant limits for their frequencies in the spectrum. Sometimes, you even want only to turn off evaluation at these positions (e.g. when the frequency of an order source shows up in the spectrum of a shaft where it does not belong to). Order sources usually appear in more than one synchronous spectrum (which belong to the two or more meshing wheels). The noise level should be the same in each spectrum, therefore you do not need to set tight limits in each





spectrum. You set the tight limit for only one spectrum and set high limit for the others to avoid double evaluation.

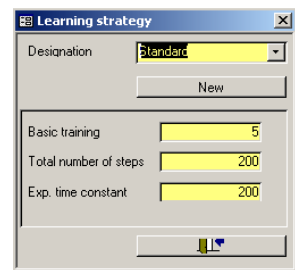
Since the positions of these frequencies in the spectrum depend on the aggregate type (especially on the number of teeth for the components), it is not possible to simply integrate the spectral values in the minimum and maximum polygons. Instead, the TasAlyser takes over the task of calculating the correct frequencies and positioning the spectral values accordingly, depending on aggregate type and order source.

The “hats” use the single value instrument “spectral value” in the parameter and in the result database.

## How Limits Are Learned

For calculating the mean and the standard deviation, which correspond to learned limits, a number of measurements are necessary. What does the TasAlyser do if the first aggregate of a type is waiting to be tested?

Learning is divided into two phases: Base learning and additional learning. Base learning involves a small number of aggregates (5 to 20), additional learning a much larger number (e.g. 200). Both numbers are set in the parameter database as shown in the figure on the right.



During base learning the aggregates are evaluated using the upper bound from the parameter database. If one of the first aggregates tested is *very* loud, then it will already be identified as being n.O.K.

After base learning has been completed, the first learned limit is generated using the mean value and the standard deviation calculated from the measurements of these first aggregates. The additional learning phase begins with the next aggregate.

Each subsequent aggregate is tested against the learned limit valid at that moment. If the result is n.O.K., then the aggregate is segregated out. However, if the result is O.K. then the data of this gearbox contributes to the whole and a new limit is generated.

Measured gearboxes (after restart of learning)



As the number of tested gearboxes increases with each additional gearbox, mean and standard deviation become more stable. Once the maximal number of aggregates has been reached, learning stops and the limits remain where they are. If you want, you can have neverending learning by entering the



value “-1” for the maximal number of aggregates. Then the “additional learning” phase is never left.

### **The Time Constant**

In addition to the number for base learning and the total learning number, you find a third number in the parameter database. This is the exponential time constant.

To be exact, the mean values are not calculated evenly over all the learned measurements. In contrast, the later (younger) measurements are weighted higher than older measurements.

The reason for this is that when testing more recent measurements there is a much better learned limit available than there was with earlier measurements. And, since the first aggregates during base learning were only tested against the maximal bounds, it could well be that these would not be O.K. when tested against the current limits – one would, therefore, like to lessen the influence of these first measurements.

The amount of weighting is determined using the time constant. The greater the time constant is when compared with the learning target, the more even will be the weighting of all the measurements. Small time constants weight the recent measurements more strongly.

Assume that you have a learning target of 200. With a time constant of likewise 200 the weighting of the first measurement compared to the last is only about 37%. With a time constant of 100 the weighting of the first measurement is about 14%, with a time constant of 500, however, 67%.

You can re-initiate learning at any time, either completely or selectively for only individual limit values. You will find more information in the chapter on Learn Parameters on page 89 ff.



## Noise Analysis Theory

This section describes the “scientific background” to rotationally synchronous noise analysis. Therefore, it is less relevant for operating the TasAlyser or for setting limits at first. Nevertheless, if you want to understand what the individual measured values mean and how you can deduce the causes from these numbers, then you should read on.

### Rotationally Synchronous Analysis

The precise error detection of the Rotas noise analysis is based essentially on the rotationally synchronous analysis of the noises. This makes it possible to extract the noise contributions of the aggregate’s different internal shafts and rotors from a sensor signal.

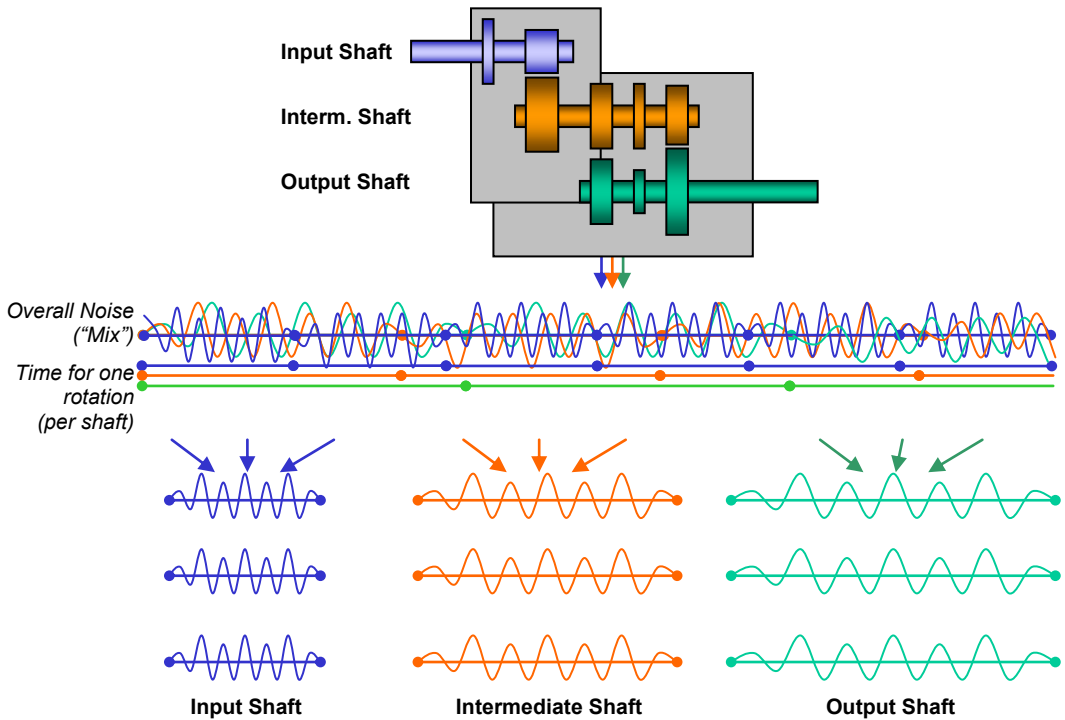
The parameter database contains construction data for all aggregate types. Given the necessary speed information (usually the speed of the input shaft), the TasAlyser is able to calculate the rotational speed of each gear wheel or rotor because it knows the internal transmission ratios.

From the relative rotational frequency of a rotor it can then be calculated how long a whole revolution will take at the current rotational speed of the reference shaft. For different rotors rotating at different speeds, the time taken for a revolution is also different. The TasAlyser cuts a copy of the total signal for each rotor into sections, which exactly cover one revolution of this rotor at a time.

Finally, after averaging over several revolutions of a rotor, a rotationally synchronous time signal is obtained, in which the noise components, which are not synchronous with this rotor (and therefore come from other rotors), are suppressed.



The diagram below illustrates graphically the principles of rotationally synchronous analysis:



### Synchronous Channels and Mix

After the rotationally synchronous analysis step there are several copies of the sensor signal. In each case, these are synchronous to a rotor and are further analyzed in parallel and independently. These processing strands are known as *synchronous channels*.

Not all the noises in an aggregate are necessarily synchronous to a rotor in the construction (an example is bearing noise). Since these noises are not synchronous to a rotor, they are suppressed in all synchronous channels. In order to prevent these noises from escaping analysis, there is another processing channel, which is known as the *mix* channel. Although the input signal is also cut into blocks for this channel which contain one or several revolutions of a reference shaft, it contains all the noise components because the averaging is omitted for these blocks.

Optionally, a further processing channel is available, a *fixed frequency* or *fix channel*. This does not refer to the revolutions of a rotor, but has a fixed scanning rate. Fixed channels are used, for example, to analyze background noises (such as gear shift noises in gearboxes). Once more, to make this quite clear: all processing channels - synchronous channels, mix channel, fixed



channel - are processed copies of one sensor signal. If the noise analysis system is equipped with several sensors, then each sensor has its own set of synchronous channels, its own mix channel and, if necessary, its own fixed channel.

## Crest & Co

The time domain analysis is the first step after calculating a synchronous (or mix) channel. In this step, different parameters are obtained from the signal of one revolution. The most important of these parameters are *RMS*, *Peak* and *Crest*.

The RMS value corresponds to the total energy of the signal – so to speak, the volume<sup>4</sup>. A high RMS value means that the aggregate is loud. If the RMS value of a synchronous channel is high, then the noise comes from this rotor. A higher RMS value in the mix channel suggests a generally loud aggregate or a cause beside the rotors. Typical RMS values lie between 1 and 10, depending upon the aggregate type and size, the rotational speed and other factors.

Occasionally the RMS value is also converted to the logarithmic dB scale, so that it is directly comparable with levels arising in the spectra. This value is known as the *master volume*. (You can find more details in Appendix B: Signal Processing on page 171ff.).

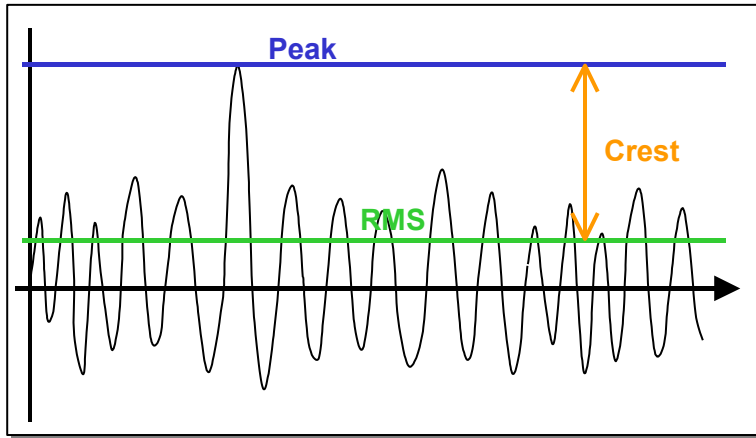
The Peak value is simply the highest occurring value, i.e. the signal spike. A single loud crashing noise during the measurement produces a high Peak value. In addition, a high Peak value will result if there is a clear “tick” during each revolution of a rotor. The signal spike will then simply appear more than once.

Because of this, the Peak value can hint to a nick of a rotor, such as a defective tooth in a gear wheel. The height of the signal spike, however, also depends on the background noise: an altogether louder aggregate or rotor (= higher RMS value, see above) will usually yield higher Peak values. On the other hand, the height of the signal spike does not necessarily have to increase with increasing rotational speed. All in all, the Peak value’s suitability for nick detection is limited.

The *Crest value* is much more reliable. It is calculated (for each revolution) as the ratio of peak value to mean value, i.e. Peak/ RMS:

---

<sup>4</sup> technically speaking, volume and total energy are completely different things



The Crest value indicates how strongly the signal spike steps out of the ground noise. A high Crest value is, therefore, a very much clearer indication of “ticking” than a high peak value. Typical Crest values lie between 4 - 8, depending upon the aggregate type.

The Crest value is calculated separately for each rotor (synchronous channel). A high Crest value in a synchronous channel hints to a nick at one of the order sources (gear wheels) on this rotor.

The Kurtosis is related to the Crest value. It increases if the signal contains many spikes. In noise terms this corresponds to a “crackle”. Defective needle bearings, for example, can cause a crackling noise.

## Frequency, Order, Harmonics

For each revolution, a spectrum is calculated from the (rotationally synchronous) time signal of each synchronous channel. (Occasionally you will see the term “FFT” = “Fast Fourier Transform”.) The characteristic frequencies of different order sources can be found in a spectrum. If the spectrum deviates from the (learned) standard, it is possible to deduce different defects from the kind of deviation.

If a time signal is processed directly in the spectral analysis, you get a frequency spectrum. If, for example, a distinctive component with 160 oscillations per second occurs in the time signal, then a line appears in the frequency spectrum at 160 Hz.

If, however, the spectral analysis is used on the rotationally synchronous time signal, then multiples of the revolution frequency are obtained instead of frequencies with the unit Hz. For example, if a distinctive component occurs with 16 oscillations per revolution in the time signal, then a line appears at 16, which means 16 times the rotational frequency or the 16th

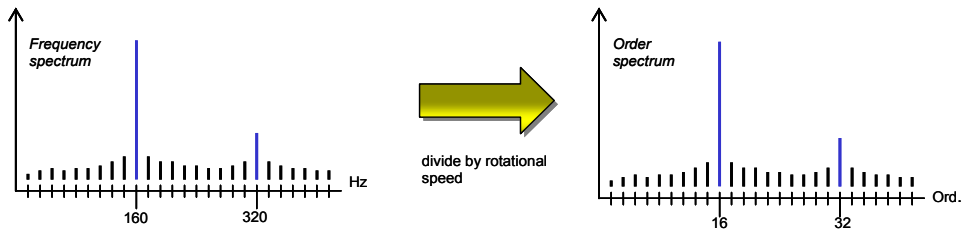


order. For this reason, the spectrum of the rotationally synchronous time signal is known as the *order spectrum*.

If the noise of a gear wheel with 16 teeth is analyzed, then 16 small “clicks” will be heard for each revolution when the teeth of the gear wheel mesh with the paired gear wheel. These 16 “clicks” produce a line in the order spectrum at the 16th order. This line is independent of the rotational frequency (rotational speed). It does not matter whether the gear wheel turns with 10 or 20 revolutions per second. The 16 “clicks” per revolution always remain and, therefore, the 16<sup>th</sup> order. That is different for the frequency in the frequency spectrum: with 10 revolutions per second the 16 “clicks” generate a frequency of 160 Hz, with 20 revolutions per second, however, 320 Hz.

This example demonstrates the advantage of the order spectrum compared with the frequency spectrum. The order spectrum is independent of the number of revolutions. You can assign the spectral components quite simply to the sources (like the 16th order to the 16 teeth of a the gear wheel).

Simple noise analysis systems produce an order spectrum by generating a frequency spectrum and then dividing the frequency axis by the rotational speed:



However, the rotationally synchronous analysis of the time signal in the Rotas system produces order spectra with very much finer resolution and can calculate a spectrum for each rotor. The result of the “simple” order analysis, however, can be compared to the “mix” channel of the Rotas system (see “Synchronous Channels and Mix” above).

### Harmonics

As described in the example, the dominating noise sources are, in particular, meshings, i.e. the noise which occurs when the teeth of gear wheels engage. Similar to a guitar string, meshing does not produce a pure sine tone with only one frequency, but, like the musical instrument, consists of root and harmonics.

The root frequency or base order (e.g. the 16<sup>th</sup> order) in particular is found in the spectrum with its multiples (32<sup>nd</sup>, 48<sup>th</sup>, 64<sup>th</sup> order etc.). In the context of Rotas noise analysis we call the base order the “first harmonic” or “H1”, the double base order “second harmonic” and/or “H2”, etc.



Harmonics can be clearly recognized in a typical gear wheel spectrum. Whether or not H1 is higher than H2, or whether H4 is still clearly recognizable, depends on the geometry and surface of the particular gear wheel. This means that no general defaults (with regard to the limit values) can be set for harmonic patterns, instead the circumstances of the project must act as a guideline.

Beside meshing orders and harmonics, usually also *sidebands* appear. High sidebands can refer to eccentricity or ovality (see “Typical Noise Patterns” below).

## The instrument “Spectral Value”

On the whole, the spectrum shows the general characteristics of a noise. In addition, individual positions in the spectrum, particularly in an order spectrum, have special meaning and deliver important information on the component to be tested. Such positions include the harmonics and their sidebands, which have already been discussed, but other positions can also be significant depending on the aggregate.

The “spectral value” tool delivers a single value, which corresponds to the value of the spectrum at a certain position – thus, for example, the height of the first harmonic as a single value.

In reality, positions in the order spectrum, which correspond to parts of the meshing order (e.g. half gear mesh order) are sometimes conspicuous. Damaged or worn out grinding wheels in gear wheel manufacturing can literally “grind in” such peculiarities on a gear wheel. These are referred to as circular pitch errors.

Compared with the spectrum in general, the spectral value has the advantage that you can specify the limit separately and independantly (see “Spectral Values: “Hats” in Spectral Limits“ on page 24) as well as independant error codes. Furthermore, single values can be evaluated statistically more easily.

Moreover, the “spectral value” instrument is not limited to only extracting an individual order from the spectrum. It can also determine the maximum value of an order band, or the total energy of an order band.

## Measurement Value Tracks

All the measured variables looked at so far have one thing in common: during testing time (for instance, over the rotational speed ramp) the values are maximized, minimized or averaged (depending upon parameterization) and deliver a final result, e.g. a spectrum. What is usually ignored here is the measured variable’s course over rotational speed, time or torque. Often irregularities do not appear during the whole testing time, but only at specific





rotational speeds or under specific torque conditions. It then disappears if only one final value is generated over the whole testing time.

In order to fill this gap, different “tracking” instruments for peak, RMS, Crest, kurtosis, spectral value and spectra exist. These instruments are used to record the measured value’s course relative to its reference signal as a curve, allowing it to be evaluated. The spectral value’s course against the reference variable for example is known as the “order track”.

When spectra courses are recorded against a reference variable, the result is called a spectrogram. With regard to the data volume, spectrograms are the “heaviest” representation, but they show an quite exact picture of noise behavior during the test.

## Second Level Instruments

The measure values covered up to now have been determined directly through averaging, minimizing, maximizing or other calculations performed directly on the data stream. The spectral value is the only exception here, because this can only be determined when the corresponding peak hold spectrum has been completed.

Instruments whose results are based on the processing of the results of other tools, are known as second level instruments. Beside the spectral value, there are other such second level instruments which can be processed only after another measure value has been calculated. The “curve interval” and the “curve polygon” belong to this category.

Both second level instruments need a tracking curve as input datum. From this curve, the curve interval calculates a single value for a specified section of the curve (maximum, minimum, mean). By applying the curve interval on a tracking curve, you have the option to divide the whole track into sections which show characteristic noise behaviour and calculate a single value for each section reflecting these characteristics. The advantage of this is that you can also evaluate track characteristics statistically (as you can for all single values).

The curve polygon is used to compare a curve with a polygon and then generate a characteristic value. In a simple case the minimum or the maximum can be determined within the polygon’s validity interval (similar to the curve interval). The new thing is that you can also calculate the area between the polygon and the curve. This kind of evaluation is carried out, for example, in the analysis of curves which represent gearshift force against displacement. The measure value, which is produced here, characterizes the gearshift work.

In parameterization, generation of limit values and evaluation, second level instruments do not differ from other instruments. The only important thing is

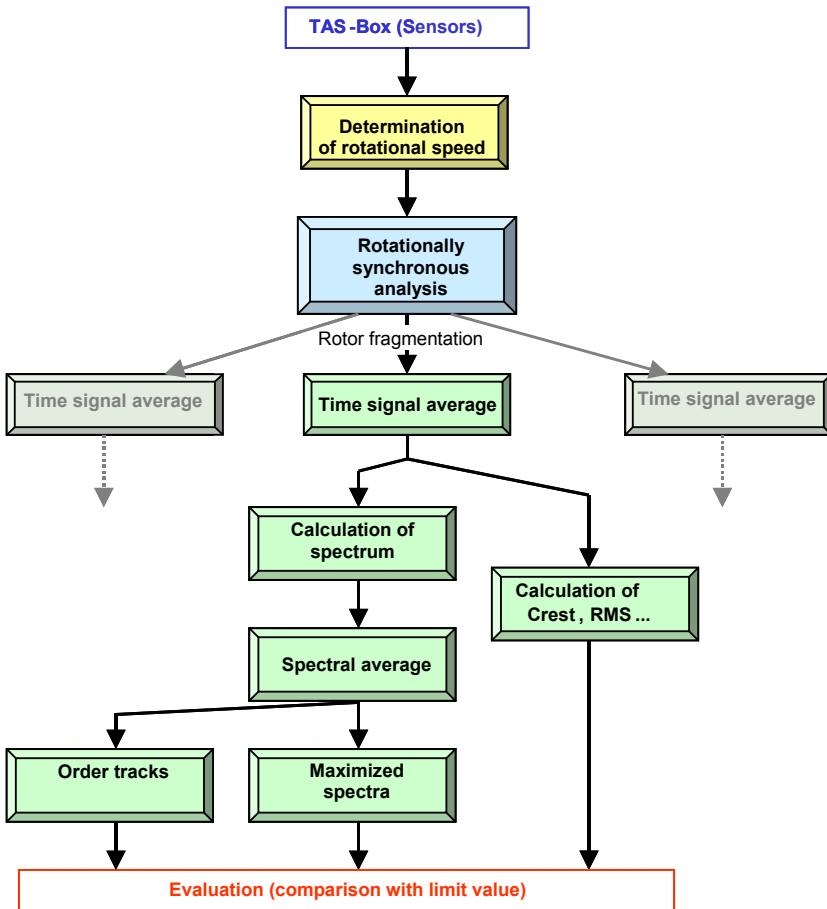


that in order to use a second level instrument (such as the curve interval), the source instrument (the measured value track) must be available as well.

### Analysis Steps

A rough operational procedure in noise analysis has already emerged from the previous remarks: rotationally synchronous analysis and separation of synchronous channels, calculation of Crest & Co, spectral analysis, secondary instruments, etc.

The following flow diagram shows the typical steps in noise analysis:



Further analysis steps can, of course, occur, depending upon your aggregate or project. This diagram can serve as orientation and also as clarification of the three important sources of evaluable measured variables: single values from the time signal, maximized order spectra and order tracks over the measuring ramp.

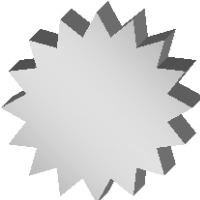
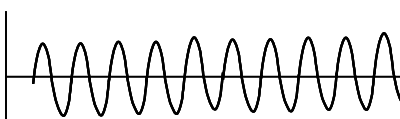
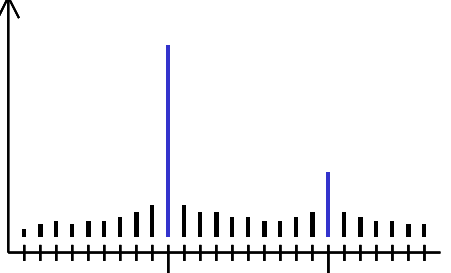
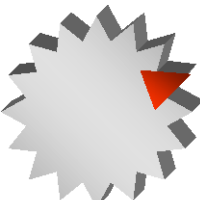
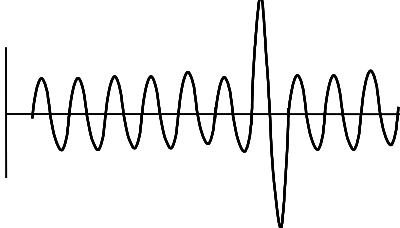
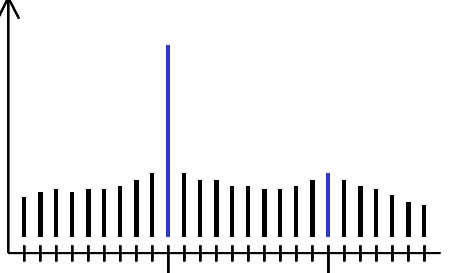
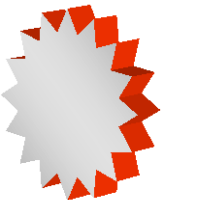
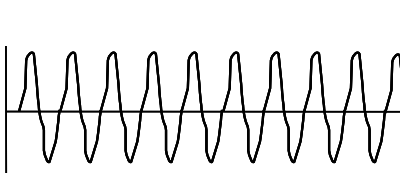
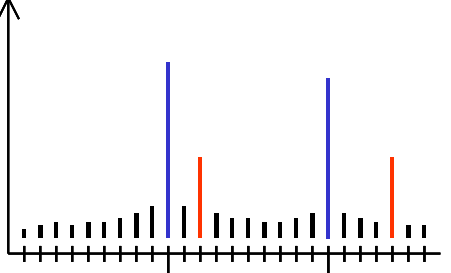


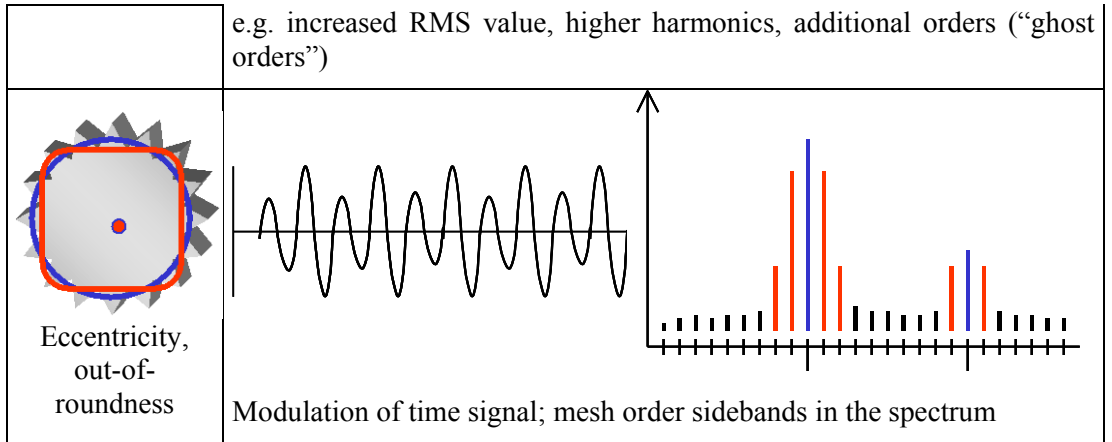
## Typical Noise Patterns

Each type of aggregate has characteristic noises and noise patterns. For this reason, a general list of the kind “If you hear this or see that in the TasAlyser then the aggregate has this or that defect” cannot be written.

This section describes only a few of the typical problems which can occur with gear wheels. It serves more as an illustration of how the results of TasAlyser’s test measurements can be interpreted.

The following table shows some common production defects in gear wheels as well as the noise signal (qualitative) and resulting order spectrum (likewise qualitative). Explanations are given on the next page.

 <p>Good Gear</p>		
 <p>Nick or similar</p>		
 <p>Bad surface</p>		



With a good gear wheel you will mainly see the meshing orders H1, H2, H3 etc. in the synchronous rotor spectrum (“synchronous channel”).

Nicks on individual teeth express themselves mainly in the time signal and are recorded by the Crest value. In the spectrum, you will possibly see a “crest” in all whole orders, but only for very bad nicks.

Surface defects such as, for example, ripples or circular pitch errors express themselves by additional spectral lines. These are known as “ghost orders”, because a gear wheel with this number of teeth does not actually exist.

Ovality and eccentricity lead to a modulation of the meshing noise and this, in turn, leads to increased sidebands near the meshing orders. To detect sidebands additional “hats” are used in the spectral limit curves (see “Spectral Values: “Hats” in Spectral Limits” on page 24).



# The TasAlyser Program

The TasAlyser program, known simply as TasAlyser or measurement program, processes sensor signals, calculates the acoustic measure values from these signals and evaluates them against limits. The TasAlyser, therefore, carries out the actual noise analysis.

Depending on the measurement project and customer requirement, the TasAlyser is individually configurable with regard to the project's analysis components and the layout of the windows. This chapter presents a "typical" measurement project with the displays, windows and control functions that most often occur.

---

## The Project Directory

The TasAlyser program opens a measurement project in the same way that you would open a Word document in the Microsoft Word program. As in Windows the TasAlyser program is usually installed in the folder `C:\Program Files (x86)\Discom5`. Without a measurement project, however, the TasAlyser is only an empty shell.

Unlike Word, a project is not, however, contained in only one file (and you cannot produce a new project as simply as a new Word document). Instead, a whole number of files belong to the project, which are all contained in a common project directory.

The project directory is usually a sub-directory of `C:\Discom\Measurement\...`, such as `C:\Discom\Measurement\MultiRot\MyProject`. The project directory contains a set of sub-folders and usually also a link to start the TasAlyser with this measurement project.

The exact contents of the project directory will be discussed later in the chapter "Further TasAlyser Functions". Individual sub-folders are discussed in the following chapters on the parameter database and on learning. For the moment it is only important to know that there exist such things as projects and that they can be very different from each other. If you do not know where the measurement project directory is stored, you can locate it using the **Project Directory** instruction in the **File** menu of the TasAlyser program.

---

<sup>5</sup> The exact installation path can be determined via the environment variable `%DiscomSoftwareRoot%`.

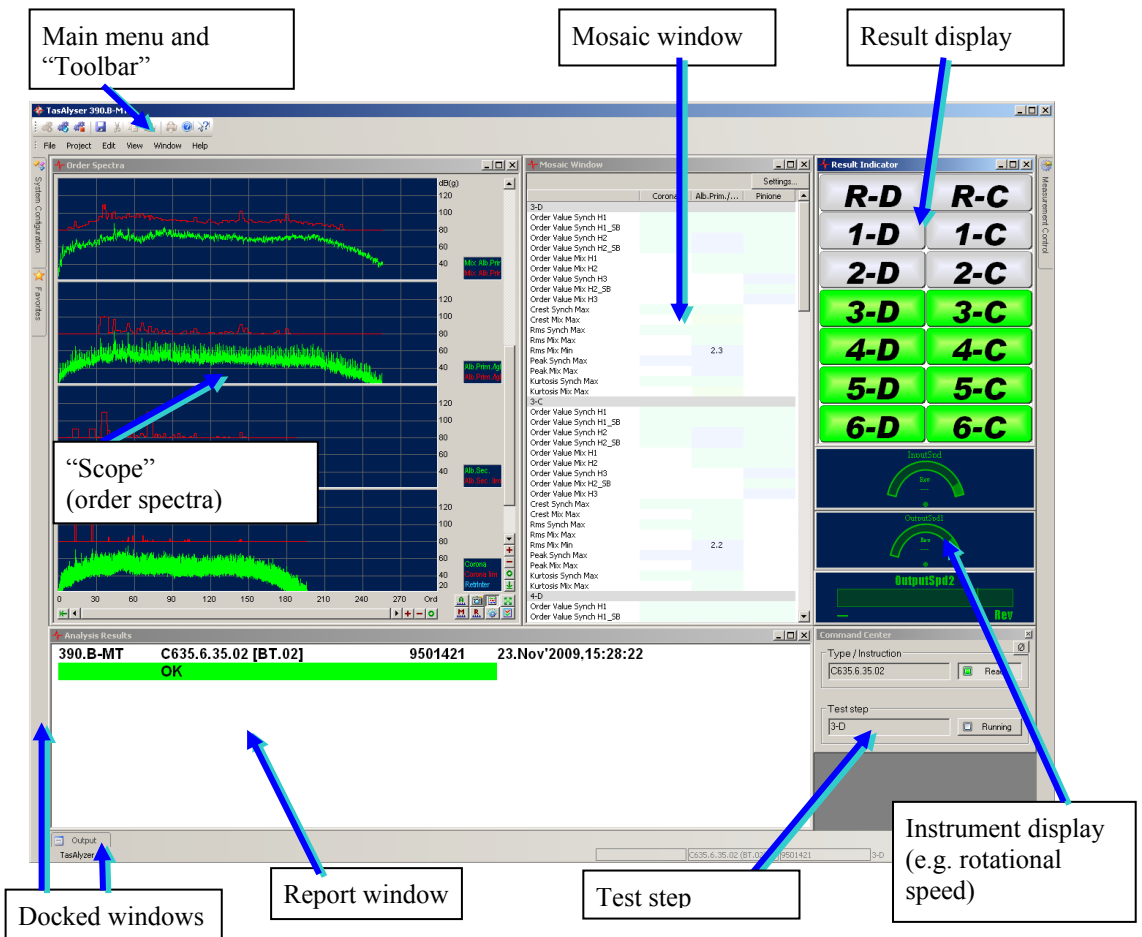


# Top View

The illustration on the following page shows a “typical” full-page screen view of the measurement program. You can see a number of different windows and displays. You can arrange, open or close these windows (and others not shown) according to preference and individual requirements. There is no fixed default regarding the screen layout.

You can even create different screen layouts, store them as window arrangement favorites and simply switch between the different views. (More information is given in the chapter (“Further TasAlyser Functions“.)

Now to the actual windows:



As is usual with Windows programs, the menu bar is found right at the top of the main window. It contains general instructions for the measurement program, as well as the toolbar with buttons to call the important menu



instructions directly. The individual menu instructions are discussed in the following sections.

Underneath the menu bar and toolbar, the figure shows a window which is very typical for the TasAlyser, a so called *Scope*. Like the graphic, the name is based on an oscilloscope. Scopes are used to display all kinds of measurement curves (such as, for instance, order spectra or tracks). Scopes can also display spectrograms. For more information on the operation of scopes refer to the section “Scopes” on page 44.

On the right in the screen shot, next to the scope, is an open *mosaic window*. This window displays the results of measured single values. Each field corresponds to a measured value, and the color indicates how close the measured value is to its limit. If a measured value exceeds its limit, the appropriate field is red. Using the button **Settings** at the above right you can configure the display, For example, you can restrict the number of measured values which are displayed.

The mosaic window appears also in another variant, the *measured value window*. Here the measured values are specified as rows of a table, from which you can directly read off the measured values, limits etc. Limit violations are also marked in red here.

The *report window* is found underneath the scope and mosaic windows, and displays the total test result (so far) as well as any errors which may have been found. For more information on the report window read the section “Operating the Windows” below.

At the top right you can see the result display, also known as the “traffic light”. It shows the evaluation results of all the test steps at a glance: Green = OK, yellow = is being tested, red = error, grey = has not yet been tested.

Underneath the “traffic light” there are three display instruments. These instruments show the values of the reference variables, i.e. rotational speed, torque etc. These instruments can display other values also, such as a sensor’s current master volume (“loudness”). You can find information on display instruments in the section “Display Instruments” on page 45.

## Test Cycle, Command Center

On the screen shot below right, below the display instruments, you can see the test step display. (The window usually headlines “command center”.) The section “The Test Cycle” on page 20 describes how a test cycle is divided into several test steps. The *command center* window displays which aggregate type is being tested and the test step in which it is being tested at that moment in time.

The *command center* window has another visual representation than shown above (see below) in which all the intended test steps are displayed as a list:

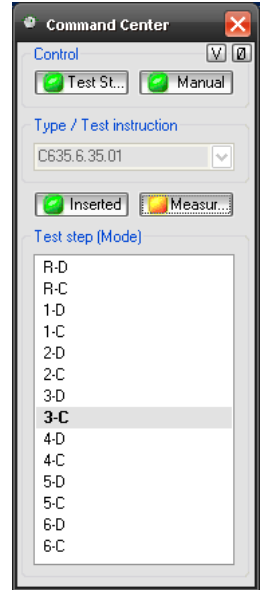


The display format can be changed using the small button with the Ø symbol at the upper right hand corner.

In its large variant, the command center window can also be used to emit commands to the measurement program (“manual operation”), hence the name “command center“.

In the menu under **Type/Test instruction**, you first select the aggregate type to be tested, and press **Insert**. Then you select a test step in the list. To start the measurement, you press **Measurement** and to terminate the measurement, you press **Measurement** again.

While the test bench control is sending commands, you can do this manually at the same time. This means that you can complete incomplete test cycles, for example, during maintenance or start-up, or supplement missing test bench commands. During normal test operations it is better to deactivate this function by turning off the **Manual** button. **Fehler! Hyperlink-Referenz ungültig.** and revert the command center to its small variant (button Ø).



## Docking Windows

In addition to the various display windows (of which the most important were specified in the previous section) there are some docking windows. These are not used to display measured values or results, but for operating the measurement program.

The docking windows **Output** and **Measurement Control** can be found on the left hand side of the bottom border of the program’s main window:



Docking windows are normally hidden except for a tab. To unhide the docking window either move the mouse to the tab and wait, or click the tab. If a docking window is unhidden, you will find control elements at the upper right corner: Via these elements you can control the behavior of the docking window. In particular, by using the “pin” (middle symbol) you can “pin down” the docking window to prevent it to be hidden automatically when you click outside of the window.







## Output

Messages and status reports appear in the **Output** docking window. The output window is divided into several sections (by appropriate tabs on the bottom window border). In **Communication** you can see, for example, a log of the control commands, which are being exchanged between test bench and measurement program.

## Measurement Sequence Control

This window contains some large buttons via which the test cycle can be controlled. These large buttons are used if the measurement program is used in mobile measurement (e.g. during a car drive) or runs on a computer with Touch screen.

## System Configuration

The tab for the **System Configuration** docking window is located at the left-hand side of the program's main window, together with the tab for the **Favorites** window.

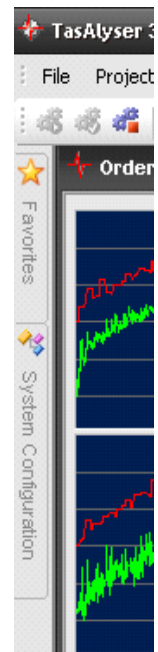
The measurement program is composed of a large number of individual software modules. Most of these modules are working in the background, and, normally, you do not have to bother about them. If, however, for some reason you do need to, then the configuration window permits access to each individual software module.

## Favorites

The most important software modules are contained in the favorites window. Here, for example, you will find the module for the order spectra scope, the report window or the wave file recorder. If you have closed the Scope window, for example, and then want to open it again, simply drop down the favorites and double-click on the appropriate entry.

You can add any module from the system configuration to the favorites. You can find further information on the organization of the favorites and the use of the system configuration in the chapter "Further TasAlyser Functions".

If you have properly closed one of the docking windows (not just hidden), and then want to re-open it, you can do this using the **View** menu of the measurement program. The docking windows are listed in the **Tool Windows** submenu.





## The Status Bar

The status bar is located on the bottom border of the program main window. On the right hand side you can see different information:

C635.6.35.03 (BT.03)	17549-34	3-C			MAN NUM
Current type (Base type, see p. 16 <b>Fehler! Textmarke nicht definiert.</b> )	Serial number	Test step	Meas. status	Addn. info	Manual operation is activated

The measurement status is indicated by a symbol: if the gear wheel has a Discom-mark then a measurement is currently active.

If **MAN** appears in the next to last field, then manual control is activated, i.e. you can control the test sequence via menu and keyboard instructions. If this field is empty, then manual control is deactivated. (We recommend deactivating manual control for normal test operations – “Activating Manual Control” on page 48).

The **NUM** display at the far right is the standard Windows display to show that the numeric keyboard is activated.

At the beginning of a test cycle, when the measurement program loads the data for the current aggregate type, a progress bar appears to the left of the current type field. From this, you can tell that the test bench has just started a new test cycle.

After a test cycle has been completed, the displays for the test step and the measurement state symbol disappear. However, type information and serial number remain so that you can see which aggregate was tested last.

---

## Operating the Windows

### Report Window

The report window shows the total test results (so far) as well as any messages about errors found.

The current test cycle is specified in the first line. The sequential order of the entries is: test bench name, aggregate type, base type in square brackets (the base type is shown only if it differs from the type itself), serial number and time stamp. This time stamp is the time when the test cycle started unless the test bench transmits otherwise.



Analysis Results			
390.B-MT	C635.6.35.03 [BT.03]	3190584	14.Sep'2009,15:58:04
not OK			
Code	Beschreibung	Wert / Grenze [MW]	Pos.
3-C			
4-C			
400	Gearmesh loud	102.29 / 80.00	[ 95.0] 45.00
500	Eccentricity	94.69 / 80.00	[ 86.6] 44.00
5-D			
5-C			
6-D			
6-C			
			Order Value Synch Alb.Prim./IglIn H1
			Order Value Synch Alb.Prim./IglIn H1_SB

The test result (so far) is given highlighted under the header.

This is followed by a list of the measured test steps and, if necessary, any error messages which have occurred. (The display of test steps in which no errors have occurred is optional - see below.)

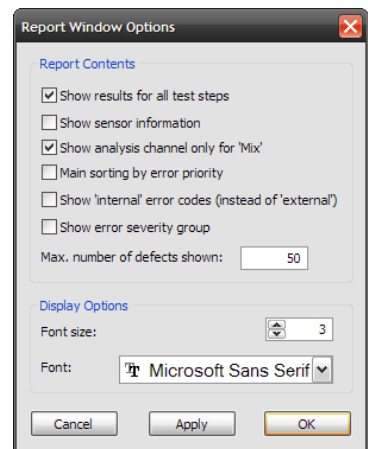
An error message contains the following: first, the error code and associated text as defined in the parameter database; second, the measured value and the associated limit value, whose violation caused the error message; third, the learned average for this measure value in square brackets. This provides additional evidence for the evaluation of the measured value's "outlier nature".

In addition, a position reference is given. The meaning of this position depends on the nature of the measure value: Positions in the order spectra are orders, positions in tracking curves are reference variables (like speed), etc. Finally, the exact characterization of the measure value is given with information on rotor, sensor etc.: the clavus of the measurement value.

By double-clicking on the window you open the options dialog:

Here you can select whether all the measured test steps are to be displayed, or only those with error messages. You can also specify whether the sensor is to be given in the error messages. (If you only use one sensor, you may not need this information.)

The error messages are normally grouped according to test step. If you activate **Main Sorting By Error Priority**, the messages are sorted according to importance (as defined in the parameter database). The test step then becomes part of the error message.



