

Condition monitoring

User's manual and data sheet X20(c)CM4810

Version: **2.21 (February 2021)**
Order no.: **MAX20CM4810-ENG**

Translation of the original manual

All values in this manual are current as of its creation. We reserve the right to change the contents of this manual without notice. B&R Industrial Automation GmbH is not liable for technical or editorial errors and defects in this manual. In addition, B&R Industrial Automation GmbH assumes no liability for damages that are directly or indirectly attributable to the delivery, performance or use of this material. We point out that the software and hardware designations and brand names of the respective companies used in this document are subject to general trademark, brand or patent protection.

1 General information.....	7
1.1 Coated modules.....	7
2 Order data.....	8
3 Technical data.....	9
3.1 Technical data.....	9
3.2 LED status indicators.....	10
3.3 Firmware update time.....	10
3.4 Pinout.....	10
3.5 Input circuit diagram.....	11
3.6 Shielding.....	11
3.7 Derating.....	11
3.8 Gain curve.....	14
3.9 Settling time.....	14
3.10 Sensor sensitivity.....	15
3.11 Using a B&R Compact CPU or fieldbus CPU.....	15
4 Register description.....	16
4.1 General data points.....	16
4.2 Function model 0 - Standard.....	16
4.3 Function model 1 - Fast master.....	19
4.4 Function model 2 - Slow master.....	21
4.5 Function model 254 - Bus controller.....	23
4.5.1 Using the module on the bus controller.....	24
4.5.2 CAN I/O bus controller.....	24
4.6 General information.....	25
4.6.1 Signal generation.....	25
4.6.2 Filter configuration.....	25
4.6.3 Frequency bands.....	26
4.6.4 Calculating the velocity signal automatically.....	26
4.6.5 Autogain, AutogainDelay and overflow.....	26
4.6.6 Term definition: Sampling rate and sampling frequency.....	26
4.7 General registers.....	27
4.7.1 ActSpeed.....	27
4.7.2 AutogainDelay.....	27
4.7.3 AutogainDelayRead.....	27
4.7.4 Control.....	28
4.7.5 SensorConfig.....	29
4.7.6 SensorConfigRead.....	29
4.7.7 Status.....	30
4.8 Analog input functions.....	31
4.8.1 Measured values.....	31
4.8.2 Characteristic value calculation in AnalogInput.....	31
4.8.2.1 Continuous mode with enable (continuous mode).....	31
4.8.2.2 Trigger mode (single shot).....	32
4.8.3 AnalogInput.....	32
4.8.4 AnalogInputConfig.....	33
4.8.5 AnalogInputConfigRead.....	33
4.8.6 AnalogInputControlByte.....	34
4.8.7 AnalogInputSamples.....	34
4.8.8 AnalogInputScale.....	35
4.8.9 AnalogInputScaleRead.....	35
4.8.10 SamplesAnalogInput.....	36
4.8.11 SamplesAnalogInputRead.....	36
4.9 Automation Runtime support.....	37
4.9.1 DataConsistentWithLockedBuffers.....	37

4.9.2 DataToggleBit.....	37
4.9.3 OverflowAnalogInput.....	38
4.9.4 OverflowCharacteristicValues.....	38
4.9.5 OverflowFrequencyBands.....	39
4.9.6 PeakHighFrequencyRef.....	39
4.9.7 PeakHighFrequencyRefCalculated.....	39
4.9.8 PeakRawRef.....	39
4.9.9 PeakRawRefCalculated.....	39
4.9.10 RmsHighFrequencyRef.....	40
4.9.11 RmsHighFrequencyRefCalculated.....	40
4.9.12 RmsRawRef.....	40
4.9.13 RmsRawRefCalculated.....	40
4.9.14 SensitivitySensor.....	40
4.10 Characteristic values.....	41
4.10.1 Sum of maximum value.....	42
4.10.1.1 PeakHighFrequency.....	42
4.10.1.2 PeakRaw.....	42
4.10.2 RMS value.....	43
4.10.2.1 Iso10816.....	43
4.10.2.2 RmsAccEnvelope.....	44
4.10.2.3 RmsAccRaw.....	44
4.10.2.4 RmsHighFrequency.....	44
4.10.2.5 RmsRaw.....	45
4.10.2.6 RmsVelEnvelope.....	45
4.10.2.7 RmsVelRaw.....	45
4.10.3 Crest factor.....	46
4.10.3.1 CrestFactorHighFrequency.....	46
4.10.3.2 CrestFactorRaw.....	46
4.10.4 K(t) value.....	47
4.10.4.1 Vdi3832KtHighFrequency.....	47
4.10.4.2 Vdi3832KtRaw.....	48
4.10.5 Kurtosis.....	48
4.10.5.1 KurtosisRaw.....	48
4.10.6 Skewness factor.....	49
4.10.6.1 SkewnessRaw.....	49
4.10.7 FrequencyBand.....	49
4.11 Characteristic values (minimum and maximum).....	50
4.11.1 CrestFactorRawMax.....	51
4.11.2 CrestFactorRawMin.....	51
4.11.3 Iso10816Max.....	51
4.11.4 Iso10816Min.....	51
4.11.5 KurtosisRawMax.....	51
4.11.6 KurtosisRawMin.....	52
4.11.7 MinMaxCounter.....	52
4.11.8 PeakHighFrequencyMax.....	52
4.11.9 PeakHighFrequencyMin.....	52
4.11.10 PeakRawMax.....	52
4.11.11 PeakRawMin.....	53
4.11.12 RmsAccEnvelopeMax.....	53
4.11.13 RmsAccEnvelopeMin.....	53
4.11.14 RmsAccRawMax.....	53
4.11.15 RmsAccRawMin.....	53
4.11.16 RmsHighFrequencyMax.....	53
4.11.17 RmsHighFrequencyMin.....	54
4.11.18 RmsRawMax.....	54
4.11.19 RmsRawMin.....	54
4.11.20 RmsVelEnvelopeMax.....	54

4.11.21 RmsVelEnvelopeMin.....	54
4.11.22 RmsVelRawMin.....	55
4.11.23 RmsVelRawMax.....	55
4.11.24 SkewnessRawMax.....	55
4.11.25 SkewnessRawMin.....	55
4.12 Frequency band configuration registers.....	56
4.12.1 HighFrequencyConfig.....	56
4.12.2 HighFrequencyConfigRead.....	56
4.12.3 MaxFrequencyEnvelope.....	56
4.12.4 MaxFrequencyEnvelopeRead.....	57
4.12.5 MaxFrequencyRaw.....	57
4.12.6 MaxFrequencyRawRead.....	57
4.12.7 MinFrequencyEnvelope.....	58
4.12.8 MinFrequencyEnvelopeRead.....	58
4.12.9 MinFrequencyRaw.....	59
4.12.10 MinFrequencyRawRead.....	59
4.13 Frequency bands.....	60
4.13.1 Broadband RMS value.....	61
4.13.2 Speed-dependent RMS value.....	62
4.13.3 Noise.....	62
4.13.4 Configuration.....	63
4.13.5 FrequencyBandMax.....	63
4.13.6 FrequencyBandMin.....	63
4.13.7 FrequencyBandConfig.....	64
4.13.8 FrequencyBandConfigRead.....	64
4.13.9 FrequencyBandDmgFreq60rpm.....	65
4.13.10 FrequencyBandDmgFreq60rpmRead.....	65
4.13.11 FrequencyBandTolerance.....	65
4.13.12 FrequencyBandToleranceRead.....	65
4.13.13 FrequencyBandLowerFrequency.....	65
4.13.14 FrequencyBandLowerFrequencyRead.....	66
4.13.15 FrequencyBandUpperFrequency.....	66
4.13.16 FrequencyBandUpperFrequencyRead.....	66
4.14 Flatstream.....	67
4.14.1 Transferring characteristic values via Flatstream.....	67
4.14.1.1 Registers for the characteristic value Flatstream.....	67
4.14.1.2 Structure of the characteristic value Flatstream.....	67
4.14.2 Uploading buffers via Flatstream.....	68
4.14.2.1 Registers for the buffer Flatstream.....	68
4.14.2.2 Buffer upload procedure.....	68
4.14.3 The Forward function on the X20CM4810.....	69
4.14.4 Flatstream communication.....	70
4.14.4.1 Introduction.....	70
4.14.4.2 Message, segment, sequence, MTU.....	71
4.14.4.3 The Flatstream principle.....	72
4.14.4.4 Registers for Flatstream mode.....	73
4.14.4.5 Example of Forward functionality on X2X Link.....	93
4.15 Using the module with on the fieldbus.....	98
4.15.1 Bus controller with FieldbusDESIGNER support.....	98
4.15.2 Bus controller without FieldbusDESIGNER support.....	98
4.15.3 B&R SG4 CPU with an interface module.....	98
4.15.4 CANIO bus controller.....	98
4.16 Assignment of the task class.....	98
4.17 Maximum cycle time.....	98
4.18 Minimum cycle time.....	98

5 Condition monitoring / Oscillation analyses.....	99
5.1 Basic information.....	99
5.1.1 What is condition monitoring?.....	99
5.1.1.1 Bathtub / Deterioration curve.....	100
5.1.1.2 Damage development and the damage chain.....	101
5.1.2 Conventional condition monitoring.....	102
5.1.3 Overview of condition monitoring methods.....	102
5.2 Oscillation measurement technology.....	103
5.2.1 Sensor technology.....	103
5.2.1.1 Basic design.....	103
5.2.1.2 Influencing variables on the sensor.....	103
5.2.1.3 Installing sensors.....	106
5.2.2 Oscillations - Overview of measuring structure-borne sound.....	108
5.2.2.1 Oscillations.....	108
5.2.2.2 Fast Fourier transforms (FFT).....	112
5.2.2.3 Envelope.....	120
5.2.2.4 Displacement, velocity and acceleration.....	123
5.2.3 Determining limits and alarm limits.....	123
5.2.3.1 Comparison with references and norms.....	123
5.2.3.2 Manufacturer's limits.....	123
5.2.3.3 Operator's limits - Experience-based values.....	124
5.2.3.4 Assessing the trend.....	124
5.2.3.5 Dynamic speed change.....	125
5.3 Practical applications of damage recognition.....	127
5.3.1 Characteristic values.....	128
5.3.2 Potential failures.....	129
5.3.2.1 Imbalance.....	130
5.3.2.2 Misalignment.....	132
5.3.2.3 Belt damage.....	133
5.3.2.4 Loose or striking parts.....	136
5.3.2.5 Slide bearing damage.....	137
5.3.2.6 Roller bearing damage.....	137
5.3.2.7 Gear damage.....	146
5.3.2.8 Electrical errors.....	148
5.3.3 Typical applications of damage recognition.....	149
5.3.3.1 Fan with rigid coupling.....	150
5.3.3.2 Fan with countershaft.....	152
5.3.3.3 Fan with belt drive.....	154
5.3.3.4 Directly coupled pump.....	157
5.3.3.5 Gearbox.....	159
5.3.3.6 Gearbox with countershaft.....	161
5.4 Further reading.....	163
6 Accessories.....	164
6.1 Sensors.....	164
6.1.1 0ACS100A.00-1.....	164
6.1.1.1 Order data.....	164
6.1.1.2 Technical data.....	165
6.1.1.3 Dimensions.....	165
6.1.1.4 Installation direction.....	166
6.1.2 0ACS100A.90-1.....	167
6.1.2.1 Order data.....	167
6.1.2.2 Technical data.....	167
6.1.2.3 Dimensions.....	168
6.1.2.4 Installation direction.....	168
6.1.3 General information.....	169
6.1.3.1 Pinout.....	169

6.1.3.2 Frequency response.....	169
6.2 Sensor cables.....	170
6.2.1 Order data.....	170
6.2.2 Technical data.....	170
6.2.3 Sensor cables with female M12 connector.....	170
6.2.4 Cable diagram.....	171

1 General information

This compact module provides a system for measuring and analyzing vibrations for use in applications such as condition monitoring for machinery and equipment.

The measurement unit consists of 4 vibration inputs with 24-bit converter resolution and a sampling frequency of 51.562 kHz. Each of the inputs has a separately activated constant current source to supply IEPE sensors.

Signal conditioning algorithms integrated in the module include configurable high and low-pass filters, an envelope curve generator and a Fourier transform, among many others.

Various parameters and configurable frequency bands are available to ensure seamless signal evaluation. The module performs calculations internally to minimize the load on the bus and CPU.

Whether used for traditional condition monitoring or as a means of process optimization, the maximum evaluable frequency of 10 kHz and maximum frequency resolution of under 63 mHz make it suitable for almost any application. If desired, it can also be used as a standard input module.

The ability to upload all values in the time and frequency domain also allow measurements to be evaluated externally or archived.

Information:

The X20CM4810 module is supported starting with Automation Studio 3.0.90.x and Automation Runtime versions \geq J3.09, J4.01 and O4.02.

- 4-channel vibration measurement and analysis module
- 24-bit resolution at 51.562 kHz
- IEPE supply
- Internal calculation of an extensive range of condition parameters
- Data buffer upload

1.1 Coated modules

Coated modules are X20 modules with a protective coating for the electronics component. This coating protects X20c modules from condensation and corrosive gases.

The modules' electronics are fully compatible with the corresponding X20 modules.

For simplification purposes, only images and module IDs of uncoated modules are used in this user's manual.