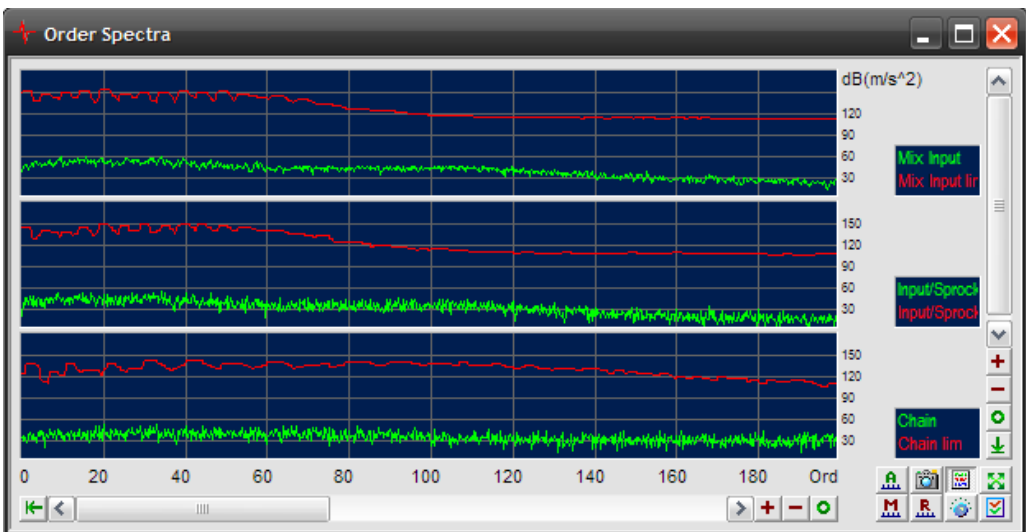


## The “traffic light”

The “traffic light window” shows panes for each test step. Each pane shows the result of the corresponding test step with its color: grey = not yet evaluated, green = ok, red = defect found. Depending on the project settings, other result colors may appear. The field of the current test step shows a yellow color.

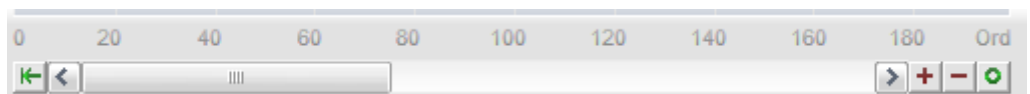
## Scopes

A scope window can display one or more curves or spectrograms. In the measurement program the scope window will mainly show order spectra and tracks. Nevertheless, it can also be found in numerous other places like the calibration control or in the adjustment of the rotational speed detector.



The curves can be distributed within the scopes on several surfaces (“Panes”). On the right of each pane, you will see a legend with the designations of the curves in that pane.

Use the scrollbar on the bottom and to the left to set the  $x$ - and/or  $y$ - axis, zoom or access other areas:




Use the actual scrollbar as usual to adjust the displayed cutout. The buttons + and – are used to zoom in and out. The  $\bigcirc$ -button scales this axis automatically so that all the curves are completely visible. The button on the far left (under the  $y$ -axis) shifts the axis so that it begins with 0 (while keeping the current zoom settings).



Since the  $y$ -axes of all the panes are coupled, there is only one  $y$ -scroll bar. If the scope displays a spectrogram, there is also a  $z$ -scroll bar on the left side.

At the bottom right of the scope window there is a group of control buttons. These buttons have the following functions:

Automatic scaling of both axes	“Still image”	Hide or unhide legend	Data monitoring tool
Store current scale settings	Retrieve stored scale settings	Open dialog of scope settings	Open dialog for curve colors

You can change the colors of the curves and their distribution on the panes in the dialog for curve colors (button bottom right). (Press the **Save** button of the tool bar in the program main window to store these changes.) 

A right-click in a pane opens the context menu for this pane, a right-click outside the pane opens the context menu with functions for the whole scope. This allows you, among other things, to export the curves of the scopes - either as diagram or as data series in Excel format. Double-click on a legend opens a window with information on the values of that particular curve.

Scopes showing continuous data (e.g. the order spectra) can be overlain with a cross-hair pointer to display values. This can be done by moving the mouse within the pane with a pressed left mouse button.

Some further functions of the scopes are discussed in the chapter “Further TasAlyser Functions”.

## Display Instruments

A display instrument window displays a control parameter, e.g., the current rotational speed. The task of this instrument is to execute this display in a graphically pleasing way.

Unlike normal windows, instrument windows do not have title bars. This saves screen space. To shift an instrument window you have to “grab” it with the left mouse button anywhere inside it.

From the instrument window you have access to two setting ranges: one for setting the instrument’s appearance, and the other for producing the value (for instance, rotational speed capture). To access the setting dialog for value production, double-click somewhere inside the instrument window with the left mouse button. You can find further details on reference variable capture in the chapter “Further TasAlyser Functions”.



Right-clicking in the instrument window, opens the dialog window for configuring the appearance:

This window does not have a title bar either. It always appears just below the instrument to which it belongs and can be moved to a preferred position simply by “grabbing” it.

Select the instruments’s graphic appearance (bars, as shown here, hand, digital display...) and the color scheme.

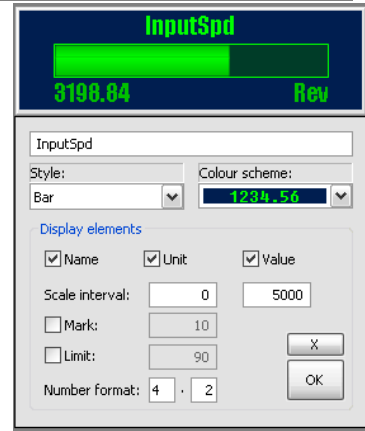
In addition to a graphic representation of the value (like hand or bar), each instrument can also display the name of the control variable, its current numerical value and the unit. These options are activated over the appropriate check boxes.

You must define a scale range for the graphic. In the graphic above, the bar ranges from 0 to 5000 (in this case Rev). For values outside the scale range, the graphic display stops at the appropriate full-scale reading.

Any form of the display instrument can also show yet another marker at any value, and an upper limit. Since the graphics do not possess a scale, markers and limits are useful for emphasizing specific values. In the above graphic, for example, neither marker nor limit is active. Finally, you can specify the format of the value’s numeric display (before and after the decimal point).

Usually, you want to show the value directly. But if you like, you can damp the display like with a mechanical device.

Use the **X** button to close the instrument window since the instrument window itself has no header with its own closing button. **OK** confirms your settings and only closes the setting dialog.





## Test Bench Connection

In a test bench environment the TasAlyser is controlled by instructions from the test bench control (see “The Measurement Computer” and “Communication with the Test Bench “ on page 11ff.). In most cases, measurement computers and test bench will be connected by a classical serial line, although a connection over network using appropriate protocols (UDP, TCP/IP) is also possible. The measurement computer can be equipped with a profibus interface card, or even communicate with the test bench using a “low level” parallel bit interface.

The advantage of instruction-based communication is that, on the one hand, it can be monitored and understood easily, and, on the other hand, it is relatively simple to extend the instruction set. With bit-based communication it must always be kept in mind which bit stands for what. Adding more instructions proves to be quite difficult, due to a lack of bits.

All kinds of test bench connections are translated within the TasAlyser into internal standard commands. This task is performed by a decoder module, which can be equipped with supplements, “plug-ins”, in order to implement additional instructions. Communication with the test bench is logged in the **Output** window, section **Communication**, (see “Docking Windows” on page 40).

The interface settings can be accessed in the system configuration (see “System Configuration” on page 41). Here you can see a tree diagram of all the modules in the measurement program. Open the section **Main Thread**



and in it the node **Control Center** (or **Command Center**). Here you will find one or more interface modules, for example, a module designated **Serial Interface**. Open the setting dialog by double-clicking on the icon in the system configuration tree.

If communication is functioning normally, you can observe the test steps during a test cycle, the aggregate type being loaded, the test steps being activated, etc., in the command center window. Note: Here in the command center you can deactivate test bench control! (In which case the button at the upper left is red.)

If you have closed the window of the command center, you can re-open it by double-clicking on the **Command Center** entry in the system tree (You have already met





this entry as super-module of the interface modules.) In addition, you can usually find the command center under the favorites.

You can read more about the command center window in "Measurement Sequence Control" on page 41.

## Manual Control

If the TAS system is built into a test bench, the test sequence is normally controlled by the test bench, which transmits instructions to the measurement program (such as, "the next aggregate type is...", "the next test step is..."). The course of this communication can be observed in the output window in the section **Communication** (see section "Docking Windows: Output" on page 41).

However, if the TasAlyser is used in the context of mobile measurement, the test sequence must be controlled manually. In addition, it can also be necessary to control the measurement program manually during test bench start-up or for maintenance purposes. The basic control instructions (see also "The Test Cycle" on page 20) are:

- Publication of the aggregate type and, at the same time, starting a new test cycle
- Inserting a test step
- Starting and terminating the measurement
- Terminating (or canceling) the test cycle.

In addition, the serial number and other supplementary information can be entered.

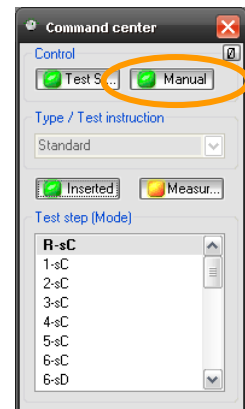
The control instructions can be given by means of menu commands, controls or shortcuts, as described in the following. All the procedures are equivalent and can be mixed (for instance, starting the test cycle over menu instructions, selecting the test step over the window "control center" and starting measurement by means of keyboard **F7**).

## Activating Manual Control

Generally, manual control must be activated first (normally it should be deactivated to prevent inadvertent incorrect operation during the automatic test operation).

Manual control can be activated in two ways:

- by activating the command **Manual Control** in the **Project** menu.







- by activating the manual button in the **Command Center** (see opposite).

If manual control is activated, then you will see the symbol **MAN** in the next to last field in the TasAlyser's status bar (at the main window's bottom border, on the right hand side, see illustration on page 42).

## Command Center

If manual control is activated, then you can control the test sequence directly via the **Command Center** window. First select the aggregate type to be tested in the selection list under **Type/Test instruction**, and press **Ready**. Then select a test step in the list. To start the measurement, press **Measurement** and to terminate the measurement, re-press this button again. Then select another test step. At the end of the test cycle deactivate the **Ready** button.

Make sure that you deactivate **Measurement**, before selecting the next test step. If you do not do this, then the measurement in the previous test step is rated as being cancelled and all the measured values are rejected. (From a different point of view, you can use this method to cancel a failed measurement, e.g. during a car drive, in order to repeat it almost immediately afterwards).

## Docking Windows for Test Sequence Control

As already described on page 41, the **Test Sequence** docking window contains large buttons for calling up the basic control instructions mentioned above. The window was designed for operation on computers using touch screen, but otherwise calls up the same functions as in the menu and keyboard instructions described below. The keys for the commands are also given on the buttons in the docking window as additional information.

## Menu Commands

The commands for controlling the test sequence are contained in the **Project** menu. Here you will find instructions for setting serial numbers and for canceling the test cycle, as well as commands which you can also perform via the **Command Center** window.

Some control instructions are hidden in order to maintain a clear menu layout. To make all instructions visible, let the cursor stand for a moment on the menu title ("**Project**") or click on the drop-down symbol at the bottom of the menu.

Some of the control instructions are also available as buttons in the Toolbar (located directly under the menu bar).



## Keyboard Operation

The following table provides an overview of the control commands using the keyboard. For some commands it is possible to use more than one key:

Key	Function
<b>F2</b>	Enter or change serial number and supplementary information.
<b>F3</b>	Cancel measurement in current test step and re-start immediately.
<b>F4</b>	Terminate measurement, insert the next test step in the list and start measurement.
<b>F5 Ctrl+I</b>	Start test sequence (a window appears in which the aggregate type can be selected)
<b>F6</b>	Select test step from the list. The test step can be entered using the keyboard or selected using the arrow key ↑/↓.
<b>F7 Space bar</b>	Measurement start /finish. (Only when a test step is inserted and only once per test step)
<b>Alt+F7</b>	Cancel measurement.
<b>F8 / Ctrl+R</b>	Terminate test sequence (regularly).
<b>F9 / Alt+Ctrl+R</b>	Cancel test sequence
<b>PgUp key↑</b>	Select previous test step in the list
<b>PgDn key↓</b>	Select next test step in the list

### Details on F3, F4, PgUp Key ↑ and PgDn Key ↓

Primarily, these keyboard commands terminate the measurement in the current test step.

**F3**, is among the keys mentioned the only one that *cancels* the current test step (because it shall be repeated). After that, the current test step is re-inserted and the measurement re-started immediately.

The other three instructions result in the *regular termination* of the measurement in the current test step. Then either the previous or the next test step in the list is inserted. Which one the previous or the next test step is, depends on the test step sequence in the list. The sequence can be read in the command center window or in the selection dialog box, which is opened

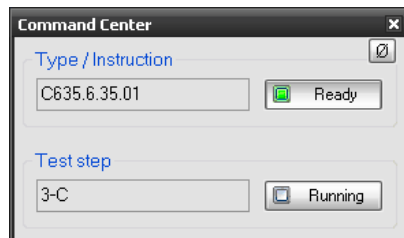


with **F6**. It is specified in the parameter database tables and can be changed if desired.

The key **F4** starts the next measurement immediately after inserting the test step. **F4** is used in the test cycle to allow quick measurement of test steps which follow one another without time loss (c.f. “Appendix A: Rotas Mobile” on page 168).

The two **PgUp/Dn** keys terminate the current test step measurement and move on to the next and/or previous test step, without starting the measurement. They are used if the measurement in the next test step should not directly follow the end of the previous test step.

Note that during keyboard operation the test sequence window (“Test Cycle, Command Center” page 39) in its large form should not have “input focus”, since this window “catches” keystrokes. To avoid this click on another window (e.g. a Scope), or change the command center window to the small variant:





## User Rights and Authorization Levels

The TasAlyser Program uses three authorization levels: normal user, “power” user and administrator.

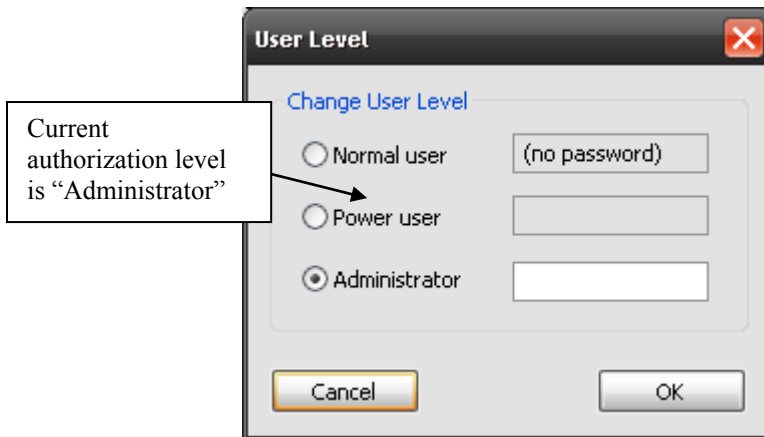
The *normal user* has only very restricted operation options. He can open and close the display windows but has no access to the system configuration or the settings dialog. Changes to window position or settings are not saved even when the **Save** button in the toolbar is pressed.

You should let the program run at the normal user level in normal test bench operations in order to avoid inadvertent changes in the settings.

The *power user* can access the system configuration and most of the setting dialogs. For example, the power user can call up the calibration function (see “Using the Calibration Function” on page 141) or change the communication settings in test bench control. However, some basic settings, such as parameterization of the A/D channels in the TAS Box, are not accessible to the power user.

The *administrator* has full access to all TasAlyser settings and functions. If you have just installed the TasAlyser then it will automatically be at the administrator level.

To change authorization levels, call up the **User Rights** command in the TasAlyser **Project** menu. The current authorization levels are marked in the dialog box:



Simply select the appropriate level. A password is required for power user and administrator but not for the normal user. If you do not wish to change the authorization levels then you do not need to enter a password (even if you remain at administrator or power user level).

The password default setting for power user and administrator is `discom`

Other passwords are entered using plaintext in the `Application.sea` file under `UserLevelPassword1` for the power user and `UserLevelPassword2` for the administrator, such as:

```
UserLevelPasswort1: e12345  
UserLevelPassword2: a54321
```

You can find the `Application.sea` file in the project directory in the **Application** folder (to be opened, e.g., with Editor Notepad).



Editor

Please contact us if you need any assistance in setting up or changing passwords.

# Parameter Administration with TasForms

The Rotas System uses a database-supported parameter administration. Without database, the Rotas System cannot work. Limit value administration, in particular, takes place in the database (c.f. section “How Limits Are Generated” on page 22). Therefore, it is important to be acquainted with the basic concepts of parameter administration.

This chapter shows how new aggregate types are created, existing ones administered, and also how to set limit values.

---

## The Database in the Overall System

### Database and User Interface

The parameter database is an Access database and consists of a single file (file ending .mdb), which contains the parameter tables. To set the parameters, you do not open the database itself (which is possible with Access) but make use of an user interface.

This chapter describes the “TasForms” user interface – which is also an Access database. TasForms only contains forms but no data (parameters); at start up it links itself to a parameter database (a “project”). After that, the project’s parameters can be edited.

### Parameter Cache

During the test, the measurement program does not access the parameter database directly, but uses data from the parameter cache. This parameter cache consists of several files in its own directory. It may be erased at any time (and, under certain circumstances, must be), since it can be restored from the database at any time.

Depending on the parameter administration’s concept the cache files can be created from the parameter database in two ways. With the common method 1 the parameter cache data are produced decentrally by the TasAnalyser on the test computer. If the test bench starts a new test, the measurement program checks the parameter database to see whether data has been changed for the requested type. If this is the case, then the parameter cache is updated for this type (which takes a short moment). If this is not the case, then the existing data is used and the measurement program is immediately ready to test.

Method 2 uses a central computer (server) for data administration. The parameter database is physically stored there and is read by an auxiliary program which creates the parameter cache data and distributes these over

the network to the test benches. The TasAlyser programs on the measurement computers never access the database directly in this case.

In any case, the TasAlyser program reads Cache-Files/ Database only at the beginning of a test run. Changes during a running test take effect not before the start of the next test run.

### Starting TasForms

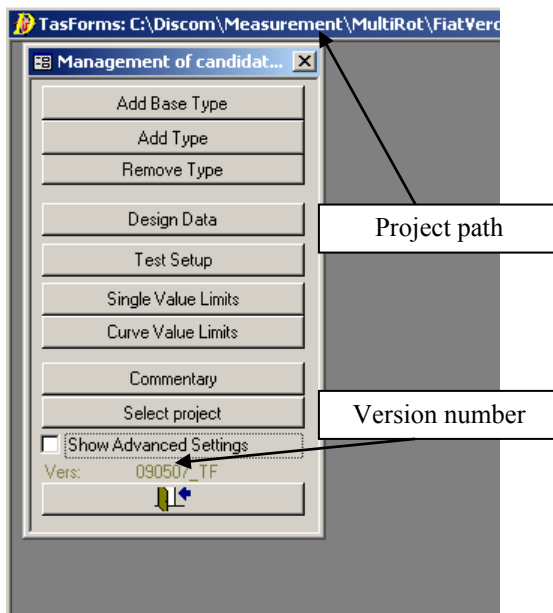
A link, symbolized by a yellow “D”, is set up on the desktop to start TasForms:



This link is either to be found on the desktop or in a folder on the desktop with the name “Rotas for Experts”. A double-click on the symbol opens the parameter administration with the start-up screen shown on the right:

In the header of the main window you can see the path locating the parameter database file which is currently being accessed. The smaller window consists almost exclusively of buttons which open further windows for data processing. Above the exit button with which you can leave the program, the version number is given. If you have questions on parameter administration, you need to know the program version. This avoids any misunderstandings, should there be newer versions with changed functionality.

With the checkbox “Show Advanced Settings” you can extend the list of selection buttons. Since these settings assume advanced knowledge, they will not be discussed in detail at this early stage.



---

## Security and Maintenance Measures

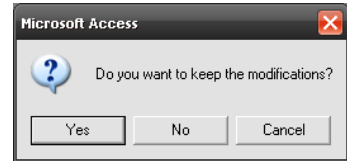
### Database Backup

If you change entries in the database and make a mistake, then many entries can become unusable. Restoring these entries is often tedious or impossible,

and you wish to have an “Undo” button in order to return to the original state. Although there is no explicit button for this, the appropriate function, however, does exist.

When the parameter administration is started, its first action is to produce a backup of the current data file. Undesired modifications, therefore, can always be taken back by restoring the “old” (saved) database file.

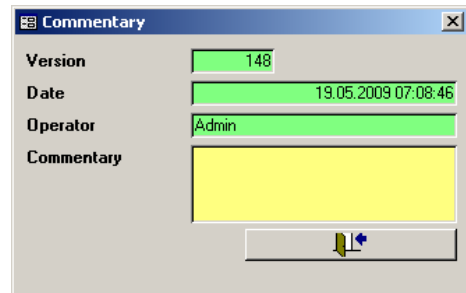
If modifications have been made, the following prompt appears when you try to close the program.



Answering with “No” ensures that the last backup (provided at parameter administration start-up) is restored and the current database file is rejected. “Yes” retains the current modified settings. The button “Cancel” cancels the operation and you return to the parameter administration. Furthermore, a “full screen” display mode is cancelled and the normal display mode activated.

Please be aware: Any modifications to the parameter administration forms take immediate effect, meaning they are immediately written into the database file. If the measurement program requests current data from the database while these are being modified then it is sometimes possible that inconsistent data are copied into the cache files.

When the above confirmation prompt is answered with “Yes” then the form on the right opens. Here you can comment on the modifications which have been made. You reach this form also by clicking the **Commentary** button. However, you will then not be able to enter a commentary, but only see the comments of the last modifications.



## Database Files and Backups

The database files are found in the project directory (use menu instruction **File – Project Directory** in the TasAlyser’s program in order to open the directory) in the folder ParamDb. A Backup directory is automatically created in this folder, in which older versions of the database are stored. For each version number, there may be a database file (as shown in the above form).

You can restore an older version simply by taking the appropriate file out of the backup folder and erasing the version number in front of the file name.



(You must, of course, first rename, erase or move the current database.) If you replace the database file manually, you should erase the contents of the cache file, which can be found in the project directory in `Locals\CacheData`.

## Database Defragmentation

If you have carried out many modifications in the database (in particular, if many entries have been erased), it is recommended that you defragment the database. When you erase data, the database file fragments. This can be explained in practical terms: Let's assume that the database file contains data records 1 to 5. If you erase data records 3 and 4, data record 5 does not automatically move to the end of data record 2. The space formerly occupied by data records 3 and 4 remains vacant and the database file is not reduced in size because it contains these unused areas. Access can decide whether or not these unused areas are used for other purposes.

A fragmented database file occupies more space on the hard disk than necessary and data access is slower. Therefore, it is recommended to perform defragmentation after you deleted many data records. The appropriate button is found in the **Advanced Settings** section on the start form.

If you want to send the database file by email, e.g., to Discom for advice, then you should also first defragment the database. Then you can compress the database file (e.g. with "7Zip" which is already installed on each measurement computer) and send the result.

---

## Creating and Erasing a Type

### A New Type or a New Base Type?

A frequent task in connection with the parameterization of the noise analysis system is the creation of a new type. As already discussed in section "Type and Base Type" on page 16, small modifications of the gearbox housing can cause the test bench to transmit a new type identification. Completely new gearboxes for a new vehicle platform are also common in reality.

The reason for a new type is not important: the first question to ask is whether or not a new base type is required for the new type. From measurement system's point of view, a new base type is necessary if, among the existing base types, none uses exactly the same order sources with the same base orders (numbers of teeth). If the new type is identical to an existing type concerning number of teeth, then, usually, only a new type name for this base type is needed. In exceptional cases, it may be that you want to create a new base type, although the new type has the same number

of teeth as an existing base type. This will be the case when you expect the new type to have major acoustic differences to the existing base type.

It is also possible to convert a type, which was only a name up to now, into an own base type later on.

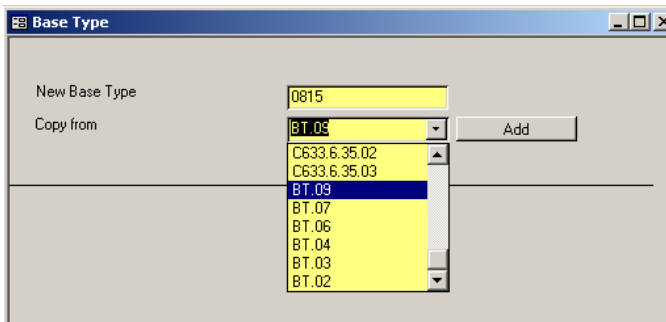
### Names

The names of types, base types (and also of all other designatable database objects) can contain letters and numbers as well as hyphens, underscores and periods. Spaces and colons are not allowed; upper and lower cases are differentiated.

## Creating a New Base Type

Let's first assume that the new type differs in the number of teeth from all existing base types. This means that we need to create a new base type. The appropriate function is accessed using the upper button in the start form: **Add New Base Type**.

This opens the form shown below:



In the input field “New Base Type” you can enter a name for the base type. In this example, the base types are all designated with names like BT.xx. The new

base type is now to receive the name 0815.

Since it is not a good idea to start with a completely empty data record for the new base type, you are forced to select an existing base type as copy source. Its data will then be taken over for the new base type. You should select a base type which is as similar as possible to the new base type for this process. The more similar the two types are, the less the new base type needs to be adapted afterwards.

When you have chosen an existing base type and selected it in the list, then click on the **Add** button. The new base type is created and the existing base type's data is copied. The program sends a message, **Done**, when the copy action has been finished. After the message has been confirmed with **OK**, the message window and the input form for the new base type will close.

Since a base type alone is not enough to make the data accessible for the measurement program, a type name is also created for the base type at the

same time (i.e., to base type 0815, a type 0815 also exists immediately, which is assigned to that base type). Since it is possible that the name of the base type already exists as type name for another base type, this action can fail. In this case, the base type has been created but is not accessible over a type name at that time. In particular, the measurement program cannot access this base type. The problem is solved when a new type is created and assigned to this base type.

In this context there are two circumstances under which a new base type cannot be created. Either the program protests when the **Add** button is pressed without previously selecting an existing base type in the list; or when the new base type already exists. In each case, an error message is given and it is clear what needs to be done. There is no message if no new base type is entered. However, in this case, the program does nothing.

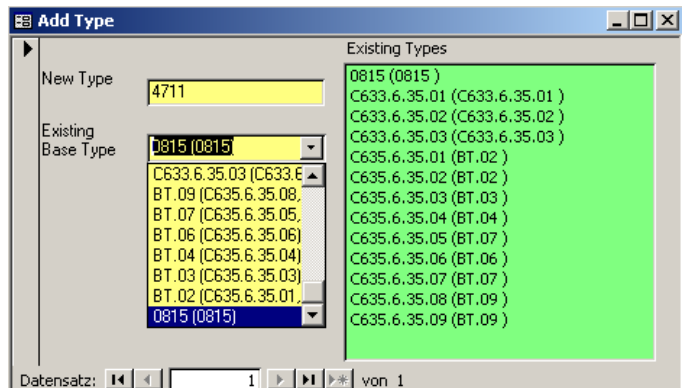
Please note that the form shown above is not only used by the program to create a new base type, but also when other new objects need to be added. Behavior and operation are then similar to the above.

If you create other objects, it can be useful to change the option “Copy all connected data” to “Copy only basic information”. If you change this option, only the really basic informations are copied from the copy source, but not, for example, connected measurement values. By default, the option “Copy all” is selected. Then *all* data connected to the copy source are also copied.

## Creating a New Type

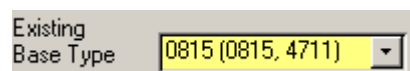
If an existing base type can be used for the new type, then you only need to assign the new type to this base type. The appropriate function can be reached using the button **Add New Type** in the parameter administration start form. Clicking this button opens the following form:

The input field and the selection field on the left hand side resemble the form for creating a new base type. In addition, there is a list on the right hand side which shows the existing types with assigned base type in brackets.



Existing Types
0815 (0815)
C633.6.35.01 (C633.6.35.01)
C633.6.35.02 (C633.6.35.02)
C633.6.35.03 (C633.6.35.03)
C635.6.35.01 (BT.02)
C635.6.35.02 (BT.02)
C635.6.35.03 (BT.03)
C635.6.35.04 (BT.04)
C635.6.35.05 (BT.07)
C635.6.35.06 (BT.06)
C635.6.35.07 (BT.07)
C635.6.35.08 (BT.09)
C635.6.35.09 (BT.09)

The procedure for creating a new type is very simple now: Enter the new type in



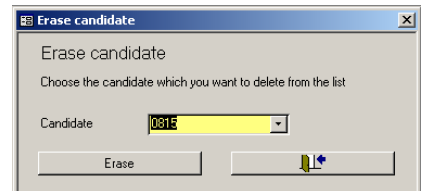
the input field, select the base type from the list (the types already assigned are listed in brackets after the base type), and press the **Add Type** button. If the action is successful, the new type appears in the list of the existing types. In addition the new type is supplemented in the selection list in the brackets behind the base type.

This action can fail if the name for the **New Type** already exists. The program will then point this out with the appropriate error message.

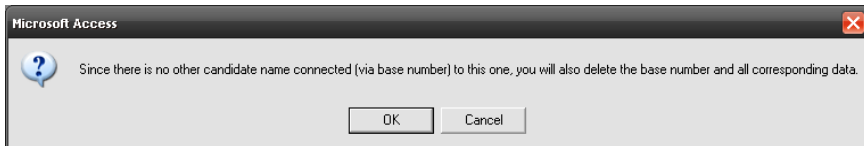
## Erasing Types and Base Types

Occasionally, aggregates are designed for production but then not built as planned. If types have already been created for these aggregates you want to get rid of them. The appropriate function is reached using the **Erase** button in the parameter administration start form. Pressing this button, opens the form:

With the help of the selection list, select a type (not a base type!). Clicking on **Erase** removes this type from the parameter database and it then becomes unknown, especially in the measurement program.



If the type to be erased is the last and only type which is assigned to a specific base type, then the following message appears:

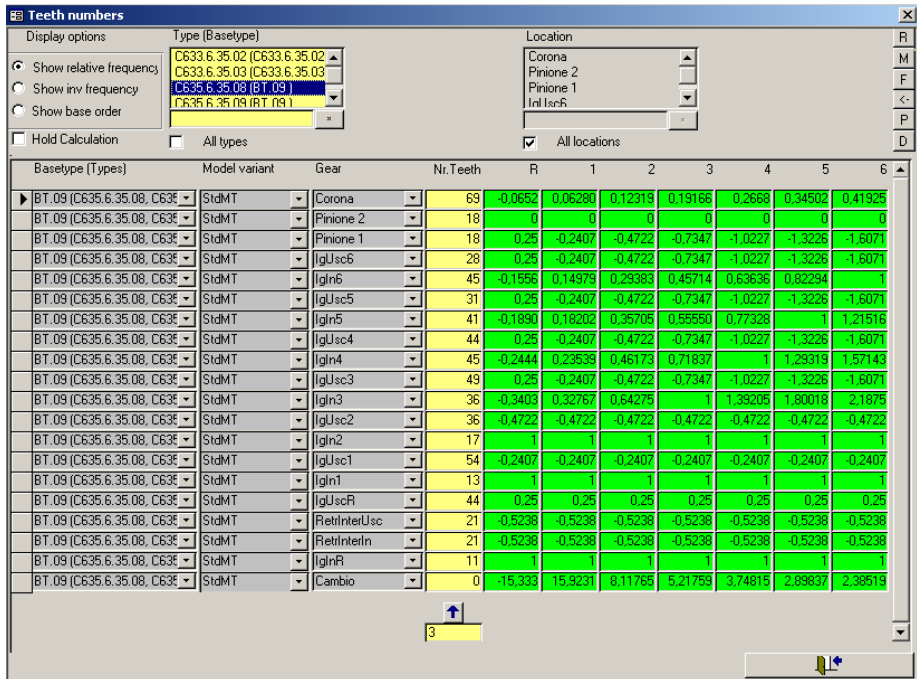


If this prompt is confirmed with **OK** then the type *and* its base type will be erased. Any settings which have been made for this base type will then be lost (limits, number of teeth, etc.) If the prompt is confirmed with **Cancel** then nothing happens. Both the type and the base type remain in the database.

This behavior makes sense since usually you do not want parameters in the database which cannot be accessed by the measurement program. However, as described above, this can occur when creating a base type. In order to get rid of such “invisible” base types, you first have to create a type name (see Creating a New Type), after which you can erase it with the “Erase Type” form.

## Changing the Number of Teeth

One of the first things that needs to be done after creating a new base type is to modify the number of teeth, since the new base type usually has - at least some - other teeth numbers. The appropriate function is accessed by clicking on the **Construction Data** button in the parameter administration start form. This opens a variant of the following form:



Basetype [Types]	Model variant	Gear	Ni, Teeth	R	1	2	3	4	5	6
BT.09 (C635.6.35.08, C635)	StdMT	Corona	69	-0.0652	0.06280	0.12319	0.19166	0.2668	0.34502	0.41925
BT.09 (C635.6.35.08, C635)	StdMT	Pinione 2	18	0	0	0	0	0	0	0
BT.09 (C635.6.35.08, C635)	StdMT	Pinione 1	18	0.25	-0.2407	-0.4722	-0.7347	-1.0227	-1.3226	-1.6071
BT.09 (C635.6.35.08, C635)	StdMT	IglJsc6	28	0.25	-0.2407	-0.4722	-0.7347	-1.0227	-1.3226	-1.6071
BT.09 (C635.6.35.08, C635)	StdMT	IglIn6	45	-0.1956	0.14979	0.29393	0.45714	0.63636	0.82294	1
BT.09 (C635.6.35.08, C635)	StdMT	IglJsc5	31	0.25	-0.2407	-0.4722	-0.7347	-1.0227	-1.3226	-1.6071
BT.09 (C635.6.35.08, C635)	StdMT	IglIn5	41	-0.1890	0.18202	0.35705	0.55550	0.77326	1	1.21516
BT.09 (C635.6.35.08, C635)	StdMT	IglJsc4	44	0.25	-0.2407	-0.4722	-0.7347	-1.0227	-1.3226	-1.6071
BT.09 (C635.6.35.08, C635)	StdMT	IglIn4	45	-0.2444	0.23539	0.46173	0.71837	1	1.29319	1.57143
BT.09 (C635.6.35.08, C635)	StdMT	IglJsc3	49	0.25	-0.2407	-0.4722	-0.7347	-1.0227	-1.3226	-1.6071
BT.09 (C635.6.35.08, C635)	StdMT	IglIn3	36	-0.3403	0.32767	0.64275	1	1.39205	1.80018	2.1975
BT.09 (C635.6.35.08, C635)	StdMT	IglJsc2	36	-0.4722	-0.4722	-0.4722	-0.4722	-0.4722	-0.4722	-0.4722
BT.09 (C635.6.35.08, C635)	StdMT	IglIn2	17	1	1	1	1	1	1	1
BT.09 (C635.6.35.08, C635)	StdMT	IglJsc1	54	-0.2407	-0.2407	-0.2407	-0.2407	-0.2407	-0.2407	-0.2407
BT.09 (C635.6.35.08, C635)	StdMT	IglIn1	13	1	1	1	1	1	1	1
BT.09 (C635.6.35.08, C635)	StdMT	IglJscR	44	0.25	0.25	0.25	0.25	0.25	0.25	0.25
BT.09 (C635.6.35.08, C635)	StdMT	RetInterUsc	21	-0.5238	-0.5238	-0.5238	-0.5238	-0.5238	-0.5238	-0.5238
BT.09 (C635.6.35.08, C635)	StdMT	RetInterIn	21	-0.5238	-0.5238	-0.5238	-0.5238	-0.5238	-0.5238	-0.5238
BT.09 (C635.6.35.08, C635)	StdMT	IglInR	11	1	1	1	1	1	1	1
BT.09 (C635.6.35.08, C635)	StdMT	Cambio	0	-15.333	15.9231	8.11765	5.21759	3.74815	2.89837	2.38519

The order sources specified in the “Gear” column depend on the respective project, as do the columns “R”, “1”, “2”, etc. for the physical gears of a gearbox.

In the “Type (Basetype)” column you must select one or more base types in order to display the appropriate construction data.

In the form above only the column with the number of teeth is open for input. The relative frequencies of the gear wheels for each gear are displayed in the green fields for information. The frequencies are given with reference to a “calculation” speed of 1. This calculation speed of 1 often corresponds with the input or output speed of the gearbox. With the gearbox in the above screen shot the relative frequencies are given with reference to the gearbox input speed.

Beside the relative frequencies, the overall ratio of the transmission for each gear is also calculated in most cases. In the screen shot above, you can not see such an entry. The overall ratio is a relatively useful piece of information

because it is easy at the test bench, to find out the transmission ratio by measurement, for example. Usually, it is displayed in the construction lists as well. If the gear ratio calculated by the parameter administration does not match with the gear ratio at the test bench or in the construction lists, then the settings should be checked. Either the wrong type has been selected or the number of teeth has been entered incorrectly (e.g. driving wheel and driven wheel have been mixed up). Experience shows that the displayed gear ratios should exactly match the construction defaults.

If you want to check partial gear ratios, you can switch the view by selecting **Show Inv Frequency** at the upper left. This displays the gearbox gear ratio from the reference speed up to the respective part.

Furthermore, you can select **Show Base Order**. The display then shows the product of the number of teeth and the relative frequency of the different parts. This value is interesting for order analysis. If you have a spectrum which is scanned with relative factor 1 with regard to the reference rotational speed, then the displayed value is the position in the spectrum at which the noise participation of the corresponding wheel should appear. (More details on positions in the spectrum will be given later.)

Finally, you can change the calculation reference (the “1”) with the combo box “base”. If you select something in this box, all views explained above temporarily change their calculation base to the selected base. (The selected base then is the “1” in the system for calculation.) This is useful if, for example, you want to display the base orders with reference to output speed whereas the default “1” references them to input speed. Moreover, this function can help you when you suspect base orders to show up in synch spectra where you do not expect to see them. (Change the base to reference the speed of the spectrum where you see such unexpected orders.)

After this explanation of the different viewing options, we return to the task of modifying the number of teeth. As soon as the number of teeth has been modified and the appropriate input field has been exited, the parameter administration re-calculates the relative frequencies of the wheels. Depending upon complexity of the gearbox and/or performance of the computer this can take a moment. If many numbers of teeth need to be modified (possibly for different types as well) and the immediate calculation drastically delays the input process, then it is possible to deactivate the frequency calculation temporarily by activating the check box **Suppress Calculation**. Postponed, however, is not abandoned. Since the frequencies *must* be calculated by the program, this calculation is carried out as soon as the check box is deactivated again or the form is closed.

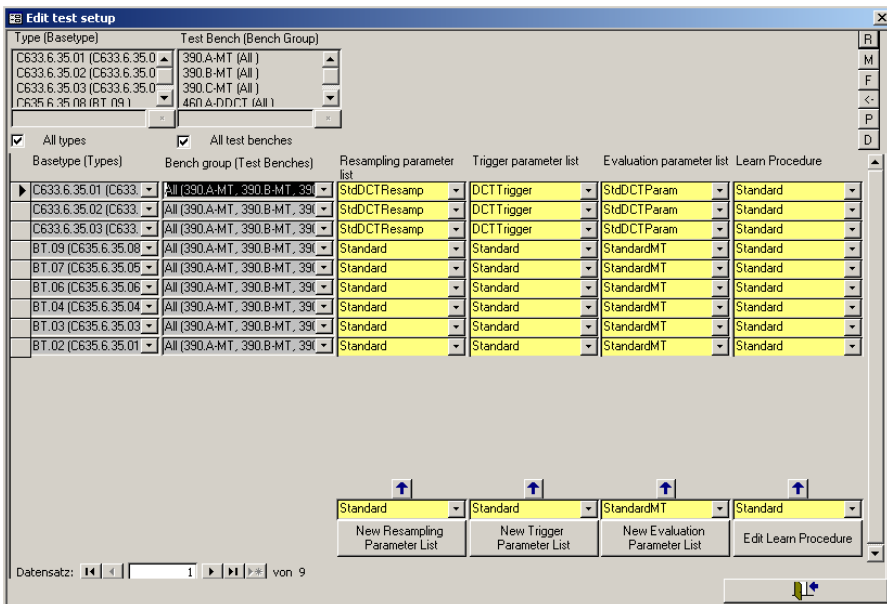
# General Form Functions

## Data Organization in Lists

Whoever has the job of data maintenance knows that it pays off to have as few data records as possible to maintain. This means, in general: as many data records as possible with far overlapping validity and as few highly specialized classified data records as possible. We have already met one approach to attain this goal by aggregating types to base types and test benches to test bench groups. If you look at this procedure from a database point of view, this means that you introduce an intermediate key “Base type” and “test bench group” which the keys “Type” and “Test Bench” can then be assigned to.

The same idea is behind the three important lists used in parameterization: the resampling parameter list, the evaluation parameter list and the trigger parameter list. The idea here is to apply as many of the same parameters as possible to all test benches for all types. Therefore, you create a parameter list and assign it to the individual types per test bench as necessary.

You reach the form for these settings, by clicking the **Test Setup** button in the parameter database. This opens the following form:



Basetype (Types)	Bench group (Test Benches)	Resampling parameter list	Trigger parameter list	Evaluation parameter list	Learn Procedure
C633.6.35.01 (C633...)	All (390.A-MT, 390.B-MT, 390...)	StdDCTResamp	DCTTrigger	StdDCTParam	Standard
C633.6.35.02 (C633...)	All (390.A-MT, 390.B-MT, 390...)	StdDCTResamp	DCTTrigger	StdDCTParam	Standard
C633.6.35.03 (C633...)	All (390.A-MT, 390.B-MT, 390...)	StdDCTResamp	DCTTrigger	StdDCTParam	Standard
BT.09 (C635.6.35.08)	All (390.A-MT, 390.B-MT, 390...)	Standard	Standard	StandardMT	Standard
BT.07 (C635.6.35.05)	All (390.A-MT, 390.B-MT, 390...)	Standard	Standard	StandardMT	Standard
BT.06 (C635.6.35.06)	All (390.A-MT, 390.B-MT, 390...)	Standard	Standard	StandardMT	Standard
BT.04 (C635.6.35.04)	All (390.A-MT, 390.B-MT, 390...)	Standard	Standard	StandardMT	Standard
BT.03 (C635.6.35.03)	All (390.A-MT, 390.B-MT, 390...)	Standard	Standard	StandardMT	Standard
BT.02 (C635.6.35.01)	All (390.A-MT, 390.B-MT, 390...)	Standard	Standard	StandardMT	Standard

Buttons at the bottom: New Resampling Parameter List, New Trigger Parameter List, New Evaluation Parameter List, Edit Learn Procedure

Many forms in the parameter administration look similar to the one given above but display different data. You read in the following chapter how to operate the controls of these forms.

## Control Area and Data Area

All the forms which are developed according to above schema, are split into two areas: at the top there is the *control area*, where you can influence the display in the *data area*. On the right of the control area is a row of six buttons, whose function will be described later. To the left of these are the key selection fields with which you can make a selection from the different setting options. The corresponding key fields in the data area have a grey background and cannot be edited. Fields in the data area which have a yellow background contain data which can be changed. The size of the form is variable, i.e. the fields in the data area, in particular, adjust their size according to the form size so that the space available is best used.

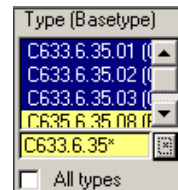
In case, you have only one entry selected in the key selection fields, the corresponding key column is completely hidden to gain additional space.

## Base Types and Test Bench Groups

As already described in detail, only one data record exists for all test benches in a test bench group, and/or for all types belonging to a base type. Since selection is, however, usually made according to type, and/or test bench, the key selection field lists in the control area contain other entries than the corresponding fields in the data area. Whereas in the key selection fields type and test bench can be selected (base type and/or test bench group are given in brackets), in the data area the base type and/or the test bench group are given, as well as all assigned types and/or test benches (in brackets). When a specific type is selected, then the other types for which the data record is also valid can always be seen in the data area.

## Operating the Key Selections Fields

With the help of the key selection fields, you can make a selection in a number of ways. The effect of the check box **All Types** is clear: a check mark here deactivates all the other possibilities of the corresponding key selection field and the data area shows all data records without restriction for that key field.



If only individual entries are required, then no checkmark is set at “all types”. Subsequently, one or more entries in the list can be marked (several at one time by pressing the Ctrl or Shift key, as usual in Windows) or with the help of the input field and the “\*” button. In the above example the asterisk button was used to mark all the entries in the list which fitted the format “C633.6.35\*”. To specify the format, the usual substitute symbols “\*” and “?” can be used. (For MS Access Profi: The asterisk button calls up the “Like” function (SQL).)



Valid for all setting buttons in selection lists: If the expected result does not occur with the first click then press the button again. Under certain circumstances, which are not clear, MS-Access needs a second trigger.

## Changing the Entries of a Whole Column

Often one would like to change all the entries of a column at one go. This function is provided by the field on the right, whose function is quite simple: You enter something, click the arrow button and the corresponding column of the current selection is filled with this value.



If the column contains a numerical value, an auxiliary function provides the option of changing values *relatively*. That means that the column is not filled with a fixed value, but calculated with the existing value. If you want to do this, you enter the arithmetic operation which you want to execute in the input field. The following options are available: +X (add value X), -X (subtract value X), \*X (multiply value by X). If you want to subtract a value using “-X” then the program will query to specify more exactly which is the action to be carried out because the minus sign can mean either the arithmetic operator or a prefix operator.

## Sorting the View

It is not always sufficient to restrict the display to just specific data. Sometimes you also want to see sorted data. In order to do this, you can use an MS Access function: Click with the right mouse button in any field of the column to be sorted. Beside other available Access functions, entries can be sorted in ascending or descending order. This function is available for all columns, whether key columns or data columns.

## Copying, Printing, Comparing

There are six buttons on the right hand side of the control area, which have been mentioned briefly above. With their assistance some powerful functions can be initiated, among which are copying, printing and comparing data.

The button **P** (for print) is used to print the current selection (in landscape format). Otherwise, the lettering of the buttons is analogous to the keys of a pocket calculator, i.e. the functions of the buttons **M** and **R** are obvious: With **M** a selection is memorized, and with **R** restored (press twice, if necessary).

With many forms a further form is opened when the **M** button is pressed. This form can also be opened separately using the **F** button (field selection). Here you can select different column fields e.g. for copying. Only the data of the selected columns are then copied.

With the <- button the copying action is finally performed as follows: The data corresponding to the memorized (using **M**) selection are read and inserted into the current selection (if possible).

The comparison function, which is initiated with the **D** button (difference) operates in a similar way. It compares the memorized data with the data of the current selection and shows the differences. With large data sets and low computing power this is to be used with caution, since the query which needs to be executed is quite complicated.

To show that the data area does not represent the data of the current selection, the control area changes its background color to brown. In this mode all the differences found between the selections are displayed.

Pressing the **D** button again, changes the color to purple. The lettering on the **M** and **D** buttons also changes to purple, in order to show that only the data records are now indicated, which are present in both selections and have different entries.

A further click on **D** displays background and **M** button in pink. In this mode the data area shows all data records which are in the memorized selection, but not in the current selection.

Finally, the reverse situation (all data records, which are in the current selection, but not in the memorized) is obtained with another click on **D**. Here the color code is a turquoise background and **D** lettering. Another click on **D**? No problem, we start again with brown...

---

## Test Setup

We have already had a glance at the **Edit Test Setup** form reached over the **Test Setup** button in the parameter administration start form. Here you can see that the test parameters for a base type are not set directly but that you select a separate list for each of the four different areas.

### Four Lists of Test Parameters

The **resampling parameters list** specifies in particular, which *locations* or *rotors* (see above) are to be resampled with which *sensors*. For further processing, the system receives data blocks (esp. Spectra) only for those location/sensor combinations which are parameterized in this list. The exact content of the data blocks (length, number of revolutions per block, etc.) is also specified here. Changes to this list are very rarely necessary, but should it be necessary then we recommend that you consult Discom. If several gearbox families are parameterized in the database, then there is a separate list of resampling parameters for each family.

The **trigger parameters list** specifies, in particular, with what degree of accuracy specific measurements are accomplished, often also within what area (see the explanation above on the term “command variable”). Since the measurement program often starts and stops the main measurements by itself over speed, these rotational speeds are also found in this list. Several lists are necessary here if different test benches or different types are to be tested in different rotational speed ranges.

The **evaluation parameters list** specifies which measure values (see above for explanation) are to be measured. How entries are modified will be discussed later. If several gearbox families are parameterized in the database, then each family has its own evaluation parameter list.

Finally, the **learning procedure list** specifies the general frame conditions under which the measure values are learned - see the section “How Limits Are Learned” on page 25.

Remark: Due to further development of the software and because of project specific reasons it can be that the evaluation parameters list or the learning procedure list is missing in the test setup form. If the evaluation parameters list is missing, the corresponding settings can be made in the forms for single value limits and for curve limits. If the learning procedure list is missing, this is being integrated into the learn parameters.

Finally, the complete list of learn parameters can be integrated into the forms for single value limits and for curve limits. This can be helpful when test bench and test bench group are identical.

---

## Setting Limits

As already discussed in the section “How Limits Are Generated“ on page 22, limit values are generated by a combination of learned data and fixed defaults. In the parameter database, therefore, there is no direct setting of limits but only the rules for generating them.

### Limits for Single Values

The limits for single values can be set using the following form, opened using the **Limit Single Value** button in the parameter administration start form. This form and the form for parameterizing the limit curves (see below) are the ones which have the most key selection fields in the control area.

Limit value settings											
Type (Basetype)	Test Bench (Bench)	Test state	Instrument	Channel	Signal	Location	Measurements				
C633.6.35.01	390.A-MT (All)	R-D	Order Value	Synch	VS	Corona	Abweichung				
C633.6.35.02	390.B-MT (All)	R-C	Crest	Mix	InputSpd	RetInterUsc	Differenz				
C633.6.35.03	390.C-MT (All)	1-D	Rms	FixFs	OutputSpd1	RetInterIn	General				
C635.6.35.08	460.A-DDCT (A)	1-C	Peak	Mix R	OutputSpd2	Inl. Isc	H1				

Basetype (Types)	Bench group (Test Benches)	Test state	Instrument	Channel	Signal	Location	Parameter	Eval on/off	Min	Max
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Peak	Mix	VS	Alb.Prir	Max	Off	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Rms	Synch	VS	Corona	Max	On	20	20
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Rms	Synch	VS	Alb.Prir	Max	On	20	20
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Rms	Mix	VS	Alb.Prir	Max	On	20	20
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Rms	Mix	VS	Alb.Prir	Min	On	0,5	0,5
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Peak	Synch	VS	Corona	Max	Off	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-C	Order v	Synch	VS	IgUsc	H1	On	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Peak	Synch	VS	Alb.Ser	Max	Off	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Crest	Synch	VS	Alb.Prir	Max	On	15	15
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Ratio T	Mix	InputSp1	Cambic	Abweic	On	0,5	0,5
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Diff Te	Mix	RearSp	Cambic	Differen	On	10	10
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-C	Order v	Synch	VS	Corona	H1	On	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-C	Order v	Synch	VS	Corona	H1_SB	On	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-C	Order v	Synch	VS	Corona	H2	On	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	6-D	Order v	Synch	VS	Corona	H1	On	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Peak	Synch	VS	Alb.Prir	Max	Off	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Order v	Mix	VS	Alb.Prir	H1	On	110	120
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Order v	Synch	VS	Alb.Prir	H1_SB	On	105	105
C633.6.35.01	All (390.A-MT, 390.B-MT)	5-D	Order v	Synch	VS	Alb.Prir	H2	Off	110	120

Whether or not evaluation should be carried out or what limits should be valid for learning (see above) is parameterized here as data (yellow fields). If the upper and lower limits are set as equal then learning is, to all intents and purposes, switched off.

### Limit Curves

Limit Curve Settings											
Type (Basetype)	Test Bench (Bench)	Test state	Instrument	Channel	Signal	Location	Measurements				
C633.6.35.01	390.A-MT (All)	R-D	Order Track	Synch	VS	Pinione	H1_M	On	StdMini	Pin-4C	
C633.6.35.02	390.B-MT (All)	R-C	Crest Track	Mix	VS	Alb.Prir	H2_M	On	StdMini	StdMas	
C633.6.35.03	390.C-MT (All)	1-D	RMS Track	Mix	VS	Alb.Prir	H2_M	On	StdMini	StdMas	
C635.6.35.08	460.A-DDCT (A)	1-C	Peak Track	Mix R	VS	Alb.Prir	H2_M	On	StdMini	StdMas	

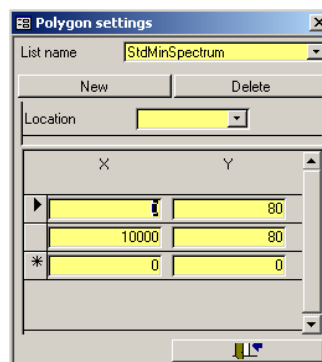
Basetype (Types)	Bench group (Test Benches)	Test state	Instrument	Channel	Signal	Location	Parameter	Eval on/off	Min	Max
C633.6.35.01	All (390.A-MT, 390...)	4-C	Order T	Mix	VS	Pinione	H1_M	On	StdMini	Pin-4C
C633.6.35.01	All (390.A-MT, 390...)	6-D	Order T	Mix	VS	Alb.Prir	H2_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	6-D	Order T	Mix	VS	Alb.Prir	H2_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	6-C	Order T	Mix	VS	Alb.Prir	H2_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	6-C	Order T	Mix	VS	Pinione	H2_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	6-C	Order T	Mix	VS	Pinione	H1_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	6-C	Order T	Mix	VS	Pinione	H2_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	6-D	Order T	Mix	VS	Alb.Prir	H1_M	On	Ing-6C	Ing-6C
C633.6.35.01	All (390.A-MT, 390...)	6-D	Order T	Mix	VS	Pinione	H1_M	On	Pin-6D	Pin-6D
C633.6.35.01	All (390.A-MT, 390...)	6-D	Order T	Mix	VS	Alb.Prir	H1_M	On	Ing-6D	Ing-6D
C633.6.35.01	All (390.A-MT, 390...)	5-C	Order T	Mix	VS	Pinione	H1_M	On	Pin-5C	Pin-5C
C633.6.35.01	All (390.A-MT, 390...)	5-C	Order T	Mix	VS	Alb.Prir	H1_M	On	Ing-5C	Ing-5C
C633.6.35.01	All (390.A-MT, 390...)	R-D	Order T	Mix	VS	Alb.Prir	H1_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	5-D	Order T	Mix	VS	Alb.Prir	H1_M	On	Ing-5D	Ing-5D
C633.6.35.01	All (390.A-MT, 390...)	5-D	Order T	Mix	VS	Pinione	H2_M	On	StdMini	StdMas
C633.6.35.01	All (390.A-MT, 390...)	4-C	Order T	Mix	VS	Alb.Prir	H1_M	On	Ing-4C	Ing-4C
C633.6.35.01	All (390.A-MT, 390...)	4-D	Order T	Mix	VS	Alb.Prir	H1_M	On	Ing-4D	Ing-4D
C633.6.35.01	All (390.A-MT, 390...)	3-C	Order T	Mix	VS	Alb.Prir	H1_M	On	Ing-3C	Ing-3C
C633.6.35.01	All (390.A-MT, 390...)	3-D	Order T	Mix	VS	Pinione	H1_M	On	StdMini	StdMas

The form for limit curves looks almost exactly like the one for single value limits. The only difference is that no individual values for learning delimitation can be entered here, only a polygonal traverse. In order to modify or create polygons, you must first select the instrument for which the polygon is valid in the appropriate key selection field (because of differing units in the  $x$ -axis of different instruments). This activates the **Polygons** button, located in the lower left corner of the form. Clicking on the button opens the polygon administration form. Similar to the lists mentioned above, the polygons are also parameters, which are independent of type and bench. They only have meaning for a test when they are used in the limit curves form.

## Defining Polygons

It is recommended to use the Talimer tool (described in the chapter starting on page 94) to edit polygons because Talimer can better visualize the polygons than TasForms can do. Nevertheless, TasForms allows you to get the same effect as Talimer (you only need a bit of imagination).

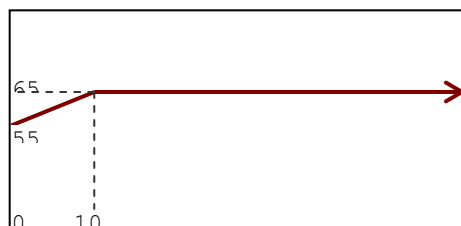
The following form shows, as an example, the settings for the polygon “StdMinSpectrum“. This is defined for the spectrum evaluation instrument.



X	Y
10000	80
0	0

The settings are to be read as follows. In each case, a line with **X** and **Y** values belong together. The sequence depends on the **X**-values (the smallest **X**-value is always at the top). The polygon is created in the measurement program by linearly connecting the registered bases according to this sequence. In the above example a horizontal line is defined as a polygon, which has the value 65 between the **X**-values 0 and 10000 (inclusively).

To clarify what we mean with the linear connection, we modify the above polygon as follows (X/Y values are noted in pairs): (0/55), (10/65), (10000/65). This polygon begins with  $X = 0$  and  $Y = 55$ , rises at  $X = 10$  to  $Y = 65$  and then continues horizontally further.



Here it must be emphasized that because the polygon is sorted according to the  $X$ -values, there cannot be two entries for the same  $X$ -value in the polygon. In order to parameterize a “step” (e.g. to the left of 100 the value is

50, to the right the value is 70) the input X-values must differ slightly (in the example given this can be achieved by input points (100/50) and (100.01/70)).

In the definition of the polygon neither the unit nor the relationship of the X and Y values are known at the start. The fact that the polygons are linked to an instrument restricts the possibilities somewhat. Nevertheless, a polygon, which is considered to define a tracking curve e.g. Crest track, can have X-values referencing, e.g., rotational speed, time or torque, depending on which reference variable is parameterized.

There can also be different references with the polygons for spectrum evaluations. The first substantial difference is that this reference can be a fixed frequency (X-values in Hz) or an order spectrum (X-values in order). Since a order spectra always has an implicit reference to rotational speed, this can be specified in the definition of the polygon as the location (the order values of the polygon are then relative to the rotational speed of that location).

---

## Parameterizing Measured Values

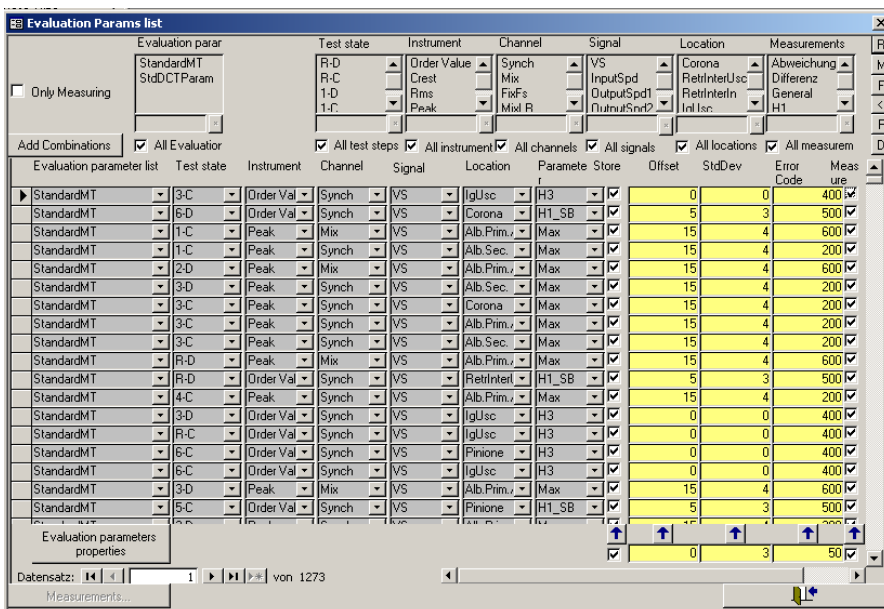
The analysis system can calculate a multiplicity of standard measured values. These have already been introduced in the section “Noise Analysis Theory” on page 27ff. Discom can include the calculation of further measured variables if required. The most common errors which occur in gearboxes, engines or during the gear wheel test, however, can already be found with the standard procedures.

In the parameter database, not only the limits for the measured values are set, as described above, but also which values are to be measured.

If you are new to noise analysis, Discom has already made a pre-parameterization of measure values. Measure value setup is an advanced function and so you can skip this section (and the rest of the chapter up to page 80) on first reading.

### General Measurement Values

The following form allows to specify general measure value parameters. It can be reached by activating the **Advanced Setup** checkbox in the start form and then clicking the **Measurement Value Setup** button.



Evaluation parameter list	Test state	Instrument	Channel	Signal	Location	Measurements	Offset	StdDev	Error Code	Measure
StandardMT	3-C	Order Val	Synch	VS	IglUsc	H3	0	0	400	✓
StandardMT	6-D	Order Val	Synch	VS	Corona	H1_SB	5	3	500	✓
StandardMT	1-C	Peak	Mix	VS	Alb.Prim.	Max	15	4	600	✓
StandardMT	1-C	Peak	Synch	VS	Alb.Sec.	Max	15	4	200	✓
StandardMT	2-D	Peak	Mix	VS	Alb.Prim.	Max	15	4	600	✓
StandardMT	3-D	Peak	Synch	VS	Alb.Sec.	Max	15	4	200	✓
StandardMT	3-C	Peak	Synch	VS	Corona	Max	15	4	200	✓
StandardMT	3-C	Peak	Synch	VS	Alb.Prim.	Max	15	4	200	✓
StandardMT	3-C	Peak	Synch	VS	Alb.Sec.	Max	15	4	200	✓
StandardMT	R-D	Peak	Mix	VS	Alb.Prim.	Max	15	4	600	✓
StandardMT	R-D	Order Val	Synch	VS	RetIntert	H1_SB	5	3	500	✓
StandardMT	4-C	Peak	Synch	VS	Alb.Prim.	Max	15	4	200	✓
StandardMT	3-D	Order Val	Synch	VS	IglUsc	H3	0	0	400	✓
StandardMT	R-C	Order Val	Synch	VS	IglUsc	H3	0	0	400	✓
StandardMT	6-C	Order Val	Synch	VS	Pinione	H3	0	0	400	✓
StandardMT	6-C	Order Val	Synch	VS	IglUsc	H3	0	0	400	✓
StandardMT	3-D	Peak	Mix	VS	Alb.Prim.	Max	15	4	600	✓
StandardMT	6-C	Order Val	Synch	VS	Pinione	H1_SB	5	3	500	✓

As already mentioned above, this is independent of type and test bench. Different lists can be created and used for diverse types or test benches.

With this list, the following parameters are set for the different measurement values: error code (given with negative evaluation), whether the measured value is currently to be measured or not (allowing measure values to be deselected without having to be erased), and whether the measured value is to be stored in the result data or not (not storing makes sense for purely intermediate results, which are only used to calculate another measured value). Two further parameters are set in the columns **Offset** and **StdDev**, which considerably influence the creation of learned limits (see **How Limits Are Learned** on page 25).

In practical terms, the function of these parameters is as follows: With *Offset* you can shift a learned limit. If the value of the calculated limit is too close to the measured values and an incorrect n.O.K. evaluation results, then the *Offset* value can be modified in order to get out of the critical range with the calculated limits. With *StdDev*, however, it can be specified to what extent the fluctuation of the measured values is taken into account in the calculation of the limits. If a high value is entered here, then the calculated limit keeps further away from the measured values should these fluctuate strongly, and so a n.O.K. evaluation can be avoided with fluctuation in the tolerance range.

Changes to these settings should be made with caution since they are in principle valid for many types and test benches. Changing one of these parameters changes it for all types and test benches wherever the relevant list is used!

## Adding Measure Values

The Evaluation parameter form is the first form we look at that has an **Add Selection** button to the left of the key selection. The function of the button is obvious: adding entries to the list of the measured values. This is a powerful function and should, therefore, only be used after careful deliberation. Thoughtlessly clicked, the database can be quickly filled unexpectedly with numerous false entries.

To add entries, you must first of all specify, using the key selection fields, which keys in the list have not yet emerged. We recommend that beginners deactivate **all** the key selection fields' checkboxes which activate the entire list (e.g. "all test steps"). Then, by moving from left to right, choose only the desired entries from the lists. We further recommend that, at the most, you activate only the whole of one of the lists with the help of the check mark. Multiple selections in other lists are to be avoided, if possible. The reason for this recommendation: If you don't watch out, the database is filled with false entries!

Why this can happen so quickly is shown by the following example: A new measured value for the spectral value instrument, parameter H5, is to be added. This measured value should be active for all sensors (S1 and S2), as well as the locations GearwheelIn, GearwheelOut and ReverseIdler in all matching test steps. Anyone wanting a quick setup selects "all test steps", instrument: spectral value, "all channels"; "all signals", location: GearwheelIn, GearwheelOut and ReverseIdler, parameter: H5, quickly clicks on "add selection" and receives the cross product of the selected keys. In detail this leads to the following false entries:

- Unwanted entries for the key "sensor". Since for some measured values the rotational speed or the torque makes sense as sensor, these entries are present in the key selection list for the sensor. The most frequently used sensors are, however, typically the noise sensors. Therefore, if both (= all) noise sensors are to be *displayed*, then "all sensors" is the preferred selection. With "addition", however, this means that entries are made for all sensors, even those, such as rotational speed or torque, which, in this context, are useless.
- Unwanted entries for the key "location", dependent on the test step. The reason for this is as follows: In a normal gearbox, GearwheelIn and GearwheelOut are the two gear wheels of the gearbox which correspond to the current gear. These are linked together, and run under load. The reverse idler wheel runs between them in order to reverse the rotation in the reverse gear. This wheel, however, only runs under load in the reverse gear, so it does not make sense, under normal circumstances, to parameterize it for measurement in other gears.



## Erasing Measured Values

It is possible to erase entries in the measured values list which are not needed, perhaps the result of a too hasty click on “Add Selection”. If you look at the form closely, you can see a triangle located in the first column of the data area (even left of the list name). This is known as the “data record marker”. It marks that data record in the visible form that is selected for editing. In the first column, it is possible to mark more than one entry for processing (as in Windows e.g. with the shift key); the column for the data record marker is then underlaid). Pressing the **Del** key on the keyboard instructs the parameter administration to remove the marked data. After confirmation of a prompt the marked entries disappear completely from the database.

## Interaction Between Evaluation Parameter List and Limit Value Setups

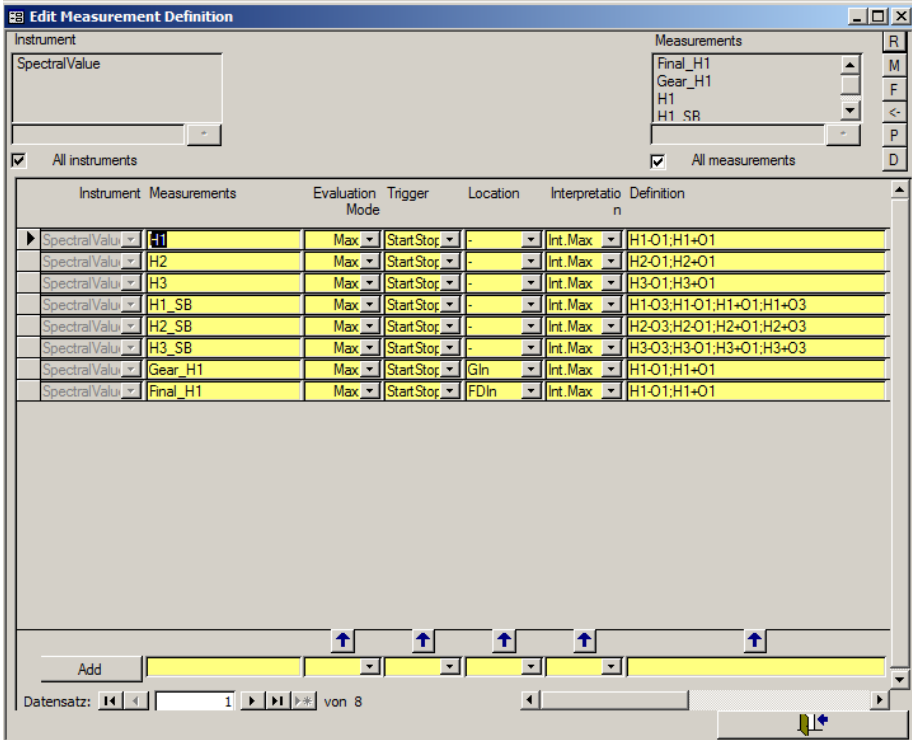
To avoid having to add measured values in more than one place, the parameter administration assumes that each measured value shall also be evaluated. Consequently, when adding measured values to the evaluation parameter list, appropriate entries are also made in the limit value setups. This takes place relatively quickly. When measured values are deleted, redundant entries are also erased from the limit value setups. However, since deletion is a rather unpleasant function for database systems, this takes longer than addition. If you need to delete measured values then you need to exercise a little patience here (depending on size of the database and the performance of the computer carrying out the task).

A consequence of this alignment of the different tables is that it is possible for the limit value list to contain entries which are not measured. Example: Two lists of evaluation parameters, “List1” and “List2”, exist. In list 1, for example, a measured variable, “H5”, is used which is missing in list 2. List 2, uses a measured variable “H5\_SB”, which is, however, missing in list 1. Both lists are in use for at least 1 type/test bench. Regardless of which list is used for the type/test bench, both entries are found in the limit values list.

This situation is justified as follows: Let’s assume that list 1 is in use for a type/test bench and is set up well with regard to the limits. For a test measurement, we would now like to use and limit another list of evaluation parameters. Once the test measurement has been preformed, the old limits should be valid again. The only way that the parameter administration can do this is to retain both entries with the limit value setups, regardless of which of the two lists is in use at the time.

## Creating a New Measured Value

Now that it is clear how new measured values are added to a list of evaluation parameters, we now explain how new measured values are *defined* at all. For this purpose and for clarification, select the spectral value (and only this) in the key selection list for the instrument. This releases the **Measured Values** button at the bottom left of the form. Clicking on it opens the following form:



Instrument	Measurements	Evaluation Mode	Trigger	Location	Interpretation	Definition
SpectralValue	H1	Max	StartStop	-	Int. Max	H1-O1;H1+O1
SpectralValue	H2	Max	StartStop	-	Int. Max	H2-O1;H2+O1
SpectralValue	H3	Max	StartStop	-	Int. Max	H3-O1;H3+O1
SpectralValue	H1_SB	Max	StartStop	-	Int. Max	H1-O3;H1+O1;H1+O3
SpectralValue	H2_SB	Max	StartStop	-	Int. Max	H2-O3;H2+O1;H2+O3
SpectralValue	H3_SB	Max	StartStop	-	Int. Max	H3-O3;H3+O1;H3+O3
SpectralValue	Gear_H1	Max	StartStop	Gln	Int. Max	H1-O1;H1+O1
SpectralValue	Final_H1	Max	StartStop	FDIn	Int. Max	H1-O1;H1+O1

At the bottom of the table, there is an 'Add' button and a row of input fields for defining a new measurement.

Like the other forms, you can make entries in the fields in the bottom row for a whole column. In this form, the bottom row has another function: If you want to enter a new definition, you enter it in this row and click the button **Add** to add it to the list of measurement definitions. Please note that the program insists that all fields are filled out before you add the row. If it allows you to proceed faster, you can enter temporary values first and change the entry later e.g. by modifying the whole column. Only the name of the measurement should be correct from the beginning.

Only after a measured value (to be more precise, the parameter for a measured value) has been inserted here, it can be added and used in the evaluation parameter list.

Vice versa: A parameter entry from the definitions list only has effect in the measurement program after this parameter has been added to the list of evaluation parameters as well.

For each instrument (e.g. Crest, spectrum, order track, etc.) other, basically similar forms for the definition of new parameters exist. What specialties need to be setup in each case results from the instrument's function.

## Different parameters and their meaning

All measure value entries contain the parameter **Evaluation Mode**. This parameter sets what evaluation should be done with the corresponding measure value. Usually this parameter is set to *Max*, because you mostly intend to get an n.O.K. result with error code when the valid (possibly learned) limit is exceeded. This is the right setting to find "too loud" noises.

Nevertheless, some tests require not to check for too high values but also for too low values. These tests use the evaluation mode *Min*. You then get an n.O.K. result if the value is below the limit.

Many measured values are gained by continually calculating data to a final value during measurement. The most important of these calculations are: Gaining the *maximum* value, gaining the *minimum* value and calculating the *mean* value. The corresponding parameter is called **calculation** and is especially used with the instruments Crest, Rms and Peak. Like evaluation, the usual setting for this parameter is *Max* to find loud noises during measurement.

**Parameter for the instrument Spectral Value:** In the screen shot above, you see that this instrument has three parameters **Location**, **Interpretation** and **Definition**. All three together define an area in the spectrum from which a result value is calculated according to the setting in interpretation. The area can be a point or a combination of some intervals and the value can be the *maximum* value, the *minimum* value or the mean value of points in that area. Furthermore, the calculation of the energy of the points (energetic sum) is also possible. The spectral value instrument operates on the spectrum which has been calculated during measurement (average spectrum or peak hold spectrum) and calculates a single value according to the settings.

**The Syntax of a Definition:** First of all, the definition string is a list of points in the spectrum which are separated by semicolons. The easiest way of defining such a point is to give an explicit order like "O26" for order 26 or to give a frequency in case of a fixed frequency spectrum (like F1500 for 1500 Hertz).

Usually you find Harmonics (see also page 31) of a base order in an order spectra (like the mesh order of meshing gears). Since you do not want to

explicitly specify the possible orders for the different transmissions, you use an alias like “H1” (first harmonic = base order). If the other settings (details follow) allow the measurement to calculate the base order, H1 will be replaced by the order belonging to the transmission.

Using H1 as reference, you can define further, relative order positions like H1-O3 (three orders left of H1).

When specifying orders, the speed reference is always important. If you specify orders using the letter “O” the speed reference is the one of the spectrum where you use it. Alternatively, you can use the letter “M”. Then the orders refer to another speed reference (details follow).

Finally, you can use the letter L in a definition. With L you can count order lines in the spectrum. The effective result order depends on the resolution of the spectrum on which the spectral definition works.

Points, defined with the components above, can form intervals which contribute to the calculation of the result, according to the parameter interpretation. We already named the possible calculation variants above: Maximum, minimum, mean value and energetic sum. For the parameter interpretation, these calculation variants are combined with the information how the list of spectral points has to be interpreted: As points (only implemented for a single point) or as intervals. Example: Specifying “Interval Max” for the list of points “H1; H2; H3; H4” has the effect, that the maximum value of the intervals “H1; H2” and “H3; H4” is being calculated (all points between H1, H2 and H3, H4 respectively contribute to the result). If you want to get the maximum of the single points (as perhaps guessed), you have to specify the following list: "H1;H1;H2;H2;H3;H3;H4;H4".

Specifying references: Like indicated above, Hx terms and Mx terms need a reference. You specify this reference using the parameter “Location”. You can select from all locations which have a base order and a relative speed assigned to it. As you can already see in the screen shot above, a fully specified measure value can look like this:

Instrument	Measurements	Evaluation Mode	Trigger	Location	Interpretation	Definition
SpectralValu	Final_H1	Max	StartStor	FDIn	Int.Max	H1-O1;H1-O1

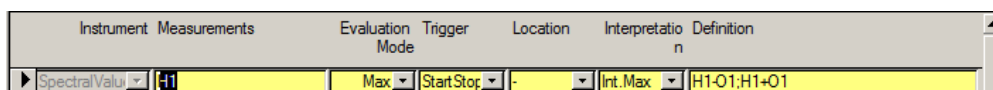
## Spectral Values and Spectra

Since spectral values are a secondary evaluation for spectra, the instrument needs to know the source spectrum for its work. The general rule for this is that a spectral value is valid for the spectrum having the same key (beside instrument and measure value) as the spectral value. This implies that spectra must not use more than one measure value definition. Otherwise, the spectral value instrument can not uniquely find the desired spectrum.

Like indicated above, the evaluation of spectral values can turn off the evaluation of the spectral limit curve at the corresponding positions (see chapter *The instrument “Spectral Value”* on page 32). This is true when spectral value and spectrum have the same evaluation mode and the calculation mode matches the evaluation mode. (This means that the spectral value needs to calculate a maximum value for maximum evaluation or to calculate a minimum value for minimum evaluation.) Deactivating the spectral limit curve is especially important for harmonics since gear meshes usually have a different noise behaviour than other noise sources of a transmission.

### Simple Definition of Harmonics

In case the test object is quite simple (e.g. a cogwheel or an axle), you can use a simplified form for the definition of harmonics. This means in detail that you do not specify the reference location in the measurement definition. The measurement program finds the reference due to other settings (example follows). In the measurement definition, such a definition looks as follows:



### Tracking Curves and Trigger

The parameter settings for instrument which generate tracking curves only slightly differ from the settings for the corresponding single value instruments (e.g. Crest compared with Crest track or Spectral Value compared with Spectral Track). Although a **Trigger** is specified for single value instruments also, this setting is not really important for these instruments. On the other hand, for tracking curves this setting is extremely important, since it defines the x-axis for the corresponding track and the resolution for the recording (see also chapter “Speed and other Command Variables” on page 113).

### Parameter of the Instrument Track-Interval

Evaluations based on tracking curves are usually more exact than evaluations of the corresponding single values. Nevertheless, they have the disadvantage that you cannot as easily use statistical functions or get an idea what’s going on as easily (stray bands of tracking curves). The instrument track-interval allows to get a compromise.

The track interval instrument is a single value instrument which calculates – as the name suggests – a single value from a track. The basis can be the whole track or only a part of it.

The parameters **Evaluation Mode**, **Trigger** and **Calculation** have already been discussed before. **Source-Instrument** and **Source-Parameter** connect

the track-interval with the track. Like with spectral values and spectra, the tracks of track interval and track must match except for instrument and parameter (explicitly specified for the track interval). Finally, **Min** and **Max** specify the range of the tracking curve which shall contribute to the track interval value. This range depends on the x-axis of the tracking curve which is the basis for the track interval. If you want to include the whole curve then you set a range which is significantly wider than the possible x-range of the tracking curves.

## Examples for Measurement Value Definitions

The following chapter explains some typical definitions of measurement values in an example situation. The transmission design for the example is as follows: An input shaft named “Prim”, a secondary shaft named “Secondary”, an output shaft named “Diff”. The cogwheels are named “GearIn” for the driving gearwheel, “GearOut” for the driven gearwheel. Likewise “FinalIn” and “FinalOut” for the wheels of the final drive gear set.

***Example 1: Spectral values for all base orders (H1) on all shafts in the synchronous channel and in the Mix-channel in 4-D.***

Definition of the measured values

Name	Eval.Mode	Location	Interpretation	Definition
Gear_H1	Max	GearIn	Pts.Max	H1
Final_H1	Max	FDIn	Pts.Max	H1

Corresponding entries in the evaluation parameter list (spectrum and spectral value):

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectrum	Synch	Secondary	Max
4-D	Spectral value	Synch	Secondary	Gear_H1
4-D	Spectral value	Synch	Secondary	Final_H1
4-D	Spectrum	Synch	Prim	Max
4-D	Spectral value	Synch	Prim	Gear_H1
4-D	Spectrum	Synch	Diff	Max
4-D	Spectral value	Synch	Diff	Final_H1
4-D	Spectrum	Mix	Prim	Max

4-D	Spectral value	Mix	Prim	Gear_H1
4-D	Spectral value	Mix	Prim	Final_H1

The measurements of the list above collect the values for H1 for both gearsets in all spectra where you expect to see them.

***Example 2: „Unexpected“ Base orders (H1) in synchronous spectra***

Sometimes, certain transmission ratios have the effect that a gear mesh order can be seen in a spectrum where you would not expect to see it. If the transmission ratio is a whole number, even the order levels match. That means that the gear mesh orders are as clearly visible in spectra where you do not expect them as in spectra where you expect them. (When the transmission ratio is “almost” a whole number, you will see a more or less lower level for the order in the “foreign” spectrum. The more the transmission ratio is away from a whole number, the more the level will be lower.) For such a situation, you will have to **add** the following entries to the list in example 1:

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectral value	Synch	Prim	Final_H1
4-D	Spectral value	Synch	Diff	Gear_H1

Concerning the **evaluation** of the spectral values in example 1 and 2, you will not set equally tight limits for all entries corresponding to Gear\_H1 (or Final\_H1). For the n.O.K. recognition it is not important whether Gear\_H1 is n.O.K. in Mix, Synch Prim, Synch Secondary or even Synch Diff. The important thing is that if a n.O.K. evaluation is detected at these positions, most likely the gear mesh is the reason.

Practically speaking, you usually set only one tight (probably fixed) limit for Gear\_H1 in one channel. The other spectral values for Gear\_H1 either get fixed, high limits to rather turn off the evaluation at these positions or not so tight, learned limits. One reason for this is to reduce the number of entries in the error list to the essential minimum.

***Example 3: H1-Side bands for the detection of eccentricities***

Eccentricities of a cogwheel have the result that the gear mesh noise does not have a constant frequency at constant speed but varies above and below the expected gear mesh frequency of the wheel. Because of this fact, you expect eccentricities to show up in the orders directly above/ below H1 (the “side bands”). The following definitions include 3 orders above and below H1 for the calculation of the spectral value. Again, the list is meant to be **added** to the list in example 1.

Definition of measured values:

Name	Eval.Mode	Location	Inter-pretation	Definition
Gear_H1_SB	Max	GearIn	Int.Max	H1-O3;H1-L1;H1+L1;H1+O3
Final_H1_SB	Max	FDIn	Int.Max	H1-O3;H1-L1;H1+L1;H1+O3

Corresponding entries in the evaluation parameter list:

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectral Value	Synch	Secondary	Gear_H1_SB
4-D	Spectral Value	Synch	Secondary	Final_H1_SB
4-D	Spectral Value	Synch	Prim	Gear_H1_SB
4-D	Spectral Value	Synch	Diff	Final_H1_SB

Usually, the measurement values to detect eccentricity are not used in the mix spectrum but only in the synchronous spectra where you expect to see these orders.

Please note, that the definition of the measured value contains *two* intervals. The points H1-O3 and H1-L1 define the interval *below* H1, the points H1+L1 and H1+O3 define the interval *above* H1. The result value is the maximum of the points in *both* intervals.

Furthermore, the definition for H1\_SB depends on how H1 has been defined. Since H1 has been defined as point only, H1\_SB must define the neighboring points as H1-L1/ H1+L1 to avoid gaps.

***Example 4: Alternative Definition (Visibility of the limit)***

The limits for spectral values are integrated into the spectral limit curve for the positions where the spectral value’s limit is valid instead of the spectral limit curve’s limit. That way, the user can directly see which limit is valid at which position of the spectrum, disregarding whether it is a spectral curve or spectral value limit. Sometimes, the limits for spectral values are not clearly visible, especially when the definitions use only a point (like H1 in our example). To avoid this, H1 is sometimes defined as an interval instead of a point as follows:

Definition of measured values:

Name	Eval.Mode	Location	Interpretation	Definition
Gear_H1	Max	GearIn	Int.Max	H1-O0.5; H1+O0.5



Final_H1	Max	FDIn	Int.Max	H1-00.5; H1+00.5
----------	-----	------	---------	------------------

As a consequence, the definition of H1\_SB must also be changed:

Name	Eval.Mode	Ort	Inter-pretation	Definition
Gear_H1_SB	Max	GearIn	Int.Max	H1-03;H1-00.5;H1+00.5;H1+03
Final_H1_SB	Max	FDIn	Int.Max	H1-03;H1-00.5;H1+00.5;H1+03

If you look closely at the definitions, you see that the points H1-00.5 and H1+00.5 are covered by both spectral values (H1 and H1\_SB). To be exact, the definitions for the side bands had to be like “H1-00.5-L1”, etc. But in order to keep the side band definitions string short, you accept the double evaluation at these points. Concerning practical relevance, you do not expect that a synchronous spectrum shows the highest peak at a position H1-00.5 since synchronous spectra usually have much less level at positions which do not correspond to whole numbers.