The coating has been certified according to the following standards:

- Condensation: BMW GS 95011-4, 2x 1 cycle
- Corrosive gas: EN 60068-2-60, method 4, exposure 21 days



2 Order data

Model number	Short description	Figure
	Other functions	
X20CM4810	X20 analog input module, for vibration measurement and analysis of condition monitoring tasks, 4 IEPE analog inputs, 51.5625 kHz sampling frequency, 24-bit converter resolution	
X20cCM4810	X20 analog input module, coated, for vibration measurement and analysis of condition monitoring tasks, 4 IEPE analog inputs, 51.5625 kHz sampling frequency, 24-bit converter resolution	
	Required accessories	
	Bus modules	
X20BM31	X20 bus module for double-width modules, 24 VDC keyed, internal I/O supply continuous	
X20cBM31	X20 bus module, coated, for double-width modules, 24 VDC keyed, internal I/O power supply connected through	
	Terminal blocks	
X20TB12	X20 terminal block, 12-pin, 24 VDC keyed	
	Optional accessories	
	Sensor cable	
0ACC0020.01-1	Cable for accelerometer, length 2 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0050.01-1	Cable for accelerometer, length 5 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0100.01-1	Cable for accelerometer, length 10 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0150.01-1	Cable for accelerometer, length 15 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0200.01-1	Cable for accelerometer, length 20 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0500.01-1	Cable for accelerometer, length 50 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC1000.01-1	Cable for accelerometer, length 100 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
	Sensors	
0ACS100A.00-1	Accelerometer, nominal sensitivity 100 mV/g, top exit	
0ACS100A.90-1	Accelerometer, nominal sensitivity 100 mV/g, side exit	

Table 1: X20CM4810, X20cCM4810 - Order data

3 Technical data

3.1 Technical data

Model number	X20CM4810	X20cCM4810
Short description		
I/O module	X20 4-channel analog input module for vibration measurement and analysis of condition monitoring tasks	
General information		
Insulation voltage between channel and bus	500 V _{eff}	
Nominal voltage	24 VDC ±20%	
B&R ID code	0xC8F9	0xE7F0
Status indicators	Run, error, vibration inputs 1 to 4	
Diagnostics		
Module run/error	Yes, using LED status indicator and software	
Power consumption		
Bus	0.01 W	
Internal I/O	2.5 W	
Certifications		
CE	Yes	
ATEX	Zone 2, II 3G Ex nA nC IIA T5 Gc IP20, Ta (see X20 user's manual) FTZÜ 09 ATEX 0083X	
UL	cULus E115267 Industrial control equipment	
HazLoc	cCSAus 244665 Process control equipment for hazardous locations Class I, Division 2, Groups ABCD, T5	
EAC	Yes	
Analog inputs		
Quantity	4	
Input type	IEPE sensor: Acceleration	
Digital converter resolution	24-bit	
Open-circuit detection		
Per channel	Yes	
At minimum supply voltage ¹⁾	Starting at 17 V (or greater) for more than 1 ms	
At nominal supply voltage ²⁾	Starting at 21.3 V (or greater) for more than 1 ms	
At maximum supply voltage ³⁾	Starting at 25.5 V (or greater) for more than 1 ms	
Permissible input signal	±10 VAC	
Conversion procedure	Sigma-delta	
Type	Vibration input	
Sampling frequency	51.5625 kHz	
Input high pass cutoff frequency	34 mHz	
Input low pass cutoff frequency	19.75 kHz	
Downsampling	200 Hz, 500 Hz, 1 kHz, 2 kHz, 5 kHz, 10 kHz (configurable)	
Frequency resolution of spectra	0.0629 Hz, 0.1574 Hz, 0.3147 Hz, 0.6294 Hz, 1.5736 Hz, 3.1471 Hz	
Sensor power supply	IEPE, 5 mA constant current source (4.9 - 5.5 mA), can be switched off for each channel	
Electrical properties		
Electrical isolation	Channel isolated from bus Channel not isolated from channel	
Operating conditions		
Mounting orientation		
Horizontal	Yes	
Vertical	Yes	
Installation elevation above sea level		
0 to 2000 m	No limitation	
>2000 m	Reduction of ambient temperature by 0.5°C per 100 m	
Degree of protection per EN 60529	IP20	
Ambient conditions		
Temperature		
Operation		
Horizontal mounting orientation	-25 to 50°C	
Vertical mounting orientation	-25 to 45°C	
Derating	See section "Derating".	
Storage	-40 to 85°C	
Transport	-40 to 85°C	

Table 2: X20CM4810, X20cCM4810 - Technical data

Technical data

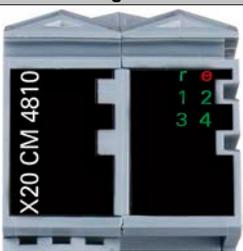
Model number	X20CM4810	X20cCM4810
Relative humidity		
Operation	5 to 95%, non-condensing	Up to 100%, condensing
Storage	5 to 95%, non-condensing	
Transport	5 to 95%, non-condensing	
Mechanical properties		
Note	Order 1x terminal block X20TB12 separately. Order 1x bus module X20BM31 separately.	Order 1x terminal block X20TB12 separately. Order 1x bus module X20cBM31 separately.
Pitch	25 ^{+0.2} mm	

Table 2: X20CM4810, X20cCM4810 - Technical data

- 1) Input voltage: 19.2 V
- 2) Input voltage: 24 V
- 3) Input voltage: 28.8 V

3.2 LED status indicators

For a description of the various operating modes, see section "Additional information - Diagnostic LEDs" in the X20 system user's manual.

Figure	LED	Color	Status	Description	
	r	Green	Off	No power to module	
			Single flash	RESET mode	
			Double flash	BOOT mode (during firmware update) ¹⁾	
			Blinking	PREOPERATIONAL mode	
	e	Red	On	RUN mode	
			Off	No power to module or everything OK	
	e + r		Red on and green single flash	Warning, error or reset status	
	1 - 4		Green	On	Status of the respective acceleration sensor (no open circuit)

- 1) Depending on the configuration, a [firmware update](#) can take up to several minutes.

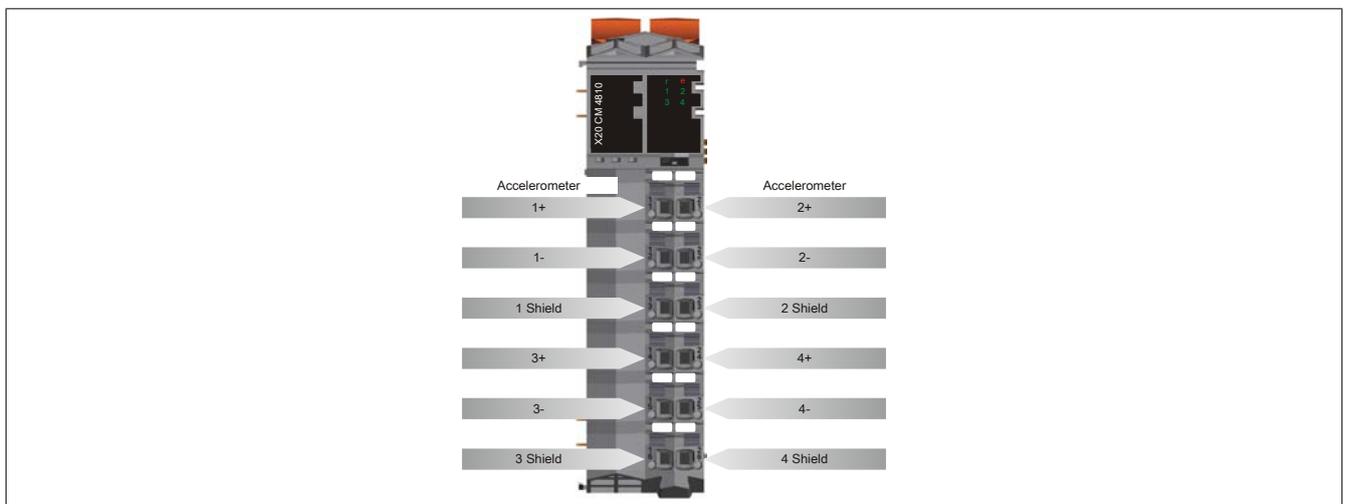
3.3 Firmware update time

Due to the large size of the firmware, the firmware update takes some time.

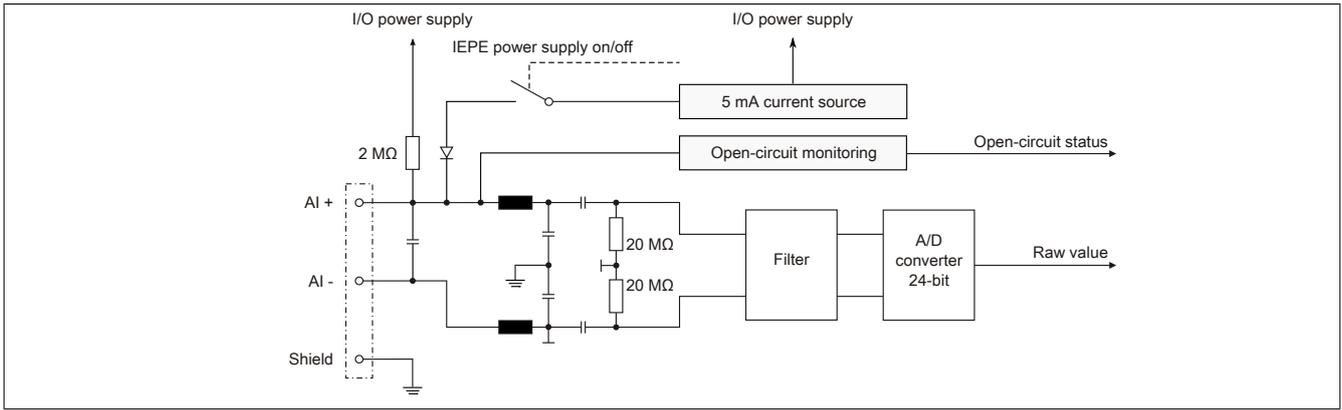
The following firmware update times can be expected depending on the configured bus cycle time:

Bus cycle time	Update time
400 μ s	Approx. 3 min
2 ms	Approx. 15 min

3.4 Pinout



3.5 Input circuit diagram



3.6 Shielding

There are 2 ways to shield the cable for the acceleration sensors:

- Shielding possibilities on the terminal block. (Contact "Acc. sensor 1 - 4 shield")
- X20 shielding on the bus module. (See the "Shielding" section in the X20 system user's manual)

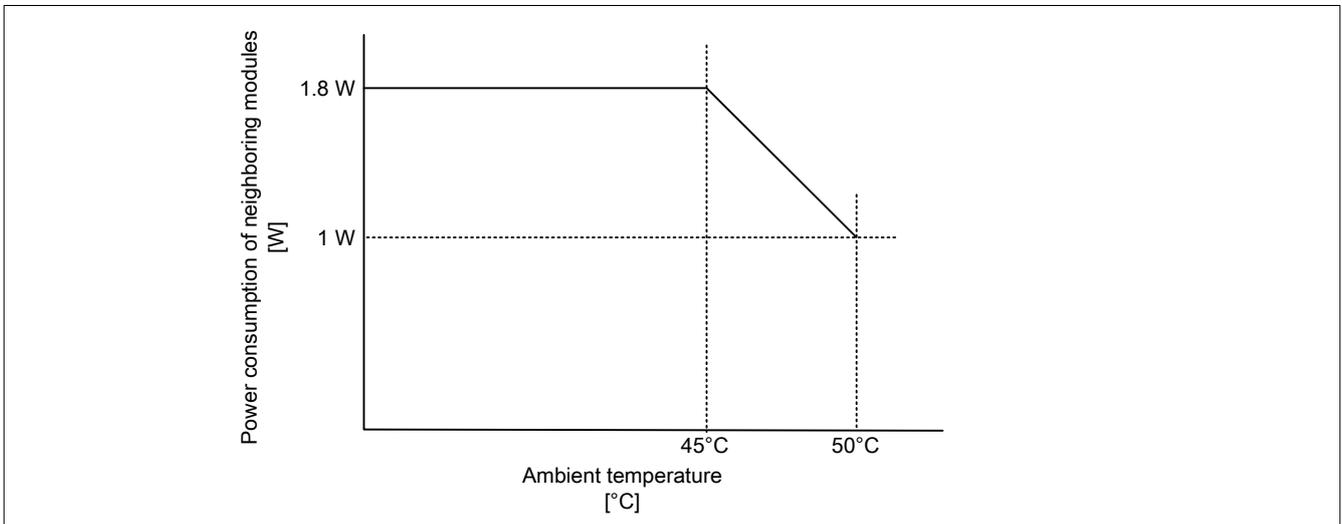
Shielding via the terminal block is perfectly sufficient for environments with no special EMC requirements.

However, if the module is used in an environment with special EMC requirements and high frequency disturbances, both options should be used together.

3.7 Derating

Horizontal installation

At ambient temperatures higher than 45°C, the X20CM4810 requires power derating:



If the X20CM4810 is to be operated with ambient temperatures up to 50°C, then the neighboring modules must not consume more than 1 W.

For an example of calculating the power dissipation of I/O modules, see section "Mechanical and electrical configuration - Power dissipation of I/O modules" in the X20 user's manual.

Example: ambient temperature up to 50°C

Technical data

Power consumption 1.8 W
Power consumption 1.8 W
Power consumption 1 W
X20CM4810
Power consumption 1 W
Power consumption 1.8 W
Power consumption 1.8 W

X20CM4810
Power consumption 1.8 W
Power consumption 1.8 W
Power consumption 1.8 W

Power consumption 1.8 W
X20CM4810
X20 CPU e.g. X20CP1486

X20 double-width modules are an exception. It is also possible, for example, to place multiple X20CM4810 modules next to one another.

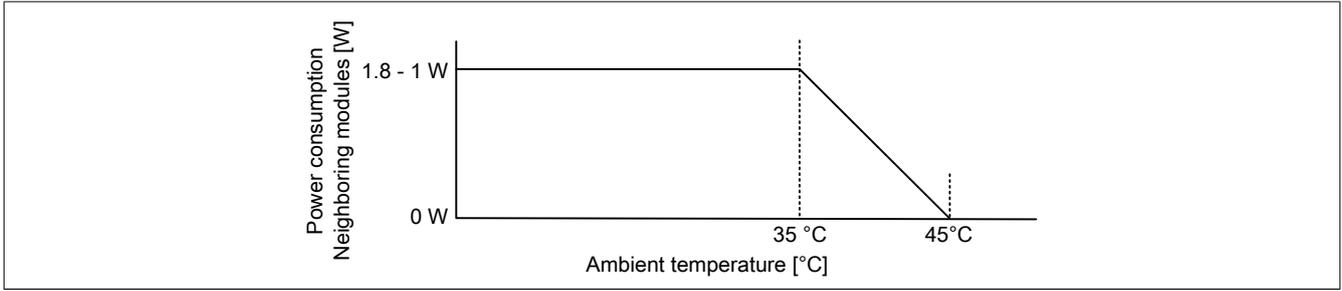
X20CM4810
X20CM4810
X20CM4810

Example: ambient temperature up to 45°C

Power consumption 1.8 W
Power consumption 1.8 W
Power consumption 1.8 W
X20CM4810
Power consumption 1.8 W
Power consumption 1.8 W
Power consumption 1.8 W

Vertical installation

At ambient temperatures higher than 35 °C, the X20CM4810 requires power derating:



If the X20CM4810 is to be operated with ambient temperatures up to 45°C, then the neighboring modules must not consume power.

Example: ambient temperature up to 45°C

X20CM4810	No power consumption e. g. X20ZF0000	X20CM4810	No power consumption e. g. X20ZF0000	X20CM4810	No power consumption e. g. X20ZF0000
-----------	---	-----------	---	-----------	---

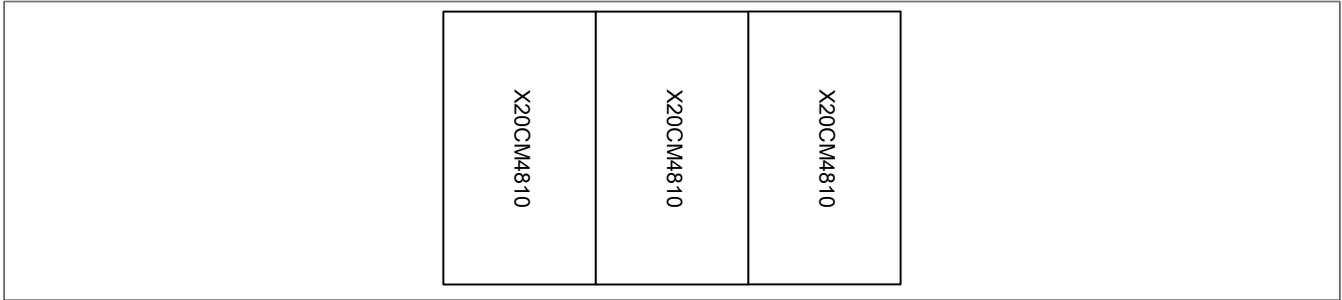
Example: ambient temperature up to 35°C

X20CM4810	Power consumption 1.8 W	X20CM4810	Power consumption 1.8 W	X20CM4810	Power consumption 1.8 W
-----------	-------------------------	-----------	-------------------------	-----------	-------------------------

X20CM4810	Power consumption 1.8 W	X20CM4810	Power consumption 1.8 W
-----------	-------------------------	-----------	-------------------------

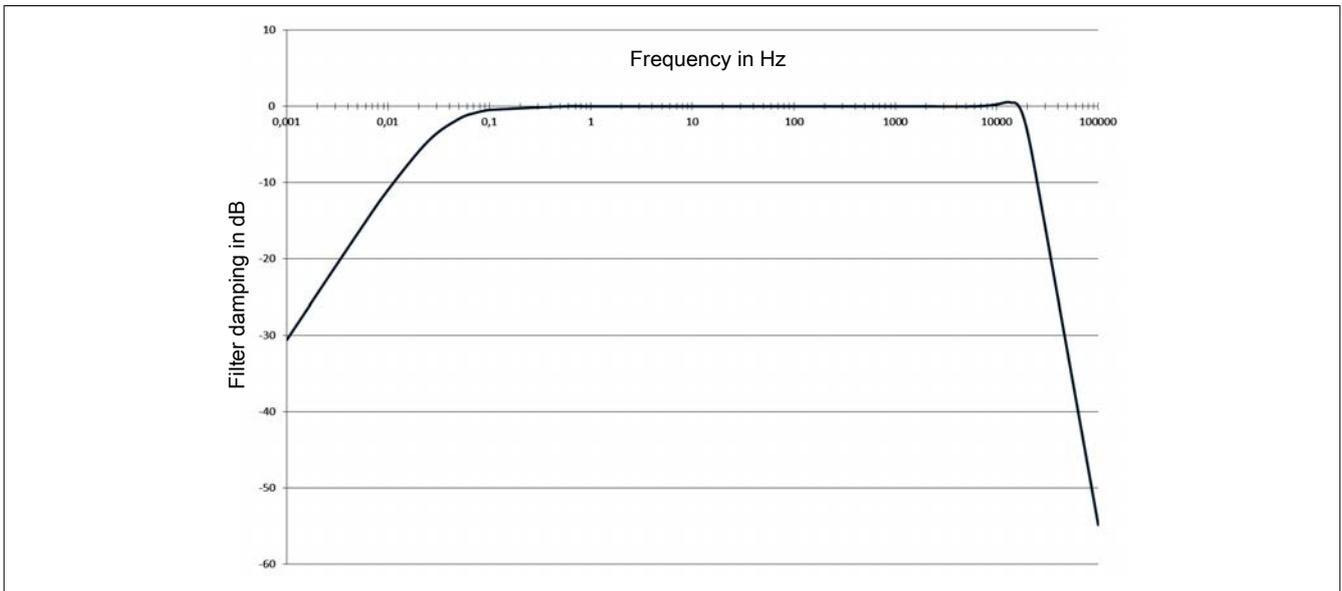
X20CM4810	Power consumption 1.8 W	X20 CPU e.g. X20CP1486
-----------	-------------------------	---------------------------

X20 double-width modules are an exception. For example, multiple neighboring X20CM4810 modules can be operated up to an ambient temperature of 30°C.



3.8 Gain curve

The following diagram shows a typical gain curve for the module.



3.9 Settling time

The input high-pass cutoff frequency of the AC voltage input (limit frequency of 34 MHz) means that a certain amount of settling time is required after changing the DC offset of the pending signal.

- Settling time of approximately 30 seconds at 100 mV/g sensor sensitivity and a 24 V supply voltage with an accuracy of 0.4 g
- Settling time of approximately 60 seconds at 100 mV/g sensor sensitivity and a 24 V supply voltage with an accuracy of 0.001 g
- The respective settling time must be allowed to pass in order to achieve accurate measurement results when an open line occurs. Because of this, all characteristic values and analog input values are set to 0 by the module during the first 30 seconds after a restart or wire breakage.

3.10 Sensor sensitivity

The module always assumes a 100 mV/g acceleration sensor on the input. When using [Function model 0 - Standard](#), it is possible to configure a different sensor sensitivity using the "SensitivitySensor" on [page 40](#) register.

If a different function model is used, for example an SGC controller or bus controller, any conversion to a different sensitivity must be performed manually.

Example

Factor = 100 / (sensor sensitivity in mV/g)

All values must be multiplied by the calculated factor. This also applies to the analog characteristic values if the characteristic value calculation is enabled as well as for uploaded time signals and amplitude spectra. Exempt from this are all characteristic values without any units, such as "KurtosisRaw" on [page 48](#), "CrestFactorRaw" on [page 46](#) and "SkewnessRaw" on [page 49](#).

3.11 Using a B&R Compact CPU or fieldbus CPU

Due to the size of the firmware for the module, only CPUs with sufficient ROM (> 1MB) are supported. (X20CP0292 or X20XC0292)

4 Register description

4.1 General data points

In addition to the registers described in the register description, the module has additional general data points. These are not module-specific but contain general information such as serial number and hardware variant.

General data points are described in section "Additional information - General data points" in the X20 system user's manual.

4.2 Function model 0 - Standard

This is the default function model for the module. The calculated module characteristic values are streamed from the module via Flatstream every 300 ms and prepared for the user by Automation Runtime. If the streamed data is not collected by the next transfer, the characteristic values of the next measurement are lost. For this reason, the maximum cycle time must be observed for an error-free evaluation.

Analog inputs are provided as cyclic data points.

To help the user, all of the module's characteristic values – such as Flatstream handling for characteristic values, unit scaling and so on – are prepared in this function model using Automation Runtime and then made available to the user. See "[Automation Runtime support](#)" on page 37.

With this function model, it is also possible to upload data from the module via another Flatstream data buffer. Library **AsIOVib** is available for uploading the buffers from the module. For a description of the library, see "Automation Help - Programming - Libraries - Direct I/O access - AsIOVib".

In this function model, the module can only be configured using the I/O configuration. No registers are permitted to be reconfigured acyclically.

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Module - Configuration						
-	Cycle time	-				
General registers						
1310	AutogainDelay01	UINT				•
526	AutogainDelay01Read	UINT		•		
0	Control01	UINT			•	
514	SensorConfig01	UINT				•
	SensorConfig01Read			•		
0	Status01	UINT	•			
Analog input functions						
2 * N	AnalogInput0N (index N = 1 to 4)	INT	•			
1330	AnalogInputConfig01	UINT				•
570	AnalogInputConfig01Read	UINT		•		
2	AnalogInputControlByte01	UINT			•	
22 + N* 4	AnalogInputSamples0N (index N = 1 to 4)	UINT			•	
1298	AnalogInputScale01	UINT				•
546	AnalogInputScale01Read	UINT		•		
1310 + N*4	SamplesAnalogInput0N (index N = 1 to 4)	UINT				•
526 + N* 4	SamplesAnalogInput0NRead (index N = 1 to 4)	UINT		•		
Additional registers and characteristic values generated by Automation Runtime						
-	ActSpeed0N (index N = 1 to 4)	REAL			•	
-	CrestFactorHighFrequency0N (index N = 1 to 4)	REAL	•			
-	DataConsistentWithLockedBuffers0N (index N = 1 to 4)	BOOL	•			
-	DataToggleBit01	BOOL	•			
-	OverflowAnalogInput0N (index N = 1 to 4)	BOOL	•			
-	OverflowCharacteristicValues0N (index N = 1 to 4)	UINT	•			
-	OverflowFrequencyBands01	UDINT	•			
-	PeakHighFrequencyRef0N (index N = 1 to 4)	REAL			•	
-	PeakHighFrequencyRefCalculated0N (index N = 1 to 4)	REAL	•			
-	PeakRawRef0N (index N = 1 to 4)	REAL			•	
-	PeakRawRefCalculated0N (index N = 1 to 4)	REAL	•			
-	RmsHighFrequencyRef0N (index N = 1 to 4)	REAL			•	
-	RmsHighFrequencyRefCalculated0N (index N = 1 to 4)	REAL	•			
-	RmsRawRef0N (index N = 1 to 4)	REAL			•	
-	RmsRawRefCalculated0N (index N = 1 to 4)	REAL	•			

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
-	SensitivitySensor0N (index N = 1 to 4)	REAL			•	
-	Vdi3832KtHighFrequency0N (index N = 1 to 4)	REAL	•			
-	Vdi3832KtRaw0N (index N = 1 to 4)	REAL	•			
Characteristic values (transferred in Flatstream)						
-	CrestFactorRaw0N (index N = 1 to 4)	REAL	•			
-	Iso10816_0N (index N = 1 to 4)	REAL	•			
-	KurtosisRaw0N (index N = 1 to 4)	REAL	•			
-	PeakHighFrequency0N (index N = 1 to 4)	REAL	•			
-	PeakRaw0N (index N = 1 to 4)	REAL	•			
-	RmsAccEnvelope0N (index N = 1 to 4)	REAL	•			
-	RmsAccRaw0N (index N = 1 to 4)	REAL	•			
-	RmsHighFrequency0N (index N = 1 to 4)	REAL	•			
-	RmsRaw0N (index N = 1 to 4)	REAL	•			
-	RmsVelEnvelope0N (index N = 1 to 4)	REAL	•			
-	RmsVelRaw0N (index N = 1 to 4)	REAL	•			
-	SkewnessRaw0N (index N = 1 to 4)	REAL	•			
-	FrequencyBandN (index N = 1 to 32)	REAL	•			
Minimum and maximum characteristic values						
2690	MinMaxCounter01	UINT		•		
3588 + N*8	CrestFactorRawMax0N (index N = 1 to 4)	UDINT		•		
2948 + N*8	CrestFactorRawMin0N (index N = 1 to 4)	UDINT		•		
3332 + N*8	Iso10816Max0N (index N = 1 to 4)	UDINT		•		
2692 + N*8	Iso10816Min0N (index N = 1 to 4)	UDINT		•		
3556 + N*8	KurtosisRawMax0N (index N = 1 to 4)	DINT		•		
2916 + N*8	KurtosisRawMin0N (index N = 1 to 4)	DINT		•		
3492 + N*8	PeakHighFrequencyMax0N (index N = 1 to 4)	UDINT		•		
2852 + N*8	PeakHighFrequencyMin0N (index N = 1 to 4)	UDINT		•		
3684 + N*8	PeakRawMax0N (index N = 1 to 4)	UDINT		•		
3044 + N*8	PeakRawMin0N (index N = 1 to 4)	UDINT		•		
3428 + N*8	RmsAccEnvelopeMax0N (index N = 1 to 4)	UDINT		•		
2788 + N*8	RmsAccEnvelopeMin0N (index N = 1 to 4)	UDINT		•		
3364 + N*8	RmsAccRawMax0N (index N = 1 to 4)	UDINT		•		
2724 + N*8	RmsAccRawMin0N (index N = 1 to 4)	UDINT		•		
3524 + N*8	RmsHighFrequencyMax0N (index N = 1 to 4)	UDINT		•		
2884 + N*8	RmsHighFrequencyMin0N (index N = 1 to 4)	UDINT		•		
3652 + N*8	RmsRawMax0N (index N = 1 to 4)	UDINT		•		
3012 + N*8	RmsRawMin0N (index N = 1 to 4)	UDINT		•		
3460 + N*8	RmsVelEnvelopeMax0N (index N = 1 to 4)	UDINT		•		
2820 + N*8	RmsVelEnvelopeMin0N (index N = 1 to 4)	UDINT		•		
3396 + N*8	RmsVelRawMax0N (index N = 1 to 4)	UDINT		•		
2756 + N*8	RmsVelRawMin0N (index N = 1 to 4)	UDINT		•		
3620 + N*8	SkewnessRawMax0N (index N = 1 to 4)	DINT		•		
2980 + N*8	SkewnessRawMin0N (index N = 1 to 4)	DINT		•		
Frequency configuration						
1302	HighFrequencyConfig01	UINT				•
550	HighFrequencyConfig01Read	UINT		•		
1306	MaxFrequencyEnvelope01	UINT				•
558	MaxFrequencyEnvelope01Read	UINT		•		
526	MaxFrequencyRaw01	UINT				•
554	MaxFrequencyRaw01Read	UINT		•		
522	MinFrequencyEnvelope01	UINT				•
566	MinFrequencyEnvelope01Read	UINT		•		
518	MinFrequencyRaw01	UINT				•
562	MinFrequencyRaw01Read	UINT		•		
Frequency bands						
3716 + N*8	FrequencyBandMaxN (index N = 01 to 32)	UDINT		•		
3076 + N*8	FrequencyBandMinN (index N = 01 to 32)	UDINT		•		
506 + N*24	FrequencyBandNConfig (index N = 01 to 32)	UINT				•
1194 + N*24	FrequencyBandNConfigRead (index N = 01 to 32)	UINT		•		
514 + N*24	FrequencyBandNDmgFreq60rpm (index N = 01 to 32)	UINT				•
1202 + N*24	FrequencyBandNDmgFreq60rpmRead (index N = 01 to 32)	UINT		•		
522 + N*24	FrequencyBandNLowerFrequency (index N = 01 to 32)	UINT				•
1210 + N*24	FrequencyBandNLowerFrequencyRead (index N = 01 to 32)	UINT		•		
518 + N*24	FrequencyBandNTolerance (index N = 01 to 32)	UINT				•
1206 + N*24	FrequencyBandNToleranceRead (index N = 01 to 32)	UINT		•		
526 + N*24	FrequencyBandNUpperFrequency (index N = 01 to 32)	UINT				•
1214 + N*24	FrequencyBandNUpperFrequencyRead (index N = 01 to 32)	UINT		•		
Flatstream						
2311	BufferForward01	USINT				•
2318	BufferForwardDelay01	UINT				•
2368	BufferInputSequence01	USINT	•			

Register description

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
2400	BufferOutputSequence01	USINT			•	
2368 + N	BufferRxByte0N (index N = 1 to 5)	USINT	•			
2400 + N	BufferTxByte0N (index N = 1 to 4)	USINT			•	
263	ParameterForward01	USINT				•

4.3 Function model 1 - Fast master

The characteristic values calculated by the module are streamed to the master every 300 ms via Flatstream. If the streamed data is not collected by the next transfer, the characteristic values of the next measurement are lost. For this reason, the maximum cycle time must be observed for an error-free evaluation.

With this function model, it is also possible to use an additional Flatstream to read data buffers from the module.

Analog inputs are provided as cyclic data points.

This function model can only be used on Ethernet-based masters and SGC or fieldbus CPUs. However, it must be ensured that Flatstream handling is implemented on the master and that the module changes the data in the Flatstream in every X2X cycle.