***Example 5: Band width of gear mesh noise***

Since the spectra, which are the basis for the spectral values, are a result of averaging or holding the peak value during the measurement, it can be that certain noise phenomena which disturb in car measurements do not show up in these spectra. Such phenomena are gear mesh noises which are audible only in a certain speed range like: “4<sup>th</sup> gear loud between 1500rpm – 2500rpm”, especially when the level of the 4<sup>th</sup> gear order is usually higher at higher speed (eg. 3500rpm). To be more exact: When all transmissions have a high level at 3500rpm, you do not see in the spectra that some transmission have the same high level already at 2000rpm whereas others do not. First of all, this leads to spectral tracks.

Furthermore, when noise in the car is involved, the human ear usually does not hear this noise “order exact”. That means, it is less important for the human ear whether the disturbing noise comes from order 45, 46 or 47 (whereas order 46 is H1). Therefore, it makes sense to include side bands for spectral tracks. Furthermore, the source for order tracks is usually the mix channel in order to have as less filtering as possible for this task.

Definition of measured values:

Name	Eval.Mode	Location	Interpretation	Definition
Gear_H1	Max	GearIn	Int.Sum	H1-M1;H1+M1

Final_H1	Max	FDIn	Int.Sum	H1-M1;H1+M1
----------	-----	------	---------	-------------

Corresponding entries in the evaluation parameter list (spectrum and spectral value):

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectrum	Mix	Prim	Max
4-D	Spectral-Track	Mix	Prim	Gear_H1
4-D	Spectral-Track	Mix	Prim	Final_H1

If you take these entries as addition to example 1, it is no problem that the measurement values for the spectral track have the same name as those for the spectral value. But you have to keep in mind that the definitions differ.

When defining the points for the spectral track, the letter „M“ was used instead of „O“. The reason for this is as follows: Assuming the final drive has a ratio of 11/66. Due to the fact that the intermediate shaft runs 6 times faster than the output shaft (whole number), you see all orders of the intermediate shaft (final drive at order 11) in the spectrum of the output shaft (final drive at order 66) as well. If you now have an eccentricity order at order 10 of the intermediate shaft, where will it show up in the output shaft spectrum? Answer: At order 60.

Since this position shift depends on the transmission ratio, using the letter „M“ forces the measurement system to include this fact in the calculation of the correct positions. In other words: If you want to cover the same range of orders relative to a certain reference with the same measure value in different spectra, you have to use the letter “M” and enter the corresponding reference. Definition H1-M1 relative to FDIn (from the spectral track definition) with final drive 11/66 results in order 10 on the intermediate shaft and order 60 on the output shaft which is the same absolute order, whereas the definition H1-O1 relative to FDIn results in order 10 on the intermediate shaft again, but order 65 on the output shaft. (only the position of H1 is being rescaled).

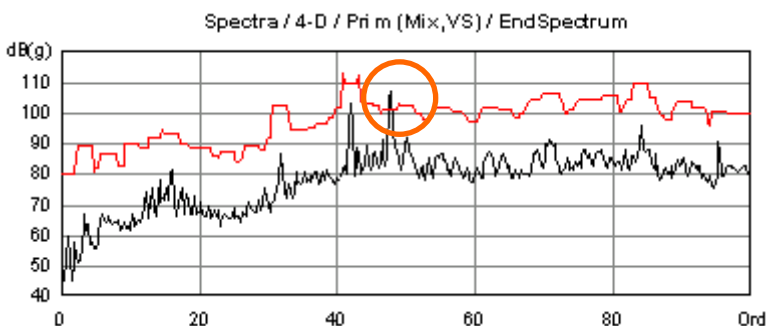
If you use M-order definitions, you usually take the faster wheel of a gearset as reference. For gear gearsets (like 1<sup>st</sup> gear, etc.) this sometimes leads to the problem that the driving wheel is the faster one in lower gears, but in higher gears it is vice versa. You can solve the problem by introducing a “virtual” gear “gear” which is equivalent to GearIn or GearOut depending on the gear. (Since this setting is part of the kinematics and has other effects on the settings, I skip the details. If you need this, please ask.)

Although it does not fit with physical theory, definitions which mix harmonics with orders relative to the destination spectrum (O-terms) are frequently used with side band definitions for eccentricity detection (see

example 3 and 4). One reason is that you can control the effect of the definitions in the spectrum better. Nevertheless, sometimes misunderstandings happen when the area of Final\_H2\_SB overlaps with Gear\_H1 on the intermediate shaft resulting in unexpected defect messages for certain positions. (Example: FDI<sub>n</sub> 18 teeth, Gear<sub>Out</sub> 37 teeth. Area of Final\_H2\_SB:  $2 \cdot 18 - 3 = 33$  to  $2 \cdot 18 + 3 = 39$ . Gear\_H1 having order 37 is almost perfectly in the middle.)

**Example 6: How to set up bearing defects, etc.**

When other parts beside gear meshes make noise, you usually see orders in the mix channel, like for example order 48 in the following example:



In order to get distinguished limits/ error codes, you cover this order with a measure value for the instrument spectral value defined as follows:

Definition of the measured value:

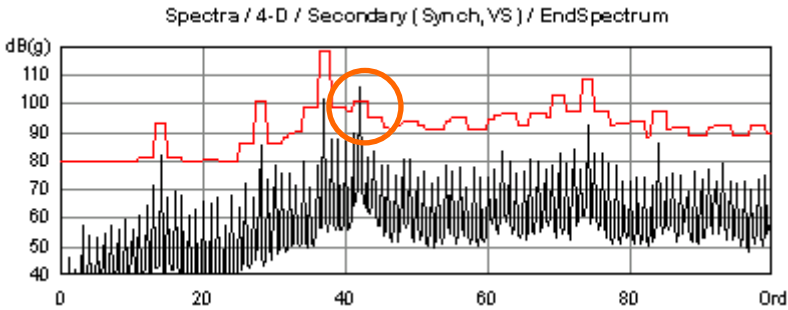
Name	Eval.Mode	Location	Interpretation	Definition
Bearing_O48	Max	-	Int.Max	O47;O49

Corresponding entries in the evaluation parameter list (spectrum and spectral value):

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectrum	Mix	Prim	Max
4-D	Spectral Value	Mix	Prim	Bearing_O48

This definition directly covers the situation as you can see it in the mix spectrum and you have a solution with the desired effect. Nevertheless, it may be that this is not the best way to do it. The real reason for mix order 48 from the example was a component defect on the intermediate shaft. The

spectrum of the intermediate shaft shows the same noise at order 41 as follows:



The same defect showing up with order 41 in the spectrum of the intermediate shaft will show up with different orders in the mix spectrum depending on the ratio of the current transmission. Therefore, it is not the best solution to solve this problem like above. You can better do is as follows:

Definition of the measured value:

Name	Eval.Mode	Location	Interpretation	Definition
Bearing_O41_Sec	Max	FDIn	Int.Max	M40;M42

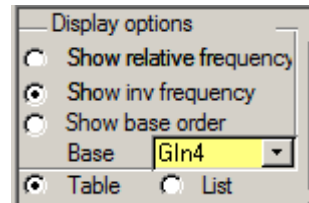
With this definition, you specify orders 40-42 relative to the frequency of FDIn (that is the reference of the intermediate shaft). First of all, this measured value is added to the evaluation parameter list for the intermediate shaft:

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectrum	Synch	Secondary	Max
4-D	Spectral Value	Synch	Secondary	Bearing_O41_Sec

Since order 48 in the mix spectrum is the same noise, this measured value is added for the mix channel as well (instead of defining and using O48 directly). Using the same measured value makes clear that it is the same noise showing up in a different spectrum. The essential effect is the same like using Bearing\_O48 from above.

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectrum	Mix	Prim	Max
4-D	Spectral Value	Mix	Prim	Bearing_O41_Sec

In order to show the connection between intermediate shaft and mix channel best, it is good to know the reason for strange orders in the spectrum. If you know that a mix order has its origin on the intermediate shaft, because some bearing has a problem there, you would like to re-calculate the orders between the different spectra (Having order X in the mix-spectrum, which order is this in the spectrum of the intermediate shaft?), then define a spectral value with reference to the intermediate shaft but use it in the mix channel as well (as we have done above). You can get the necessary information in the design data form by changing the display options to **Show inv.frequency**. This has the effect that the form now shows the calculation factor from the selected base to a different spectrum. Which order is mix-order 47 in the spectrum of the intermediate shaft? With the display settings shown above, you see the correction factor in the row for GearOut4 (intermediate shaft) and in the column for the 4<sup>th</sup> gear. With factor 0.881 this results in order 41.4 like seen in the spectrum of the intermediate shaft.



**Example 7: Simple definition of harmonics**

Harmonics and their side bands can also be defined in simple form. For the examples 1 and 3, this looks as follows:

Definition of measured values:

Name	Eval.Mode	Location	Interpretation	Definition
H1	Max	-	Pkt.Max	H1
H1_SB	Max	-	Int.Max	H1-O3;H1-L1;H1+L1;H1+O3

Corresponding entries in the evaluation parameter list (spectrum and spectral value):

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectrum	Synch	Secondary	Max
4-D	Spectral Value	Synch	GearOut	H1
4-D	Spectral Value	Synch	FDIn	H1
4-D	Spectrum	Synch	Prim	Max
4-D	Spectral Value	Synch	GearIn	H1
4-D	Spectrum	Synch	Diff	Max
4-D	Spectral Value	Synch	FDOut	H1

4-D	Spektrum	Mix	Prim	Max
4-D	Spectral Value	Mix	GearIn	H1
4-D	Spectral Value	Mix	FDIn	H1

Side bands:

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spectral Value	Synch	GearOut	H1_SB
4-D	Spectral Value	Synch	FDIn	H1_SB
4-D	Spectral Value	Synch	GearIn	H1_SB
4-D	Spectral Value	Synch	FDOut	H1_SB

Like in example 1, the spectral values are listed after the spectrum where they are used. Especially when look at the side bands, you do not see at once in which spectrum a certain spectral value will end up. Exception: When you have a “simple” transmission like an axle the names for the wheels (like FDIn) usually are the same as for the shafts (instead of secondary, etc.).

Concerning the mix channel, you have the problem to decide for which wheel you want to add the measured value H1, especially concerning spectral tracks like in example 5:

Definition of the measured value

Name	Eval.Mode	Location	Interpretation	Definition
H1	Max	-	Int.Summe	H1-M1;H1+M1

Corresponding entries in the measurement parameter list:

Mode	Instrument	Channel	Location	Meas.Quant.
4-D	Spektrum	Mix	Prim	Max
4-D	Spectral-Track	Mix	GearIn	H1
4-D	Spectral-Track	Mix	FDIn	H1

Since the spectral track uses a „M“-term in the definition it makes a difference whether you use FDIn or FDOut. If you use FDIn, the orders refer to FDIn, if you use FDOut, the orders refer to FDOut, re-scaled to the mix channel. The resulting interval (according to the definition) differs significantly!

Furthermore, especially for spectral values, you have the risk of adding *both* wheels by mistake. In worst case, this can end in a double evaluation of the same measure value (in effect) with different limits.

Concerning the other examples: Example 4 (visibiltiy of the limit) can also be done in the same way with simple definitions, whereas example 2 and example 6 do not have a simpler form and do not change. Especially example 2 *cannot* be defined in simple form. Therefore, if you must expect that such a situation can occur because of the complexity of the transmission, you should *not use* the simple definition. Otherwise it could happen that you have the need to add mesh orders like in example 2 *beside* gear mesh orders defined in the simple way. Two definitions for the same thing can be confusing...

### ***Example 8: Sensor signal check***

Usually, some functional checks are included to make sure that the measurement system has really carried out a correct measurement. A n.O.K. because of such a check is not meant to be a defect of the candiate but because of something wrong *during the test*. If the setup is complete this means on the other hand that an OK result means that both the candidate and the test run have been correct according to the valid tolerance limits.

The sensor signal check is usually done with the Rms instrument. The parameter **Calculation** is set to Max, the parameter **Evaluation Mode** however is set to Min. This definition ensures that the measured value takes the highest value during measurement but checks this value for being above a limit. If there is no sensor signal, the calculated maximum should be far below the set limit and lead to a n.O.K. evaluation. If you set a learned limit you have to make sure that the learned limit never falls down to 0.

The sensor check makes most sense used in the Mix or in the Fix channel.

Sometimes, the sensor check via Rms is too unspecific to safely detect a missing sensor signal. As an alternative you may also set a lower limit for a gear mesh order like Final\_H1, if you can be sure that the final drive *always* produces a significant gear mesh noise (H1).

### Example 9: Checking speed and other, similar signals

As already mentioned, the noise emission of a transmission directly depends on the speed. But other control values like torque or temperature of the transmission also have influence on the noise emission. In order to make sure that each test has similar conditions, these control values can also be checked and result in a n.O.K. result if they differ too much.

The instruments for checking control values are CV-Value (for single values) and CV-Track (for tracking curves) where CV is the abbreviation for "Command Variable".

If you check a command variable using the CV-Value instrument and combining the parameter calculation/ eval.mode, you can get the following informations about the command variable:

- Eval: Max, Calculation: Max: This setting stores the maximum of the command variable during measurement and compares it against an upper limit. This check ensures that a command variable does not get too high during measurement.
- Eval: Min, Calculation: Max: This setting takes the maximum value during measurement and compares it against a lower limit. This check ensures that the maximum value does not get too small (e.g. by cancelling a rising speed ramp before reaching maximum speed).
- Eval: Min, Calculation: Min: This setting takes the minimum value during measurement and compares it against a lower limit. This check ensures that the command variable does not get too small during measurement (e.g. because of missing pulses for the speed sensor).
- Eval: Max, Calculation: Min: This setting takes the minimum value during measurement and compares it against an upper limit. This check ensures that the minimum value does not get too high (e.g. by cancelling a falling speed ramp before reaching minimum speed).

In most cases, only the checks using “Max/Max” and “Min/Min” are used to ensure that the command variable remains in a certain interval during measurement.

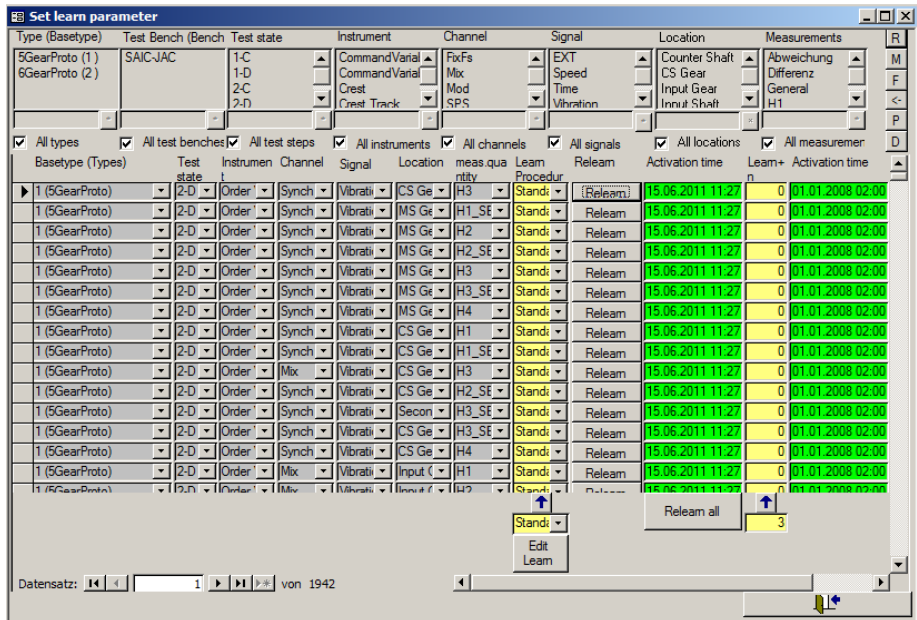
For practical reasons, the check of command variables is usually not done with the CV-Value instrument directly, but by evaluating a CV-Track with the track interval instrument. That way, you make sure that if you get a n.O.K. evaluation, you also have a tracking curve as reference (except when you unchecked the **Store** checkbox explicitly for these curves). Such a curve is very helpful to get an idea what is the reason for a failure because it makes a difference whether a command variable gets strange at the beginning/ end of a measurement (maybe not fitting start/stop settings) or somewhere in the middle.



# Learn Parameters

As already mentioned in the chapter for the test setup settings, the list to specify the learn procedure can be integrated in the list of learn parameters. In this case, the learn procedure can not only be set for test bench group and base type but much more detailedly. The list of learn parameters holds an entry for each entry in the evaluation parameter list for each base type and each test bench (not test bench group!). This makes this list the one with far the most entries in the database.

The form for this list looks as follows:



Sometimes, it happens that useless data have been learned for one test bench or one type (sometimes even only partially). In order to fix resulting useless limits, it is useful to restart the learning process for the corresponding entries in the database, in short “relearn” the measured value. In this case, the measurement program has to “forget” the learned data (meaning mean value and standard deviation) and start “from the beginning”. It is necessary for this process to have the desired results that the measurement PC and the PC where the parameter administration is running are synchron regarding their system time (time zones are respected). The time, when the relearning process has been initiated last is displayed in the form in the column “activation time” beside the Relearn button. This is the information the measurement program gets from the database.

Beside mean value and standard deviation, the measurement program enters into the learn archives the time when the learning process started (effectively). In comparison with the activation time from the database, the measurement program knows exactly whether learning has to be restarted or not.

## Re-learning globally

Sometimes, it is not enough to initiate a relearning from the database (e.g. when the learn archives themselves have been damaged). Another reason can be that basic settings like test run, speed intervals, torque settings, etc. have been changed and you have to “clear up” everything. In this case, it is often more effective to remove the learn archives than to initiate the relearn from the database.

The measurement program of a test bench holds its own learn archives. For each base type, a learn archive file exists holding mean values and standard deviation values. You find these files in a subfolder of the project folder on the measurement PC.

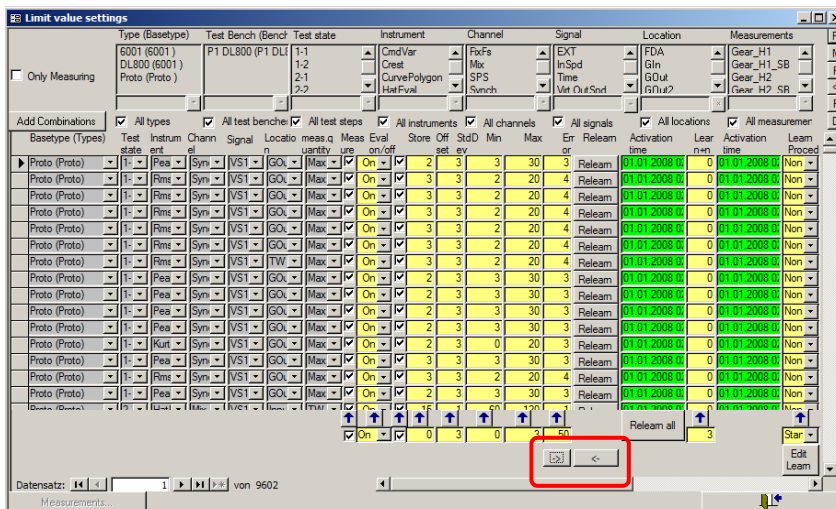
In order to re-learn all the limits for an aggregate type, you can delete these learn archive files. The way to do this is as follows:

1. Use the menu instruction **File – Project Directory** in the TasAlyser program to open the project directory with Windows Explorer.
2. Terminate the TasAlyser program.
3. Go into the subfile `Locals\LearnData` in the project directory
4. Delete the learning file of the aggregate type in question (or all learning files in order to re-learn all types).
5. Re-start the TasAlyser.

Beginning with the next aggregate the TasAlyser will learn new limits.

## Three in one

As already mentioned above, in some projects the evaluation parameter list and/ or the list of learn parameters are integrated into the list for limit curves and limit values. In this case, the form for limit values looks as follows:



As described for the evaluation parameter list, now you find in this form the buttons “add combinations” and “Measurements”. They work in the same way as described above. In the same way, the learn control works as described for the list of learn parameter.

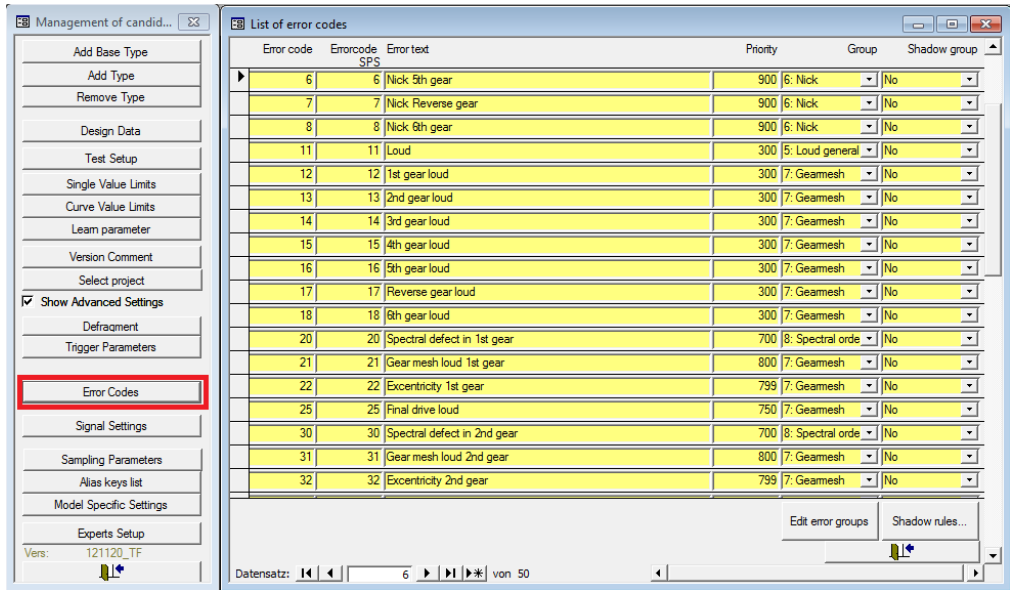
Since the amount of data as shown in the figure above requires a wide screen to be displayed adequately, you can partially hide some data for a better overview. This is being done with the two buttons marked with the red rectangle. If necessary, you can hide with these buttons the limit settings (then you only see the learn parameters) or the learn parameters (then you only see the limit settings). You need rarely both settings at the same time.

When you open the limit settings list with integrated learn parameters, the learn parameters are hidden by default, because re-learning is needed not as often as changing limits. Then you find one of the buttons on the bottom right corner to “un-hide” the learn parameters.

## Defining Error Codes

Each measured value (be it a single value or a curve) is associated with an error code. When the measured value violates a limit, the associated error code is raised and the associated message is shown in TasAlyser’s report window (and later in the evaluation software).

Error codes and error messages are defined in the parameter data base and can be changed as needed. They are listed in the Error Codes form:



## Changing the error messages

The error texts associated with the codes can be changed at any time. There is no limitation to the length of error texts (besides practical considerations about the readability and the on-screen display size). If you are using a language we Discom people are not fluent in reading (like e.g. Chinese), we recommend using dual-language texts like “齿轮啮合响 Gearmesh loud”. So when you ask us for advice and send us measured data (see “Help from Discom” on page 159), we will be able to understand the problem.

## Adding Errors

You can add lines to the error code table as needed. (You can also delete error codes you do not need any more.) Each line consists of the following entries:

<b>Error Code</b>	The error code can be any positive number less than 2147483648. The used codes do not need to be sequential (you do not need to define error 99 if you wish to use error 100). Error code 0 (Zero) is not allowed.
<b>External code (SPS code)</b>	When the test stand control asks TasAlyser for the error messages, it gets the <i>external</i> code as a reply. In most cases, the external code will be equal to the error code.  As an example, the external code can be used to map several errors (like all errors for one gear) to the same test stand code, because the test stand can handle less error codes than the measurement system.
<b>Error text</b>	The text displayed in the report window. See remarks above.
<b>Priority</b>	The errors from one test run are sorted by priority (highest priority goes first). For the production statistics, the first error (the one with the highest priority) is evaluated.
<b>Group/Severity</b>	Errors can be sorted into groups with ascending severity. The severity group can be used for additional statistics or for advanced techniques like re-typing of aggregates.
<b>Shadow group</b>	Shadow groups can be used to implement relationships between errors. So for example, nicks tend to elevate the spectrum and raise a number of spectral errors in addition to the nick error message. Shadow groups can be used to suppress the spectral errors in these cases.

To add an error, just select one line, copy it (**Ctrl+C**) and paste it at the end of the table (**Ctrl+V**) and then change the entries as needed. After you have added the error code, you can use it in the measurement value definition.

## Test Stand Errors

The test stand can send error codes to the measurement system. These “test stand errors” are treated as normal errors: they are displayed together with the acoustical errors, they render a “not OK” overall result and they are stored in the result data base.

Any error code the test stand wants to send must be defined in the Error Codes table. (If the code sent by the test stand must be mapped to a different TasAlyser error code, this can also be achieved.) Please refer to the documentation about the Test Stand Commands for more details about sending test stand errors.



# The Talimer



As described above, the evaluation limits in the measurement program are a combination of learned values and settings from the database. The learned values represent the statistic of each measurement value. The settings from the database are also specified individually for each measurement value.

The user interface for the parameter database – TasForms – allows to edit the limit settings. But since TasForms is Access-based, its graphical abilities are limited. Nevertheless, it is useful for displaying/ editing tables.

Consequently, for the editing of polygons, necessary to give bounds to learned limit curves, TasForms only gives a table view of the anchor points of such polygons. A graphical display is not available.

For this task, you use Talimer. Talimer has a user interface that allows to edit polygons in the database graphically. Moreover, Talimer can read result data and allows you to shape a polygon well to the real need. With Talimer you can edit every polygon, disregarding whether it is a spectrum, a tracking curve or a parameter polygon for shift force evaluation.

Therefore, Talimer is not an alternative for TasForms, but a specialized additional tool for editing limit *curves*.

## Starting Talimer

Talimer works – as TasForms does – on the parameter data base file (which you usually will find in the folder `ParamDb` of your project directory – see “The Project Directory” on page 37). Please note that only one of the tools may operate on the database at the same time. Talimer will not open a parameter database which is open in TasForms at the same time.

There are two possibilities to start Talimer (except from starting it directly from the `discom-program` folder): Open a parameter database or open a Talimer project file.

The Talimer project files (“\*.talip”) mainly contain a link to the parameter database file and some additional information. Usually, you will not create Talimer project files, since they are mainly used to transport information from e.g. Web.Pal to Talimer. Sometimes you find such a file in the “Rotas For Experts” folder to give you something to click on.

The other way is to right-click on the parameter database file in Windows file explorer and select `Open With...` from the context menu. Please select **Talimer** to edit the parameter database with Talimer. (If Talimer does not occur in the list, you can add it with the last entry in the menu. You find the

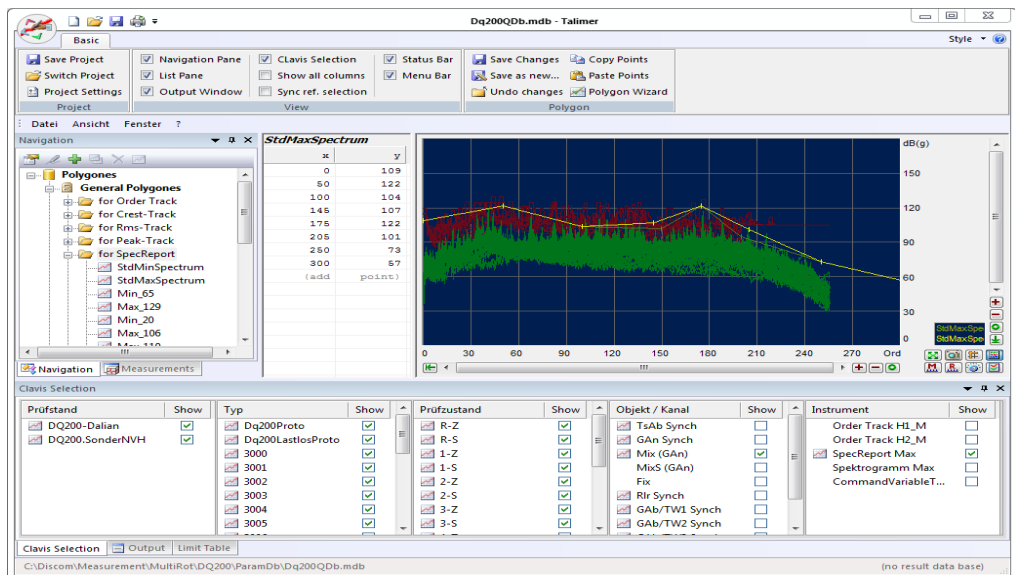


Talimer program in the Discom program folder (C:\Program Files (x86)\Discom\bin).

The Talimer settings are stored in the parameter database file, not in the project file.

## Overview

When you start Talimer, you see a main window with a “Scope”, where you edit polygons and show reference data. On the left and on the bottom, you find some docking windows which control which polygon you edit, and which reference curves you see.



In the multi function bar (“Ribbon”) you find the section View. There you find the multiple docking windows, in case you have closed one and want to re-open it.

The different docking windows are described in the following sections, except one: At the very bottom, you find a window “Output”: This window is usually hidden but you can make it visible by clicking its tab. The output windows contains status and error messages. If you have for example problems accessing the database or problems loading reference measurements, the output window usually contains details to the circumstances or problems.



---

## Navigation

In the docking window **Navigation** (on the left border) you select the polygon you want to edit.

Polygons are sorted into three categories: Polygons which can be used as learn boundaries, polygons which set fixed limits and polygons which are used as parameter for measure values.

The polygons for learn boundaries are called *general polygons*, since they can be generally used for all fitting measured values. In other words, these form a polygon collection from which you can choose for specific measured values. Nevertheless, the collection is separated for each instrument. That means, if you want to set a limit for a spectrum, you can only choose from polygons that were designed for a spectrum. You cannot set a polygon which was designed for a track. (Usually, the polygons for tracks and spectra will have totally different values and areas. This restriction tries to make polygon settings easier.)

Such a general polygon can be used for many measured values. In contrast to that, a fixed limit is specific for just that measured value (and that test step, candidate, test stand, etc.). Therefore, these polygons are called *individual polygons*. If your database contains fixed limits, you will see that the corresponding branch of the navigation tree shows each measured value with its full key.

The *parameter polygons* are not used to form a limit curve. Instead, they are used to calculate a value. For example, they are used for gear shift force evaluation. There you calculate a value that describes the area between a polygon and the gear shift force curve. (To be more exact: The area where the shift force curve is *above* the polygon.) The result of this calculation is a *single value*.

The **navigation** window lists all polygons. Its counterpart is the window **clavis selection** which is usually docked at the bottom side. In this window, you see which measured value uses which polygon. (This selection also controls the content of the docking window **limit table**. See more below.)

---

## Editing a polygon

Open a parameter database with talimer. In the navigation window, select the instrument below general polygons for which you want to edit a polygon and open the corresponding branch in the navigation tree. Double-click the corresponding entry for that polygon.





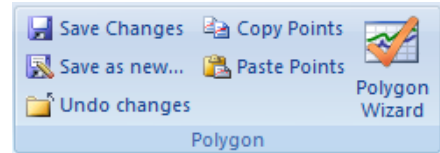


in the list at its correct position. Now you can enter the y-value for the new point.

To delete a point via the table, click on the x-value and erase it (by entering an empty text). After pressing enter the point is being removed.

## Saving changes

In the multi-function bar (“Ribbon”) you find the section **Polygon**.



When you edit a polygon, the functions **Save Changes** and **Undo changes** are being unlocked. Click **Save Changes** to get the changed entries written into the database or click **Undo changes** to revert the polygon to its original form.

If you change the polygon selection after editing a polygon without having saved or undone the changes, Talimer ask whether the changes shall be saved or undone.

## Exchange with Excel

You can transfer the table of polygon points into an excel table or make a polygon from an excel table quite easily.

To transfer the points, simply press **Copy Points** in the **Polygon** ribbon panel. Now, go to your Excel spreadsheet, select a cell and use the standard **Paste** command (keyboard shortcut **Ctrl+V**).

The other way round, you can get your polygon points from any Excel table: If your spreadsheed only consists of a single table with two columns, you directly can call the standard **Copy** command in Excel (keyboard **Ctrl-C**). If your spreadsheet contains more data, select the two column cell range containing your polygon points and **Copy** these. Now switch to Talimer, open the polygon you want to change for editing (as in the previous section) or create a new one using the Navigation window tool bar button, and press **Paste Points** in the **Polygon** ribbon panel.

The points copied and pasted from Excel completely replace the existing polygon points. You will get a preview of the points and will be asked for confirmation before the points are actually replaced. And you still can use the **Undo Changes** button.

## Store the database

If you start editing a polygon with Talimer (or change the contents of the parameter database by other means), Talimer automaically makes a copy of



the original database. (This copy is located in the same folder like the original with the addition “.talimer-bak”).

All changes are being made in the current database. If the measurement program access the database during your work, the changes will get active at once.

When you close Talimer or when you call the command **Save database**, you have two options:

- **Keep changes.** In this case, the copy of the database is being moved to the subfolder Backup and renamed according to the internal version number. In case you notice that you have to revert a change only after closing the database, the backup folder always contains previous versions.
- **Revert changes.** The database with the changes is being erased and the copy renamed back to the original name.

If you want to keep changes, you can enter a short commetary about the changes being made. This is the same thing as you know from TasForms.

Since the settings of Talimer (e.g. Scope settings or connection to a result database) are being stored in the parameter database itself, reverting changes has also influence on these settings. By reverting a previous database the Talimer settings are also reverted to its original state.

---

## Polygon Management

If you want to delete a polygon, search for it in the navigation tree, right-click and select the command **Delete** from the context menu. Alternatively, you can use the button in the tool bar of the navigation window (see below).

If you want to create a new polygon (general or parameter polygon), go to a branch in the navigation tree which is as close to your desired polygon as possible. This can be the branch “for spectrum”, if you want to create a new polygon for spectral limits or it can be the main branch “parameter polygons” if you want to create the first parameter polygon for a certain instrument.

Right-click the branch and select **new polygon** from the context menu or use the corresponding button from the tool bar of the navigation window.

You are now prompted to give a name for the new polygon and for which instrument you want to create the new polygon (if necessary). The polygon names must be unique within their scope.



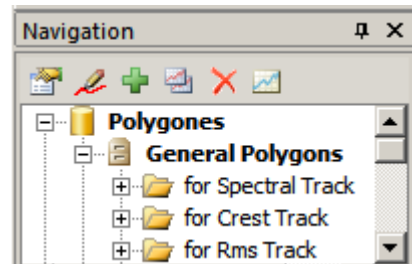
At first, the new polygon does not have any points. Use the methods described above to give points to the polygon. The measurement program does not accept polygons with less than two points.

With the corresponding command of the context menu or the tool bar you can also copy an existing polygon. Give a name for the copy as described above and adapt the points of the new polygon according to your needs.

Via the context menu command **Properties** or with the corresponding tool bar button you can rename a polygon. More about polygon properties below.

## The Navigation Toolbar

The docking window **Navigation** has its own tool bar:



The function of the buttons correspond to the commands of the context menu. Most of the commands are available only if a polygon has been selected. The buttons of the tool bar have the following functions (from left to right):

**Properties:** Display the properties of the selected polygon. This includes especially the display of measured values for which the polygon is currently used. Moreover, you can rename the polygon here.

**Edit:** Selects the polygon for editing. This function corresponds to the double-click in the navigation tree. The polygon is being displayed in the scope and you can edit the polygon points.

**New Polygon:** Add a new polygon to this section of the navigation tree as described above.

**Duplicate:** Makes a copy of the selected polygon as described above.

**Delete:** Deletes the selected polygon as described above.

**Reference:** You can display a second polygon as reference in the scope display. For example: If you edit a maximum polygon you can show the corresponding minimum polygon.

## Tidy up the polygon collection

When you have been working with the parameter database for a while, it can easily be that your database contains polygons which you do not longer use. Reason for that can be that base types have been deleted which used certain polygons or measured values have changed which used certain parameter polygons.



For this purpose there exists a tidy up function. You find it in the multi-function tool bar **Maintenance**.

## Reference Measurements

You can display reference measurements from measurement archives in the Talimer. Talimer can then automatically load the measurements curves of those measurements which use this polygon. That way you can easily see whether the polygon fits to the data. Moreover, you can use the measure curves as basis for the creation of a polygon using the polygon wizard.

### Loading reference measurements

Reference measurements can be loaded from archive files directly or via the result database.

If you want to use an archive file directly, you only need to move it from the windows explorer into the talimer window. Talimer reads out the contents and displays it in the docking window **Measurements**. (You find this window at the same place as the **Navigation** window. You can switch between both windows with the tab at the bottom.)

The tool bar of the **Measurements** window contains also a button which lets you open an archive file. Another button lets you query the result database for measurements or “forget” all loaded measurements.

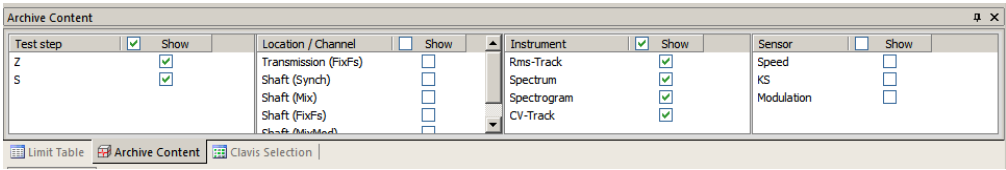
The **Measurements** window lists all measurements of all loaded archive files. Use the **Archive Content** window to see which measurements are in these archives or to manually select one for displaying it.


Time Stamp	Serial No.
2012-01-18.14:05:40	006NRQR010951D120...
2012-01-18.14:10:29	006NRQR010951D120...
2012-01-18.14:15:19	006NRQR010960D120...
2012-01-18.14:27:51	006NRQR010969D120...
2012-01-18.14:34:08	006NRQR010973D120...
2012-01-18.14:40:29	006NRQR010989D120...
2012-01-18.14:46:45	006NRQR010991D120...
2012-01-18.14:59:33	006NRQR010997D120...
2012-01-18.14:53:01	006NRQR011002D120...
2012-01-18.15:06:01	006NRQR011004D120...
2012-01-18.15:24:55	006NRQR011012D120...
2012-01-18.15:12:22	006NRQR011013D120...
2012-01-18.15:18:39	006NRQR011017D120...
2012-01-18.15:37:29	006NRQR011023D120...
2012-01-18.15:31:12	006NRQR011026D120...
2012-01-18.15:43:46	006NRQR011037D120...
2012-01-18.15:56:22	006NRQR011042D120...
2012-01-18.15:50:05	006NRQR011052D120...

### Archive Content

The docking window **Archive Content** is usually located at the bottom of the main window together with the windows **Clavis Selection** and **Limit Table**. (If your monitor is large enough you can also place the docking windows above each other. You do not need to switch between the windows, then.)

The Archive Content window lists in the corresponding columns for which test steps, types, sensors, instruments, etc. data is available:

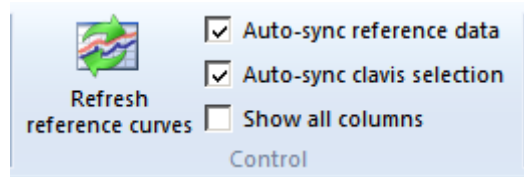


If you select a polygon in the navigation tree (double-click), the corresponding entries in the Archive content window which use this polygon are being marked with the symbol . Moreover, the corresponding measured curves are being loaded from the archives and displayed in the scope window together with the polygons.

## Auto-Sync and Display of Various Data

This auto mechanism is not always desired. Sometimes, you see much more measurement curves than you want to see. (For example: It can be that you only want to see spectra from the mix channel and not also those of the synch channels even if they also use the same polygons). On the other hand, it may be that you want to see data which does not seem to have anything to do with the selected polygon (e.g. spectral track polygon compared with a RMS track).

Therefore, the automatic sync between polygon selection and reference measurement can be turned off. You find the corresponding function in the section **Control** of the multi function bar.



The checkbox **Auto-sync reference data** turns off the automatic synchronization between polygon selection and loaded reference curves. (The checkbox **Auto-sync clavis selection** does the same between clavis selection and content of the limit table.)

When auto sync reference data is turned off, you can select various measured curves in the docking window archive content (by entering green checks in the corresponding boxes). Then press the button **Refresh reference curves** as displayed above to see the curves in the scope.

To do this, it is not necessary to select a polygon. That means you can also use Talimer to display the content of various measurement archives in case you want to have a quick glance at the result curves.

The third check box **Show all columns** has effect to the docking windows **Archive content** and **Clavis Selection**. If **Show all columns** is unchecked, columns which have only one entry are hidden in these windows. (In the screen shot above, the columns *test bench* and *type* have been hidden.) This



function helps to concentrate on those entries where there you indeed have to select something.