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Module - Configuration						
-	Cycle time	-				
General registers						
2 + N*2	ActSpeed0N (Index N = 1 to 4)	UINT			•	
1310	AutogainDelay01	UINT				•
526	AutogainDelay01Read	UINT		•		
0	Control01	UINT			•	
514	SensorConfig01	UINT				•
	SensorConfig01Read			•		
0	Status01	UINT	•			
Analog input functions						
2 * N	AnalogInput0N (Index N = 1 to 4)	INT	•			
1330	AnalogInputConfig01	UINT				•
570	AnalogInputConfig01Read	UINT		•		
2	AnalogInputControlByte01	UINT			•	
22 + N*4	AnalogInputSamples0N (Index N = 1 to 4)	UINT			•	
1298	AnalogInputScale01	UINT				•
546	AnalogInputScale01Read	UINT		•		
1310 + N*4	SamplesAnalogInput0N (Index N = 1 to 4)	UINT				•
526 + N*4	SamplesAnalogInput0NRead (Index N = 1 to 4)	UINT		•		
Transferring characteristic values via Flatstream						
-	CrestFactorRaw0N (Index N = 1 to 4)	1)	•			
-	Iso10816_0N (Index N = 1 to 4)	1)	•			
-	KurtosisRaw0N (Index N = 1 to 4)	1)	•			
-	PeakHighFrequency0N (Index N = 1 to 4)	1)	•			
-	PeakRaw0N (Index N = 1 to 4)	1)	•			
-	RmsAccEnvelope0N (Index N = 1 to 4)	1)	•			
-	RmsAccRaw0N (Index N = 1 to 4)	1)	•			
-	RmsHighFrequency0N (Index N = 1 to 4)	1)	•			
-	RmsRaw0N (Index N = 1 to 4)	1)	•			
-	RmsVelEnvelope0N (Index N = 1 to 4)	1)	•			
-	RmsVelRaw0N (Index N = 1 to 4)	1)	•			
-	SkewnessRaw0N (Index N = 1 to 4)	1)	•			
-	FrequencyBandN (index N = 1 to 32)	1)	•			
Minimum and maximum characteristic values						
2690	MinMaxCounter01	UINT		•		
3588 + N*8	CrestFactorRawMax0N (Index N = 1 to 4)	UDINT		•		
2948 + N*8	CrestFactorRawMin0N (Index N = 1 to 4)	UDINT		•		
3332 + N*8	Iso10816Max0N (Index N = 1 to 4)	UDINT		•		
2692 + N*8	Iso10816Min0N (Index N = 1 to 4)	UDINT		•		
3556 + N*8	KurtosisRawMax0N (Index N = 1 to 4)	DINT		•		
2916 + N*8	KurtosisRawMin0N (Index N = 1 to 4)	DINT		•		
3492 + N*8	PeakHighFrequencyMax0N (Index N = 1 to 4)	UDINT		•		
2852 + N*8	PeakHighFrequencyMin0N (Index N = 1 to 4)	UDINT		•		
3684 + N*8	PeakRawMax0N (Index N = 1 to 4)	UDINT		•		
3044 + N*8	PeakRawMin0N (Index N = 1 to 4)	UDINT		•		
3428 + N*8	RmsAccEnvelopeMax0N (Index N = 1 to 4)	UDINT		•		
2788 + N*8	RmsAccEnvelopeMin0N (Index N = 1 to 4)	UDINT		•		
3364 + N*8	RmsAccRawMax0N (Index N = 1 to 4)	UDINT		•		
2724 + N*8	RmsAccRawMin0N (Index N = 1 to 4)	UDINT		•		
3524 + N*8	RmsHighFrequencyMax0N (Index N = 1 to 4)	UDINT		•		
2884 + N*8	RmsHighFrequencyMin0N (Index N = 1 to 4)	UDINT		•		
3652 + N*8	RmsRawMax0N (Index N = 1 to 4)	UDINT		•		
3012 + N*8	RmsRawMin0N (Index N = 1 to 4)	UDINT		•		
3460 + N*8	RmsVelEnvelopeMax0N (Index N = 1 to 4)	UDINT		•		
2820 + N*8	RmsVelEnvelopeMin0N (Index N = 1 to 4)	UDINT		•		
3396 + N*8	RmsVelRawMax0N (Index N = 1 to 4)	UDINT		•		
2756 + N*8	RmsVelRawMin0N (Index N = 1 to 4)	UDINT		•		
3620 + N*8	SkewnessRawMax0N (Index N = 1 to 4)	DINT		•		

Register description

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
2980 + N*8	SkewnessRawMin0N (Index N = 1 to 4)	DINT		•		
Frequency configuration						
1302	HighFrequencyConfig01	UINT				•
550	HighFrequencyConfig01Read	UINT		•		
1306	MaxFrequencyEnvelope01	UINT				•
558	MaxFrequencyEnvelope01Read	UINT		•		
526	MaxFrequencyRaw01	UINT				•
554	MaxFrequencyRaw01Read	UINT		•		
522	MinFrequencyEnvelope01	UINT				•
566	MinFrequencyEnvelope01Read	UINT		•		
518	MinFrequencyRaw01	UINT				•
562	MinFrequencyRaw01Read	UINT		•		
Frequency bands						
3716 + N*8	FrequencyBandMaxN (Index N = 01 to 32)	UDINT		•		
3076 + N*8	FrequencyBandMinN (Index N = 01 to 32)	UDINT		•		
506 + N*24	FrequencyBandNConfig (Index N = 01 to 32)	UINT				•
1194 + N*24	FrequencyBandNConfigRead (Index N = 01 to 32)	UINT		•		
514 + N*24	FrequencyBandNDmgFreq60rpm (Index N = 01 to 32)	UINT				•
1202 + N*24	FrequencyBandNDmgFreq60rpmRead (Index N = 01 to 32)	UINT		•		
522 + N*24	FrequencyBandNLowerFrequency (Index N = 01 to 32)	UINT				•
1210 + N*24	FrequencyBandNLowerFrequencyRead (Index N = 01 to 32)	UINT		•		
518 + N*24	FrequencyBandNTolerance (Index N = 01 to 32)	UINT				•
1206 + N*24	FrequencyBandNToleranceRead (Index N= 01 to 32)	UINT		•		
526 + N*24	FrequencyBandNUpperFrequency (Index N = 01 to 32)	UINT				•
1214 + N*24	FrequencyBandNUpperFrequencyRead (Index N = 01 to 32)	UINT		•		
Flatstream						
2311	BufferForward01	USINT				•
2318	BufferForwardDelay01	UINT				•
2368	BufferInputSequence01	USINT	•			
2400	BufferOutputSequence01	USINT			•	
2368 + N	BufferRxByte0N (Index N = 1 to 5)	USINT	•			
2400 + N	BufferTxByte0N (Index N = 1 to 4)	USINT			•	
263	ParameterForward01	USINT				•
270	ParameterForwardDelay01	INT				•
320	ParameterInputSequence01	USINT	•			
352	ParameterOutputSequence01	USINT			•	
320 + N	ParameterRxByteN (Index N = 1 to 13)	USINT	•			

1) For information about the data format, see "Structure of the characteristic value Flatstream" on page 67.

4.4 Function model 2 - Slow master

This function model was developed specifically for operating the module with "slow masters" and for conserving resources on the PLC.

With this function model, it is not possible to upload data buffers from the module.

Analog inputs are provided as cyclic data points. Scaling of the characteristic values must be performed manually.

Characteristic values are calculated by the module every 300 ms and can only be read via acyclic access. In order to keep all characteristic values consistent with one another, they can be locked as they are read. This function model does not allow for seamless measurements. The Min/Max functionality can be used for seamless measurements, however. See "Characteristic values (minimum and maximum)" on page 50

This function model is recommended for all slow buses and masters. It is important to note that acyclic register access must be implemented on the master if a B&R master is not being used.

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
General registers						
2 + N*2	ActSpeed0N (Index N = 1 to 4)	UINT			•	
1310	AutogainDelay01	UINT				•
526	AutogainDelay01Read	UINT		•		
0	Control01	UINT			•	
514	SensorConfig01	UINT				•
	SensorConfig01Read					
0	Status01	UINT	•			
Analog input functions						
2 * N	AnalogInput0N (Index N = 1 to 4)	INT	•			
1330	AnalogInputConfig01	UINT				•
570	AnalogInputConfig01Read	UINT		•		
2	AnalogInputControlByte01	UINT			•	
22 + N*4	AnalogInputSamples0N (Index N = 1 to 4)	UINT			•	
1298	AnalogInputScale01	UINT				•
546	AnalogInputScale01Read	UINT		•		
1310 + N*4	SamplesAnalogInput0N (Index N = 1 to 4)	UINT				•
526 + N*4	SamplesAnalogInput0NRead (Index N = 1 to 4)	UINT		•		
Characteristic values						
828 + N*8	CrestFactorRaw0N (Index N = 1 to 4)	UDINT		•		
572 + N*8	Iso10816_0N (Index N = 1 to 4)	UDINT		•		
796 + N*8	KurtosisRaw0N (Index N = 1 to 4)	DINT		•		
732 + N*8	PeakHighFrequency0N (Index N = 1 to 4)	UDINT		•		
924 + N*8	PeakRaw0N (Index N = 1 to 4)	UDINT		•		
668 + N*8	RmsAccEnvelope0N (Index N = 1 to 4)	UDINT		•		
604 + N*8	RmsAccRaw0N (Index N = 1 to 4)	UDINT		•		
764 + N*8	RmsHighFrequency0N (Index N = 1 to 4)	UDINT		•		
982 + N*8	RmsRaw0N (Index N = 1 to 4)	UDINT		•		
700 + N*8	RmsVelEnvelope0N (Index N = 1 to 4)	UDINT		•		
636 + N*8	RmsVelRaw0N (Index N = 1 to 4)	UDINT		•		
860 + N*8	SkewnessRaw0N (Index N = 1 to 4)	DINT		•		
Minimum and maximum characteristic values						
2690	MinMaxCounter01	UINT		•		
3588 + N*8	CrestFactorRawMax0N (Index N = 1 to 4)	UDINT		•		
2948 + N*8	CrestFactorRawMin0N (Index N = 1 to 4)	UDINT		•		
3332 + N*8	Iso10816Max0N (Index N = 1 to 4)	UDINT		•		
2692 + N*8	Iso10816Min0N (Index N = 1 to 4)	UDINT		•		
3556 + N*8	KurtosisRawMax0N (Index N = 1 to 4)	DINT		•		
2916 + N*8	KurtosisRawMin0N (Index N = 1 to 4)	DINT		•		
3492 + N*8	PeakHighFrequencyMax0N (Index N = 1 to 4)	UDINT		•		
2852 + N*8	PeakHighFrequencyMin0N (Index N = 1 to 4)	UDINT		•		
3684 + N*8	PeakRawMax0N (Index N = 1 to 4)	UDINT		•		
3044 + N*8	PeakRawMin0N (Index N = 1 to 4)	UDINT		•		
3428 + N*8	RmsAccEnvelopeMax0N (Index N = 1 to 4)	UDINT		•		
2788 + N*8	RmsAccEnvelopeMin0N (Index N = 1 to 4)	UDINT		•		
3364 + N*8	RmsAccRawMax0N (Index N = 1 to 4)	UDINT		•		
2724 + N*8	RmsAccRawMin0N (Index N = 1 to 4)	UDINT		•		
3524 + N*8	RmsHighFrequencyMax0N (Index N = 1 to 4)	UDINT		•		
2884 + N*8	RmsHighFrequencyMin0N (Index N = 1 to 4)	UDINT		•		
3652 + N*8	RmsRawMax0N (Index N = 1 to 4)	UDINT		•		
3012 + N*8	RmsRawMin0N (Index N = 1 to 4)	UDINT		•		
3460 + N*8	RmsVelEnvelopeMax0N (Index N = 1 to 4)	UDINT		•		
2820 + N*8	RmsVelEnvelopeMin0N (Index N = 1 to 4)	UDINT		•		
3396 + N*8	RmsVelRawMax0N (Index N = 1 to 4)	UDINT		•		
2756 + N*8	RmsVelRawMin0N (Index N = 1 to 4)	UDINT		•		
3620 + N*8	SkewnessRawMax0N (Index N = 1 to 4)	DINT		•		

Register description

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
2980 + N*8	SkewnessRawMin0N (Index N = 1 to 4)	DINT		•		
Frequency configuration						
1302	HighFrequencyConfig01	UINT				•
550	HighFrequencyConfig01Read	UINT		•		
1306	MaxFrequencyEnvelope01	UINT				•
558	MaxFrequencyEnvelope01Read	UINT		•		
526	MaxFrequencyRaw01	UINT				•
554	MaxFrequencyRaw01Read	UINT		•		
522	MinFrequencyEnvelope01	UINT				•
566	MinFrequencyEnvelope01Read	UINT		•		
518	MinFrequencyRaw01	UINT				•
562	MinFrequencyRaw01Read	UINT		•		
Frequency bands						
3716 + N*8	FrequencyBandMaxN (Index N = 01 to 32)	UDINT		•		
3076 + N*8	FrequencyBandMinN (Index N = 01 to 32)	UDINT		•		
956 + N*8	FrequencyBandN (Index N = 01 to 32)	UDINT		•		
506 + N*24	FrequencyBandNConfig (Index N = 01 to 32)	UINT				•
1194 + N*24	FrequencyBandNConfigRead (Index N = 01 to 32)	UINT		•		
514 + N*24	FrequencyBandNDmgFreq60rpm (Index N = 01 to 32)	UINT				•
1202 + N*24	FrequencyBandNDmgFreq60rpmRead (Index N = 01 to 32)	UINT		•		
522 + N*24	FrequencyBandNLowerFrequency (Index N = 01 to 32)	UINT				•
1210 + N*24	FrequencyBandNLowerFrequencyRead (Index N = 01 to 32)	UINT		•		
518 + N*24	FrequencyBandNTolerance (Index N = 01 to 32)	UINT				•
1206 + N*24	FrequencyBandNToleranceRead (Index N = 01 to 32)	UINT		•		
526 + N*24	FrequencyBandNUpperFrequency (Index N = 01 to 32)	UINT				•
1214 + N*24	FrequencyBandNUpperFrequencyRead (Index N = 01 to 32)	UINT		•		

4.5 Function model 254 - Bus controller

This function model can only be used with a CANIO bus controller. It includes the same functionality as [Function model 2 - Slow master](#).

Differences:

- The order of cyclic registers is somewhat different on the bus.
- Since the AnalogInputToggleBit01-04 data points cannot be consistently transferred to the "AnalogInput" on page 32 data points, they are not available in this function model. The user must watch for changes in the value of the data points "AnalogInput" in order to determine whether a new value is available.