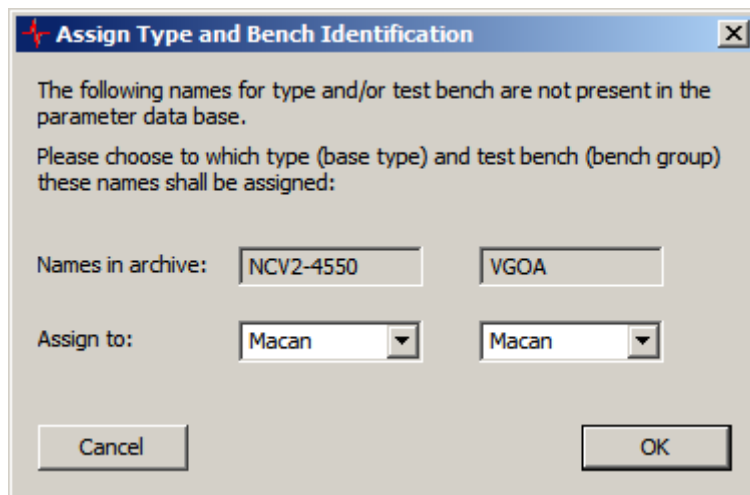
## Foreign Archives, Foreign Names

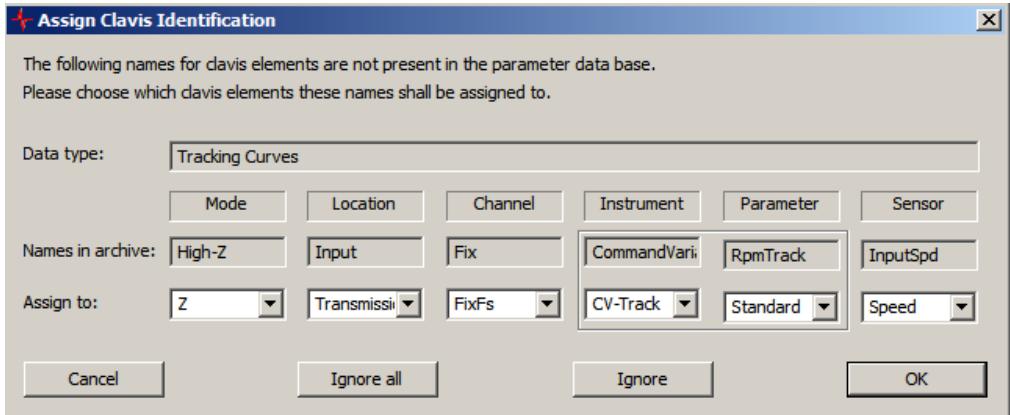
It can happen, that you want to load reference measurements which have not been measured on the same test bench (or test line) as the parameter database is designed for. The reason can be that you want to display data from another factory producing similar aggregates or that the archives have been written by TasAlyser's predecessor.

Talimer generally can only work with measurements belonging to the currently used parameter database. Only the test steps, types, sensors, instruments, etc. can be used to find reference measurements.

When you load an archive that contains different test steps, types or other measurement specifications that are not in the parameter database, Talimer asks what to do with them.

If Talimer finds unknown types or test stand names, it will ask which of the known types/ test stands shall be used instead. The same is true for unknown measured values.





Clicking Cancel stops reading that archive.

Assigning measurement values has two more options: If you click **Ignore** the current measurement value is being skipped (and will not appear in the archive content list). **Ignore all** skips all measured values of the current data type (which is displayed in the upper part of the window).

Talimer stores the assignments that you selected. If you want to load another archive from the same source, you will not be prompted again. (You will be prompted only for those entries that you **ignored** the previous time.)

These settings are also stored in the parameter database. In case you select not to keep changes when ending Talimer, the assignments done since the last start will also be reverted.

In the multi function bar section **Maintenance** you find a button where you can check, edit or reset these assignments if necessary.

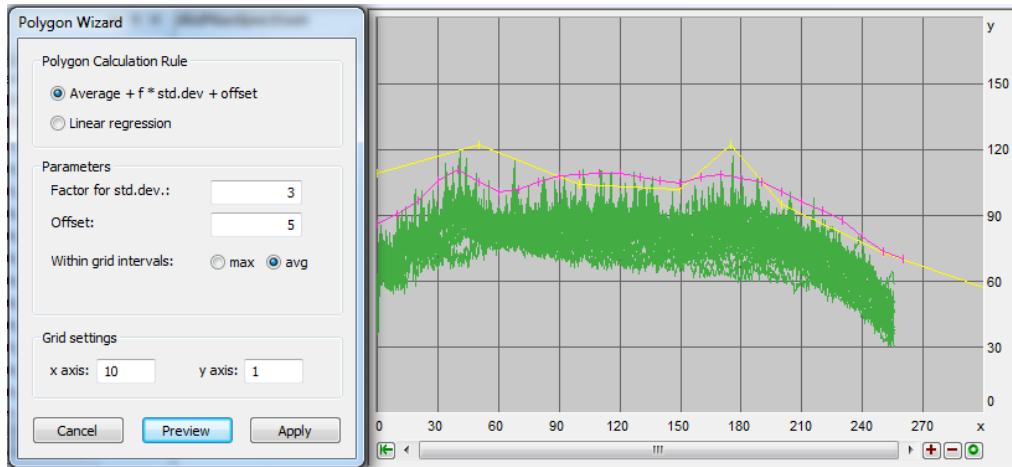
## Polygon Wizard

Instead of manually fitting the polygon points to measurement data curves, you can summon the polygon wizard. The wizard creates a polygon from the displayed measurement curves, which you can refine afterwards.

You summon the wizard with the according button in the **Polygon** ribbon panel. The wizard is only available when there are reference measurements loaded – see above on how to load reference measurements.

Polygon wizard knows different methods to create a polygon from your reference data. Just try and press the button **Preview** (bottom center). In the scope, in addition to the current polygon (in yellow) and the reference measurements (dark green) you will see the current wizard proposal (in pink).





Change the calculation parameters and press Preview again until you are content. Only when you press Apply in the wizard window, the wizard stores the polygon as if you edited the points manually. (That means that you can still cancel the changes with the corresponding button of the multifunction bar.)

Moreover, you can fine tune the points after wizard has done its work.

## Calculation methods

### Grid Size

A polygon usually shall be designed with less points than the measure curves. For this reason you set a grid size. The grid size in x-direction defines the distance between the polygon anchor points. For example, if you define a x-grid of 100 the polygon has anchor points at 1100, 1200, 1300, etc. depending on the area of your result curve. The y-grid defines how much these values are being rounded. A grid of 2 has the effect that the value will be rounded to multiples of 2. If the calculation results in a point (100, 17.2) and the y-grid is 2, the polygon will get the point (100,18). If you enter the value 0, the y-values are not being rounded.

The polygon wizard knows three different ways of creating a polygon based on the reference values.

### Polygons from Average and Standard Deviation

The calculation from average and standard deviation works basically the same way it does for learned limits. The wizard also calculates Average and standard deviation for each point from all measurement curves loaded via  $p = \text{Average} + \text{Offset} + f \times \text{Standard deviation}$ . Finally, the polygon points are combined according to grid size. The points that are combined correspond to the x-grid. Two methods for this summary are available: Maximize or



Average. Please try yourself the effect of both methods for the polygon and **Preview** the result to get a feeling for which method is best in which situation.

### Polygon from Linear Regression

This procedure is also applied within the  $x$  grid intervals. For each interval a regression line is drawn and connected at the interval bounds by averaging. Finally, a “general offset” is being added (which can also be negativ to generate a minimum polygon).

If you set the  $x$  grid width to 0 or a value higher than the reference data length, you get a polygon with two points. Then you have *the one* regression line built from all data points.

As an option, you can give a gradient for this regression line or a fix point. The wizard will then take care that the result line will either use that gradient or will hit the fix point.

### Polygon by maximization

The polygon wizard simply calculates the maximum of all reference curve points within reach  $x$  grid interval. This gives the polygon point for this interval. The **overall offset** is applied afterwards. However, there is one distinctive feature in case the reference curves contain less points than the  $x$ -grid defines. Polygon points are set only at positions where there are measurement curve points as well. If the reference curve has only two points at 1000 and 4000, the polygon will also have only these two points even if the  $x$ -grid is set to 10.

All other calculation methods interpolate values according to the  $x$ -grid. Witch averaging in the example above, the polygon would have 301 anchor points a 1000, 1010, 1020,...,3990, 4000

## Genral Settings

The multi function bar contains on the left the button **settings**. With this button you open a dialog where you can make some general settings.

- Grid size: Defines grid size for each instrument in  $x$  and  $y$  direction (see above).
- Reference measurements: The maximum number of measurements (not measurement values but whole test runs) from which reference curves shall be loaded from. Furthermore you can set whether the limits shall be displayed as well which were valid for evaluation at the time of measurement.
- Result database: Connect to a result database and load fitting reference curves from there. The name of the result database is



being displayed in the status bar at the bottom right of the Talimer window

As already mentioned, these settings are stored in the parameter database and are lost when you decide not to keep changes at the end...

## Individual Polygons

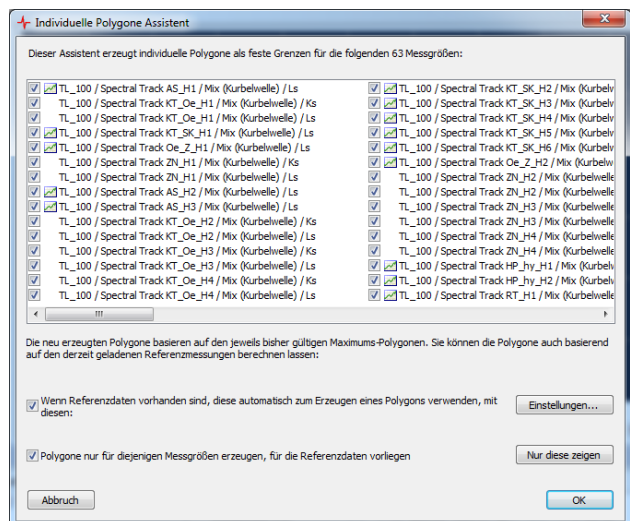
As described above, individual polygons are used as fixed limits for single measured curves.

Existing individual polygons can be edited like general polygons, including the use of the polygon wizard.

If you create a new individual polygon, this means that you change the setting “learned limit” to “fixed limit” for the corresponding measured curve. Since there can be as many individual polygons as measured curves, Talimer offers to create many individual polygons at one go (as well as deleting them). That means, you can call the function **New Polygon** in the navigation tree at any sub-branch below “Individual Polygons” to create individual polygons for all measured curves below this branch.

Nevertheless, this usually leads to much too many polygons. Therefore, you can limit the list of polygons to create.

1. Select a branch of the navigation tree and call **New Polygon**. For example, you call this function for the branch representing a certain test step of a certain type. Then, at first the list of polygons to create contains of all polygons for this test step.
2. Talimer also uses the settings in the window **Clavis-Selection**. (You have to turn off the function **Auto-Sync**, if you want to limit the list of polygons to create that way.) For example you could turn off all sensors but one in the clavis selection. Then the list contains all polygons from step 1 restricted to those of the selected sensor.
3. All measured curves are removed





from the list which already have a fixed limit set.

4. Now Talimer lists all measured curves that are still in the list. When reference curves are loaded, Talimer marks those which correspond to reference curves. With the additional filter “Show only those” (with reference curves present) you can restrict the list even more.
5. Finally, you can uncheck entries in the list to forbid the creation of individual polygons for those.

If you want to create polygons from reference curves, you should check the **settings**. They correspond to those of the polygon wizard (since this is the tool creating the reference curves).

If you now click **OK**, the wizard starts to work. The following action is carried out for each measured curve:

6. For the measured curve the setting “learned limit” is changed to “fixed limit”.
7. If you selected to create the polygons from reference curves, for each measured curve a polygon is being calculated using the reference curves.
8. If no reference curves have been loaded (or you did not select to use them), Talimer checks whether there already exists an individual polygon for this measured curve in the parameter database. If yes, this polygon is being set active again and the wizard is finished with this measured curve.
9. If no former polygon exists, the wizard creates a new individual polygon and copies the points of the maximum polygon which were used before. That means, the new fixed limit as a copy of the former maximum polygon and can be changed afterwards.

If you use the navigation tree function **Show as reference** on a sub tree (e.g. the sub tree representing a test step) the marks and check boxes in the clavis selection and archive content window are adapted according to this branch.

## Deleting Individual Polygons

The deletion of individual polygons also affects many entries, if you call the function on a sub tree in the individual polygon’s section. In the same way as described above, a list of affected polygons is created which you finally can check or uncheck.

Nevertheless, the affected polygons are not really removed from the database. Only the setting “fixed limit” is set back to “learned limit”. You can return to fixed limit any time you want (e.g. using the limit table or TasForms).



## Limit Table

The window **Limit Table** (sharing space with **Clavis Selection** and **Archive Content**) is similar to the **Curve Value Limits** form in TasForms. In case your database has learn settings or evaluation params integrated into the limit table, the window only shows the limit settings (e.g. “Offset” will be hidden).

Type	Test step	Location / Channel	Instrument	Sensor	Limit type	Min Lim	Max/Fix Lim
					*	*	*
Macan	Z	FixFs	Rms-Track	KS	(off)	StdMinRmsTrack	StdMinRmsTrack
Macan	Z	Shaft Synch	Spectrum	KS	learned	StdMinSpectrum	StdMaxSpectrum
Macan	Z	Mix (Shaft)	Spectrum	KS	learned	StdMinSpectrum	StdMaxSpectrum
Macan	Z	FixFs	Spectrum	KS	learned	StdMinSpectrum	StdMaxSpectrum
Macan	Z	MixMod (Shaft)	Spectrum	Modulation	learned	StdMinSpectrum	StdMaxSpectrum

The content of the **Limit Table** window is controlled by the **Clavis Selection** window.

If **Auto-sync clavis selection** is checked in the **Control** ribbon panel, the list shows the entries using the currently selected polygon. If **Auto-sync clavis selection** is switched off, you can set the **Clavis Selection Show** check marks manually and press the **Refresh** button above the limit table.

In every line, you can switch between “no evaluation”, “learned limit” and “fixed limit”. For learned limits you can select the corresponding polygons for min and max, for individual polygons, the name of the polygon is being displayed, but you cannot change it (since it is the individual part of just this measurement).

With the ?-fields in the first row, you can set that entry to a specific value vor *all* measurements in the list. For example, you can set the same minimum polygon for all entries. If you change the setting evaluation to “fixed limit”, this operation is being done only for these entries which already have an individual polygon (meaning which once already had a fixed polygon). If you change from “fixed polygon” to “learned” or “no evaluation”, the individual polygon remains. It is just not being used anymore.



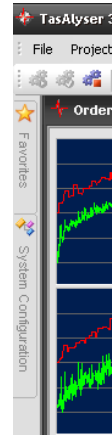
# Further TasAlyser Functions

Various functions of the TasAlyser program that are occasionally required in normal operations, are described in this chapter.

## Configuration, Favorites and Windows

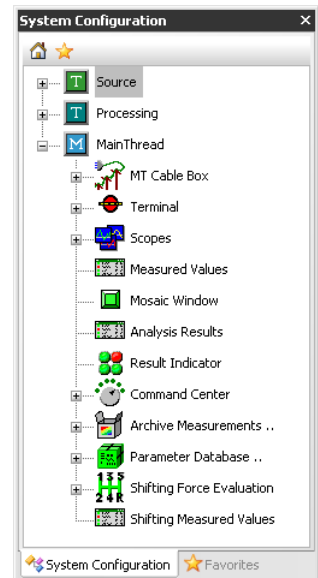
Both of these docking windows were introduced briefly in the chapter “The TasAlyser Program” on page 41. In this section, we go into the functions and applications in more depth

As has already been suggested on various occasions, the TasAlyser program consists of a large number of individual software modules, which do different tasks for the measurement process. Depending on the functions necessary for a particular measurement project, the required modules are combined to a *configuration*. Some modules, such as e.g. the control center or the learning control, are always present. Other modules are present depending upon the requirements - hence there can be, for example, several rotational speed modules, a sensor channel averaging or others.




All the modules together form the *system configuration* and are represented as a tree in the appropriate window. The basis knots of the tree are **Source**, **Processing** and **Evaluation** (in some projects also **MainThread**). In **Source** you will find, among other things, the modules for the TAS Box control, for recording and playback of Wave files and for the rotationally synchronous re-organization of the signals. In **Processing** the processing chains, described in the section “Analysis Steps” (page 34), are calculated for all sensors and rotors. Under **Evaluation** you will find the modules for display, evaluation, test bench communication etc.

The operation of the system configuration is very simple: You pull out the docking window (e.g. by clicking on the tab), fold out the tree knots (by clicking on the + boxes) and locate the desired module. A double-click on the module’s name or icon opens the module’s control dialog (if there is one) or the corresponding display window. Some modules possess auxiliary functions. A right-click on the module in the system will call up a context menu, in which the auxiliary functions will appear, if available.



## Favorites

There are, of course, modules which you need more often, and others which you hardly ever visit. Since it can be quite tedious to locate a specific module in the configuration tree, you can setup *favorites*, which are simply a collection of the modules which you expect to call up most often. You can add any module from the system configuration to favorites by selecting the module and pressing the  button in the toolbar over the system tree (or by calling up the appropriate command from the module's context menu (right-click)).

In the docking window, the list of modules which you have selected as favorites is accessed under **Favorites**. Using the buttons in the toolbar at the top of the favorite window you can re-sort the list and also remove modules:



In other respects, the operation is the same as with the system configuration: you double-click on a module to open its window and right-click to call up the context menu.

## Window Layout

The TasAlyser is able to display a large number of measurement values and curves, from rotational speeds to time signals of the sensors, order and fixed frequency spectra, up to tables of results and evaluations. This leads to an almost unmanageable multiplicity of windows.

Sometimes it is difficult to find a specific window again. At first, you can call again the command to open the window (usually double-click on an entry in favorites). The window in question is then brought into the foreground.


Furthermore, you can open the **Window** menu in the TasAlyser. Here all the open windows (scopes, table window etc.) are listed automatically. The selection of an entry in that menu brings that window into the foreground. However, control windows (such as, e.g., the control center) and also display tools for reference variables do not appear in this menu.

The docking windows, **Favorites**, **System Configuration** and **Output** do not appear in the **Window** menu either. These are found in the **Toolbar and Docking Windows** submenu of the **View** menu.

## Fixing Window Positions

After you have worked a while with the TasAlyser, you will have arranged a preferred screen view for yourself. Store this view by clicking on the **Save**



button  in the toolbar. Then go into the **View** menu and activate the **Fix Window Position** function.

If this function is switched on, you can open, close and shift windows while the program is running but when you terminate and re-start the TasAlyser, all the windows will appear again in the same position as they were when you pressed the **Save** button.

If you press the **Save** button again, then the current screen view is fixed and restored with the next program start.

Note: If the current authorization level is set at normal user (see “User Rights and Authorization Levels ” on page 52), then any changes of the window positions are *never* stored (not even when the **Save** button is used)!

### Window Position Favorites

You can also store several preferred screen views and switch between them - for instance, a view for normal test operation and one for the investigation of special noises in manual operation. The appropriate function is also found in the **View** menu as **Window Position Favorites**.

All the favorites which have been created so far are listed in the corresponding control window - this is at least one for the current screen view. Select an entry from the list and press the **Switch** button to change the screen view. (You can also double-click on the list entry.)

To create a new favorite, proceed as follows: position all the windows as required, call up the window position favorites, enter a name for the new favorite under **Administration**, and press **Create New**. In order to get rid of an existing favorite, select it in the list and press **Erase**. (You cannot erase the current active favorite; you must first switch to another favorite.)

If you open the control window, the current active favorite is always selected automatically. If you re-arrange the windows and press the **Save** button, this changed view is stored and the current favorite is also changed.

### Printing

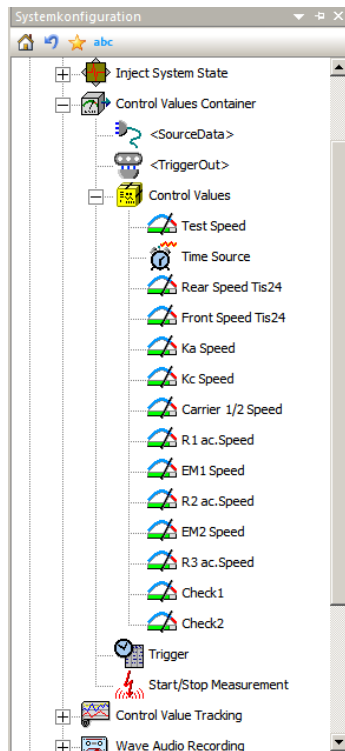
Most of the TasAlyser’s windows - in particular, the scopes and the report window with error messages - can be printed. First, bring the relevant window into the foreground (e.g. by clicking the title bar). Then go into the **File** menu (as is usual in Windows) and select **Print**. Using the **Print Preview** menu command you can get a preview of the printed result first.



## Speed and other Command Variables

A command variable is a measured signal which can control or reference measurements. For example, if you run a speed ramp between 1000 rpm and 4000 rpm, the speed is a command variable. There are some frequent command variables: Speed, torque and time. Sometimes, the temperature also appears as command variable. However, this signal usually does not control or reference measurements but is only recorded. Other command variables are force and positions which occur with shift force evaluation.

All command variables are calculated at a central point in the measurement program: The Control Values Container. You find it in the system configuration in the section **Source**.



Usually, the most important control values can also be found in the favorites window.

If you double-click one of the control value entries in the favorites or in the system configuration, the display instrument for that control value is being opened or brought into foreground. These display instruments were already described on page 45. Please refer to that chapter concerning issues of appearance and usage.

In this chapter, the settings for the generation of the control values are described.

If you want to get to the settings for the generation of control values, you have to double-click with the left button on the display instrument. (Alternatively, you can right-click on the entry in the system configuration or in the favorites and select

Options from the context menu.)

### Simple Control Values

The „simple control values“ are those control values where the value corresponds directly to the voltage on the input connector and therefore can easily be calculated by rescaling of the voltage value using the calibration factor. Examples for such control values usually are torque, force, position or temperature. For these control values, the input connector on the Tas-Box is set to DC (see also Configuring the TAS Box on page 134).



On the other hand, speeds usually are not simple control values, since they have to be calculated from the frequency of a pulse signal.

The settings dialog for a simple control value is as simple:

Mainly, you select the sensor signal being the source for the value of this control value. (Usually, the signal and the module calculating the control value have the same name.)

Please note, that the name of the signal cannot be displayed just after the start of the TasAlyser. You must have started a test run once (have done an “Insert”). Then the TasAlyser knows the names of the sensor channels.

Moreover, you can set the time resolution for the calculation. In the figure above, a time resolution of 20 milliseconds has been set. That means, that the measured input values of for 20 milliseconds are averaged to one torque value on the output. If you have a signal which changes fastly, this may influence the result, but if you have a signal which is distorted by another high frequency signal this can be useful.

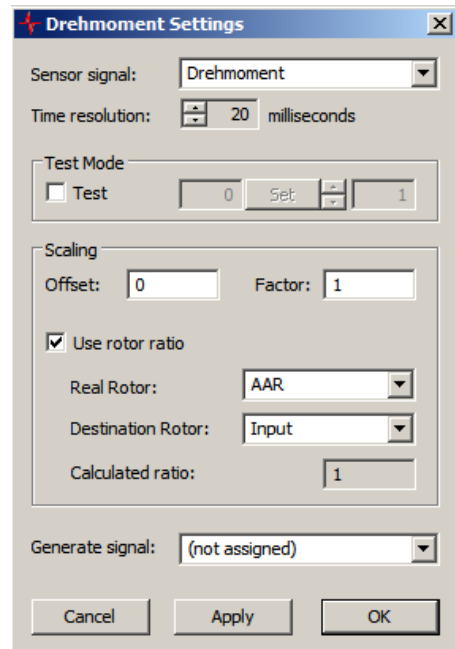
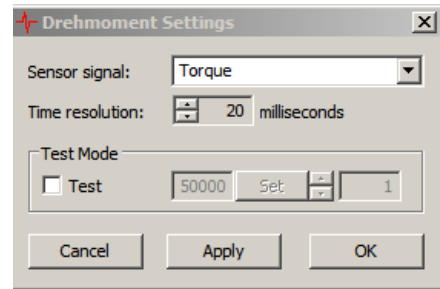
If you enable test mode, you get a fixed output value instead of a value corresponding to the input value. You can enter a desired value and click the **Set** button, or you can use the arrow buttons to change the value in steps corresponding to the value in the field on the right.

## Scaled Control Values

A variant of the simple control values are the „scaled control values“. Their settings dialog contains additional settings:

You can set the scaling directly (using an offset value and a factor, both can also be negative) or you can use the transmission ratio between two rotors as scaling factor. Usually, this ratio depends on the test step and is displayed in the field **Calculated ratio**.

If **Generate signal** shows (**not assigned**), the scaled value replaces the original control value (set at the



top of the settings dialog). Otherwise, the scaled value is sent as the entered signal.

### **Time as Control Value**

Often, the time is used as control value to start or stop measurements (like measure for 10 seconds and record a value each 0,05 seconds). For this control value, no settings dialog is necessary. The measurement program calculates the time according to the A/D data stream. If the base sampling frequency is set to 100kHz, counting 500.000 sample values correspond to 5 seconds.

## **Speeds**

Most often, speeds are calculated from pulsed signals where the pulse frequency correspond to the current speed. Typically, such a pulse sensor is connected to a certain shaft and generates a certain number of pulses per 1 revolution of the shaft. Since there are many different pulse sensors available having different characteristics, the measurement program allows different settings for the calculation of the desired speed.

### **Speed Capturing Using A TIS Module**

The easiest way to connect a pulsed speed signal is to connect to a TAD96-channel. But this connections has certain limits.

The speed detector needs 4 values per pulse to safely detect the speed value. If you have a base sampling frequency of 100kHz this results in a maximum pulse rate of 25kHz. If the pulse sensor gives 100 pulses per revolution, this results to a maximum revolution frequency of 250 Hz which corresponds to 15.000 revolutions per minute. This range is suitable for most measurement situations. On the other hand, if the speed sensor gives 1000 pulses per revolution and the base sampling frequency is only set to 50kHz, then the maximum speed is only 750rpm. In this situation, you have to use a TIS-module in the Tas-Box to get a suitable speed range.

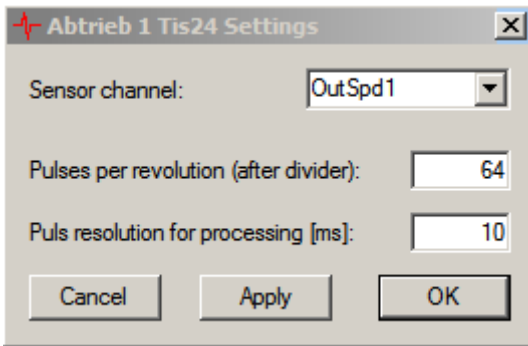
The sampling frequency of the TIS-module is independant from the base sampling frequency. A maximum pulse frequency of 10Mhz is possible resulting in about 300.000rpm with a pulse frequency of 2000 Pulses per revolution... A more than acceptable range...

Moreover, the TIS module can capture up to four speed signals. (The standard TAD96 A/D-module only has two inputs.) The disadvantage of the TIS module is that it only works well for “clean” signals like TTL signals. In contrast to that, the standard speed detector module can even deal with pulse gaps and varying pulse lengths.

Therefore, the settings dialog only offers to set the number of pulses per revolutions.

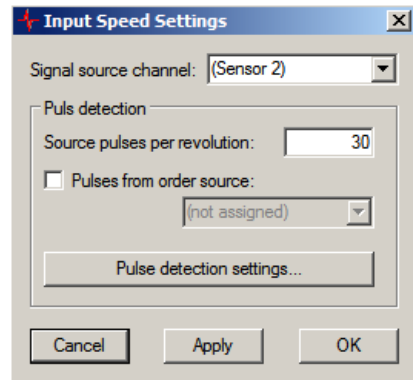


In addition to that, the time resolution can be set analog to the simple control values.

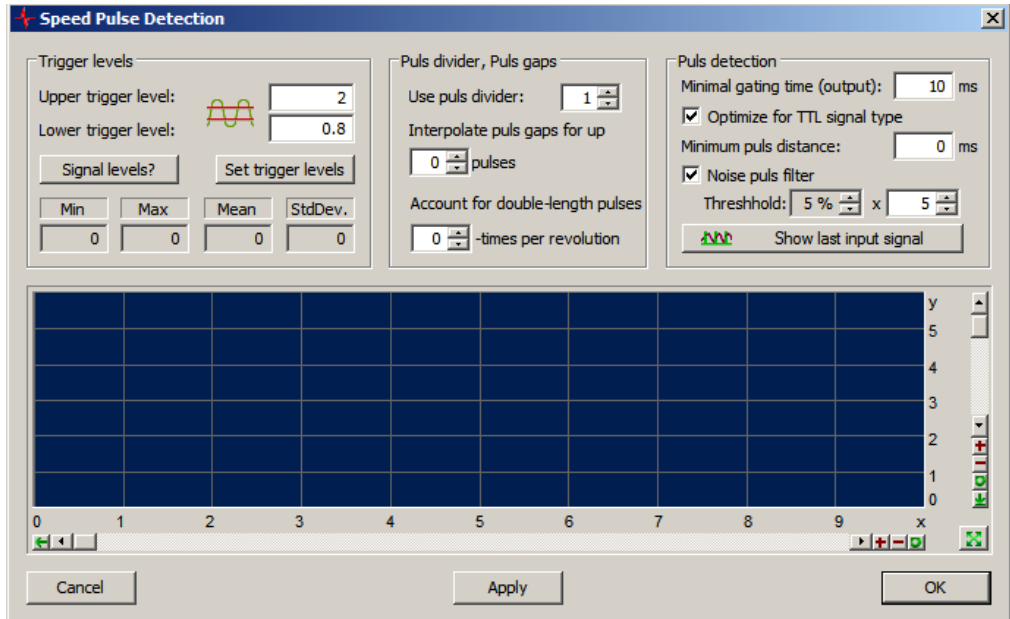


### The Standard-Speed Module

For the standard speed module, you also have to set the number of pulses per revolutions. This module offers the option to set this number depending on the current type by enabling **Pulses from order source** and selecting an appropriate order source. The base order of that order source (teeth number for gear wheels) is then used as number of pulses for the speed calculation. Of course, the parameter database must provide this information.



Please click on the button **Pulse detection settings** to open the settings dialog for pulse detection:



Basically, the pulse detection works like a Schmitt-Trigger. You set an upper trigger level and a lower trigger level. A pulse is being detected when the signal has passed the upper trigger level. Then, the signal has to fall below the lower trigger level and rise again above the upper trigger level for the detector to detect the next pulse.

The trigger levels are being set in the upper left corner of the dialog. The buttons **Signal levels?** and **Set trigger levels** can help you finding appropriate settings. If you want to use this help, please verify that you currently receive a speed signal (meaning that you started a test run and the speed shaft is turning). Then press **Signal levels?** In the fields below the buttons, the properties of the received signal are being displayed. In addition to that, you can use the button **Show last input signal** on the right side to see the signal of one data block displayed in the scope view.

You can then decide to manually set trigger levels according to the value interval displayed or you click **Set trigger levels** and then **Apply**. That way, the trigger limits are set automatically. The result is being displayed in the dialog. Now the speed instrument display should show the correct speed.

When you click **Show last input signal** again, you will see the original signal in magenta and the detected pulses indicated in green. The lower line of the green pulse curve is at the lower trigger level, the peaks reach to the upper trigger level and indicate when a speed pulse has been detected after applying of pulse divider and other settings.



Sometimes it happens that the signal level of the speed sensor depends on the frequency. Then you should try the procedure described above at different speeds until you found levels which work for every speed. Sometimes it also helps to change the input settings of the TAS Box (see Configuring the TAS Box on page 134) from DC to AC (or vice versa).

The area **Pulse detection** on the upper right allows to fine tune the speed detection. Changes of the settings are only necessary for very bad speed signals.

In the center area, you make the settings for pulse gap or pulse length correction. Sometimes, speed sensors miss one or more pulses to identify the 0° position. For example, if you get a pulse per 6° revolution (corresponding to 60 pulses per revolution) but have one pulse missing, you have to set 60 as pulse number per revolution and **Interpolate pulse gaps for up 1 pulses**.

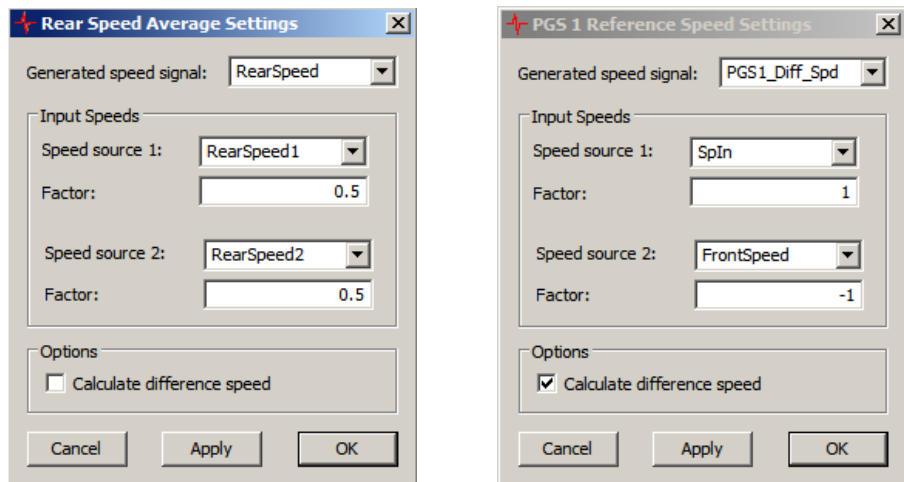
Likewise set **Account for double-length pulses** if one pulse per revolution has double length (instead of a gap).

If your pulses do not have the same distance length (but not a gap or a double-length pulse), you can correct this by using a **divider**. For example, if you get 4 pulses per revolution at 0°, 85° (instead of 90°), 185° (instead of 180°) and 270°, the speed detector calculates varying speed (for each revolution) although the original speed is constant. The detector will calculate more exact speed if you set a divider of 4 and reduce the number of pulses per revolution to 1.

## Calculated Speeds

Not every speed in a transmission is available for direct measuring, for example, the output speed. The available speeds correspond to the right or left wheel, but you cannot measure the output speed inside the transmission, so to say: on the other side of the differential. But you can calculate this speed by averaging the available output speeds.

For these situations, you have speed detector modules which have the ability to calculate a speed using other speed signals and/or rotors.



The figure above on left shows the situation of calculating the speed on the other side of a differential “Rear Speed” using the real speeds RearSpeed 1 and Rear Speed 2. Instead of real speeds, rotors or other speeds can used (even if they also were calculated). The measurement program reads out the information about the rotors from the parameter database and calculates the current rotor speed accordingly.

The figure on the right shows the calculation of a signal „PGS1\_Diff\_Speed“ later used as reference for the gear mesh frequencies of a planetary gear set. The gear mesh frequency of a planetary gear set usually depends on the difference speed of two components, determining the speed of the third (e.g. difference of ring and carrier determine sun speed). FrontSpeed again is a real speed but SpIn is a rotor calculated using RearSpeed. In the situation above, FrontSpeed drives the Ring of a planetary gear set, SpIn drives the carrier of that gear set.

Since the shaft speed of the sun is also needed, another speed calculator, not shown calculates it using rotors referenced to RearSpeed and PGS1\_Ref\_Speed...

Sometimes, calculated speeds are used to change the speed reference for certain measurements. If you want to specify measurements according to output speed, you need a speed signal output speed. If you only have input speed available, you can calculate output speed using the kinematics of the transmission.

## Trigger

The purpose of a trigger is to check control values (speed, torque, etc.) and add marks to the signal flow on certain circumstances. Such a circumstance usually is the passing of a certain value. The marks then can be used to

initiate other actions like starting or stopping a measurement or the recording of a point for a tracking curve.

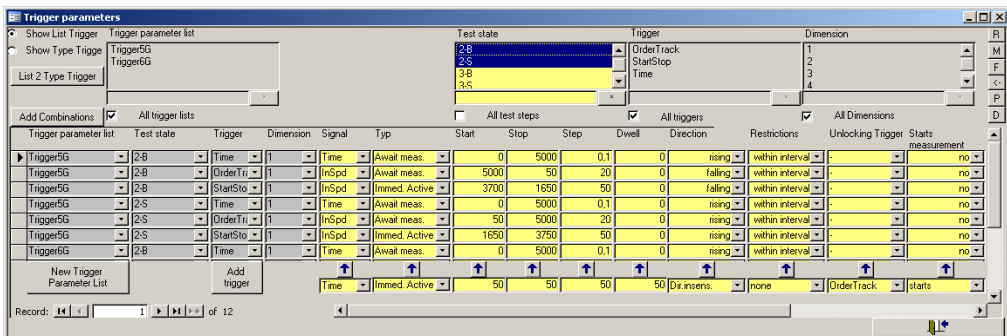
## Where Trigger are used

The most frequent use of a trigger is for recording tracking curves depending on a control value, typically an order track or spectrogram relative to input speed. With reference to time, control values (like speed) are tracked during measurement to check the measurement conditions. In both situations, you must specify a trigger to get results.

In case the test stand does not send these commands, trigger are also used to start and stop measurements. If in doubt, the use of a trigger is preferable because the noise analysis usually checks control values with finer resolution than the test stand. Moreover, you do not need to take the command sending and processing time into consideration (which also may not be exactly the same at all times).

## Trigger setup

The trigger parameters are set in the corresponding dialog in the parameter database section **Trigger Parameters**.



## Trigger-Fields

By specifying **start**, **stop** and **step**, a segmented control value interval is being defined. Generally, when the control value enters the interval, a trigger is being started. When a segment boundary is passed, a mark is being sent, and leaving the interval stops the trigger. We call the interval *field* and the segments *cells*.

If the difference between **start** and **stop** is not dividable through the **step** size, the higher value is being rounded up.

## Directions

The behavior of control values that leads to trigger events can be specified more detailedly with further settings. If a certain **direction** is specified, the





trigger has to reach the interval following the specified direction in order to start the trigger. The same is true for the segment boundaries. Although the trigger can skip segments on sudden changes of the control value, if the control value leaves the segment in the wrong direction this does not lead to another trigger mark. Instead, the trigger waits until the control value enters another segment (relative to the last trigger point) in the right direction before sending the next mark.

## Restrictions

The behaviour described above belongs to a trigger having the **restriction “within interval”**. If this restriction is changed to **none**, the trigger marks each change of the control value in the right direction. Basis for calculation of the segments are **start** and **stop** value as usual, although they are ignored as such.

## Dwell time

Under certain circumstances, for example if the control value is not stable, it can be useful to specify a minimum **dwell** for a trigger cell. This is being seen in units of data blocks. The trigger counts the number of data blocks for which the value of the control value remains inside the cell. When the **dwell** time has been reached, the mark is being sent. If the control value leaves the cell (in whatever direction), no mark is being sent and the counter resetted. The cell is then considered as not reached.

Please note that you create a time shift when using a dwell time greater than zero. Moreover, some calculations may behave differently. Crest, Rms, etc calculate values on the basis of the predecessor signal blocks when using a dwell of zero. If you use a higher dwell, the calculation is restarted when entering a trigger cell. In this case, the calculation includes only those blocks where the control value was inside the corresponding cell.

## More Than One Control Value

Although the trigger usually follows on control value, it is able to follow four of them. The trigger field then gets multidimensional and each **dimension** is set up in the database separately. The entries **typ**, **dwell**, **unlocking trigger** and **starts measurement** which are common for all dimensions are specified with the entries in the first dimension.

## Trigger-Types

Basically, triggers have to be „unlocked“. Trigger starting measurements are **immediately active**, that means they follow their control values and send marks on corresponding events without the need of further activation.

On the other hand, the **Await measurement** type needs further activation. Its control values can have every possible value, it does not send marks before a measurement has been started (manually, by test stand or by another trigger).

The third type is the **Follower** trigger. This type is unlocked when the **unlocking trigger** has reached its end condition. With this trigger, trigger cascades can be built.

The last type is the **continuous** type. This type is used to record *each* measured value, disregarding trigger points. Therefore, this is just formally a trigger. Continuous triggers are basically working for spectrograms and have the following restrictions:

1. They always run between start and end measurement
2. They start always at ,0'
3. **Stop**, **Step** and **Direction** must specify a rising ramp
4. The control value for the trigger must be a *speed* signal for measurements in synchronous channels (or mix). Fixed frequency channels must use the signal *time*.
5. For trigger using time, the segments should correspond with the time resolution of the signal flow. Otherwise either data is being lost (too large segments) or you do not have data for each segment (too fine segments).

### Trigger Using Signal Time

A trigger using time determines that time using the sampling rate in the signal flow. The 0-position is the point when the trigger is being unlocked. A **immediately active** trigger has the 0-point when the mode starts, trigger of the type **Await measurement** have their 0-point when the measurement starts. Finally, **Follower** have their 0-point when their unlocking trigger ends. 0 is a valid start time.

### Further Influence of Trigger Settings On Tracking Curves

As already mentioned, trigger control when the recording of a tracking curve starts and stops. Furthermore, the passing of segment boundaries issues a mark in the signal flow. For tracking curves, this mark means that a data point shall be captured. Since these segments are mainly dependant on the step size, the step size defines when a data point shall be captured. In other words: The step size defines "how fine" a tracking curve shall be recorded.

Furthermore, the trigger also defines the x-axis for this tracking curve. If a tracking curve is recorded using a time trigger, the curve has the time as x-axis. If a tracking curve is recorded using input speed, the x-axis of that curve is the input speed. Therefore, if you want to record a curve using output speed instead of input speed, you have to use a trigger in the

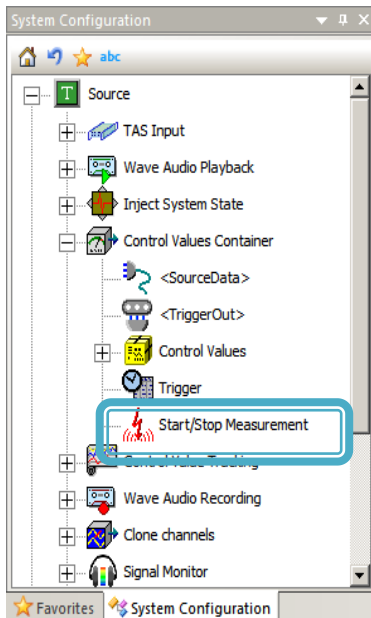


measurement definition which uses output speed. It is not possible to rescale the x-axis afterwards.