Register	Offset ¹⁾	Name	Data type	Read		Write	
				Cyclic	Acyclic	Cyclic	Acyclic
General registers							
2 + N*2	2 + N*4	ActSpeed0N (Index N = 1 to 4)	UINT			•	
1310		AutogainDelay01	UINT				•
526		AutogainDelay01Read	UINT		•		
0	2	Control01	UINT			•	
514		SensorConfig01	UINT				•
		SensorConfig01Read			•		
0	2	Status01	UINT	•			
Analog input functions							
2 * N	2 * N*4	AnalogInput0N (Index N = 1 to 4)	INT	•			
1330		AnalogInputConfig01	UINT				•
570		AnalogInputConfig01Read	UINT		•		
2	22	AnalogInputControlByte01	UINT			•	
1298		AnalogInputScale01	UINT				•
546		AnalogInputScale01Read	UINT		•		
1310 + N*4		SamplesAnalogInput0N (Index N = 1 to 4)	UINT				•
526 + N*4		SamplesAnalogInput0NRead (Index N = 1 to 4)	UINT		•		
Characteristic values							
828 + N*8		CrestFactorRaw0N (Index N = 1 to 4)	UDINT		•		
572 + N*8		Iso10816_0N (Index N = 1 to 4)	UDINT		•		
796 + N*8		KurtosisRaw0N (Index N = 1 to 4)	DINT		•		
732 + N*8		PeakHighFrequency0N (Index N = 1 to 4)	UDINT		•		
924 + N*8		PeakRaw0N (Index N = 1 to 4)	UDINT		•		
668 + N*8		RmsAccEnvelope0N (Index N = 1 to 4)	UDINT		•		
604 + N*8		RmsAccRaw0N (Index N = 1 to 4)	UDINT		•		
764 + N*8		RmsHighFrequency0N (Index N = 1 to 4)	UDINT		•		
982 + N*8		RmsRaw0N (Index N = 1 to 4)	UDINT		•		
700 + N*8		RmsVelEnvelope0N (Index N = 1 to 4)	UDINT		•		
636 + N*8		RmsVelRaw0N (Index N = 1 to 4)	UDINT		•		
860 + N*8		SkewnessRaw0N (Index N = 1 to 4)	DINT		•		
Minimum and maximum characteristic values							
2690		MinMaxCounter01	UINT		•		
3588 + N*8		CrestFactorRawMax0N (Index N = 1 to 4)	UDINT		•		
2948 + N*8		CrestFactorRawMin0N (Index N = 1 to 4)	UDINT		•		
3332 + N*8		Iso10816Max0N (Index N = 1 to 4)	UDINT		•		
2692 + N*8		Iso10816Min0N (Index N = 1 to 4)	UDINT		•		
3556 + N*8		KurtosisRawMax0N (Index N = 1 to 4)	DINT		•		
2916 + N*8		KurtosisRawMin0N (Index N = 1 to 4)	DINT		•		
3492 + N*8		PeakHighFrequencyMax0N (Index N = 1 to 4)	UDINT		•		
2852 + N*8		PeakHighFrequencyMin0N (Index N = 1 to 4)	UDINT		•		
3684 + N*8		PeakRawMax0N (Index N = 1 to 4)	UDINT		•		
3044 + N*8		PeakRawMin0N (Index N = 1 to 4)	UDINT		•		
3428 + N*8		RmsAccEnvelopeMax0N (Index N = 1 to 4)	UDINT		•		
2788 + N*8		RmsAccEnvelopeMin0N (Index N = 1 to 4)	UDINT		•		
3364 + N*8		RmsAccRawMax0N (Index N = 1 to 4)	UDINT		•		
2724 + N*8		RmsAccRawMin0N (Index N = 1 to 4)	UDINT		•		
3524 + N*8		RmsHighFrequencyMax0N (Index N = 1 to 4)	UDINT		•		
2884 + N*8		RmsHighFrequencyMin0N (Index N = 1 to 4)	UDINT		•		
3652 + N*8		RmsRawMax0N (Index N = 1 to 4)	UDINT		•		
3012 + N*8		RmsRawMin0N (Index N = 1 to 4)	UDINT		•		
3460 + N*8		RmsVelEnvelopeMax0N (Index N = 1 to 4)	UDINT		•		
2820 + N*8		RmsVelEnvelopeMin0N (Index N = 1 to 4)	UDINT		•		
3396 + N*8		RmsVelRawMax0N (Index N = 1 to 4)	UDINT		•		
2756 + N*8		RmsVelRawMin0N (Index N = 1 to 4)	UDINT		•		
3620 + N*8		SkewnessRawMax0N (Index N = 1 to 4)	DINT		•		
2980 + N*8		SkewnessRawMin0N (Index N = 1 to 4)	DINT		•		
Frequency configuration							
1302		HighFrequencyConfig01	UINT				•
550		HighFrequencyConfig01Read	UINT		•		
1306		MaxFrequencyEnvelope01	UINT				•

Register description

Register	Offset ¹⁾	Name	Data type	Read		Write	
				Cyclic	Acyclic	Cyclic	Acyclic
558		MaxFrequencyEnvelope01Read	UINT		•		
526		MaxFrequencyRaw01	UINT				•
554		MaxFrequencyRaw01Read	UINT		•		
522		MinFrequencyEnvelope01	UINT				•
566		MinFrequencyEnvelope01Read	UINT		•		
518		MinFrequencyRaw01	UINT				•
562		MinFrequencyRaw01Read	UINT		•		
Frequency bands							
3716 + N*8		FrequencyBandMaxN (Index N = 01 to 32)	UDINT		•		
3076 + N*8		FrequencyBandMinN (Index N = 01 to 32)	UDINT		•		
956 + N*8		FrequencyBandN (Index N = 01 to 32)	UDINT		•		
506 + N*24		FrequencyBandNConfig (Index N = 01 to 32)	UINT				•
1194 + N*24		FrequencyBandNConfigRead (Index N = 01 to 32)	UINT		•		
514 + N*24		FrequencyBandNDmgFreq60rpm (Index N = 01 to 32)	UINT				•
1202 + N*24		FrequencyBandNDmgFreq60rpmRead (Index N = 01 to 32)	UINT		•		
522 + N*24		FrequencyBandNLowerFrequency (Index N = 01 to 32)	UINT				•
1210 + N*24		FrequencyBandNLowerFrequencyRead (Index N = 01 to 32)	UINT		•		
518 + N*24		FrequencyBandNTolerance (Index N = 01 to 32)	UINT				•
1206 + N*24		FrequencyBandNToleranceRead (Index N = 01 to 32)	UINT		•		
526 + N*24		FrequencyBandNUpperFrequency (Index N = 01 to 32)	UINT				•
1214 + N*24		FrequencyBandNUpperFrequencyRead (Index N = 01 to 32)	UINT		•		

1) The offset specifies the position of the register within the CAN object.

4.5.1 Using the module on the bus controller

Function model 254 "Bus controller" is used by default only by non-configurable bus controllers. All other bus controllers can use other registers and functions depending on the fieldbus used.

For detailed information, see section "Additional information - Using I/O modules on the bus controller" in the X20 user's manual (version 3.50 or later).

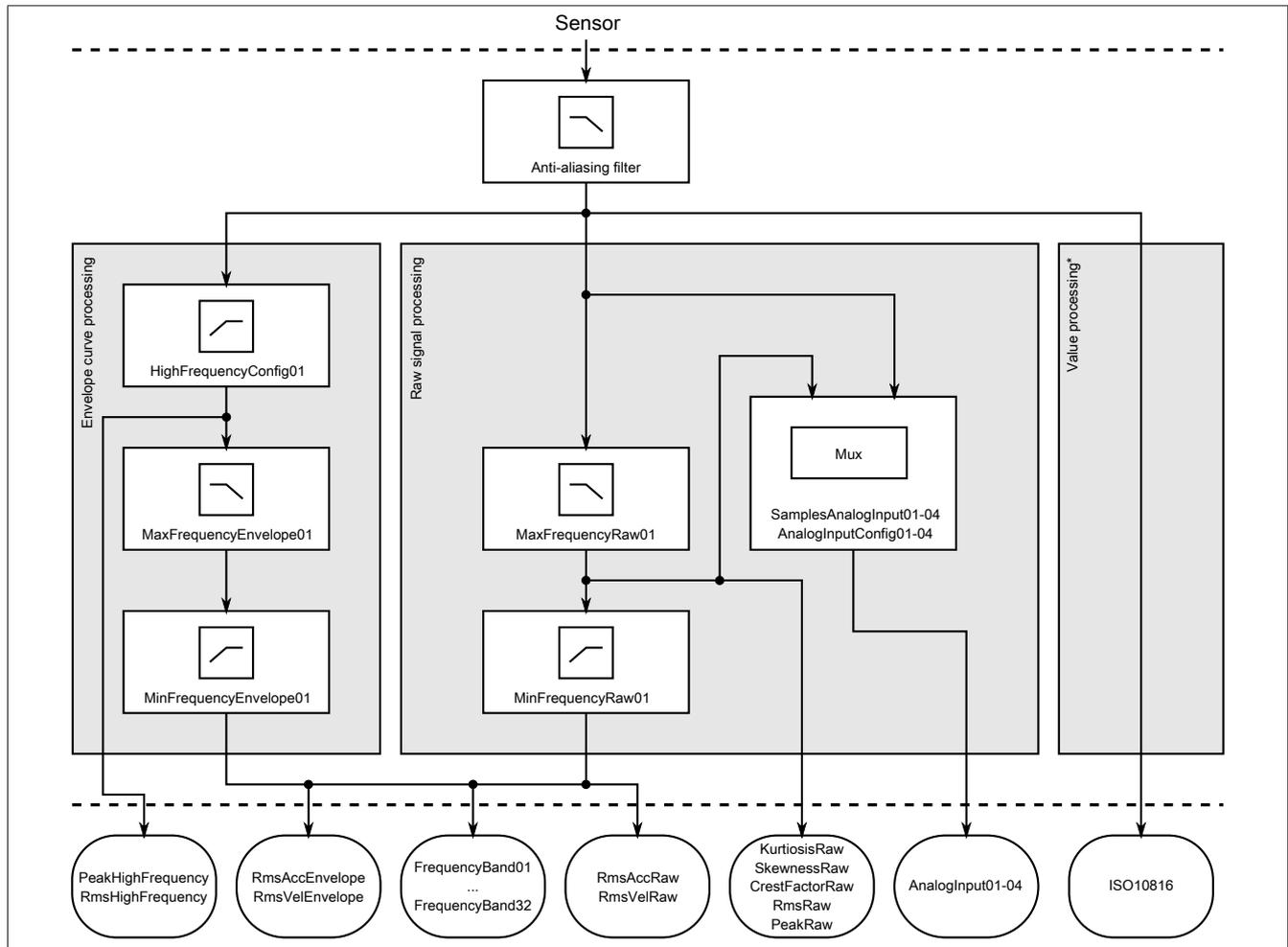
4.5.2 CAN I/O bus controller

The module occupies 2 analog logical slots on CAN I/O.

4.6 General information

4.6.1 Signal generation

The following signals and characteristic values are calculated from the acceleration sensor's input signal:



* Processing per ISO 10816-3

4.6.2 Filter configuration

The module has a number of configurable filters.

There is a configurable high-pass filter for the whole module that can be configured using register "[HighFrequencyConfig](#)" on [page 56](#). Possible settings are 500 Hz, 1 kHz and 2 kHz. This high-pass affects all high-frequency and envelope characteristic values of all module channels.

In addition, there are 2 adjustable low pass filters per channel.

- Filtering of the raw signal. This filter is configured using register "[MaxFrequencyRaw](#)" on [page 57](#). Possible settings are 200 Hz, 500 Hz, 1 kHz, 2 kHz, 5 kHz and 10 kHz.
- Filtering of the envelope signal. This filter is configured using register "[MaxFrequencyEnvelope](#)" on [page 56](#). Possible settings are 200 Hz, 500 Hz, 1 kHz and 2 kHz.

These low-pass filters affect all calculated characteristic values of the respective signal, i.e. the raw or envelope signal. They can be used to increase the frequency resolution of the FFT. For the characteristic value calculation in the analog input, however, it can be selected whether the characteristic values are to be calculated from the direct input signal or the low-pass filtered raw signal.

4.6.3 Frequency bands

It is possible to individually configure up to 32 frequency bands where the RMS value (RMS) or the noise of a quadrant is calculated.

Parameter	Settings		
Enable	Off RMS Noise		
Channel	1 2 3 4		
Source	Raw acceleration signal Raw velocity signal Enveloped acceleration signal Enveloped velocity signal		
Calculation of harmonics (RMS only)	Yes No		
Rotation-dependent (RMS only)	On	Selects the data point for velocity (" ActSpeed " on page 27)	[1/100 Hz]
		Standardized damage frequency at 60 rpm	[1/100]
		± Width of the frequency band (tolerance band)	[1/100 Hz]
	Off	Lower frequency	[1/4 Hz]
Upper frequency		[1/4 Hz]	
Quadrant (noise only)	1st quadrant 2nd quadrant 3rd quadrant 4th quadrant		

4.6.4 Calculating the velocity signal automatically

The module can calculate the velocity signal from the signal provided by the acceleration sensor. This calculation is disabled by default since it can reduce the accuracy of the acceleration signal.

Reason

When converting from acceleration to velocity, low frequency portions become very large. As a result, the [autogain](#) is decreased by a few levels, which then further degrades accuracy.

If this calculation is not enabled, 0 is output for all characteristic values calculated from the velocity spectrum. Characteristic value "[Iso10816](#)" on page 43 is not affected by this.

4.6.5 Autogain, AutogainDelay and overflow

The module automatically adjusts the measurement signal dynamically (autogain) to ensure that it is measured with the highest possible accuracy. This adjustment is made in multiple steps. Each step amplifies the input signal more than the last. If the signal was very small for a long time and an impact suddenly occurs, an overflow may occur with some calculated characteristic values. This is indicated by the overflow bit for the respective channel (Overflow01-04 in the "[Status](#)" on page 30 register) being set and the affected module characteristic values being set to their maximum.

With [Function model 0 - Standard](#), there are also the "[OverflowCharacteristicValues](#)" on page 38 and "[OverflowFrequencyBands](#)" on page 39 registers. These are automatically generated by Automation Runtime and directly indicate the overflow status of the individual characteristic values and frequency bands.

If an overflow occurs or if an internal threshold is exceeded, the autogain for the next measurement is reduced by one level. If no overflow occurs for a certain number of measurements (adjustable using the "[AutogainDelay](#)" on page 27 register), or the signal stays below the internal threshold, the autogain is increased by one level again.

If overflows are occurring frequently, increasing AutogainDelay may help.

4.6.6 Term definition: Sampling rate and sampling frequency

The terms sampling rate and sampling frequency are used in this document. The following is a definition of these terms:

Term	Definition
Sampling rate	Number of times an analog signal is sampled per unit of time. The unit of time is generally 1 second. Example: 100 samples per second
Sampling frequency	Sampling of an analog signal with respect to 1 second. Specified in Hertz [Hz]. Examples: <ul style="list-style-type: none"> • Sampling an analog signal once per second corresponds to a sampling frequency of 1 Hz. • Sampling an analog signal once per millisecond corresponds to a sampling frequency of 1 kHz.