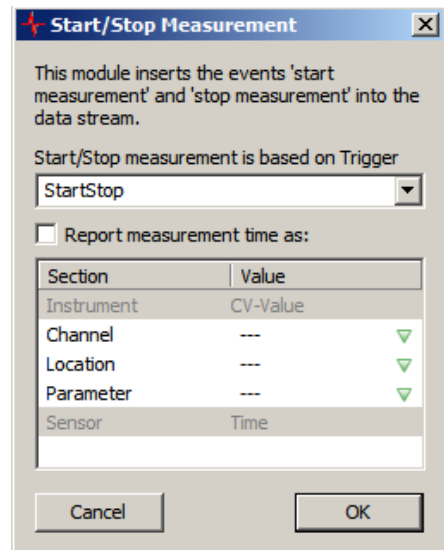
Whether the values of the single points differ or not, does not depend on the trigger. If a trigger is set to record a spectral point each 20ms but the spectrum changes only after 200ms (because of resampling settings/ speed), the track will contain 10 points with the same value but different positions on the x-axis. In other words: Recording data with a certain step size does not necessarily mean that you get more detailed information about the noise behaviour. Sometimes you have to adapt other settings also to get this effect.

## Start and Stop Measurement Using Trigger

A measurement can be started using a trigger with the type **immediately active** (and *only* using this type). On the other hand, every trigger can end a measurement.



In the system configuration (see figure on the left) you find the CStartStopMeasmt-Module. Because of historical reasons, you can select a trigger in its combobox which shall control the measurement:



When a trigger has been specified in this module, all other settings from the parameter database are ignored. If no trigger has been specified in this module, the database settings are valid. In the database, you can set for each trigger whether it shall start or stop the measurement (or both). Since you can specify more than one trigger to start or stop the measurement, the rule is: First come, first served. This is also true if the measurement is also controlled otherwise (test stand, manually, etc.).



## How Exact Are Trigger Marks

Generally, trigger work exact for each sample. The marks follow the signal block that contain a control value which issued a trigger event and describe the event and the corresponding position within the block. If more than one event occurs within a data block, a mark is added for each event. Please note that the control values themselves have limited exactness. Furthermore, the trigger does not calculate values in between but marks the cell of the trigger field which belongs to the event. From these cells, the value of the control value is reconstructed later. In other words, the exactness of these values rely on the size of the trigger cells.

Those modules which calculate tracks using trigger events, usually do their job per block. That means, that the exact position of the trigger event is usually not being used.

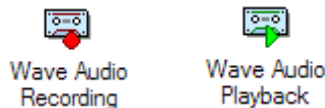
Furthermore, the trigger module does its job before resampling. During resampling, all trigger marks which belong to a resampled data block are added after that block.

In addition to that, most modules ignore trigger marks. That means that they add the triggermarks which they received after the input blocks to their output blocks. This is also true for modules with “memory” like the averaging module.

---

## Wave Audio Recording and Playback

The TasAlyser can record complete test cycles or concatenated cuts as Wave files. Control commands, such as test step selection, are embedded in the Wave file, so that later, during playback with the TasAlyser, not only is the noise signal reproduced, but the complete test cycle is repeated. The modules for recording (*Recorder*) and playback (*Player*) are found in the favorites (or in **Source** in the system configuration):



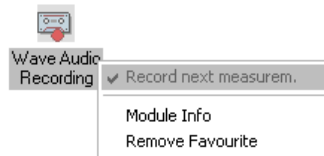
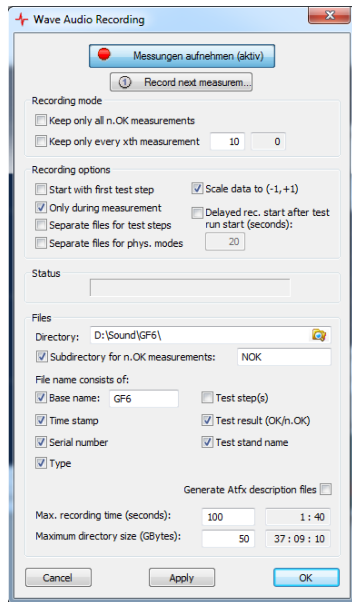
### Wave Recorder

The Wave recorder writes the complete signal current of all sensors and channels into a file in WAVE format. In addition, information about the channels, such as calibration data, is filed in the header data (the meta data) so that, when read with the Wave player, the signal current is restored exactly the same as when it originally came from the TAS Box. Read the

section “Audacity” and “TasWavEdoitor” for more information on the Wave files.

The Wave recorder can produce a recording either for each measurement, for every  $n$ th measurement, for only n.O.K. measurements, or for only the current/next measurement. However, test cycles which have been cancelled or in which no measurement has been carried out, are either not recorded or the recordings are erased automatically.

Open the Wave recorder dialog for the setup:



Switch on the **Rec** button in order to activate the recording of Wave files. Below, in recording mode, you can specify that not all measurements are to be kept, but only n.O.K. measurements, or only every  $n$ th measurement as sample. (You can combine both options.) If you switch off the **Rec** button during a recording, the current recording is being terminated immediately.

If you set a checkmark at **Record Next Measurement**, only the subsequent test cycle is recorded. If you checkmark during the current test cycle, only the remainder of that test cycle is recorded. The alternative to setting the checkmark in the dialog is to call up the context menu of the module under the favorites (using a right-click) and activate the **Record Next Measurement** function (graphic above).

Furthermore, you can activate different **Recording Options**.

- Activate **Start With First Test Step** if your test cycles have a long lead-in time before the actual noise analysis. Otherwise, recording begins when the test bench signals the start of the test cycle (with *Insert*; see “The Test Cycle” on page 20).
- **Only During Measurement** is to be recommended if there is a long pause between the ramps. **Separate Files for Test Steps** is used if the individual ramps and/or test steps are very long, i.e., recording the entire test cycle would result in an unmanageably long file.



- The option **Scale Data To +/-1** must be used if you want to examine the produced data later using the Audacity program (see next but one section).
- As with the option **Start With the First Test Step**, you can activate the option **Delayed Rec....** to suppress the recording of uninteresting parts at the beginning of a test cycle.

Depending on the situation different combinations of the options are appropriate. **Start With the First Test Step** and **Keep only n.O.K. Measurements** are the usual options for normal measurement series. For measurements with a mobile system in a vehicle on the road it is usual to activate **Only During Measurement** and **Separate Files for Test Steps** and activate the **Test Steps** option for the construction of the file name.

In the lower dialog area under files, select the directory in which the Wave files are to be stored, as well as elements for the construction of the file name. If **Generate Atfx Description Files** is activated, a file with the same name in Asam Atf format is produced for each Wave-file, which describes the contents of the Wave file. This allows the data (including channel description etc.) to be imported into Asam compatible programs.

Below the options for the construction of the file name you can enter a maximum recording time. This time is restricted by the maximum possible wave file size in Windows (1 gigabytes). This may seem a lot but you must remember that the Wave recorder stores raw data (and not MP3); with a 100 kHz scanning rate that is approximately 400 MB per channel per second! If you want to get the max. possible recording time, you can also enter a 0.

### Maximum Directory Size

Finally, you can specify a maximum directory size. The Wave recorder ensures that the Wave files, corresponding to the name pattern currently in use, do not, in total, exceed the indicated gigabytes. If necessary, the Wave recorder will erase the oldest files, if the directory becomes too large. Adjacent to the input field for the gigabytes is the approximate recording duration (hours, minutes, seconds) which corresponds to the fixed directory size. For this calculation, the Wave recorder also takes the number of sensor channels and the base scanning rate into account.

If you activate the option **Subdirectory for n.ok measurements** and give a name for this subdirectory, all n.ok measurements are stored in this directory. The maximum directory size is checked for this folder separately, meaning that the subdirectory can not get larger than specified. The superdirectory, however, can than get twice the size (once for the subfolder, once for the folder itself).

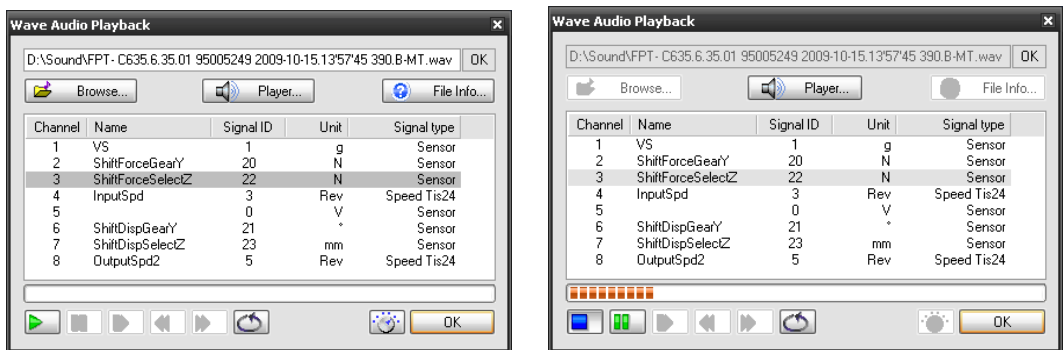
Furthermore, the maximum directory size in each case is checked not before the end of the recording. During recording, therefore, the sum of all the files can exceed the maximum size. Only when recording has finished, old files are erased until the total volume is again smaller than the delimitation.

Note in addition that, under Windows, there is not only a restriction on the maximum size of a file (resulting in a maximum recording time for each recording), but also a maximum in the permitted number of files in a directory. If you create a very large directory in which many short recordings are stored, then it could be that you will reach the upper limit.

## Wave Player

The Wave player complements the Wave recorder: it plays back the recorded Wave files and, at the same time, reproduces (if activated) recorded test cycle events, such as a test step change.

At the top of the Wave player window (see the figure below) you can select a file for playback. The **Player** button leads you directly to the audio monitor (see “Audio Monitoring” on page 140), because you can also listen to the noises over the loudspeakers during the Wave playback, exactly the same as with a true measurement. **File info.** opens a window, in which some basic characteristics of the file are given, such as, duration.



The channels which are found in the Wave file are listed in the center of the window. This list corresponds to what was attached to the TAS Box on recording. It is possible to change the signal IDs and the units in the list (by direct editing of the table fields). This is not necessary for playing back normal recordings, but is used, for example, if you want to play a recording in another TasAlyser project, which is parameterized for other sensors.

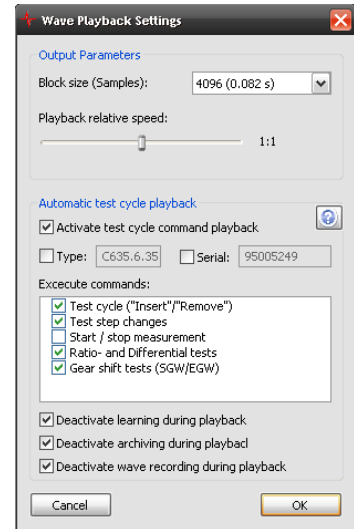
The lower part of the window contains the playback controls. These correspond to those of a typical audio player. The button for “single step”, “return” and “forward” are only available if you press “pause/break”.



The adjustment knob next to **OK** brings you to the dialog for activating the test cycle reproduction:

At the top of the window you can use the slide control to adjust the playback speed. In the middle position “1 : 1” the recording is played at normal speed. Push the control to the right to slow down playback (up to 1/4 normal speed) and to the left to increase the speed (up to 4 x normal speed). If you push the control completely to the left (“max”) then the full processing power of the computer is used to calculate the data as fast as possible. (If you activate the audio monitoring at the same time as the Wave playback then playback will automatically be at normal speed.)

In the lower part of the dialog you can control the automatic procedure, i.e. the reproduction of the test cycle. Here you can switch playback on and off. You can also select whether the aggregate type with its original serial number as stored in the Wave file, is to be used, or (by setting the checkmarks and changing the values) another type or another serial number should apply.



In the list you can specify whether or not different kinds of test cycle events are to be reproduced. Usually you do not want “Start/stop Measurement”, for example, to be reproduced because the measurement is controlled by the reference variable in Wave playback as well (c.f. “The Test Cycle” on page 20).

For the rest of the measurement program, the Wave file playback cannot be differentiated from an actual measurement - even if playback is performed at “maximum speed”. To prevent “canned” playback from influencing the learned limit or being archived like new measurements and then added to the result database, these functions are normally suppressed artificially during Wave playback. Nevertheless, sometimes you do want to get an archive file or learn limits. For this reason, you can re-activate these deactivated functions.

## Audacity

The program *Audacity* is a free audio processing program. It is installed as standard on TAS measurement computers and can be downloaded under [www.audacity.de](http://www.audacity.de).







Audacity can open and process multi-channel Wave files which have been created by the TasAlyser. You can look at the signal tracks of all channels, listen to individual channels and, if necessary, also modify the signal (such as by filtering). Audacity is, however, not able to evaluate the channel and test cycle information which the TasAlyser has additionally stored in the Wave files. This information is ignored. If you modify and then store a Wave file with Audacity, then such information is (unfortunately) lost.

The Wave files created by TasAlyser have as many channels as there are sensors connected and activated. The data are stored as 32 Bit floating-point numbers (“IEEE float”). The Windows Media Player, for example, cannot do anything with these data (although the Wave standard permits such files). Audacity can open these files, but only if the floating point values are scaled in the range [-1, +1]. This is the reason for the corresponding Wave recorder option.

There are other audio programs which can also open TasAlyser Wave files, such as, for example, *Audition*, a commercial product from Adobe. This program can also cope with general floating-point values. If you use Audition or a similar program, you should deactivate the +/-1 option because then you can read off the actual signal values (around 0.05 g) directly from the graph of the audio signal.

If your TAS Box contains a TIS rotational speed module, then its data will be written in raw format in the Wave file as two channels. These data contain binary coded rotational speed information and cannot, therefore, be analyzed properly with any audio processing program.

---

## TasWavEditor

The program TasWavEditor is not an editor for the noise data itself (like Audacity is) but for the meta information which the TasAlyser also stores in the wave files.



In the folder “Rotas for Experts” on the desktop you can find a link to the TasWavEditor program. Another way to start this program is to right-click in the windows explorer on a wave file and select the entry **Open with** from the context menu. If you do this for the first time, you must then search the TasWavEditor.exe file which you will find in the c:\Program Files (x86)\Discom\bin folder. Windows remembers this choice and shows it automatically the next time you use the **Open with** command to open a wave file.

In the main section of the TasWavEditor program (see figure below), you see a display of the time signals of all sensor channels. For speed channels, the decoded speed signal is shown in the same panel as the raw data. (For TIS channels and CAN data channels, only the decoded speed is shown.)



There are three docking windows on the left (stacked above each other):

**General Information** shows the length of the recording etc. together with information about the test subject (type, serial number and so on). You can change the test subject information by selecting a value and editing the text.

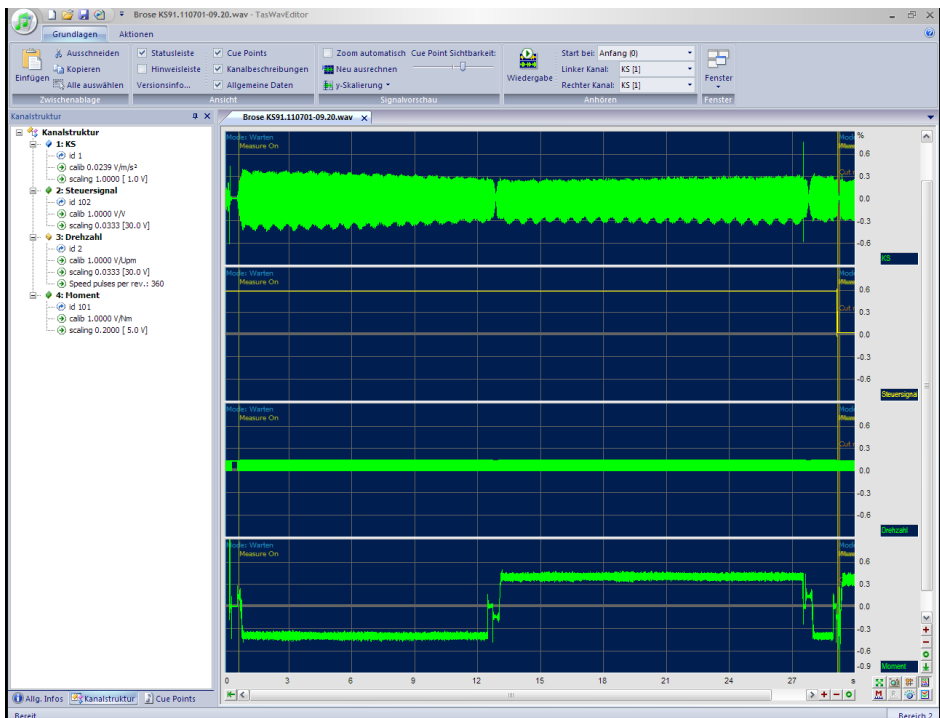
The **Channel Structure** tree shows all channels present in the wave file together with information on each channel like the sensor name, calibration factors and more.

y shows the list of events during the recording. You can edit a cue point by double-clicking on it in the list.

The TasWavEditor uses the scope window to display the sensor signals. You already know the scope window from the measurement program (see chapter Scopes on page 44). With the control buttons at the end of the scroll bars you can change the current section and zoom into the signal that way (the x-axis shows the time).

If you zoom into a small time section, you see two curves for each sensor: the current preview and an overview curve which is used when you use the x-scroll bar.

The “cue points”, the events of the test run, are marked in the time signals.

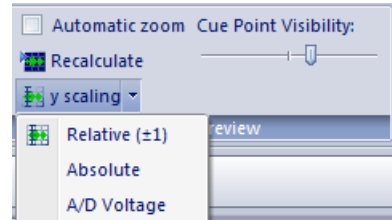




Please note: If the wave file has been recorded with the option **only during measurement** active in the measurement program then you will not see a continuous time signal but concatenated sections of the measurement time. (The x-axis then refers to the time in the available data, not the time of the test run!).

You can select different scaling options for the y-axis (in the corresponding area of the multi function bar):

**Relative** means that all signals are scaled to their own full scale. This is the same scaling that you see in Audacity.



**Absolute** shows the original units for each signal. This means, Structure bourne noise signals will show g or  $m/s^2$ , torque signals will show Nm, speed signals usually show Volt.

**A/D Voltage** shows the original values of the sensors in Volt. With this setting you can control the gain settings, if necessary.

## Speed Signals

The TasWavEditor has the ability to decode speed signals, analogue ones and TIS channels. The speed curve for analogue signals is displayed in the scope in the same color as the pulse signal. TIS-speeds are all collected in a scope pane of their own. All speed curves have the sign  $\omega$ .

TasWavEditor needs to know the correct number of pulses per revolution to be able to calculate the correct speeds. This number is being displayed in the channel description. If need be, you can also correct it there.

Depending on the selected y-scale, the speed curves are scaled as well. For **absolute** y-scale, you can read the correct speed values at the curve. (For example: If the speed  $\omega$  curve is at 6000, this means that the speed has 6000 rpm). If the y-scale is **relative**, the speed is being divided by 10000 in order to be in the area between +1 and -1. The speed of 6000rpm then corresponds to a curve value of 0.6. If you select **A/D-voltage** the speed is being divided by 1000 in order to get a result in the usual range of A/D voltages. Each rescale is being done to be able to compare sensor signal and speed curve.

## Changing Meta Data

You can edit the additional informations of the wave file using the TasWavEditor, for example type or serial number. The corresponding entries are listed in the docking window **Basic data** where they can also be changed. Click in the right column on the entry you want to change and enter the new value.



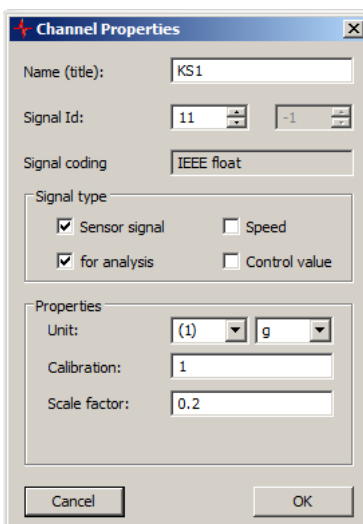
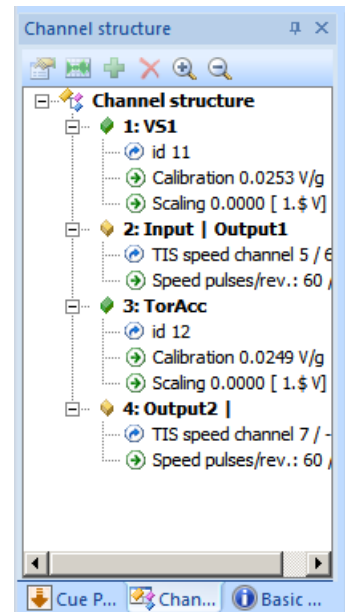
If you load a wave file into the TasWavEditor which has not been recorded by the TasAlyser, this file will not contain additional information. You can convert such a wave file to a “TasAlyser wave file” by adding the missing data manually in the **Basic data** window. Enter typ information, serial number, etc and activate the **Format** check box **Discom Data**. This sets an internal mark which allows the TasAlyser program to identify its own wave files. Next thing is to correct the channel descriptions.

As soon as you change the additional information (or channel information or cue points), a “\*” is being added to the file name in the title bar to indicate that data has being edited. If you close TasWavEditor, you will be asked whether you want to save these changes. You can save changes under the same or a different file name.

## Channel Properties

The docking window **Channel Structure** shows information concerning the channels in the wave file. For each channel you can see the name of the sensor signal, the calibration factor, etc.

The docking window has a tool bar of its own with which you can reach functions for the selected channel. Click on the name of a channel to select it. The second button in the tool bar allows to hide this channel in the scope. The third button adds a copy of the selected channel to the wave file. The fourth button deletes the selected channel. (Both changes only have effect when you save the wave file.) The first button of the tool bar opens the channel properties:



For each channel, you can edit the channel properties like calibration factors, unit, etc. The **Signal Id** must correspond to a signal id number defined in the parameter database of the project where you want to use the wave file for. For TIS speed channels you must specify two ids for the two sub channels.

## Cue Points

“Cue Points” are markers for events in the wave file which describe the test run like



selection of a test step or start/stop measurement.

Cue Points can be edited directly in the docking window where they are displayed. Double-click an entry for editing (e.g. correcting its position). The buttons in the Cue Points' window tool bar allow to create new cue points or delete old ones.

## Export Functions

In der multi function bar **Toolbox** you can find the different export functions.

### Metadata Ex-/Import, Atfx

The button **Metadata Export** allows to export all additional information (channel description, cue points, etc.) in a separate wave file. This export file does not contain sensor data, only additional information. The button **Metadaten Import** then allows to add these additional information to another wave file. The button **Generate Atfx** writes an Atfx description file for the current wave file. (When you changed channel informations, you should first of all save the wave file before you generate an Atfx file.) The atfx file is written in the same folder as the wave file with the same name (only suffix .atfx instead of .wav).

### Channel Export and Cutting

With **Channel Export** you can save selected channels from the current wave file into a new wave file. This function in addition can save decoded speed signals as analog channels in the new wave file. By doing this and generating an Atfx description, you can import TIS speeds into software from other manufacturers.

**Cutting** can export part of the current recording (for example a single test step). It also exports selected channels and can reduce the size of the exported wave file by reducing the sample rate.

The option **Generate standard-(16 Bit) stereo-wave format** is only available when you selected only one or two channels for exporting and these channels do only contain analog signals (that means sensor data, no digitally coded data).

## Listen to the Sound

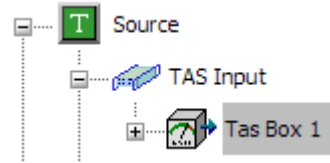
Finally, TasWavEditor allows to play back the sensor signals so you can listen to them Select the channels which shall be played back on your sound card in the multi function bar **Basics**, section **Play back**. Only Cue Points are possible starting points for play back.



# Configuring the TAS Box

Usually the TAS Box will have already been configured at the initial setup, so that you do not need to change anything here. Sometimes, however, you may wish to change the sensor configuration. For example, with the mobile system you sometimes want to make measurements with less than the 4 possible microphones.

Access to the TAS Box can be found at the top of the system configuration in the **Source** section. Expand the **TAS Input** entry and double-click on **TAS Box 1**. (It is technically possible to connect more than one TAS Box to a measurement computer in order to realize applications with a large number of sensors. Then you will find all the existing TAS Boxes listed here.)



Note that a type (any type) must be loaded so that you can make settings for the sensors, since the sensor names and their characteristics are stored in the parameter database. (Press **F5** to load a type manually before you open the TAS settings dialog.)

The settings dialog for the TAS Box has several sections:

**TAS Settings**

A/D Channel Settings | RPM Properties | Base Clock Settings | Special Commands | Firmware versions

Channel	Active	Signal Source	Input	Coupling	Sensitivity	TAS Factor
A.1.1	<input checked="" type="checkbox"/>	VS	Single En...	ICP	1V	1
A.1.2	<input checked="" type="checkbox"/>	InputSpd	Differential	AC	1V	1
A.3.1	<input type="checkbox"/>	.	Single En...	ICP	2V	1
A.3.2	<input type="checkbox"/>		Differential	AC	10V	1
A.4.1	<input checked="" type="checkbox"/>	ShiftForc...	Differential	DC	10V	1
A.4.2	<input checked="" type="checkbox"/>	ShiftDisp...	Differential	DC	10V	1

OK Cancel Apply Help

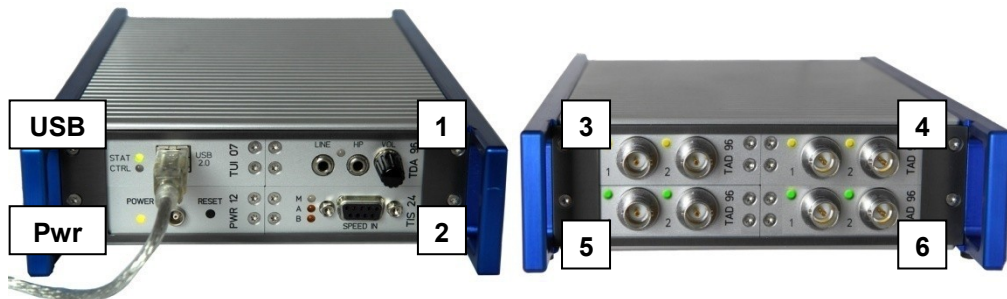
Which sensors are plugged into which of the TAS Box’s ports is specified in the first section **A/D Channel Settings**.

## Identifying the Channels

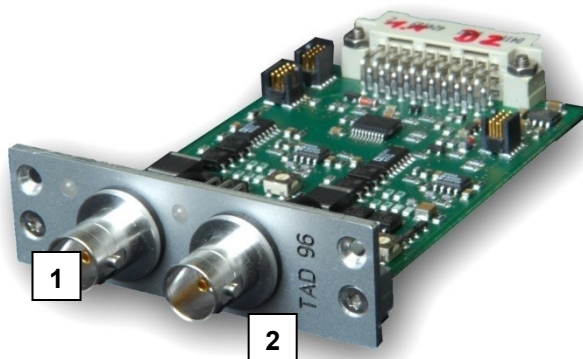
For making settings in this dialog for the right channel, you need to know which connector of the TAS box is meant by “**A3.1**”, for example.



The first number identifies the module in the TAS Box, the second number the channel. The modules are numbered as follows:



Under A/D channel settings only A/D transducer modules appear (no rotational speed board, D/A transducer or similar). Each A/D transducer module has two channels:



The BNC port on the left is channel 1, on the right channel 2. (To help you identify the channels these are labeled 1 and/or 2 underneath the sockets, not visible in the photograph).

In the TAS settings dialog, the name “**A3.1**” simply means “module 3, port 1” i.e. the left port of the module at the rear upper left. (“A” in “A3.1” stands for “TAS Box A”. Only if you use more than one TAS Box, will you get to see “B” and “C” here.)

## Assigning Sensors

In the dialog for the TAS settings on the previous page, you specify which sensor is connected to which port. Just select the appropriate sensor in the corresponding line in the **Source** selection box. You should also enable the **Active** checkmark, if you want to use the sensor. Vice versa, in case of the microphones of the mobile system, this means that you must enable only those microphone channels where you really want to connect a microphone. In the mobile application, the single microphone signals are calculated to a



common signal. If you include a disconnected microphone into this calculation (by leaving the **Active** flag enabled), you would get “white noise” calculated to the common signal.

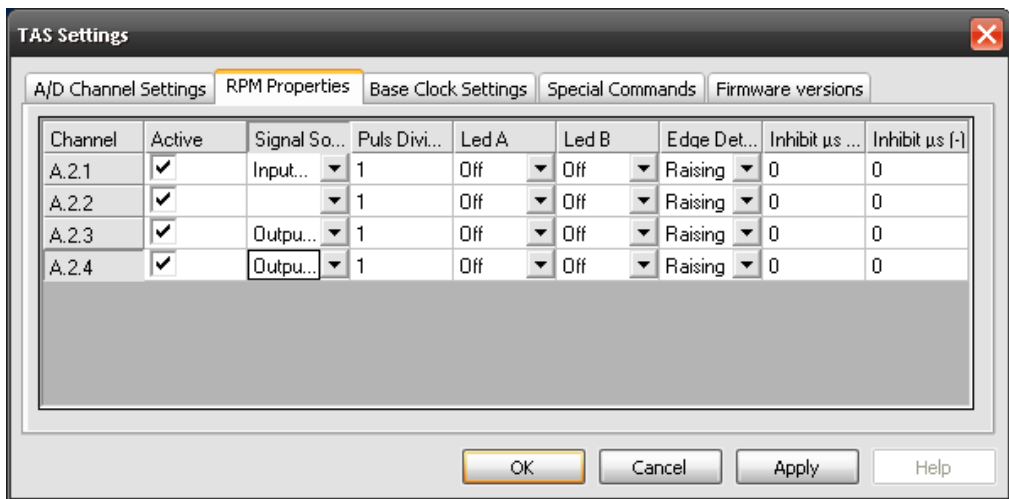
As you can see from the graphic, the TAS Box ports do not have to be occupied in sequence; you can allocate them as desired. If you want to connect a sensor to another connector, simply copy the appropriate settings to another line of the A/D channel settings sheet. Other settings (in other locations in the TasAlyser) are not necessary.

Make sure that you select **ICP** for the accelerometer and appropriate microphones in the **Coupling** column, in order to switch on the supply voltage. If the coupling is only on **AC**, you will not receive an useful sensor signal (as if the noise were extremely quiet).

Enter a suitable sensitivity range in the **Range** column. During the test cycle, open the signal monitor (see “Audio Signal Monitoring and ” on page 139), so that you can check the recording level and optimize this range, if necessary.

## TIS Speed Properties

There is a separate section in the settings dialog for configuring the TIS rotational speed module:



Using the **Signal Source** column you can assign the rotational speed encoders to the TIS input channels. For very high pulse rates the use of a pulse divisor may be necessary. With the columns **Led A** and **Led B** you can configure that the two LEDs on the TIS board display the receipt of rotational speed pulses for one of the channels.





The TIS board channels can only be activated (**Active**) in pairs. If you use only one of the paired channels, simply leave the **Source** of the other channel empty (see graphic).

## More Settings

In the **Base Clock Settings** section you can set the base sampling rate. The TAS Box offers diverse base scanning rates up to a maximum of 100 kHz.

Under **Special Commands** you will find, among other things, the **Reset TAS** button. This is the same as pressing the **Reset** button, which is attached to the TAS Box's power module. However, if the TAS is built into the measurement computer, you will probably not be able to reach this button to press it. The dialog function enables you to execute this command. (Afterwards you should terminate the TasAlyser program and re-start it.)

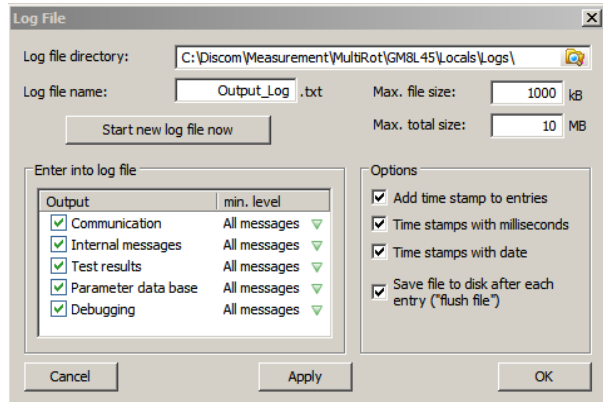
The version numbers of the components built into the TAS Box are displayed in **Firmware Versions**. In rare cases, Discom may request this information if updates of the firmware exist.

## Log File

The TasAlyser writes all the texts which appear in the different sections of the output window in a log file (c.f. “Output” on page 41). Should any problems occur with the TasAlyser, then it can be helpful for Discom to see this log file because it documents the test cycle. (See also “Help from Discom” on page 159).

Go into the **File** menu and call up the **Log File** instruction to get to the setup:

The log files are usually stored in a subdirectory Logs of the project folder’s subdirectory Locals. In the dialog you can read the current log’s location and name. (You can then use the **File – Project Directory** menu instruction to call up a Windows browser for the project directory and then reach the log file in the appropriate folder from there.)



First of all there is the current log (in the above example “Output\_Log.txt”), which is updated until it exceeds its maximum size (here 1000 KB). The file is then renamed as “Output\_Log-Time.txt” and a new “Output\_Log.txt”) is begun. “Time” in the file name is the time of the last entry in the log file in the form YYYY-MM-DD.HH.MM. This process is being repeated until the max.total size (10MB in the example corresponding to about 10 Log-files) has been exceeded. Then the oldest log file is being deleted.

This mechanism ensures that you always have a sufficient number of log files available in order to be able to analyse problems as well which occurred some time ago.

# Signal Monitoring, Calibration, Filter

## Audio Signal Monitoring and Signal Level

The TasAlyser is able to play the sensor signals collected by the TAS Box over the measurement computer's sound card. This allows you to listen, for example, to a structure-borne noise signal directly (over loudspeakers or headphones).

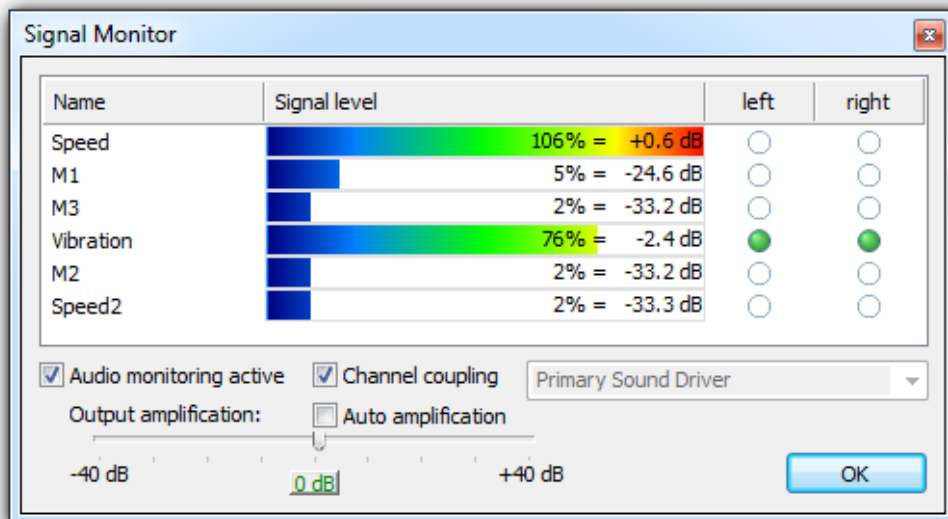
Audio monitoring is coupled with a recording level indicator, where you can read whether or not the sensor signals are possibly overmodulated or too weak.

The audio monitor is found under the favorites with the icon opposite. A double-click on the icon will open the monitor.



Signal Monitor

The signal monitor dialog displays all the analog sensor signals (not, e.g., the signals of a TIS rotational speed recording). (Note: the list is filled only when the test cycle begins.)



The colored bars in each line show the modulation amplitude. Optimally, the bars for the loudest signals should not penetrate further than the yellow area (approx. 70% = -3 dB).

A double-click on one of the colored bars switch the display to show the real measurement values (e.g. "36 out of 50g").

## Audio Monitoring

To listen to a sensor signal checkmark **Audio Monitoring Active**. (Don't forget that you will also need a loudspeaker or headphones if you want to hear something.)

Over the **left** and **right** columns you can select which sensors you want to hear on the sound card's left and/or right stereo channel. You can also set the same signal on the left and on the right (quasi mono listening). If you checkmark **Channel Coupling**, then the audio monitor ensures that you will only hear one sensor with both ears.

You can strengthen (or weaken) the signal output of the sound card by using the output amplification slide control. (The small button 0dB in the center returns the amplification to "none".) By checkmarking **Auto Amplification** you allow the audio monitor to amplify the signal depending on its level, i.e. quiet signals are strengthened, louder signals, weakened. Bear in mind, however, that you can get a wrong impression of the results, i.e. that the signal always has the same loudness.

You should only activate audio monitoring and have the monitor dialog open when you are actually using the function, because the representation of the signal amplitudes and output uses system resources, which are then no longer available for the actual noise analysis.

---

## Calibration

The greater part of signal processing in the measurement program is done digitally. However, before digitization there are the sensor (accelerometer, microphone...), a possible amplifier as well as analog signal processing in the TAS Box. At the end of this processing chain there is voltage (in volts), which is converted by the A/D transducer of the TAS Box into a digital value. But which original measured value - which acceleration, sound pressure, torque... - corresponds to 1 V A/D voltage? This conversion is carried out using the *calibration factor*.

The calibration factor states what A/D voltage comes from the analog processing chain when, at input, a specific signal (specific acceleration, sound pressure etc.) is given. For example, 0.25 V/Pa means that a Pascal sound pressure produces 0.25 V of A/D voltage.

The calibration function of the TasAnalyser is used to determine the calibration factors for all the sensors.

The analog processing chain is, by nature, subjected to fluctuations and tolerances, even when these are very small with the components used for the TAS system. It, therefore, makes sense to check the calibration occasionally. This check is also carried out with the help of the calibration function.



## Calibration Source

A known calibration signal is needed for calibrating – just as when calibrating a set of scales an exact known weight, the “standard kilogram”, is required. The calibration signal is made available by a calibration source. This is a device which emits an exactly defined oscillation signal (for accelerometers), a whistle (for microphones) or something similar.

Calibration sources can be obtained from the appropriate specialist manufacturers. The characteristics of the calibration source (such as, “the signal has a sound pressure of 0.15 Pa with 1000 Hz”) are to be found in the documentation of the calibration source (and frequently on the equipment itself).

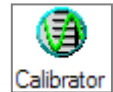
## Using the Calibration Function

### Preparing for Calibration

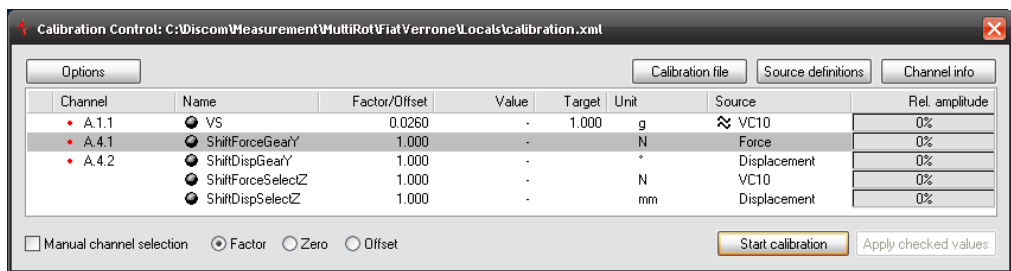
First, you must activate signal processing by means of the measurement program. This is only active during a test cycle, so you must begin one; and because you cannot normally calibrate during a regular test cycle, you must start the test cycle manually (with a deactivated test bench).

You start by activating manual control (see page 48) and then start a test cycle (e.g. by pressing **F5** on the keyboard).

Now open the calibration control. Normally you will find the icon for the calibrator module in the favorites, as shown opposite. Double-click on the icon to start calibration control.



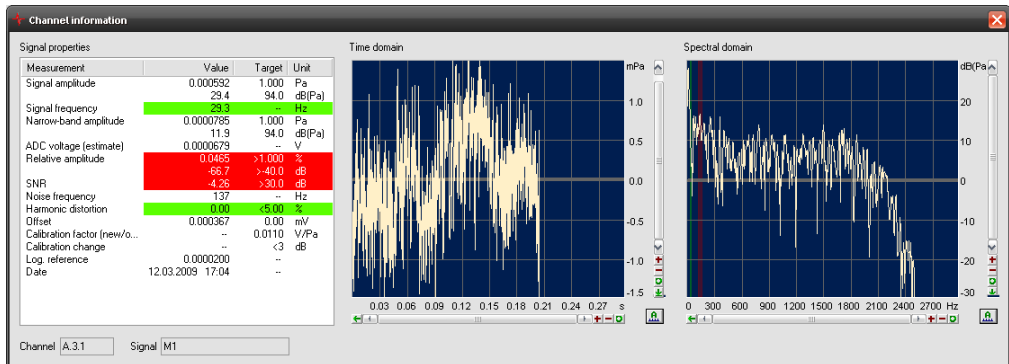
All sensor channels are listed in the calibration control window:



Note: if you open the calibration control before starting a test cycle, then the list will remain empty. The list of channels appears only after the test cycle has started. In addition: many installations in noise analysis require only a single structure-borne noise sensor. In this case, the list will, of course, contain only one line.