4.7 General registers

4.7.1 ActSpeed

Name:

ActSpeed01 to ActSpeed04

Registers for the current speed for calculating frequency bands 01 to 32 if they have been configured as speed-dependent.

The current speed must be specified in 1/100 Hz. In [Function model 0 - Standard](#), Automation Runtime handles this.

If the 4 different speed data points are not sufficient, e.g. for several different gear ratios, the speed ratio can also be included when calculating the standardized damage frequency for the frequency band ("[FrequencyBandDmgFreq60rpm](#)" on [page 65](#) register).

Characteristic value in [Function model 0 - Standard](#)

Format	Values	Unit
REAL	0 to 655.35 ¹⁾	in 1 Hz

1) The driver reduces larger values to 655.35.

Characteristic value in all other function models

Format	Values	Unit
UINT	0 to 65,535	in 1/100 Hz

4.7.2 AutogainDelay

Name:

AutogainDelay01

This register can be used to configure the autogain delay for all 4 channels.

To ensure that even smaller signals can be calculated precisely, the autogain can be increased in steps. This happens if there has been no overflow in the number of measurement cycles configured in this register and all conditions for the next step were always met. If an overflow does occur, then autogain is reduced immediately by one step.

The unit for delaying autogain is specified in measurement cycles (300 ms).

Data type	Values	Information
UINT	1 to 65,535	Number of measurement cycles. Bus controller default setting: 50

4.7.3 AutogainDelayRead

Name:

AutogainDelay01Read

Register for reading the current "[AutogainDelay](#)" on [page 27](#) configuration.

Data type	Values
UINT	0 to 65,535

4.7.4 Control

Name:
Control01

General control register for the module.

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	RequestBufferLock01 or RequestDataLock01	0	Data should not be locked.
		1	Data should be locked.
1	MinMaxUpdate01	x	The acyclic minimum and maximum values are refreshed at each edge
2 - 15	Reserved	0	

RequestBufferLock01 or RequestDataLock01

"Function model 0 - Standard" and "1 - Fast master"

RequestBufferLock01 can be used to lock all buffers and parameters on the module. When the bit is set to 1, all buffers (raw data and FFT) are locked starting at the beginning of the next measurement. Before a buffer can be uploaded, all the data it contains must be locked.

The characteristic values associated with the locked buffers are transferred in the Flatstream characteristic values as soon as BufferLockValid01 = 1.

Information:

Since the measurement is universal, the parameters associated with the buffers are only transferred once.

"Function model 2 - Slow master" and "254 - Bus controller"

RequestDataLock01 can be used to lock all parameters on the module. When the bit is set to 1, then a consistent version of all measured values is retained until the next measurement. Once all data on the module has been locked, all the calculated characteristic values can be read acyclically from the module.

Data on the module is only locked once bit BufferLockValid01 or DataLockValid01 in register "Status01" on page 30 is set.

Once the data has been uploaded, the RequestBufferLock01 or RequestDataLock01 bit can be reset to 0. Once BufferLockValid01 or DataLockValid01 in register "Status01" on page 30 are back to 0, the data on the module is no longer locked.

A new freezing is only permitted by the module if the buffers of the channel with the largest buffer length are filled again. The buffer length depends on the settings of registers "MaxFrequencyRaw" on page 57 and "MaxFrequencyEnvelope" on page 56.

MinMaxUpdate01

An edge on MinMaxUpdate01 updates all acyclic minimum and maximum values. A new cycle then starts internally to generate the minimum and maximum values that are again copied to the acyclic registers on the next edge. Once an edge has been reached, the current minimum and maximum values can be read acyclically in the next X2X cycle. Register "MinMaxCounter" on page 52 specifies how many measurement cycles were analyzed with minimum and maximum mapping. The minimum/maximum values themselves are only valid if the counter has a value other than 0.

4.7.5 SensorConfig

Name:
SensorConfig01

This register can be used to enable or disable the IEPE sensor supply for individual channels.

Data type	Values	Bus controller default setting
UINT	See the bit structure.	0

Bit structure:

Bit	Description	Value	Information
0	Channel 1: IEPE supply	0	Off (bus controller default setting)
		1	On
...		...	
3	Channel 4: IEPE supply	0	Off (bus controller default setting)
		1	On
4 - 7	Reserved	0	
8	Channel 1: EnableVelocityCalculation	0	No calculation (bus controller default setting)
		1	Calculation enabled
...		...	
11	Channel 4: EnableVelocityCalculation	0	No calculation (bus controller default setting)
		1	Calculation enabled
12 - 13	Reserved	0	
14	Buffer length	0	8192 measured values (bus controller default setting)
		1	65535 measured values
15	Selects the function model	0	Function model 2 - Slow master and Function model 254 - Bus controller (bus controller default setting)
		1	Function model 1 - Fast master

EnableVelocityCalculation

This bit can be used to enable the calculation of all characteristic values calculated on the velocity spectrum.

If this calculation is not enabled, 0 is output for all characteristic values calculated on the velocity spectrum.

To maximize the precision of the characteristic values based on the acceleration spectrum, it is recommended to only enable this bit if the velocity signals are absolutely required.

4.7.6 SensorConfigRead

Name:
SensorConfig01Read

Register for reading the current "SensorConfig" on page 29 configuration.

Data type	Values
UINT	0 to 65,535

4.7.7 Status

Name:
Status01

General status register for the module.

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Channel 1: BrokenWire01	0	No error
		1	Open circuit
...		...	
3	Channel 4: BrokenWire04	0	No error
		1	Open circuit
4	BufferLockValid01 or DataLockValid01 ¹⁾	0	Data not locked
		1	Data locked, consistent and valid
5	Channel 1: Overflow01	0	No error
		1	Overflow of one or more characteristic values
...		...	
8	Channel 4: Overflow04	0	No error
		1	Overflow of one or more characteristic values
9	Channel 1: AnalogInputToggleBit01 ²⁾	0	Does not toggle
		1	Toggles
...		...	
12	Channel 4: AnalogInputToggleBit04 ²⁾	0	Does not toggle
		1	Toggles
13 - 15	Reserved	-	

- 1) Confirmation of RequestBufferLock01 or RequestDataLock01 in register "Control" on page 28
- 2) Toggles with each new calculation and each new input value in register "AnalogInput0x" on page 32

4.8 Analog input functions

Each of the module's 4 accelerometer inputs can also be used directly as an analog input with various special functions (see register "[AnalogInput](#)" on page 32).

The resolution of the analog inputs can be set using the configuration ("[AnalogInputScale](#)" on page 35). The lower the maximum value, the higher the resolution of the register and vice versa. When the maximum value is exceeded, the register is limited to the respective maximum (positive or negative).

A toggle bit ([AnalogInputToggleBit01-04](#)) signals when a new value has been transferred.

The following functions are available:

- Normal analog input function
- Characteristic value calculation in continuous mode with enable (continuous mode)
- Characteristic value calculation in trigger mode (single shot)

4.8.1 Measured values

The last 8 measured values before the X2X cycle are always averaged and transferred on the bus. Here, the direct input signal (raw signal max. 10 kHz) with a sampling frequency of 25.781 kHz is always used and is not mean-adjusted. The value is scaled according to the configuration. (see "[AnalogInputScale](#)" on page 35 register)

4.8.2 Characteristic value calculation in AnalogInput

The following characteristic values can be indicated directly in register "[AnalogInput](#)" on page 32. In this case, it is necessary to check the configured scale.

- Mean
- Peak value (absolute)
- RMS value
- Crest factor

There are 2 signals available to calculate the configured characteristic values:

- Input signal filtered to 10 kHz with a sampling frequency of 25.781 kHz and not mean-adjusted.
- Raw signal filtered to the maximum frequency with a sampling frequency that is dependent on the "[MaxFrequencyRaw](#)" on page 57 configuration and with mean adjustment using the last 8192 samples.

Register "[SamplesAnalogInput](#)" on page 36 can be used to set how many sampled values should be used to calculate the respective parameter. The time between 2 samples depends on the maximum frequency.

2 modes are available:

- "[Continuous mode with enable \(continuous mode\)](#)" on page 31
- "[Trigger mode \(single shot\)](#)" on page 32

4.8.2.1 Continuous mode with enable (continuous mode)

This mode offers the following advantages:

- When the parameters are configured correctly, nothing can be overlooked.
- Using "enable", measurement in the module can be started after an event or events can be hidden.
- The toggle bit toggles with every new value.

The following must be taken into consideration for the configuration:

- In order to guarantee seamless measurement, the sample time (number of samples * sample rate) must be longer than the X2X Link cycle (see register "[SamplesAnalogInput](#)" on page 36).
- If a shorter sample time than the X2X Link cycle is configured, the last complete measurement is always transferred.

Information:

Values are lost in this mode. The measured values cannot be transferred to the bus because multiple values are calculated in each X2X Link cycle.

4.8.2.2 Trigger mode (single shot)

This mode offers the following advantages:

- Only one measurement is taken at a time.
- The trigger is edge-sensitive, so it can be retriggered in each X2X Link cycle.
- The toggle bit toggles with every new value.

The following must be taken into consideration for the configuration:

- A new trigger will be ignored during an ongoing measurement. The runtime on the bus can cause retriggering before the toggle bit has changed.
- If a shorter sample time than the X2X Link cycle time is configured, the last complete measurement is always transferred.
- In trigger mode, values are lost because the measured values are recorded acyclically to the X2X Link and cannot be synchronized continually.
- The trigger results in temporary synchronization with X2X Link.

4.8.3 AnalogInput

Name:

AnalogInput01 to AnalogInput04

Depending on the configuration, these registers contain

- The actual input value of the associated input averaged over the last 8 samples
- Or the characteristic value to be calculated using the configured number of samples

The value in the register is scaled according to the configuration of register "[AnalogInputScale](#)" on page 35.

If scaling the value takes it outside of the permissible range of values for INT, then it will be limited to the minimum or maximum INT value. The overflow bit for the corresponding channel is not set in this case.

Information:

In [Function model 0 - Standard](#), Automation Runtime automatically scales the analog input to mg or non-dimensional values (crest factor) while taking `SensitivitySensor` and `AnalogInputScale` into account. If the value of `AnalogInput` without sensor scaling exceeds the value range of `AnalogInputScale` related to 100 mV/g, the corresponding `AnalogInputOverflow` bit is set.

Data type	Values
INT	-32768 to 32767

4.8.4 AnalogInputConfig

Name:

AnalogInputConfig01

Register for configuring the characteristic value calculation in "AnalogInput" on page 32. This is only needed if "SamplesAnalogInput" on page 36 of the respective channel is greater than 0.

Data type	Values	Bus controller default setting
UINT	See the bit structure.	0

Bit structure:

Bit	Description	Value	Description
0 - 1	Value to be calculated in AnalogInput01	0	Mean value (bus controller default setting)
		1	Peak value
		2	RMS value
		3	Crest factor
2 - 3	Value to be calculated in AnalogInput02	x	For possible values, see bits 0-1.
4 - 5	Value to be calculated in AnalogInput03	x	For possible values, see bits 0-1.
6 - 7	Value to be calculated in AnalogInput04	x	For possible values, see bits 0-1.
8	Trigger mode of AnalogInput01	0	Continuous with enable (bus controller default setting)
		1	Once with trigger
...
11	Trigger mode of AnalogInput04	0	Continuous with enable (bus controller default setting)
		1	Once with trigger
12	Signal source for characteristic value calculation AnalogInput01	0	Raw signal filtered to the configured maximum frequency (see "MaxFrequencyRaw" on page 57) with mean value adjustment. ¹⁾ (Bus controller default setting)
		1	Raw signal filtered to 10 kHz without mean adjustment
...
15	Signal source for characteristic value calculation of AnalogInput04	0	Raw signal filtered to the configured maximum frequency (see "MaxFrequencyRaw" on page 57) with mean value adjustment. ¹⁾ (Bus controller default setting)
		1	Raw signal filtered to 10 kHz without mean adjustment

1) The average of the last 8192 samples is used for mean adjustment.

4.8.5 AnalogInputConfigRead

Name:

AnalogInputConfig01Read

Register for reading the current AnalogInputConfig01 configuration.

Data type	Values
UINT	0 to 65,535

4.8.6 AnalogInputControlByte

Name:

AnalogInputControlByte01

The control register for "AnalogInput" on page 32 is only functional if the respective "SamplesAnalogInput" on page 36 configuration register is greater than 0.

The configuration in "AnalogInputConfig" on page 33 for the respective channel determines whether the respective bit is an enable or trigger bit.

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	AnalogInputControl01	x	See Function of the bit
...		...	
3	AnalogInputControl04	x	See Function of the bit
4 - 15	Reserved	0	

Function of the bit

Function in continuous mode:

Bit to start the continuous characteristic value calculation of "AnalogInput".

0 = Calculation disabled

1 = Continuous calculation of characteristic values

Function in single-shot mode:

Bit to start a new characteristic value calculation of "AnalogInput".

0, 1 Every edge starts a new characteristic value calculation provided the previous one is already completed.

Characteristic value calculation

The characteristic value configured in "AnalogInputConfig" on page 33 for the respective channel is calculated. The number of samples configured in "SamplesAnalogInput" on page 36 is used. The calculated value is displayed in "AnalogInput" on page 32 with the scaling of the respective channel configured in "AnalogInputScale" on page 35. The value of AnalogInputToggleBit01-04 changes each time a new calculation takes place.

4.8.7 AnalogInputSamples

Name:

AnalogInputSamples01 to AnalogInputSamples04

If bit 15 of the respective "SamplesAnalogInput" on page 36 register is 1, then this register cyclically defines the number of samples used to calculate characteristic values.

Information:

If the register is changed while a measurement is running, then the current measurement is discarded (AnalogInputToggleBit0X does not change). This is especially important to remember when using [continuous mode with enable](#).

Data type	Values	Information
UINT	0	Invalid
	1 to 8191	Characteristic value calculation active for the respective channel in the corresponding analog input
	>8191	Invalid

The time between 2 samples depends on register "MaxFrequencyRaw" on page 57:

Maximum frequency	Sampling time (time between 2 samples)
10000 Hz	38.79 µs
5000 Hz	77.58 µs
2000 Hz	193.94 µs
1000 Hz	387.88 µs
500 Hz	775.76 µs
200 Hz	1939.39 µs

4.8.8 AnalogInputScale

Name:

AnalogInputScale01

This register can be used to specify the scale of the 4 analog inputs ("[AnalogInput](#)" on page 32). If the actual value is greater than the value configured in this register, the respective register for the analog input ("[AnalogInput](#)" on page 32) is limited to the positive maximum (32767).

For example, if ± 128 is configured for AnalogInput04, then the 16 bits represent a value range of ± 128 g or without units if the crest factor has been calculated.

Information:

The configured scaling value is always based on a 100 mV/g sensor. Any sensor that has a different sensor sensitivity must be reflected in the configuration.