We will go into the exact contents of the list later on. First, the practical side of things: select a channel (as in the graphic) and press the **Channel Info**

button (upper right). A second window, **Channel Information**, opens, in which you can read the properties of the channel currently being recorded by this sensor:

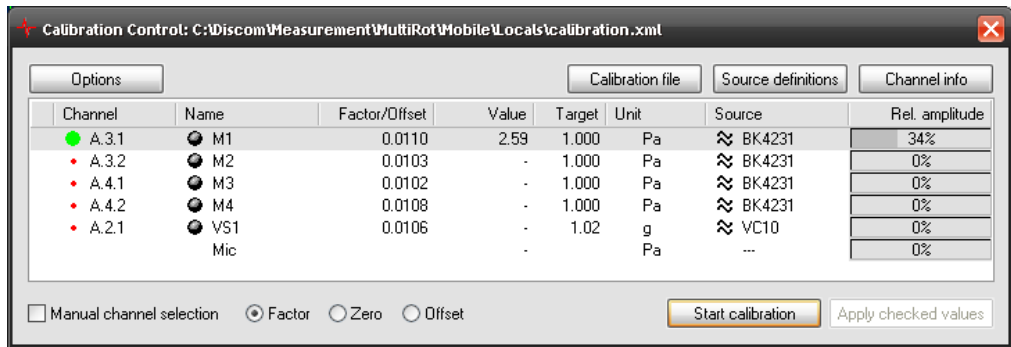


On the left, the window displays signal properties and, on the right, the time signal and the spectrum.

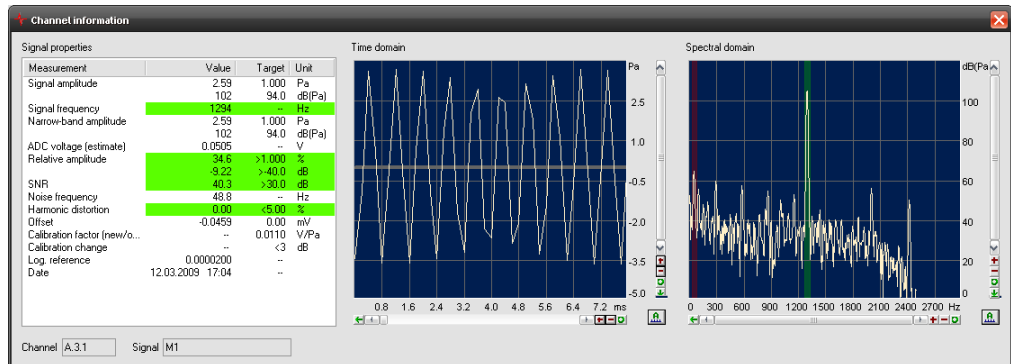
## Calibrating

Calibration control functions more or less fully automatically. Follow the procedure given below:

1. Press the **Start Calibration** button in calibration control. The calibration module now “eavesdrops” on all sensor channels for a useful calibrator signal.
2. Switch on the calibration source and move the source to the sensor (or the sensor to the source). With a structure-borne noise calibrator it is usually sufficient to press the calibrator perpendicularly against the sensor BKS03. With microphone calibration the calibration source is inverted over the microphone.
3. Watch the channel list in calibration control and the signal properties table. Calibration control should discover the signal within a few seconds, then highlight the sensor concerned in the channel list and display the signal in the channel information window:



In the **Channel** column green highlighting appears in the line of the sensor against which you are holding the calibration source. The **Value** column gives the current measured value, and the **Rel.Amplitude** column shows level control bars. The signal and the spectrum appear in the **Channel Info** window, and the yellow and red lines should become green:

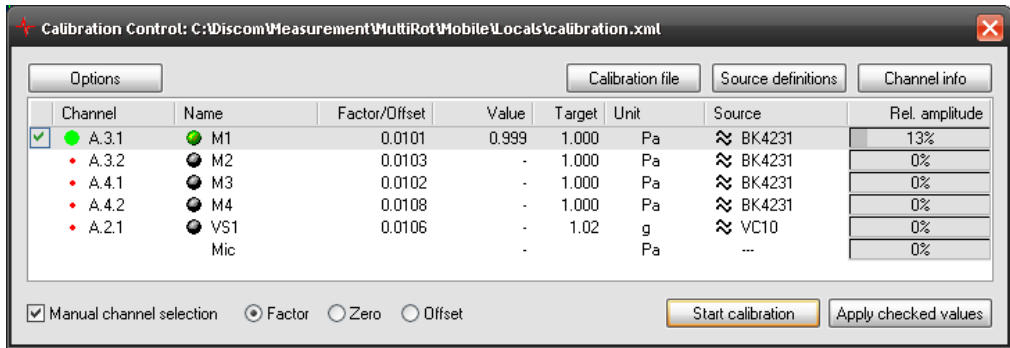


In the spectrum window the frequency band, in which the calibration signal was found, is highlighted green. The frequency band with the next highest level is highlighted red (see graphic).

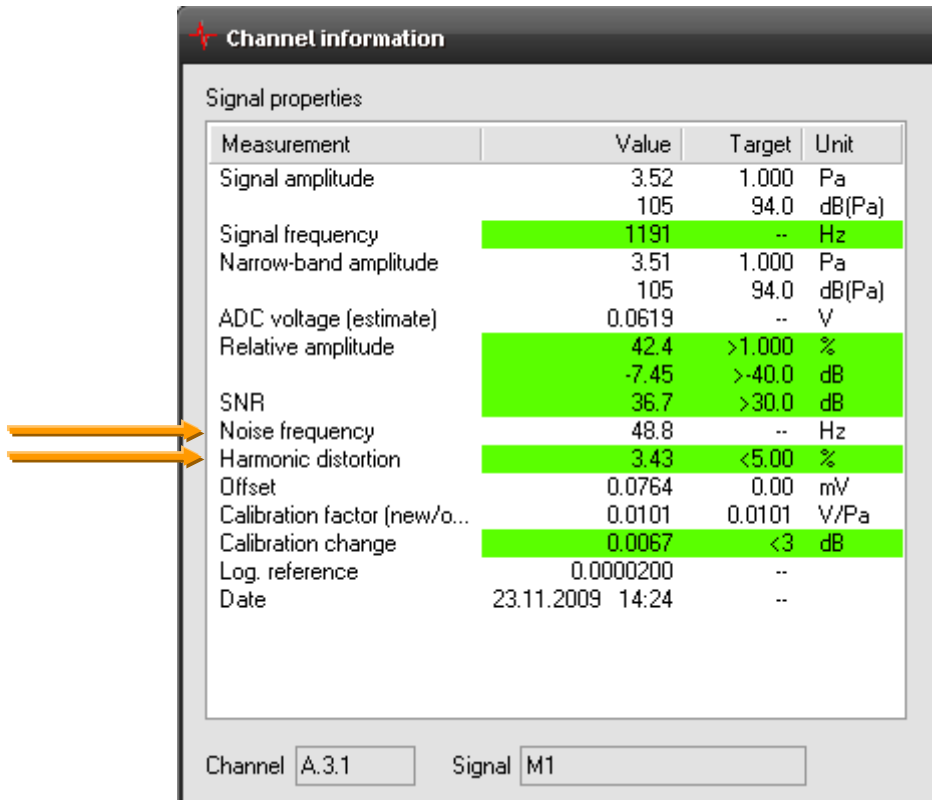
4. Make sure there is good contact between the calibration source and the sensor, so that all the colored lines in the signal properties table become green. Keep this up for some seconds.

If not all lines of the signal properties are green, read which properties the calibration control considers to be unsuitable. Try to improve the signal or the connection between calibration signal and the sensor.

5. When the calibration signal has been stable for some time, a checkmark appears in front of the corresponding sensor channel, and a green symbol in the **Name** column:



In the channel information window the new calibration factor is now displayed in the **Calibration Factor (New/Old)** line. The next line gives the change of the new factor in relation to the old factor (as factor and in dB):



- You can now hold the calibration source directly on the next sensor. Calibration control will automatically select the next sensor channel – continue with point 3.
- When you have calibrated all the sensors you want, you can accept the new calibration factors. Only factors from the lines which have

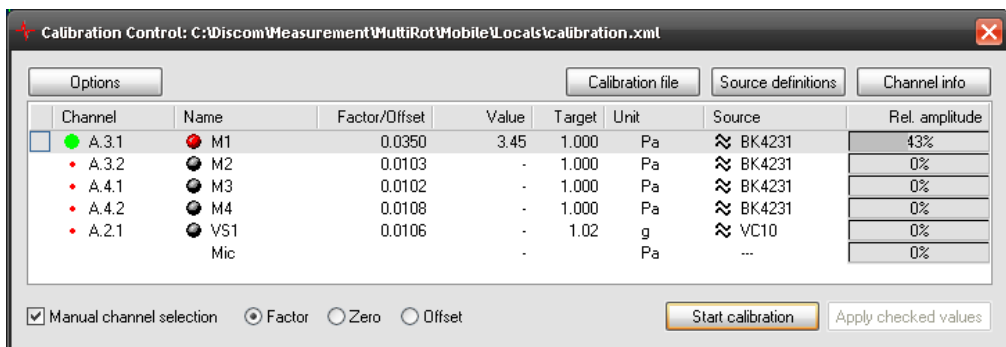




green checkmarks (see graphic under point 5) will be accepted. You do not need to calibrate all the sensors, and you do not have to accept all the newly measured calibration factors.

To accept the new calibration factors press the **Apply Checked Values** button in the calibration control window (lower right). If you do not want to accept these values, simply close the window of calibration control (using the X upper right).

If the new calibration factor differs from the old value by more than the preset factor, then *no* green checkmark will appear in the relevant line in calibration control, and the symbol in the **Name** column will be red not green:



Check if you have carried out calibration correctly and repeat the procedure, if necessary (by pressing **Start Calibration** again). When you are sure that the new calibration factor is to be applied (e.g. because you have exchanged sensors), then you must place a checkmark before the line manually, so that the new calibration factor will also be accepted when **Apply Checked Values** is pressed.

## Entering the Calibration Factor Manually

You can also enter the calibration factor in calibration control manually. Select a line and click on the value in the **Factor/Offset** column. You can now enter the numerical value directly. When you have confirmed the entry, a green checkmark for accepting the value will appear before the line.

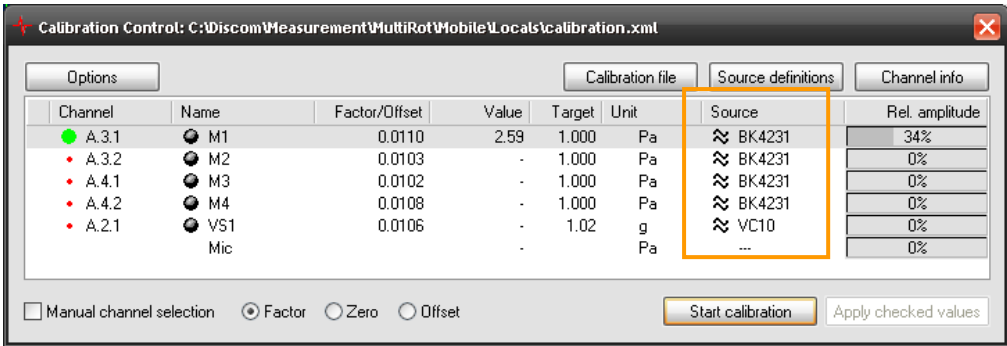
## Extended Functions

### Setting Up Calibration Sources

In order to be able to calibrate a sensor, you must, of course, make the calibration signal properties known to the calibration control. This is done by creating a calibration source definition and then assigning the correct source definition to each sensor channel.

If you only have one type of sensor, such as only structure-borne noise sensors, then you only need to create one source definition, even if you have several channels.

The allocation of calibration sources to channels occurs in calibration control using the **Source** column:

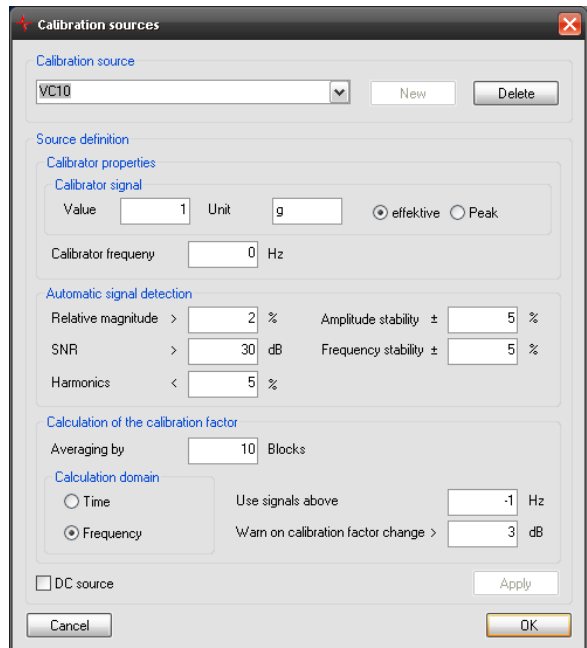


Click in the relevant line in the table under the **Source** heading. The table field changes into a selection list, which you can fold down to select the desired source definition.

To look at source definition details or input a new definition, press the **Source Definition** button in calibration control, which will allow you to access the source definition dialog:

In the **Calibration Source** list select the relevant definition. To input a new definition, enter a new name in the selection list and press the **New** button to the right.

The other dialog fields display the properties of the calibration source, which can be changed.

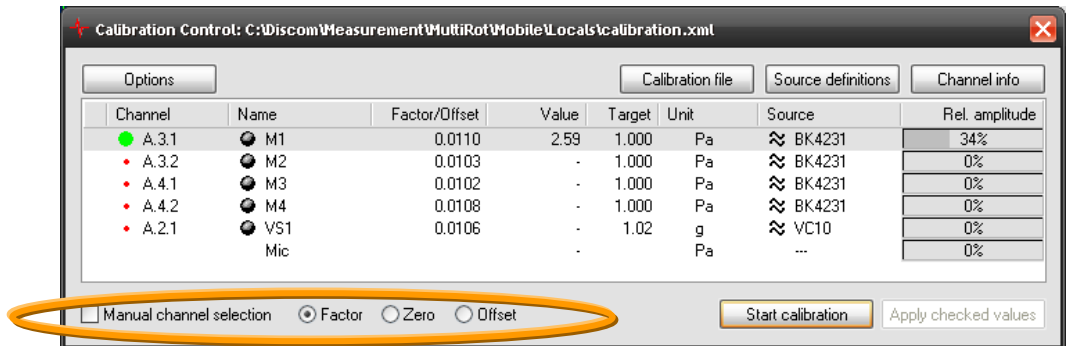


## Selecting the Channel Manually

Normally, calibration control searches all the sensor channels for useful calibration signals automatically. You can deactivate this automatic search



by setting the checkmark at **Manual Channel Selection** in calibration control (located bottom left):



You must then select the line of the sensor to be calibrated.

## DC Zero Point Calibration

Even when there is no signal present at the sensor, there can be voltage at the A/D transducer, which is generated by upstream analog electronics. This voltage is known as the DC offset. In the analysis of oscillation signals the DC offset is of secondary importance; for DC voltage signals (e.g. torques) on the other hand it is very important.

In order to determine the DC offset and subtract it from the signal recordings from now on, select the **Zero** option at the bottom of calibration control (instead of **Factor**). The contents of the **Factor/Offset**, **Value** and **Target** columns (which becomes 0) will then change.

Activate **Manual Channel Selection**, select a line, make sure that there is no signal at this sensor, and press **Start Calibration**. Calibration control determines and then displays the DC offset. If you press **Apply Checked Values**, the DC offset will be stored as well.

## Calibrating DC Sources

With DC sources, such as torque sensors, calibration also involves the measurement of the A/D voltage produced by a known calibration signal. For DC sources, however, we need two measurement points: one for the zero signal value (i.e. the DC offset described above) and one for the calibration signal.

In the source definition for the calibration source, you must place the checkmark at **DC Voltage Sources** (at the bottom left).

First, carry out the DC zero point calibration for the relevant sensor as described above. Then switch from **Zero** to **Offset** at the bottom of calibration control, input the calibration signal and press **Start Calibration** again.

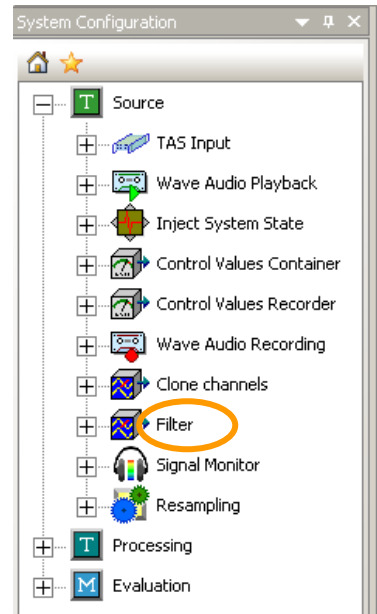
## Using Filters

Filter functions are used in different areas of the analysis system. This chapter deals with the *filter module* used in the input data stream. It is used to filter an input signal before any other processing. The typical use is the A-filtering of a microphone signal. In certain situations you can filter rotor synchronous components even before resampling.

Within the filter module, *filter groups* need to be defined. Each filter group consists of one or more single filters (e.g. A filter or band pass). You can assign a one filter group for each signal to apply the corresponding filters on the input signals. A filter group can be assigned to more than one signal (e.g. all microphones).

### Available single filter

The filter module allows to use filter which are described in a text filter. This makes it possible to freely define a filter using a filter design program, using coefficients from literature or based on own calculations. In addition to that, some algorithms to calculate filter are implemented by default. Currently, Butterworth filter and Peak/Notch filter (like those in parametric equalizers) are available. With these defaults you can even design filters online (you are not bound to filters defined in a file) and follow the effect of changes directly using the audio monitor.



### How to use the filter module

Assuming that well designed filters for the application are already defined, the filter module can be used to control the effect of the filters, especially to enable/ disable certain filter for offline analysis. This can easily be done using the “small dialog”. The same options and more are available in the main dialog as well.

#### The “Small Dialog”

The small dialog shows all filters defined for the active filter groups. Left clicking the filter entry in the dialog activates/ deactivates the filter. The change has immediate effect but is not stored. If you want the setting changed permanently, you have to enable/ disable the filter in the main dialog’s section **filter groups**.





If you right-click on one of the filter entries, the corresponding design dialog is opened (unless there is none available for that kind of filter). You can read more on filter design below.

## Interaction of “small dialog” and main dialog

When the small dialog opens on start of the application (because it was open when the application was closed) it is displayed “empty”. Clicking on the filter module in the configuration tree always opens one of the two dialogs. If no type has been loaded, or no valid filters are available, the main dialog opens. Otherwise, the “small dialog” opens. The main dialog also opens when the “small dialog” is already open.

## Setup of the filter module

The following section explains how filters are defined and which parameters need to be set. At first, some terms and the structure of the module need to be explained.

### Structure of the module

The filter module is designed to be multi-functional. For this purpose, the filter settings have been put in three main categories:

1. Assigning filter groups to *input signals*.
2. Defining *filter groups* as sets/ combinations of single filters of different kind (e.g. an A-filter as file based filter and a Butterworth filter as default filter).
3. Defining *single filters*, e.g. the coefficient’s file of the A-filter and the definition of the Butterworth filter as highpass of 2<sup>nd</sup> order with a limit frequency of 100Hz.

The parametric filters, in this definition single filters, are currently always filter of 2<sup>nd</sup> order. As such they can be combined to a filter with higher order, for example to “comb” the harmonics of a distortion signal out of a signal. Such a combination can either be used as single filter (like 3) or all elements of the combination are used in a filter group (like 2).

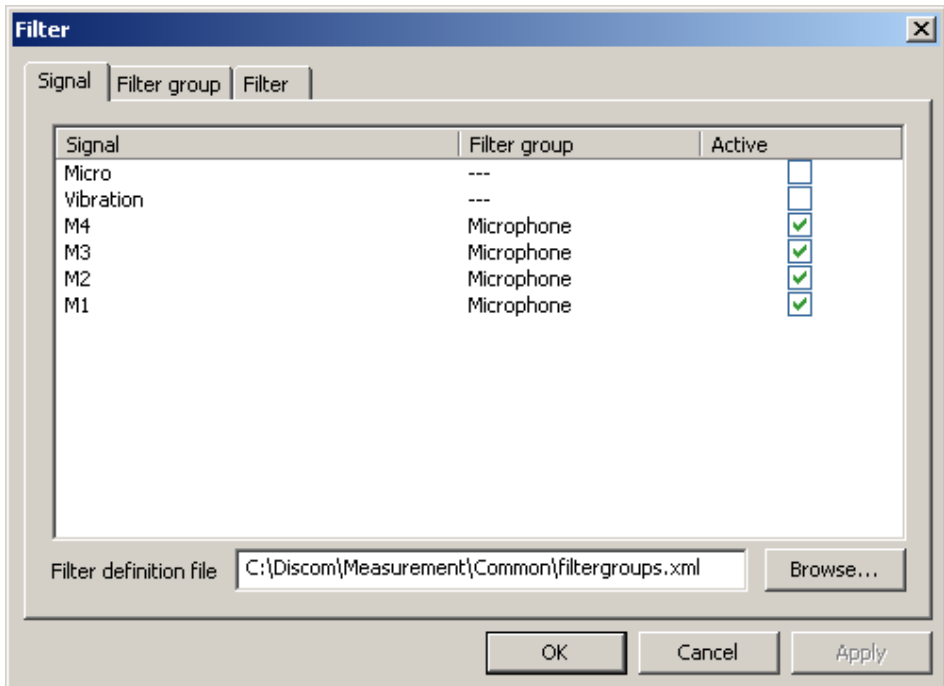
### Adiabatic changes

*Adiabatic* changes of the single filters can be monitored online. “Adiabatic” changes are such changes which does not affect the structure of the filter, e.g. changes of the limit frequency or the amplification. A non-adiabatic change is the change of the filter order or switching a filter from low pass to high pass.

## The main control window

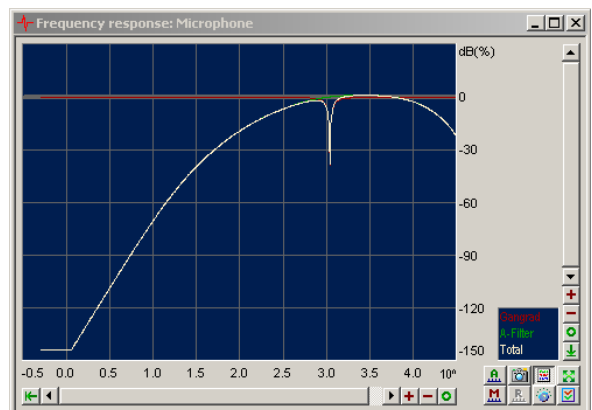
The main dialog shows the described structure in the three main sections “Signal”, “Filter group” and “filter”.

### Assigning filter groups to signals



In the section **Signal** you can assign a filter group to an input signal and activate the filtering of the signal. Furthermore, the file of the filter descriptions is set here. In this file, all filter settings are stored, except which filter groups are assigned to which signals. This allows the filter setting file to be spread over various applications. The connection between signals and filter groups on the other hand is stored in the settings file of the project. Anyway, filter groups can only be assigned after signals have been run through the application (this means not directly after start of the program).

In the column filter group, you find combo boxes where you





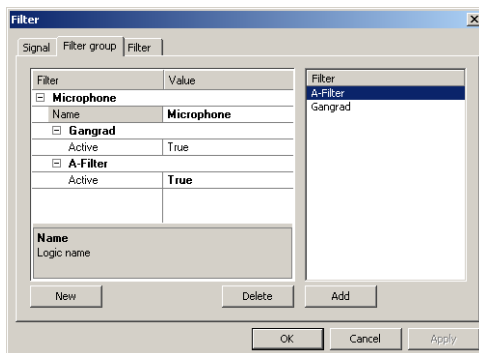
can select one of the filtergroups defined in the section **filter group**. You can assign not more one filter group for each signal.

A double click on one of the signals in the column **signal** opens a scope which shows the frequency flow of the selected filter group (white curve = **Total**).

The scaling is double-logarithmic. This means that the abscissus shows powers of 10. Beside the total frequency track, it also shows the frequency tracks of all single filters.

## Filter groups

In the section **Filter group** you can put some of the filters defined in the section **Filter** (see below) into one filter group.



Each group has a name and contains a number of (parts of) filters. Not all filter parts must be active. Click on the entry **True/False** in one of the **Active** rows to switch the usage of filter parts. The name of the filter group can be changed in the **name** row (not in the group heading).

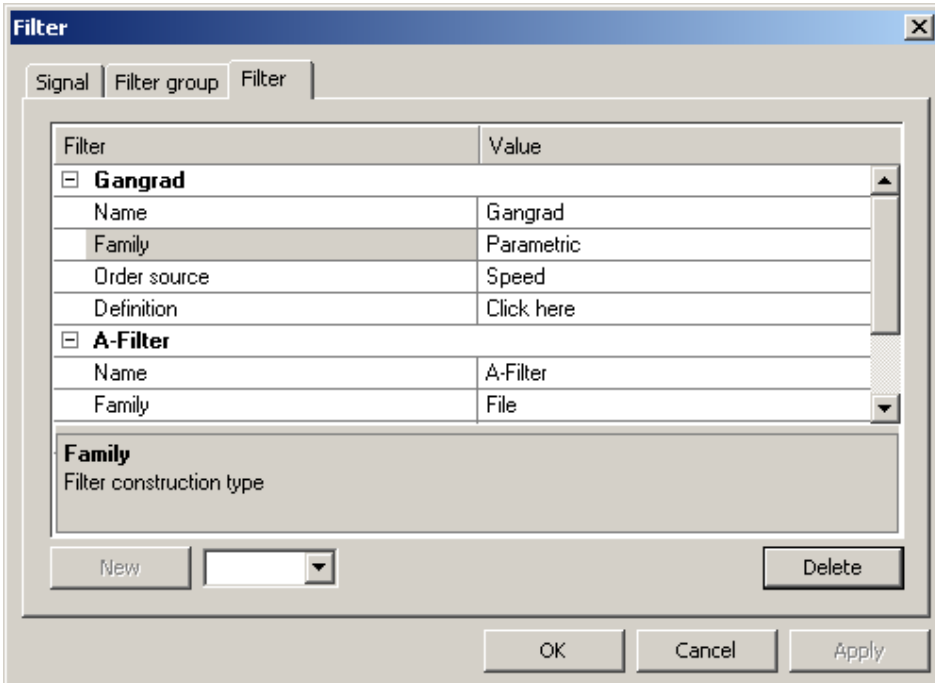
You can create new filter groups by clicking the button **New**. You

delete a filter group by selecting the *group heading* (not one of the sub entries) and clicking **Delete**.

In similar way you can assign further filter parts to a group. At first, you select the group (by selecting the heading), then you must specify the desired filter part from the list on the right and click **Add**. You can assign a single filter part to a certain filter group more than once. You remove filter parts from a group by selecting the *heading of the filter part* and clicking **Delete**.

## Filter parts

Finally, in the section **Filter**, you find the definition of the single filter parts. Each filter part has a **name**, a **family**, if necessary a **speed reference** and a **definition**.



You rename or delete filter parts in the same way you rename or delete filter groups.

Creating a new filter part, first requires to select a filter family from the combo box beside the button **New**. After that, the button gets active and can be pressed. You cannot change the family of a filter after you created the filter. This can only be done by deleting and recreating the filter with the new filter family.

A filter can use a fixed frequency reference or use an order frequency with reference to a certain speed source. This is specified in the row **Order source**. You select the dash if you want to use fixed frequency reference. Otherwise you select the corresponding speed source (referencing the rotor of that speed source). The characteristic frequencies of the filter are either Hertz (fixed frequency reference) or orders with respect of the speed source. **File** based filters are always fixed frequency filters and cannot use a speed reference.

Depending on the filter family, a click in the **definition** field opens either a file selection dialog (family *file*) or a design dialog (all other families).

If you want to define filters by yourself, you can find a short guide in Appendix C.





## Measurement Archives and Evaluation

This chapter describes how to deal with stored measurement data and gives a brief introduction to “Presentation”, the evaluation program. It is recommended that you consult the detailed manual on presentation, if you frequently work with the evaluation program.

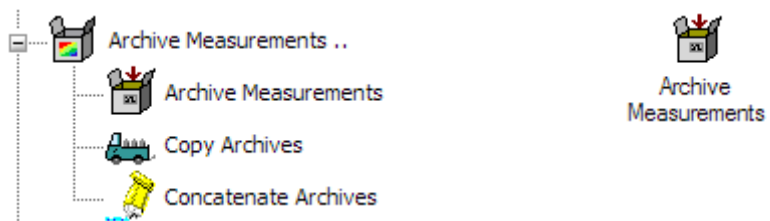
The measurement program stores the data of completed tests in *Archives* with a special file format and the file extension `.rdt`. These archives can be stored in a central place (on a server or the measurement computer) and indexed by a database.

The complement to the measurement program which stores the archives is the evaluation program, known as *Presentation*. This program allows you to read the archive files, evaluate the content (the measurement data) and display the results graphically. While the measurement program can only be started on a computer equipped with a TAS Box, the presentation program can be used on any PC.

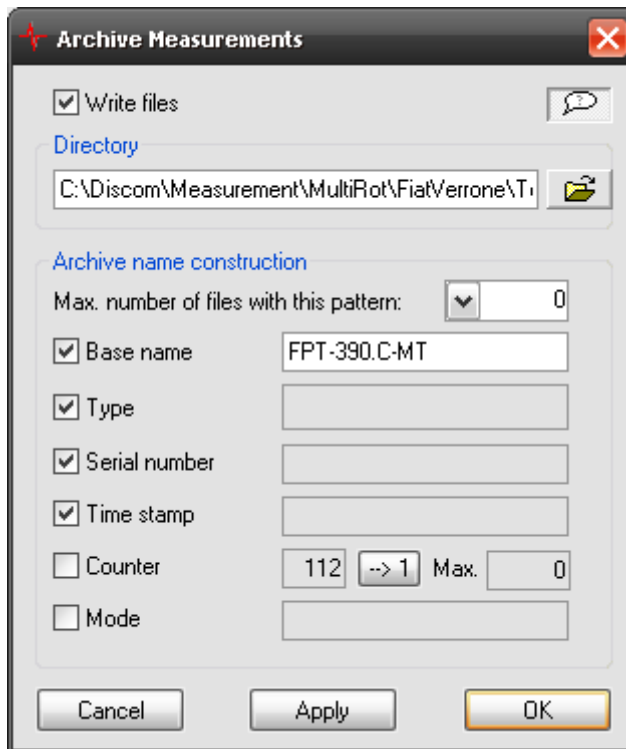
---

## Archiving in the TasAlyser

In the TasAlyser one module is responsible for the creation of the archive files. You find it in the favorites or in the system configuration below **Evaluation**.



By double-clicking on the entry you open the dialog of the archive module. Here you can specify where the archive files are to be stored and how the name is constructed:



The **Write Files** control box at the top is particularly important. Here you can completely deactivate the archiving process.

## Moving Archives to the Collector

An archive file is created by the archiving module for each test cycle, typically in the subdirectory **TempArchives** of the project directory (see dialog setting in the picture above).

Normally, the archives do not remain in this directory. Only with the mobile application they remain in the folder where they are created because they shall later be moved manually to a suitable location of choice. With applications in test bench series the TempArchives directory would soon be full and reach the maximum number of files per directory allowed by Windows.

You can specify a maximum number at files in the archiver dialog (see above). If you enter 0 in the input field, there is no upper limit; if you enter another value, the oldest archives will be deleted, if necessary.

The usual procedure, however, is to move the archives from the temporary directory to the Collector's input directory. The Collector concatenates the archives of the individual measurements into day archives and sorts these into week directories. At the same time, the Collector makes entries in the



result database for each measurement, so that you can easily find any measurement with the help of the presentation program.

If your installation does not include a result database, you can let the concatenation of day archives be carried out by a module of the TasAlyser, as described in the next section.

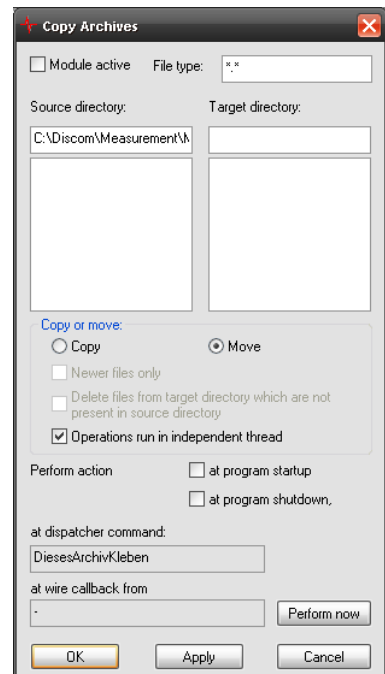
The module for moving archives to the collector and for creating day archives is found in the system configuration next to the archiver module:

The icon for the moving module is a truck, that for the concatenate module, a tube of glue.

Open the dialog of the mover as usual by double-clicking on the icon:

At the top right you can find the **Module Active** control box, with which you can deactivate the mover. If you deactivate it, the archives accumulate in the local directory (if not processed otherwise). This can be useful, if, for example, you carry out test measurements and want to access the archives directly. If you activate the mover again at a later stage in time, it will automatically move all the accumulated archives (i.e. not only the last file) at the end of the next test. You can also press the **Perform Now** button in the dialog (below right) to complete any moving tasks which are still pending.

You can see the source and target directories in the dialog, as well as their contents (there are no files in the target directory in the graphic). The **Copy** or **Move** options should usually be set at **Move**.



## Local Concatenation of Archives

If your installation does not contain a Result database, local concatenation should be active. Using the “Concatenate Archives” module, you can

concatenate the measurements by day or by week, sort measurements according to type, etc.

Activate the concatenator by checkmarking the **Concatenation Is Active** control box if you want to use it. Enter the directory in which the concatenated archives are to be stored under **Target Directory**.

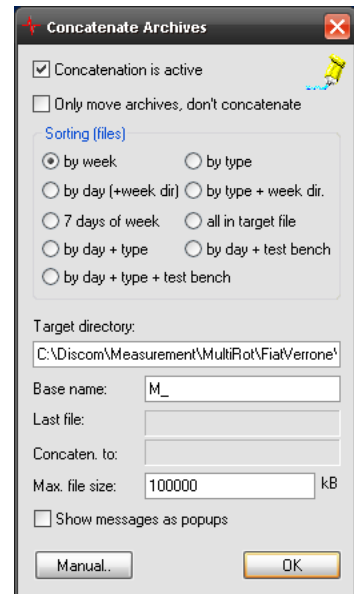
Under **Sorting**, you can setup the pattern for concatenating the individual measurements. If you select one of the options which sort according to day, then the “concatenator” will first of all create folders in the target directory which are numbered and named according to calendar week. Within these files, it will then create day files. The day archives’ names have the number of the day in the year (e.g. 51 represents February 20). If you activate **Only Move Archives, Don’t Concatenate**, week directories are created but the individual measurements are treated as individual files and not concatenated as day archives.

**Weekdays** is a special option which creates a separate file for Monday, Tuesday etc. On the following Monday the previous Monday file is deleted and a new Monday file is begun.

This option means that you only ever retain the data from the last 7 days, but, on the other hand, the total volume of the archive remains restricted.

With all of the other sorting options all measurements are retained. With the usual archive volume of less than 1 MB and hard disk volumes of several hundred gigabytes you can accumulate measurements for quite a long time before having to think about problems of disk space.

The Collector and the local concatenator can also be combined. To do this you must switch from **Move** to **Copy** in the move module dialog (see the graphic on the previous page). A copy of every archive created by the archiver is then sent to the Collector, and the archive is concatenated in the local collection after that.




---

## The Presentation Program

The presentation program is the tool with which you can look at, compare and evaluate stored measurement data. Within the presentation program you



can represent the data on *layout pages*. Each page contains graphic modules (such as text fields and curve graphs) which display the data.

The data can come from the archive files as well as from the result database. Archive files are opened directly. For database queries, the *database assistant* will help you. After you loaded archives or carried out a database query, you get a list of all loaded measurements, identified by the measurement time, the serial number, type, test bench and further information.

After measurement have been loaded, the data can be displayed in the existing layouts and graphic modules with the help of *Rapports*, which can produce a complete test record or evaluation. Rapports offer many options and possibilities for producing test records, which can even spread on several pages.

This brief introduction only describes how to use Rapports, not how to create Rapports. The creation of Rapports, as well as the interactive display of data, are described more detailedly in the presentation manual.

## Installing the Presentation Program

The presentation program is already installed on the measurement computer. You can also install the program on your personal computer to allow you to examine archive files there or access the result database.

Installation packages for this program can be found on our web server under [www.discom.de/ftp](http://www.discom.de/ftp), under names such as `Presentation_De_(Date).msi` or `Presentation_En_(Date).msi` which differ only in the language of the installed manual. The installed presentation program is identical in all packages and automatically adopts the language of the computer on which it is installed.

Download one of the packages. (If you have already installed an older version of the program, you should uninstall this first using the control panel, as is usual with Windows). Start the installation package and follow the instructions.

As well as the presentation *program* you also need – similar to the TasAlyser - a presentation *project*. A Presentation project also consists of several files, which are contained in a project folder. Normally an appropriate project can be found in the folder `C:\Discom\Analysis\Presentations` or in the project directory of the appropriate TasAlyser project on the measurement computer. You can simply copy the folder with the presentation project onto your personal computer.

You can also load an sample presentation project from our web server under [www.discom.de/ftp](http://www.discom.de/ftp), with an accompanying installation package with the name `Presentation_StdSample...msi`.

When you start the presentation program for the first time after installation, it does not yet know the project you want to work on. You will see a prompt informing you that the project base file is missing. Confirm the prompt, navigate in the **Open File** dialog to the project directory and open the base file found there (with the extension `bse`, e.g. `Presentation.bse` or `GtrPresent.bse`).

The presentation program subsequently notes which project you opened last (even if you have deleted and updated presentation in the meantime).

## Updating the Presentation

If you already installed the presentation program and only want to update it, you can download a package with the program files only from our server([www.discom.de/ftp](http://www.discom.de/ftp)), the so called binaries. These packages are named `Presentation-bin-(Date).zip`.

After you unzipped the package, you get a folder `Presentation`. This folder contains in particular the file „Update Presentation.bat“. Run this batch and the binaries are being updated. (Of course, you have to end all running presentation programs first: Furthermore you may need administrative right to do this. That means, you have to run the batch with administrative rights.)



## Help from Discom

The Discom team will do its best to help you not only with any problems you might have with the software and/or hardware, but also with strange noise phenomena, the choice of suitable parameters and any other issues to do with noise analysis.

Depending on the type of problem, you can make it easier for us to help you by making suitable information available - this information will usually be files from the measurement computer. This chapter describes how you can supply us with the necessary information and what else we need so that we can help you as efficiently as possible.

---

## Transmitting Files

### Compression

Before you send us a file or a whole directory by email, you should compress it. In the first place, this reduces the volume of data and, in the second, you can then send a whole directory including subdirectories in the package without problem.

The program *7Zip* is pre-installed on all measurement computers. This is a free compression program, which you can also download from [www.7zip.org](http://www.7zip.org).

To compress you select the relevant file or directory in Windows Explorer and call up the context menu using a right-click. Here you will find a submenu **7Zip**. From this submenu select one of the instructions “Add to Archive xxx.zip” (a suitable file name is usually offered to you) or “xxx.7z”. *7Zip* will then provide an archive file in the same location as the file or the directory. Copy the archive file onto a transportation medium, e.g. a USB stick, and transfer this to a PC with email access.

If *7Zip* is installed on a computer which has email program and accounts, then you can use the “Archive and Send” instruction in the *7Zip* context menu directly.

If you receive a compressed file from us or download one from our web page then you can use *7Zip* to unpack the files. Right-click the file, go into the *7Zip* submenu and select either the “Unpack File” or “Unpack here” instruction.

### Mailing

In most cases you can simply send us the compressed files as email attachment. It is a good idea to send separate mails (one for each file) if you



have a number of large files. Modern mail-servers usually permit attachments of several megabytes, and nowadays upload and download speeds are also sufficiently high.

However, if you have to send a *very* large file, then it is advisable to upload the files onto our web server. To do this, use `ftp` (“File Transfer Protocol”). There are several programs (including Firefox extensions, for example) with which you can upload the files per `ftp`. “Command Prompt” (to be found over Windows Start-Menu/Accessories) also commands `ftp`.

Follow this procedure: Start the command prompt.

Using the `cd` instruction access the directory where the file which is to be transferred is found. Then enter:

```
ftp discom.de
```

to link with our Web server. You will then be asked for a user name and a password. You will receive this information from us when necessary.

Then give the commands

```
binary
```

```
and
```

```
put (File name)
```

The transmission will begin. After transmission has been completed, terminate the `ftp` session using the command `bye`.

---

## When the TasAlyser Does Not Work

The first question we will ask is “What does not work?” Some examples:

- Does the program start?
- Are there error messages on startup? (If yes, what do they tell?)
- Or does the malfunction occur during normal operation? If yes, what are the circumstances? (e.g. “Always when a test cycle begins”)
- Is the program “stuck”, meaning it does not react to the mouse, does not close, or similar?
- Has the TasAlyser crashed? (Often you’ll then get an error message “Debug Assertion Failed”, which you can close with **OK**.)
- Or is it a problem of communication, meaning that the measurement program no longer reacts to (all) test bench commands?





- Although the program is working you cannot see the rotational speed any more, or you can see the rotational speed, but no signals in the scopes?

When there is an error message from the TasAlyser, the text is always helpful. If there is a communication problem with the test bench you should consult the output window, communication section, for advice (see “Test Bench Connection“ on page 47 as well as “Docking Windows” on page 40). If the rotational speed or the noise signals are suddenly missing, check the appropriate sensors and the cables between the sensors and the measurement computer. Otherwise, follow the procedure given below:

1. If the TasAlyser does not react any more, then you have to “kill” it, (using the Windows task manager). Restart the program but do not begin a test. Note any error messages at program startup.
2. If the TasAlyser reacts, or if you have restarted it in 1., use the instruction **Info on the TasAlyser** in the **Help** menu. Here you will find a version number and specification “Build” with a date. Make a note of this information.
3. After this, use the instruction **Project Directory** from the instruction **File**, in order to call up Windows Explorer, which displays the project directory. Terminate the TasAlyser.
4. Use the pre-installed 7Zip, as described in the section “Compression”, to compress the Application folder in the project directory. Send this folder to us, as well as the version information in 2. and any error messages in 1.
5. If you know of any circumstances or other details which have caused the TasAlyser’s malfunction, mention these also in your email.

Please contact us immediately. In many cases, we can state the cause of the problem immediately or after a short examination of the files and then offer a suitable remedy. Otherwise, we will discuss with you how to proceed further.

---

## Strange Noises

We are always interested in learning about new noise phenomena and investigate these with you. Other than with problems with the TasAlyser program, where we need settings and log files (see previous section), with noise phenomena we are interested in the noises themselves (Wave files) and/or associated measurement data archives.

### Wave files

Use the function for recording Wave files to record the test cycles of some interesting aggregates (see “Wave Audio Recording and Playback ” on page



124). Give these files an appropriate name and upload them on our ftp server. You can read the directory in which the measurement program has stored its recordings in the control window of the Wave Recorder.

### Measurement Data Archives

Usually, we also need the the associated measurement data archives together with the wave files. It is best to deactivate the transport of the archives to the database collector temporarily, so that the individual archives remain in the project directory's **TempArchives** folder. Compress this folder and send it to us or upload it onto our server together with the wave files.

---

## Unwanted Test Results

Sometimes it occurs that the noise analysis system rejects too many tests with NOK result. Generally speaking, a n.O.K is caused by some measure value trespassing its limit and can be fixed by shifting the corresponding limit. Before you proceed that way, you should nevertheless try to evaluate *why* this happens. Maybe, the noise analysis system is quite right in rejecting the transmissions.

First of all, you should listen to the actual noises using the audio monitor (see "Audio Monitoring" on page 140). You can do this either at the test bench (using head phones), or record the Wave files and use a copy of the TasAlyser on your desktop PC. Use the presentation program to compare n.O.K. measurements with (probably older) O.K. measurements.

For us to be able to advise you, we need measurement data archives as well as the parameter database. The latter is found in the project directory's subdirectory **ParamDb**. Go into this folder and compress the mdb file(s) which are contained there. Simply send us the result by email. (Do not compress the whole folder because if you do, you will also include the **Backup** subfolder into the package, which can be very large, but does not help us in the analysis.)



## Appendix A: Rotas Mobile

Because of its small size, robust construction and low power consumption, the TAS Box is especially suitable for use as mobile system, for example, in mobile vehicle tests.

This appendix describes the standard configuration of the mobile system and gives tips on its use in vehicles.

---

### Setting Up the Hardware

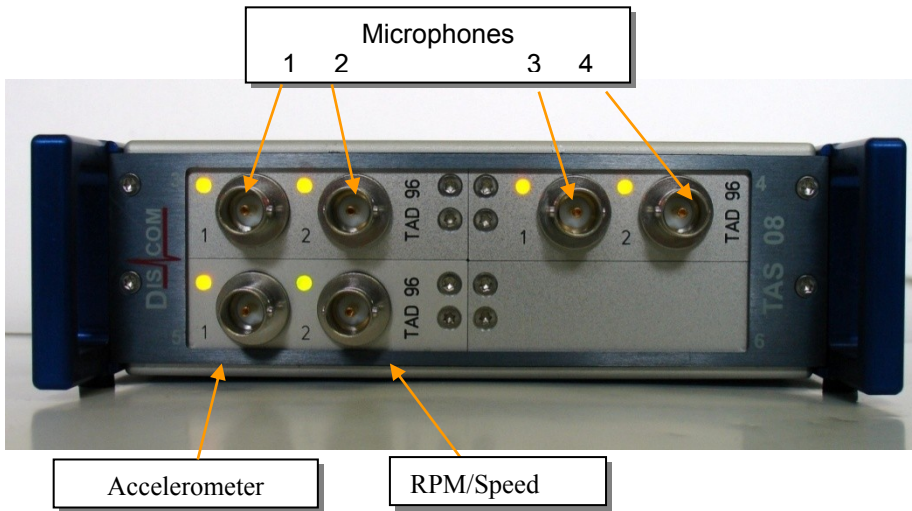
The mobile system consists of the following components:

- A TAS Box with USB cable
- A notebook, if necessary with power supply unit to connect to the vehicle's electrical system ("cigarette lighter")
- Four microphones with suction cups and BNC-cables
- A structure-borne noise sensor with amplifier and BNC-cable
- A rotational speed sensor with cable(s)

If the TAS Box is supplied with power via USB only, then up to five sensors with ICP supply voltage can be attached. The mobile system, therefore, usually uses four microphones and a structure-borne noise sensor. Although you do not have to connect all four microphones, the use of more microphones reduces the influence of interior resonances, which can falsify the measurement results if only one microphone is used.

Attach the sensors to the TAS Box. For the standard mobile project the four microphones must be attached to the upper row of the sensor ports, the structure-borne noise sensor at the bottom left with the rotational speed next to it:

In case you have a mobile system which includes a TIS-card (see above the section **Mobile system** on page 6), slot 4 holds a TIS card which you do not need for mobile measurements. The TAD96 card moves to slot 6 instead. This has the consequence that the horizontal rows change their connectors which means you have to connect noise to 3.1, speed to 3.2 and the microphones to 5.1, 5.2, 6.1 and 6.2.



## Speed acquisition

For mobile measurements, there are basically two procedures to get rotational speed: reading out the vehicle diagnostic system, or connecting an external speed sensor at an accessible axle, e.g. the drive or the Cardan shaft.

For obtaining rotational speeds from the vehicle diagnostic system you need an adaptor. These devices are specific to the vehicle manufacturers and often even specific to the model.

As external sensor, a laser sensor (Wenglor sensor) has been found very useful. If you want to use such a sensor, you have to tightly fix a reflective mark to the axle. The sensor itself is fixed on the vehicle close to the axle so that the laser points to the reflective mark. When the axle turns, the sensor detects the passing of the reflective mark and generates one pulse per revolution of the shaft.

The following pictures show the sensor, which has a magnetic foot for fastening it to the vehicle, as well as the power interface for supplying the sensor with power:



The laser sensor is attached to the power and signal interface via the **In RPM (Laser)** port. The **Out RPM** port is connected with the speed input channel of the TAS Box.



Make sure that the laser sensor is firmly fixed to the vehicle to ensure that it remains aligned with the reflective mark and does not fall off when the vehicle moves!

## Starting the TAS Box

Finally, connect the TAS Box to the notebook via the USB cable. Start the TasAlyser software only *after* connecting the TAS Box (and the computer has recognized the USB device).

Press **F5** to open the dialog for starting a test cycle. First, simply press **OK** (or the <Enter> the key on the keyboard) to load the standard test sequence. Then press **F6** to insert a test step and thereby start the A/D transducer of the TAS Box. The LEDs at the TAS Box's sensor ports should now light up.

The LEDs at the microphone ports and at the structure-borne noise port should be orange, which signals an active ICP supply voltage, while the LED at the speed port should be green:





In addition, you should be able to see signals or spectra in some of the TasAlyser’s scopes, as described below.

Once you have checked the signals, press **F9** to cancel the “test cycle”.

---

## The TasAlyser Mobile Project

In the mobile project, two groups of sensors are set up: the microphones and the structure-borne noise sensor. A fixed frequency channel (with a scanning rate of e.g. 20 kHz) and a rotationally synchronous channel (mix channel) are calculated for each sensor. The microphone channel spectra are averaged. You get a total of four channels as a result:

<p><b>Mic-FX</b> average microphone signal, fixed frequency</p>	<p><b>Mic-Ord</b> Average microphone signal rotationally synchronous</p>
<p><b>VS-FX</b> Structure-borne noise sensor, fixed frequency</p>	<p><b>VS-Ord</b> Structure-borne noise signal rotationally synchronous</p>

Maximized spectra, spectrograms, as well as level tracks, are recorded for all four channels. Since mobile measurements are usually used to investigate specific noises and then evaluate these subjectively, all evaluations (limit curves) are deactivated.

In the mobile project, the gears of a standard gearbox are pre-configured. Each gear can be measured in several test steps. Usually, you run measurements while varying the speed. This results in the two test steps “Drive” and “Coast”. Furthermore, you have at least one “Steady” test step which allows measurements without referencing the speed explicitly (except to calculate the rotationally synchronous channels).

Therefore, in the “steady” steps, you can perform measurements without a rotational speed signal, but you will then, of course, only get data for the FX channels. This means, if you connect the hardware for testing as described above, you should see signals and spectra in the displays for the FX channels after inserting a test step. Tap the microphones and the structure-borne sound sensor gently and observe the amplitudes in the displays. In this way, you can check whether all sensors are working correctly.

If you want to use less than four microphones, then you have to deactivate those channels you do not want to use. You can do this in the TAS Box setup



as described in the chapter “Further TasAlyser Functions”. Optionally, you may want to calibrate the microphones before the first measurement and/or check calibration. In this case, read the section “Calibration” starting page 140.



## Driving Measurements

The basic measurement cycle with the mobile system is similar to that on the series production test bench: load test sequence, run test steps, terminate test cycle. Usually, you control the measurement using the Notebook keyboard.

Keyboard control uses the following keys (in the order in which they are usually used during a test run).

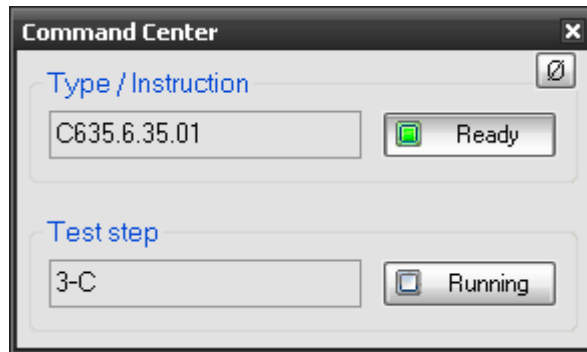
Key	Function
<b>F5</b>	Start test cycle: a window appears for selecting the test sequence. Enter a serial number and/or a comment, if necessary, and confirm the dialog using the <b>&lt;Enter&gt;</b> key.
<b>F2</b>	Enter or change serial number and supplementary information
<b>F6</b>	Select test step from the list. Use the keyboard to enter the name of the desired test step (it will be completed automatically), or select the test step using the <b>↑/↓</b> key. Confirm the dialog using the <b>&lt;Enter&gt;</b> key.
<b>Space bar</b> or <b>F7</b>	Measurement start / finish. (Only when a test step is inserted and only once per test step.)
<b>F3</b>	Cancel measurement in current test step and re-start immediately (= repeat test step).
<b>F4</b>	Terminate measurement in current test step, insert next test step and start measurement.
<b>Alt+F7</b>	Cancel measurement
<b>F8</b>	Terminate test cycle (regularly) and store measurement data.
<b>F9</b>	Cancel test cycle
<b>PgUp↑</b>	End measurement and select the previous test step in the list (no start of new measurement)
<b>PgDn↓</b>	End measurement and select the next test step in the list (no start of new measurement)

Read also the section “Keyboard Operation” on page 50 for more details on keyboard control and manual operation.





You should activate the compressed view of the command center window (see page 39) by pressing the  $\emptyset$ -button above right:



(This also secures operation because the large window absorbs some of the key commands mentioned above).

In order to perform a test run proceed as follows:

1. Start a test cycle with **F5**.
2. Start driving. At the same time, check whether the rotational speed as displayed in the TasAlyser is correct. Find a suitable stretch of road.
3. Press **F6**, and select a “Drive” test step.
4. Drive and engage the appropriate gear.
5. When the measurement is to begin, press the space bar.
6. When you have reached your maximum speed, press **F4**.
7. When the vehicle is in idle load or coasting and/or you have reached the desired lower speed again, press the space bar once more.
8. If necessary, repeat 3 - 7 for other gears.
9. If necessary, press **F2** to enter a comment to your test run.
10. Press **F8** to terminate and store the test. You can also press **F8** after each gear and then press **F5**. You will then get an archive file for each gear.

---

## Storing the Results

Like the measurement program, the mobile version also stores measurement data archives and sound files. Read “Archiving in the TasAlyser” on page 153 with regard to measurement data archives. Here you will find out how to specify folders for storing the archives.



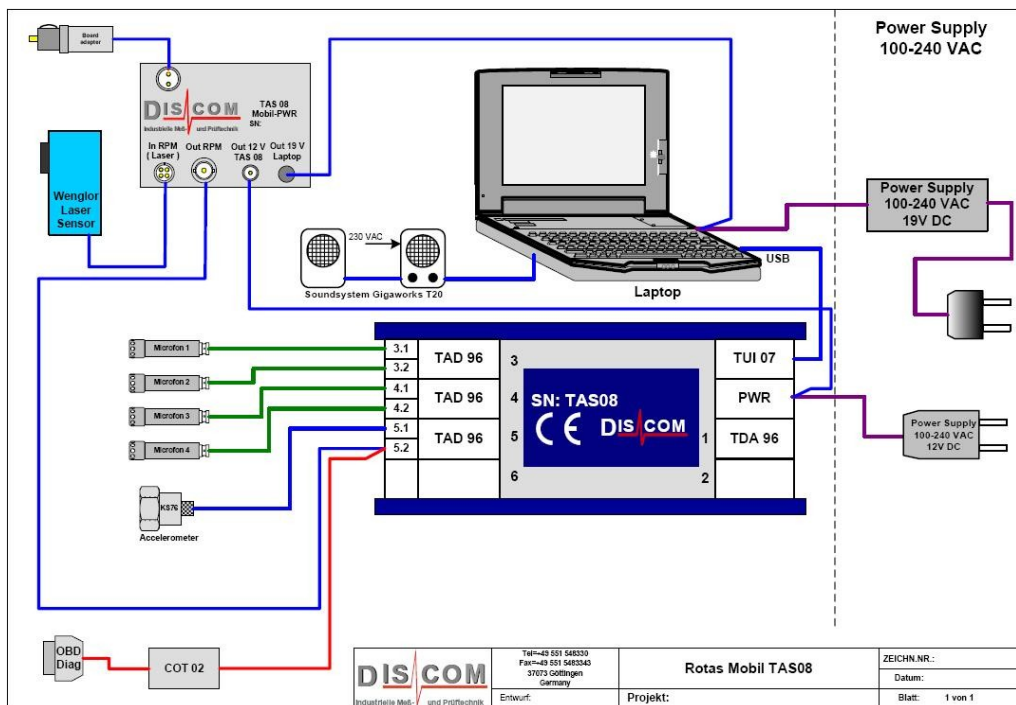
The recording of Wave files is explained in “Wave Recorder” from page 124 onwards. Open the Wave recorder’s setup dialog and select the folder for storing the sound files. Under **Options** we recommend that you activate the **Only When Measurement Is Running** and **Separate Test Steps** as well as the **Test Step** option under name construction. You will then get a sound file per test step (ramp) with an appropriate name.

Since you start the measurement manually during a mobile measurement (see previous section point 5), you also specify when recording should begin. You can begin recording before reaching the actual ramp rotational speed and then make comments into the microphones - this will, of course, be recorded at the same time and can be heard later.

In addition, you can make comments in the measuring data archives before terminating the test run - see points 9 and 10 of the previous section.

## Block Diagram

The block diagram gives an overview of the mobile system:





## Appendix B: Signal Processing

This chapter contains some general information on digital signal processing and other mathematical and physical concepts, which are of importance in noise analysis.

This is not a physics text book, so expositions are to be understood as being only references in case of ambiguity, and do not claim to be comprehensive or easily understandable.

---

### RMS, Crest, Kurtosis

The RMS value is calculated as the square root of the mean of the squares of the measured values (RMS = “root mean square”). For a noise signal the RMS value describes the total energy in the signal (per revolution, for rotationally synchronous signals). If the RMS value (similar to spectra) is converted into the dB scale (see next section), then the master volume is obtained.

The master volume is the RMS value of the mix channel or a fixed frequency channel in dB. If the RMS value of a rotationally synchronous averaged channel is calculated, then the “rotor master volume” is obtained. The *order level* is obtained from an order or order range in the spectrum (mix or synchronous).

### dB Scale

The decibel scale<sup>6</sup> converts a normal scale (e.g. sound pressure, acceleration) into a logarithmic scale. Namely, the relationship between the measured value and a reference value, the *logarithmic reference*, is considered.

If  $x$  is the measured value and  $x_0$  is the logarithmic reference, then the dB-value or *level*  $L(x)$  is defined as

$$L(x) = 10 \log_{10} \left( \frac{x}{x_0} \right)$$

Hence it follows that a difference of 10 dB between two values corresponds to a relationship of 1:10 between these values. If a spectral value in an order spectrum is 20 dB lower than another then one value is 100 times lower than the other. The logarithmic reference for structure-borne noise is usually 0.00001g; for airborne noise, 0.00002 Pa.

---

<sup>6</sup> The “Bel” comes from Alexander Graham Bell, the telephone pionier. The “decibel” has an additional factor of 10 before the logarithm.



## Crest

The Crest value is generated for a stretch of signal, by dividing the peak value by the mean. The Crest factor, therefore, indicates how far the highest point stands out from the signal.

The Crest value is particularly helpful in nick recognition, since nicks lead to individual points in the signal. The measured value formed in the TasAlyser is averaged over several revolutions and then maximized over the total ramp (testing time).

## Kurtosis

The kurtosis is the “fourth moment” in signal statistics. The more peaks a signal has and the higher these are, the higher will be the kurtosis. In this respect it is related to the Crest factor. However, whereas the Crest factor will decrease with the presence of many points, this is not the case with Kurtosis.

In the context of noise analysis Kurtosis can be imagined as a measure for “crackling”.

---

## Exponential Averaging

Whenever averaging takes place in the TasAlyser, we are almost always dealing with *moving means*. In other words, the mean is not calculated by adding up all the measured values and then dividing the sum by the number of measurements (the common *block average*), but instead the mean for the measured value  $x_{n+1}$  is calculated from the previous mean:

$$\text{Mean}_{n+1} = a \cdot \text{Mean}_n + (1 - a) \cdot x_{n+1}.$$

The factor  $a$  (which lies between 0 and 1) determines how strongly the current measured value  $x_{n+1}$  influences the new mean: the nearer  $a$  lies to 1, the weaker is the influence of  $x_{n+1}$ , and the more stable and less variable is the mean.

In the parameterization of this averaging process we usually do not specify the factor  $a$  but apply instead a time constant  $T$ . The significance of this time constant is that the influence of the old measurement value has sunk to 0.37 ( $= 1/e$ ) after  $T$  further measurements.

An example: if we specify a value of 10 (revolutions) for the averaging time constant of a rotationally synchronous averaging, then clearly more than 10 revolutions are included in the mean, because the influence of the revolution  $x_{n-10}$  on the current mean is 0.37, the influence of  $x_{n-20}$  (before 20 revolutions) is still 0.135.



## Filter Definition

### About Biquads

Every transfer function may be written as a rational function with real coefficients. According to the fundamental theorem of algebra, both polynomials of such a function can be divided to complex linear factors. These factors possess real-valued roots or constitute conjugated pairs with complex roots. The products of those pairs build up real quadratic terms. The quotient of such terms is called biquad (abbr. bi-quadratic) in filter jargon. In general, such a biquad is described by six coefficients, three for both the numerator and the denominator

The module's internal representation of filters is the representation as analog biquad, which means it lives in the frequency domain, before the application of the Z-transform.

### Partial filters

#### File-filter

For file-filters, clicking **Definition** leads to a file dialog. It accepts files with the extension `.adt`. As an example, examine a *leaky integrator*. It may be used to transform the signal of an acceleration sensor to a velocity signal. The transform function is

$$H(i\omega) = \frac{1}{i\omega + a} \quad \text{or} \quad H(s) = \frac{1}{s + a} \quad \text{with } s = i\omega + \sigma$$

The leaky term is  $a$ . For  $a = s_0$  this results in a attenuation of 3 dB with respect to the ideal integrator with  $a = 0$ . Comparing coefficients with the general description of a biquad

$$H(s) = \frac{b_0 + b_1s + b_2s^2}{c_0 + c_1s + c_2s^2} \quad \text{with real } b_n \text{ and } c_n, \text{ one found the coefficients as}$$

Coefficients/Index	0	1	2
$b$	1	0	0
$c$	$a$	1	0

The following shows a `*.adt` file defining this filter for  $s_0 = 2\pi$ , meaning a cut-off frequency of 1 Hz.

```
; Integration filter int1.adt
; 3dB Deviation from ideal integrator at 1Hz
```



```

; f0 = 1/(2\pi), b0=2\pi
;
;          b2*w^2+b1*w+b0
; biquad = -----
;          c2*w^2+c1*w+c0
;
;      Format: after " DATA":
;      nrsections
;      factor
;      f0 b2 b1 b0 c2 c1 c0
;
DATA
1
1
0.159154943 0 0 1 0 1 6.283185307

```

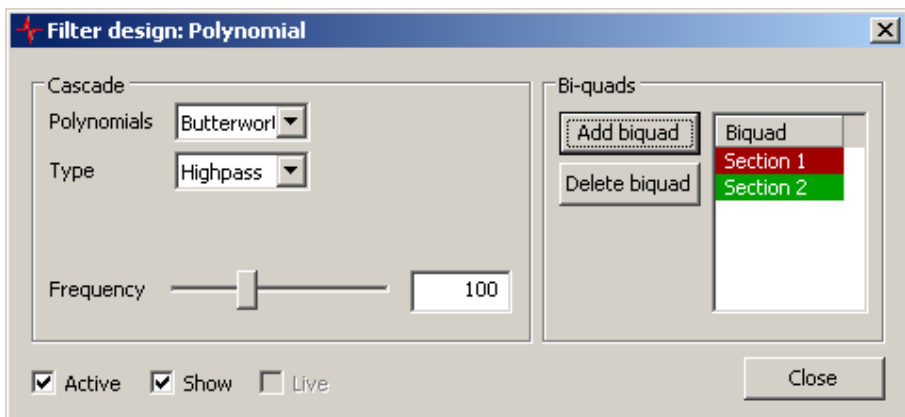
These files describe biquads in normalized frequencies  $\Omega = 2\pi f/f_g$ . By setting  $f_0 = 1/2\pi$ , the frequencies are normalized to angular frequencies  $\omega = 2\pi f$ . So they are in the domain of the just found coefficients.

Another example is the file `C:\Discom\Measurement\Common\DinAbq.adt` which defines the *A*-filter.

## Polynomial filters

Polynomial filters are filters build from the classical filter polynomials. At the time, only Butterworth polynomials are implemented. These show a maximal flat amplitude response at the pass-band. You can choose between low-pass and high-pass filters of even order. Band-pass or band-stop filters are set up by cascading the first ones.

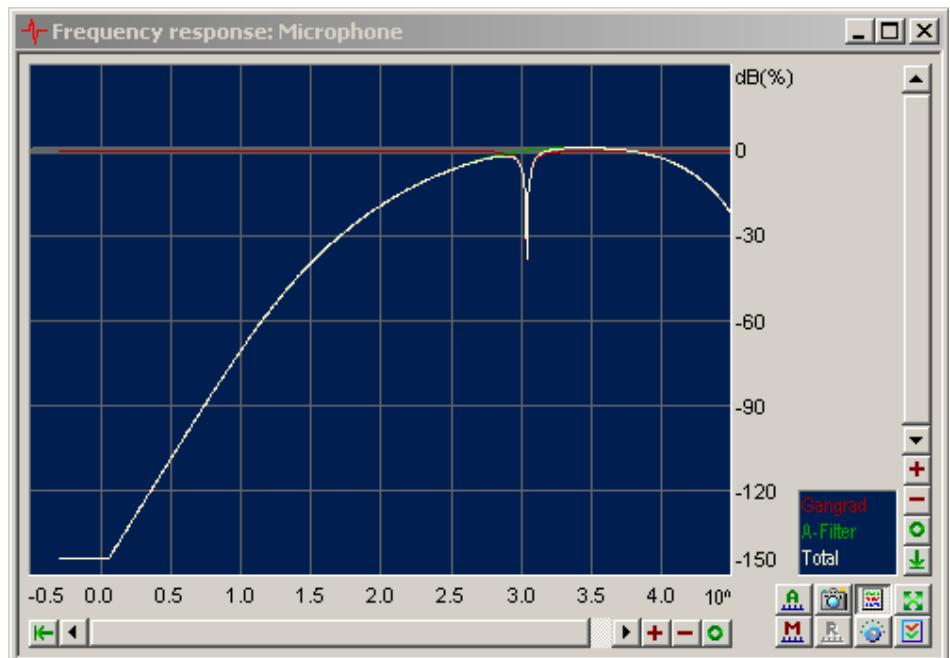
If you click the **Definition** property of a polynomial filter at the **Filter** tab, the according design dialog pops up.





Every filter needs at least one biquad. You add biquads by pressing the button **Add biquad**. Existing biquads show up at the list on the right. Each biquad raises a filter's order by two. A biquad may be dismissed by choosing it from the list and pressing **Delete biquad**. (For polynomial filters it doesn't matter which biquads are deleted, since the remaining ones are recalculated anyway.) Further, a polynomial filter needs to know the family of **Polynomials** to use (Butterworth only yet), which **Type** the filter should be (low-pass/high-pass) and what the cut-off **Frequency** is.

To get an overview of the transfer function, check **Show**. This pops up a scope showing the transfer functions of all biquads and their total.



The check-box **Active** switches the filter on and off and serves for the monitoring of the filter's effect. This change is *adiabatic*, so its effect is immediate. In addition, this change is not persistent. To activate or deactivate a partial filter permanently, you have to use the corresponding property at the **Filter groups** tab. (The check-box does the same as activation of a partial filter by clicking it at the "little" dialog.)

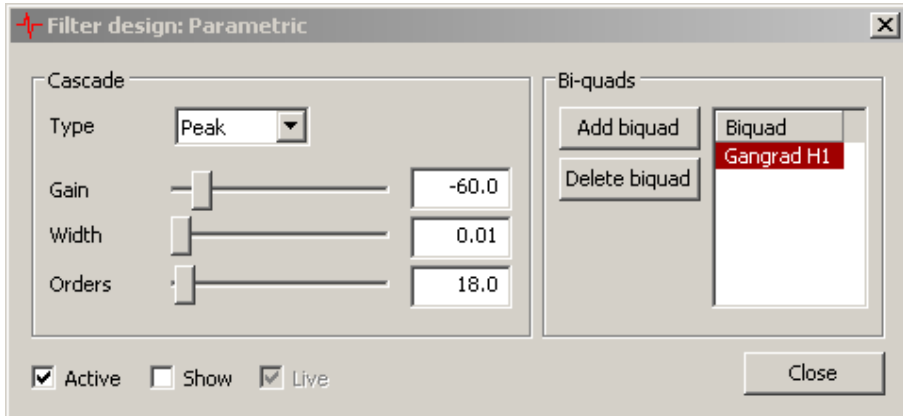
Moreover, the dialog shows a disabled check-box **Live**. It indicates whether the filter's settings and views are in synch with the module by adiabatic changes, or if it is off synch by a preceding non-adiabatic change like the addition or the removal of a biquad. The dialog is synchronized by applying or cancelling non-adiabatic changes.



## Parametric filters

Parametric filters here work like the ones known as parametric equalizers from PA- or studio equipment. They allow for the amplification or attenuation of arbitrary frequencies.

Clicking the **Definition** property of a parametric filter opens its design dialog.



Several controls are analogous to the controls of the design dialog for polynomial filters and may be looked up there. The different controls are explained in the following.

**Type** defines the kind of parametric filter. At the time, only the described equalizer filters are implemented and named **Peak**. These filters amplify a specified **Frequency** or **Order** by an arbitrary positive or negative gain (in dB) with a **Width**, which is essentially the inverse  $Q$ -Factor of the filter.

In opposition to the polynomial filters, the parameters may be chosen independently for every biquad. Default is the first one, the others may be chosen by selecting them from the list.

Additionally may the biquads here be named. To name a biquad, select it from the list and click it once again. Now, the list field is editable.

## Valid parameters

If a filter's parameter is invalid, e.g. has it a cut-off frequency of zero, it is not applied. It is *not* deactivated, so you can't recognize that at a dialog.

More over, the frequency range is limited. This is especially important for filters with speed reference: If the according speed is so low or high that the corresponding cut-off frequency exceeds  $[0.00001, 0.49999] \cdot f_s$ , the filter "sticks" at these limits. Dependent of its parameters, a filter may show unde-





irable behavior but away from these frequencies. So it is recommended to control the transfer functions at least for the expected speed interval.

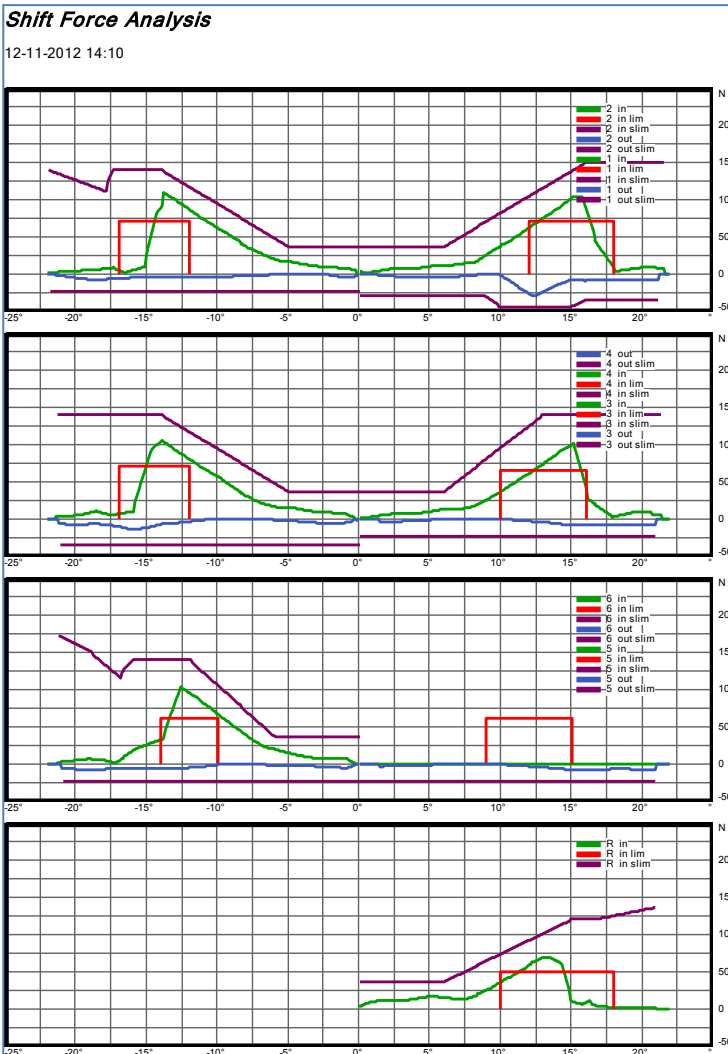


# Appendix D: Shift Force Evaluation

## Overview

Rotas shifting force evaluation uses the position and force signals of a shifting robot or similar mechanism to construct force-position-curves. These curves can be evaluated in two different ways to detect hard shiftability problems as well as missing synchron rings.

The system constructs separate curves for gear insertion and gear removal (gear in and gear out). In the example picture gear in curves are shown in green and gear out curves in blue.



Gear in curves are tested against an upper limit curve (pink curves in picture), thus detecting hard shiftability problems. The limit curves are prepared in the parameter data base and can be different for each gear.

Gear in curves are additionally checked for a minimum force in a certain position interval (red rectangle curves in picture). If this minimum force is not reached, a missing synchron ring is reported.



---

## Setup

### Input signals

The system needs as input signals the (voltage) signals from position and force sensors. Maximum voltage is +/- 25 Volts. Smaller ranges are amplified accordingly.

Positions can be measured as distance (cm, mm) or angle (degrees). It is not necessary that the position signal is Zero for the neutral position. Instead, you have to enter the position value for the neutral position in TasAlyser.

Force should be zero for neutral position, although an overall offset can be corrected by calibration. Direction of force (positive/negative values) does not matter.

The TasAlyser measurement application includes a calibration function, where the position and force signal calibration factors can be set up and checked. (Please refer to page 147ff for details.)

### Test stand commands

The test stand has to inform the measurement system about start and end of shifting operations.

Immediately before start of a shifting operation, the test stand has to send the command

SGW: a b

where  $a$  and  $b$  are gear numbers. For example, the command SGW: 3 4 initiates a shifting operation from gear 3 to gear 4. The reverse gear uses the letter R, the neutral position can be addressed as N or 0 (Zero). Thus, to start a shift force measurement from neutral position to reverse gear, the command is SGW: 0 R or SGW: N R.

The measurement system acknowledges the SGW command by sending 1 as a reply. (The reply is 0 if the measurement could not be started, for example because there was already a shifting operation in progress.)

After the shifting operation is completed, the test stand has to send the command

EGW:

This command has no parameters. The reply is 1 if the shift force measurement did not show any problems and 0 in case of any not OK results (hard shiftability, missing synchron ring).

## Display in TasAlyser and Presentation

A normal shifting operation consists of a continuous movement from one gear into the other. The system automatically separates this measurement into gear out and gear in curves. It is not necessary to tell TasAlyser in advance whether the force or position signals for one gear or the other will be positive or negative, because TasAlyser automatically concludes this information from the *SGW* command and the actual measured values.

TasAlyser converts the position data for each curve in such a way that position always runs from zero to positive, and force data in such a way that it is always positive. This makes limit parameterization (see next section) much easier because the user does not need to take into account the real positions (left or right) of gears on the shifting rod.



To produce a compact display, the gear out curves are mirrored to negative force values as shown in the picture besides. In this way, both curves for each gear can be placed into one graph.

Then, the curves for odd gears (1, 3, 5) are mirrored to the left, as can be seen on the previous page. Limit curves are mirrored together with measured curves.

As stated before, this mirroring serves only the purpose to produce a compact display. For evaluation, position and force are always positive.

---

## Evaluation methods

The shift force curves are evaluated for hard shiftability and missing synchron rings.

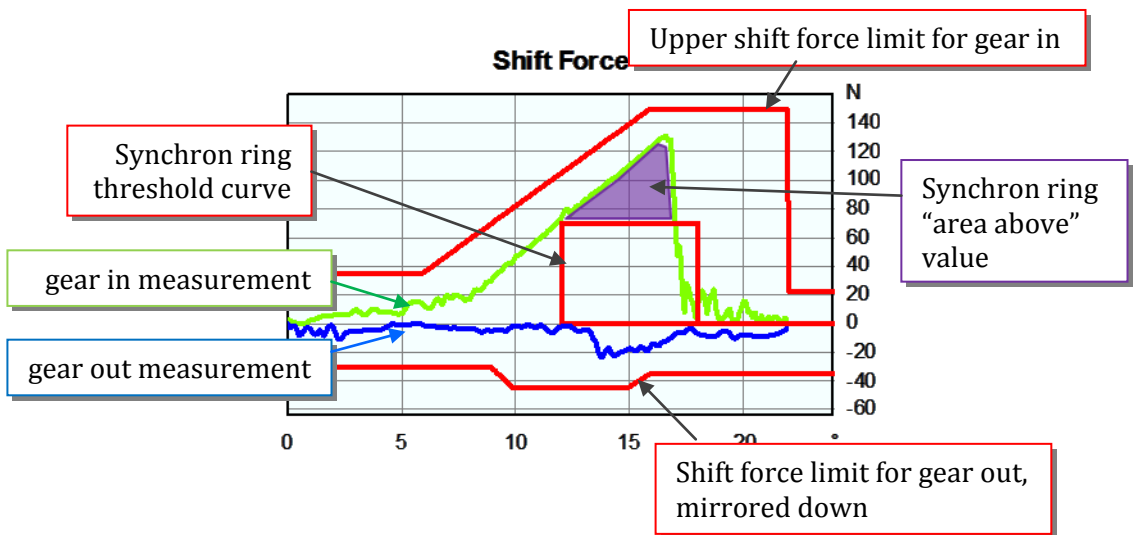
### Hard shiftability

To test for hard shiftability, gear in as well as gear out curves are checked against upper limit curves. (As explained in the previous section, in the display the gear out curves and limits are mirrored to negative forces, so the limit curves for gear out *looks like a lower limit.*) The limit curves are defined as polygons in the parameter data base (see below).



## Missing synchron rings

The presence of the synchron ring shows in the gear in curve as a characteristic force necessary to insert the gear (see picture below). The presence of this “force hill” in the curve is checked by defining a minimum force value for a certain position range (threshold curve, see picture). Then TasAlyser calculates the area of the measured curve *above the threshold curve*. This value is checked against a *lower limit*. If the “force hill” is missing or too small (indicating a missing synchron ring), the area above is small or zero and the test against the lower limit fails.



Just for clarity: the hard shiftability check uses a *limit curve*, the synchron ring a *limit value* for a *single value quantity*, which is the area above the threshold curve.



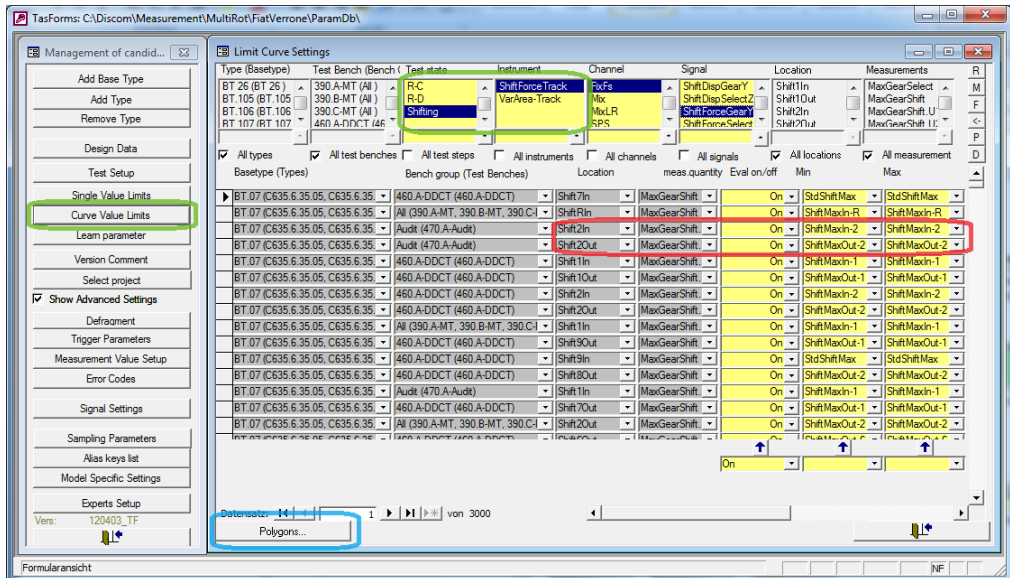
## Limit setup

The limit curves for hard shiftability and the threshold curves for synchron rings are defined as polygons in the parameter data base. (Because the measured curves for different transmissions tend to vary somewhat in the position direction, we recommend not using learned limit curves here.)

### Hard shiftability limit curves

As for all limit curves, the hard shiftability limit curves can be found in the limit curves section of the parameter data base (see picture on next side).

In the main form, select **Curve Value Limits** (marked green in the picture). In the Limit Curve Settings form, select the *test step* **Shifting** and the *instrument* **ShiftForceTrack** to reduce the list to the entries for the hard shiftability limits:



The limit curve polygons for each gear and direction (labeled *Location* in the list) are selected as minimum *and* maximum polygon.

To edit these polygons, press the **Polygons...** button in the lower left. Alternatively, you can use Talimer to edit these polygons.



## Synchron ring thresholds and limits

Because the threshold curves are not limit curves, they have to be editet in the **Measurement Value Setup** section of TasForms:

The screenshot shows the TasForms software interface. The main window is titled "Measurement value setup" and contains a table of evaluation parameters. The "Test state" column is set to "Shifting" and the "Instrument" column is set to "SvTrack Poly". The "Measurements" column lists "Synch1357", "Synch2", "Synch246R", and "Synch4".

The "Edit Measurement Definition" window is open, showing the "Instrument" set to "SvTrack Polygon". The "Measurements" column lists "Synch1357", "Synch2", "Synch246R", and "Synch4". The "Polygons" column is set to "Polygon".

The "Edit Measurement Definition" window also contains a table of instrument measurements:

Instrument	Measurements	Evaluation Mode	Trigger	Interpretation	Source Instrument	Source Parameter	Polygon
SvTrack Poly	Synch1357	Min	Standard	AreaAbove	ShiftForce Track	MaxGearShift	ShiftIn-1357
SvTrack Poly	Synch246R	Min	Standard	AreaAbove	ShiftForce Track	MaxGearShift	ShiftIn-2468
SvTrack Poly	SynchR	Min	Standard	AreaAbove	ShiftForce Track	MaxGearShift	ShiftIn-R
SvTrack Poly	Synch5	Min	Standard	AreaAbove	ShiftForce Track	MaxGearShift	ShiftIn-5

In the Measurement value setup form, select *test step* **Shifting** and the *instrument* **SvTrackPolygon** to only show the entries for synchron ring test. Press the **Measurements...** button in the lower left of tis form. Here you will find the predefined threshold settings. To change the threshold polygons, press **Polygons...** in the Edit Measurement Definition form. (Again, you also may use Talimer to change these polygons.)

The *limits* for the calculated area-above values are set (as usual) in the **Single Value Limits** form. Again, select the *test step* **Shifting** to find the appropriate entries. Remember: these are lower limits.