Data type	Values	Bus controller default setting
UINT	See bit structure.	0x8888

Bit structure:

Bit	Description	Value	Information
0 - 3	Scaling for AnalogInput01	0	Invalid
		1	± 1
		2	± 2
		3	± 4
		4	± 8
		5	± 16
		6	± 32
		7	± 64
		8	± 128 (bus controller default setting)
		9 to 15	Invalid
4 - 7	Scaling for AnalogInput02	x	For possible values, see AnalogInput01.
8 - 11	Scaling for AnalogInput03	x	For possible values, see AnalogInput01.
12 - 15	Scaling for AnalogInput04	x	For possible values, see AnalogInput01.

4.8.9 AnalogInputScaleRead

Name:

AnalogInputScale01Read

This register can be used to read the scale of the analog inputs ("[AnalogInput](#)" on page 32).

Data type	Values
UINT	0 to 65,535

4.8.10 SamplesAnalogInput

Name:

SamplesAnalogInput01 to SamplesAnalogInput04

If the corresponding SamplesAnalogInput register is equal to 0, then the "AnalogInput" on page 32 registers will provide the current input value for the analog input.

If the SamplesAnalogInput register is greater than 0, the characteristic value configured in register "AnalogInput-Config" on page 33 for the respective channel is calculated. For this, the number of samples configured in this register will be used and displayed in the corresponding AnalogInput with the configured scaling.

If bit 15 of this register equals 1, then the characteristic value configured in "AnalogInputConfig01" for the respective channel is calculated. However, the number of samples specified cyclically in register "AnalogInputSamples" on page 34 is used and displayed in the associated analog input with the configured scaling.

Data type	Values	Bus controller default setting
UINT	See the bit structure.	0

Bit structure

Bit	Explanation	Values	Information
0 - 14		0	No characteristic value calculation of the respective channel in the associated analog input (bus controller default setting)
		1 to 8191	Characteristic value calculation active for the respective channel in the corresponding analog input
		>8191	Invalid
15		0	Sampling defined cyclically according to configuration of this register
		1	Cyclic specification of samplings via register "AnalogInputSamples" on page 34 (bus controller default setting)

The time between 2 samples depends on register "MaxFrequencyRaw" on page 57:

Maximum frequency	Sampling time (time between 2 samples)
10000 Hz	38.79 μ s
5000 Hz	77.58 μ s
2000 Hz	193.94 μ s
1000 Hz	387.88 μ s
500 Hz	775.76 μ s
200 Hz	1939.39 μ s

4.8.11 SamplesAnalogInputRead

Name:

SamplesAnalogInput01Read to SamplesAnalogInput04Read

Register for reading the current "SamplesAnalogInput" on page 36 configuration.

Data type	Values
UINT	0 to 65,535

4.9 Automation Runtime support

In [Function model 0 - Standard](#), Automation Runtime also prepares some of the characteristic values calculated by the module for the user.

Information:

To ensure error-free evaluation, it is important to observe the maximum cycle time. The module streams the characteristic values it calculates every 300 ms via Flatstream. If the streamed data is not collected by the next transfer, the measured characteristic values are lost.

Other features provided by Automation Runtime support:

- **"ActSpeed"**: The module always expects a value in 0.01 Hz resolution on these data points. Automation Runtime support allows the user to specify can state the current speed directly in Hz in the "standard" function model.
- **"AnalogInput"**: The analog input is automatically scaled to the sensor resolution and with the defined AnalogInputScale. It is then made available to the user in mg. This scaling does not apply to the "Crest factor" on page 46 since it is a non-dimensional value.
- **Characteristic values and frequency bands**: All characteristic values and frequency bands calculated by the module are flat and can be connected directly in the I/O mapping. They are already scaled to the correct sensor resolution and will be displayed in mg or mm/s or as non-dimensional values (kurtosis, crest factor, skewness and Vdi3832) depending on the characteristic value.
- **Additional characteristic values**: In addition to the characteristic values calculated by the module, the following characteristic values are also provided automatically via Automation Runtime:
 - **Vdi3832KtRaw01-04** Requires PeakRawRef and RmsRawRef as reference values and outputs the reference values used in the calculation to PeakRawRefCalculated and RmsRawRefCalculated.
 - **CrestFactorHighFrequency01-04** Ratio of the absolute maximum to the RMS value ("Crest factor" on page 46) of the high-pass filtered input signal. ("PeakHighFrequency" on page 42 and "RmsHighFrequency" on page 44)
 - **Vdi3832KtHighFrequency01-04** Requires PeakHighFrequencyRef and RmsHighFrequencyRef as reference values and outputs the reference values used in the calculated to PeakHighFrequencyRefCalculated and RmsHighFrequencyRefCalculated.

4.9.1 DataConsistentWithLockedBuffers

Name:

DataConsistentWithLockedBuffers01

If the data buffers on the module are locked to prevent uploading, this bit is used to indicate the time at which all characteristic values and frequency bands are consistent with the locked buffers on the module.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Values
BOOL	0 or 1

4.9.2 DataToggleBit

Name:

DataToggleBit01

This bit changes its value whenever new characteristic values are loaded from the module and updated (approximately every 300 ms).

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Values
BOOL	0 or 1

4.9.3 OverflowAnalogInput

Name:

OverflowAnalogInput01 to OverflowAnalogInput04

Indicates whether a signal is pending on the input that is greater than the configured "AnalogInputScale" on page 35.

Information:

This is always based on a 100 mV/g sensor.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Values
BOOL	0 or 1

4.9.4 OverflowCharacteristicValues

Name:

OverflowCharacteristicValues01 to OverflowCharacteristicValues04

This register contains an overflow indicator bit for each characteristic value of the respective channel.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	PeakHighFrequency	0	No error
		1	Overflow
1	RmsHighFrequency	0	No error
		1	Overflow
2	CrestFactorHighFrequency	0	No error
		1	Overflow
3	Vdi3832KtHighFrequency	0	No error
		1	Overflow
4	RmsAccEnvelope	0	No error
		1	Overflow
5	RmsVelEnvelope	0	No error
		1	Overflow
6	RmsAccRaw	0	No error
		1	Overflow
7	RmsVelRaw	0	No error
		1	Overflow
8	PeakRaw	0	No error
		1	Overflow
9	CrestFactorRaw	0	No error
		1	Overflow
10	SkewnessRaw	0	No error
		1	Overflow
11	KurtosisRaw	0	No error
		1	Overflow
12	Vdi3832KtRaw	0	No error
		1	Overflow
13	Iso10816	0	No error
		1	Overflow
14	RmsRaw	0	No error
		1	Overflow
15	Reserved	0	

4.9.5 OverflowFrequencyBands

Name:

OverflowFrequencyBands01

This register contains an overflow indicator bit for each frequency band.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Values
UDINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	FrequencyBand01	0	No error
		1	Overflow
...		...	
31	FrequencyBand32	0	No error
		1	Overflow

4.9.6 PeakHighFrequencyRef

Name:

PeakHighFrequencyRef01 to PeakHighFrequencyRef04

Reference value (correct state) specified from the application for the peak value of the high-pass filtered input signal used to calculate the Vdi3832 K(t) of the high-pass filtered input signal.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.7 PeakHighFrequencyRefCalculated

Name:

PeakHighFrequencyRefCalculated01 to PeakHighFrequencyRefCalculated04

Reference value (correct state) used by the module for the absolute maximum of the high-pass filtered input signal used to last calculate the Vdi3832 K(t) of the peak value.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.8 PeakRawRef

Name:

PeakRawRef01 to PeakRawRef04

Reference value (correct state) specified by the application for the absolute maximum of the raw signal used to calculate Vdi3832 K(t) of the raw signal.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.9 PeakRawRefCalculated

Name:

PeakRawRefCalculated01 to PeakRawRefCalculated04

Reference value (correct state) used by the module for the absolute maximum of the raw signal used to last calculate the Vdi3832 K(t) of the raw signal.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.10 RmsHighFrequencyRef

Name:

RmsHighFrequencyRef01 to RmsHighFrequencyRef04

Reference value (correct state) specified from the application for the RMS value of the high-pass filtered input signal used to calculate the Vdi3832 K(t) of the high-pass filtered input signal.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.11 RmsHighFrequencyRefCalculated

Name:

RmsHighFrequencyRefCalculated01 to RmsHighFrequencyRefCalculated04

Reference value (correct state) used by the module for the RMS value of the high-pass filtered input signal used to last calculate the Vdi3832 K(t) of the high-pass filtered input signal.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.12 RmsRawRef

Name:

RmsRawRef01 to RmsRawRef04

Reference value (correct state) specified by the application for the RMS value of the raw signal used to calculate Vdi3832 K(t) of the raw signal.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.13 RmsRawRefCalculated

Name:

RmsRawRefCalculated01 to RmsRawRefCalculated04

Reference value (correct state) used by the module for the RMS value of the raw signal used to last calculate the Vdi3832 K(t) of the raw signal.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.9.14 SensitivitySensor

Name:

SensitivitySensor01 to SensitivitySensor04

The module always calculates the characteristic values based on a 100 mV/g sensor on the input. If using a different sensor, the sensor sensitivity can be specified in mV/g for each channel on these registers. All cyclic characteristic values are then automatically scaled to the correct sensor resolution by Automation Runtime. If this parameter is changed, then the next measurement indicated by "[DataToggleBit](#)" on [page 37](#) is invalid.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Data type	Information
REAL	Range of values depends on sensor sensitivity

4.10 Characteristic values

The following applies to all characteristic value module registers:

- These registers are only available in [Function model 2 - Slow master](#) and [Function model 254 - Bus controller](#).
- All calculated characteristic values can be locked using RequestDataLock01, which allows all registers to be read in a consistent manner.

The following characteristic values can be read from the X20CM4810 module for each channel:

Characteristic values	Description
PeakHighFrequency	Absolute maximum of the high-pass filtered input signal.
CrestFactorHighFrequency ¹⁾	Ratio of the absolute maximum to the RMS value of the high-pass filtered input signal
Vdi3832KtHighFrequency ¹⁾	Ratio between the reference values and the currently measured values of the high-pass filtered input signal in accordance with the VDI 3832 guideline
PeakRaw	Peak value (absolute) of the input signal up to the configured maximum frequency
CrestFactorRaw	Ratio of the absolute maximum to the RMS value of the input signal up to the configured maximum frequency
SkewnessRaw	Skewness (third statistical moment) of the input signal up to the configured maximum frequency
KurtosisRaw	Kurtosis (fourth statistical moment) of the input signal up to the configured maximum frequency
Vdi3832KtRaw ¹⁾	Ratio between the reference values and the currently measured values of the input signal in accordance with the VDI 3832 guideline
RmsHighFrequency	RMS value of the high-pass filtered input signal
RmsRaw	RMS value of the input signal up to the configured maximum frequency
RmsAccRaw	RMS value of the input signal's acceleration from the configured minimum frequency up to the configured maximum frequency
RmsVelRaw	RMS value of the input signal's speed from the configured minimum frequency up to the configured maximum frequency ²⁾
Iso10816	RMS value of the velocity in the frequency domain 10 Hz to 1 kHz in accordance with ISO 10816
RmsAccEnvelope	RMS value of the acceleration of the input signal's envelope from the configured minimum frequency up to the configured maximum frequency
RmsVelEnvelope	RMS value of the speed of the input signal's envelope from the configured minimum frequency up to the configured maximum frequency ²⁾

1) Only in [Function model 0 - Standard](#)

2) Only calculated if the EnableVelocityCalculation bit (configured in the "SensorConfig" on [page 29](#) register) of the respective channel is set; otherwise, 0 is output.

4.10.1 Sum of maximum value

The maximum value is often also referred to as the peak value.

The peak value of a mechanical oscillation signal indicate the maximum sum of individual impacts that come from the ambient noise. Different types of damage give rise to strong impacts, which show up in the peak value.

4.10.1.1 PeakHighFrequency

Name:

PeakHighFrequency01 to PeakHighFrequency04

Registers for reading the absolute maximum of the high-pass filtered input signal of the respective channel.

PeakHighFrequency is formed from the high-pass filtered input signal of the oscillation acceleration in the frequency domain between the value set in register "[HighFrequencyConfig](#)" on page 56 and 10 kHz.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	1/65.536	mg	256000.0

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	1/65536	g	16777215

4.10.1.2 PeakRaw

Name:

PeakRaw01 to PeakRaw04

Registers for reading the absolute maximum of the raw signal of the respective channel.

PeakRaw is formed from the raw signal of the oscillation acceleration up to the maximum frequency configured in register "[MaxFrequencyRaw](#)" on page 57.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	1/65.536	mg	256000.0

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	1/65536	g	16777215

4.10.2 RMS value

The RMS value is also known as the quadratic mean, or the root-mean-square. Along with the amplitude, it also takes the energy content of the oscillation into consideration and is the mathematical background for many characteristic values of assessment.

If the RMS is calculated to be above the oscillation velocity, this can be referred to as oscillation speed. In the RMS value, everything contributing to the oscillation is added up. The high oscillation amplitudes of an imbalance are the same as the low oscillation level of bearing damage that is just beginning to occur.

If the RMS value is measured broadly, changes in individual elements contributing to the oscillation can be masked by the averaging. The ability to detect damage early, e.g. due to defects in roller bearings, is limited.

4.10.2.1 Iso10816

Name:

Iso10816_01 to Iso10816_04

Registers for reading the RMS value (per ISO 10816) of the respective channel.

Iso10816 is formed from the raw signal of the oscillation velocity in a frequency range from 10 Hz to 1 kHz.

This broad characteristic value is often used in the assessment of the machine condition since assessment limits are specified for this characteristic value in the standard. These depend on the type of machine and type of installation (rigid or elastic). The characteristic value limits for a pre-warning or a warning are given according to their defined classification.

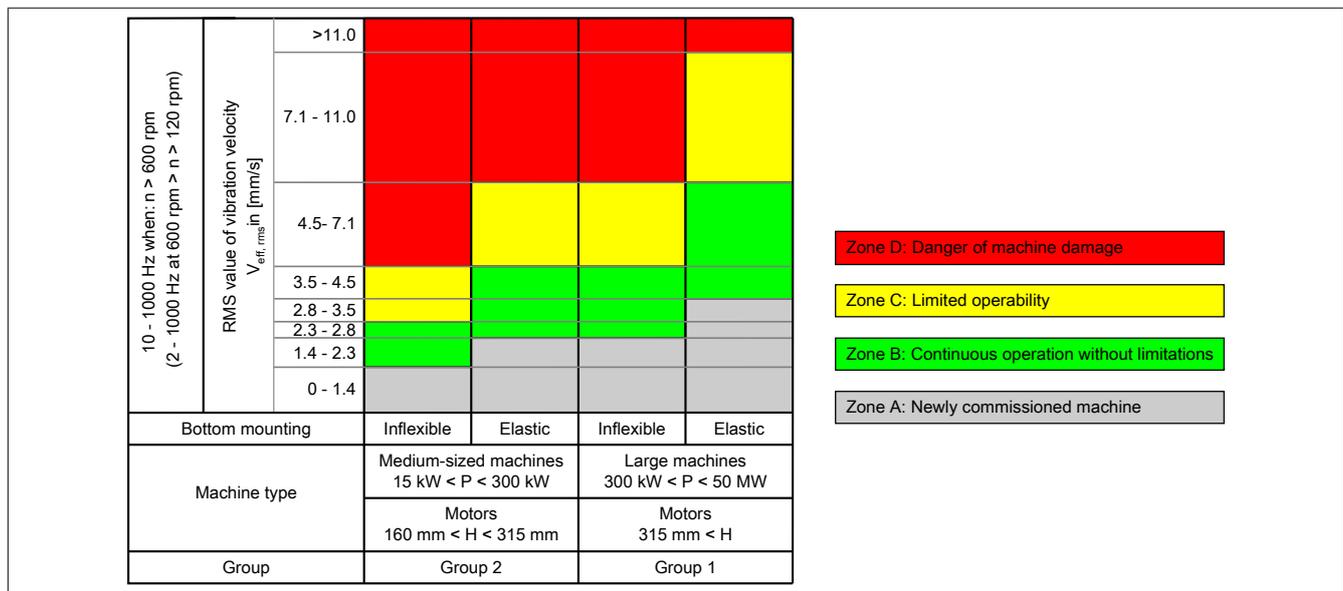


Figure 1: ISO assessment scheme

If the machine condition is in order, this characteristic value is low. If damage occurs, then this value increases severely. If the RMS is assessed in the range of the oscillation velocity, lower frequency portions such as drive speed (and associated imbalance and/or misalignment) that are emphasized more and reflected more heavily in the characteristic value.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	0.001	mm/s	16777.21

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	1	µm/s	16777215

4.10.2.2 RmsAccEnvelope

Name:

RmsAccEnvelope01 to RmsAccEnvelope04

Registers for reading the RMS value of the envelope signal of the respective channel.

RmsAccEnvelope is formed from the envelope signal of the oscillation acceleration from the configured minimum frequency ("[MinFrequencyEnvelope](#)" on page 58) to the configured maximum frequency ("[MaxFrequencyEnvelope](#)" on page 56).

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	1	mg	16777215

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	0.001	g	16777215

4.10.2.3 RmsAccRaw

Name:

RmsAccRaw01 to RmsAccRaw04

Registers for reading the acceleration RMS value of the respective channel.

RmsAccRaw is formed from the raw signal of the oscillation acceleration from the configured minimum frequency ("[MinFrequencyRaw](#)" on page 59) to the configured maximum frequency ("[MaxFrequencyRaw](#)" on page 57).

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	1	mg	16777215

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	0.001	g	16777215

4.10.2.4 RmsHighFrequency

Name:

RmsHighFrequency01 to RmsHighFrequency04

Registers for reading the high-pass RMS value of the respective channel.

RmsHighFrequency is formed from the high-pass filtered input signal of the oscillation acceleration in the frequency domain between the value set in register "[HighFrequencyConfig](#)" on page 56 and 10 kHz.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	1/65.536	mg	256000.0

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	1/65536	g	16777215

4.10.2.5 RmsRaw

Name:

RmsRaw01 to RmsRaw04

Registers for reading the raw signal RMS value of the respective channel.

RmsRaw is formed from the raw signal of the oscillation acceleration up to the maximum frequency configured in register "[MaxFrequencyRaw](#)" on page 57.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	1/65.536	mg	256000.0

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	1/65536	g	16777215

4.10.2.6 RmsVelEnvelope

Name:

RmsVelEnvelope01 to RmsVelEnvelope04

Registers for reading the RMS value of the envelope velocity of the respective channel.

RmsVelEnvelope is formed from the envelope signal of the oscillation acceleration from the configured minimum frequency ("[MinFrequencyEnvelope](#)" on page 58) to the configured maximum frequency ("[MaxFrequencyEnvelope](#)" on page 56).

This characteristic value is only calculated if the EnableVelocityCalculation bit (configuration in register "[Sensor-Config](#)" on page 29) is set for the respective channel; otherwise, 0 is output.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	0.001	mm/s	16777.21

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	1	µm/s	16777215

4.10.2.7 RmsVelRaw

Name:

RmsVelRaw01 to RmsVelRaw04

Registers for reading the velocity RMS value of the respective channel.

RmsVelRaw is formed from the raw signal of the oscillation velocity from the configured minimum frequency ("[Min-FrequencyRaw](#)" on page 59) to the configured maximum frequency ("[MaxFrequencyRaw](#)" on page 57).

This characteristic value is only calculated if the EnableVelocityCalculation bit (configuration in register "[Sensor-Config](#)" on page 29) is set for the respective channel; otherwise, 0 is output.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution	Unit	Value on overflow
REAL	0.001	mm/s	16777.21

Characteristic value in all other function models

Format	Resolution	Unit	Value on overflow
UDINT	1	µm/s	16777215

4.10.3 Crest factor

The crest factor is defined as a the quotient derived from the peak value and the RMS value. In a sinusoidal oscillation, this factor amounts to $\sqrt{2}$. This value is also known in electrical engineering as the crest factor.

In a bearing that is operating normally, the crest factor is also approximately the value $\sqrt{2}$. If the condition of the bearing deteriorates, individual impacts affect the peak value and consequently raise the crest factor. If pronounced defects are then compounded by overall wear, the RMS value is increased further. In the worst case, the crest factor can remain unchanged or even sink again, even despite increasing damage.

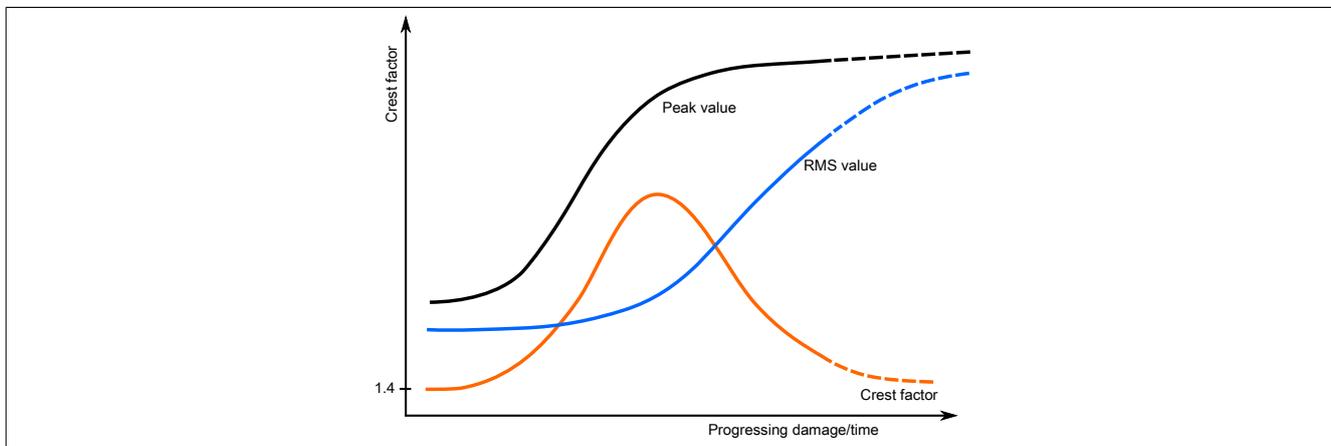


Figure 2: Relationship between the RMS value and peak value

Information:

When a recording of the crest factor is made, the peak and RMS values should also be recorded.

4.10.3.1 CrestFactorHighFrequency

Name:

CrestFactorHighFrequency01 to CrestFactorHighFrequency04

Registers for reading the raw value ratio of the absolute maximum ("[PeakHighFrequency](#)" on page 42) to the RMS value ("[RmsHighFrequency](#)" on page 44) of the respective channel.

CrestFactorHighFrequency is formed from the high-pass filtered input signal of the oscillation acceleration in the frequency domain between the value set in register "[HighFrequencyConfig](#)" on page 56 and 10 kHz.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Format	Resolution
REAL	1

4.10.3.2 CrestFactorRaw

Name:

CrestFactorRaw01 to CrestFactorRaw04

Registers for reading the raw value ratio of the absolute maximum to the RMS value of the respective channel.

CrestFactorRaw is formed from the raw signal of the oscillation acceleration up to the maximum frequency configured in register "[MaxFrequencyRaw](#)" on page 57.

Since this register value is divided by the RMS value in the module ("[RmsRaw](#)" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution
REAL	1

Characteristic value in all other function models

Format	Resolution	Value on overflow
UDINT	0.001	16777215

4.10.4 K(t) value

The K(t) is described in the VDI 3832 guideline and is calculated from the RMS value and the absolute maximum of a broadband time signal of the oscillation acceleration. For the time signal, the entire available frequency spectrum is used for calculation.

This ratio correlates to the reference values. The reference values should be measured by the operator shortly after the running-in time. These values can be classified as "Plant in order" and are therefore the initial values.

The K(t) value decreases with progressive wear. This allows it to be classified into 3 groups:

- Undamaged
- Early damage
- Pronounced damage

The advantage of the K(t) value is that does not change much, even when damage is severe.

$$K(t) = \frac{a_{\text{RMS}}(0) \cdot a_p(0)}{a_{\text{RMS}}(t) \cdot a_p(t)}$$

The following applies:

Formula symbols	Explanation	Characteristic value in the module
$a_{\text{rms}}(0)$	RMS value of the reference value	RmsHighFrequencyRef RmsRawRef
$a_p(0)$	Maximum value of the reference value	PeakHighFrequencyRef PeakRawRef
$a_{\text{rms}}(t)$	Current RMS value	RmsHighFrequency RmsRaw
$a_p(t)$	Current absolute maximum	PeakHighFrequency PeakRaw

Example

Possible progression of the (K/t) characteristic value

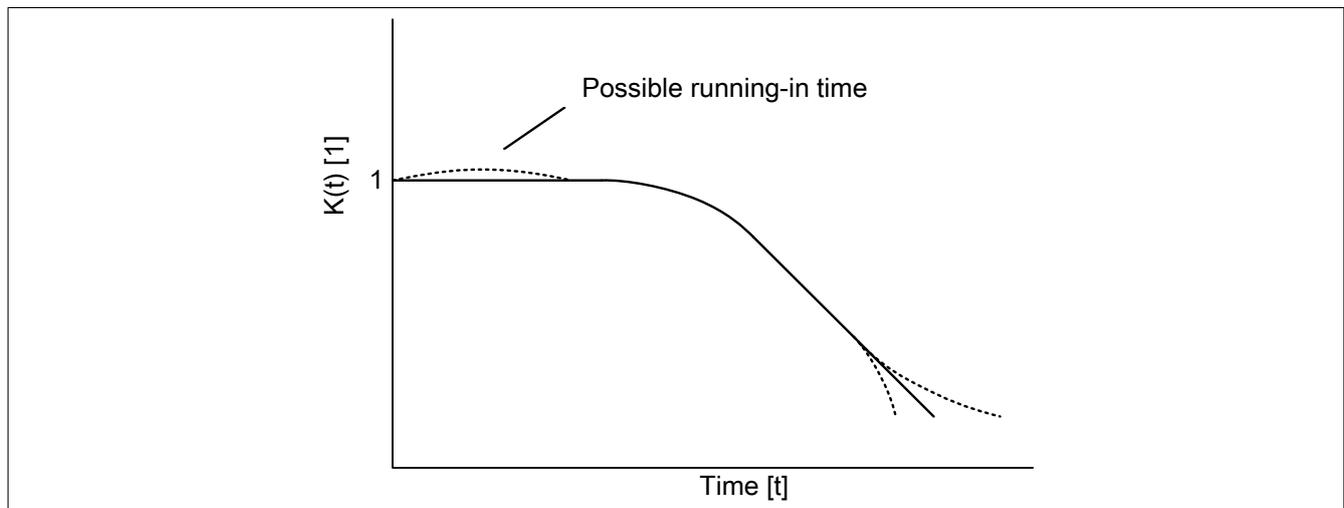


Figure 3: K(t) value progression

4.10.4.1 Vdi3832KtHighFrequency

Name:

Vdi3832KtHighFrequency01 to Vdi3832KtHighFrequency04

Registers for reading the high-pass K(t) value (per VDI 3832 guideline) of the respective channel.

Vdi3832KtHighFrequency is formed from the peak value ("[PeakHighFrequency](#)" on page 42) and RMS value ("[RmsHighFrequency](#)" on page 44) of the high-pass filtered input signal and the vibration acceleration in the frequency range between the value set in register "[HighFrequencyConfig](#)" on page 56 and 10 kHz.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Format	Resolution
REAL	1

4.10.4.2 Vdi3832KtRaw

Name:

Vdi3832KtRaw01 to Vdi3832KtRaw04

Registers for reading the raw K(t) value (per VDI 3832 guideline) of the respective channel.

Vdi3832KtRaw is formed from the raw signal of the oscillation acceleration up to the maximum frequency configured in register "[MaxFrequencyRaw](#)" on page 57.

This register is only available in [Function model 0 - Standard](#) in the Automation Studio I/O mapping.

Format	Resolution
REAL	1

4.10.5 Kurtosis

Kurtosis is an effective characteristic value for assessing the number of peaks in a signal. Kurtosis (peakedness, fourth statistical moment) is defined as the ratio of two statistical characteristic values/processes.

Kurtosis is a type of weighted [Crest factor](#). The signal peaks are valued higher than the "signal noise" (also called the carpet value) due to the fourth power of the integral. Not only is the RMS value also used, but the entire signal progression as well. This increases the informational value of this characteristic value.

Information:

The kurtosis factor is standardized at 0 in the module.

A kurtosis factor of less than 2 is typical of a machine in good condition.

4.10.5.1 KurtosisRaw

Name:

KurtosisRaw01 to KurtosisRaw04

Registers to read the kurtosis factor of the respective channel.

KurtosisRaw is formed from the raw signal of the oscillation acceleration up to the maximum frequency configured in register "[MaxFrequencyRaw](#)" on page 57.

Since this register value is divided by the RMS value in the module ("[RmsRaw](#)" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution
REAL	1

Characteristic value in all other function models

Format	Resolution	Value on overflow
DINT	0.001	8388607

4.10.6 Skewness factor

The skewness factor (skewness, third statistical moment) specifies the degree of a signal's asymmetry in reference to its standard distribution. The lower the skewness, the more evenly distributed the signal. A signal with a high skewness factor has many large amplitudes in the assessment range.

A symmetrically distributed signal has a skewness factor of 0. Depending on the direction of the slant, the skewness can be positive or negative. A considerable slant means approximately a skewness factor of ± 1 .

A large kurtosis factor combined with a large skewness factor indicates electrostatic discharge.

4.10.6.1 SkewnessRaw

Name:

SkewnessRaw01 to SkewnessRaw04

Registers to read the skewness factor of the respective channel.

SkewnessRaw is formed from the raw signal of the oscillation acceleration up to the maximum frequency configured in register "[MaxFrequencyRaw](#)" on page 57.

Since this register value is divided by the RMS value in the module ("[RmsRaw](#)" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution
REAL	1

Characteristic value in all other function models

Format	Resolution	Value on overflow
DINT	0.001	8388607

4.10.7 FrequencyBand

Name:

FrequencyBand01 to FrequencyBand32

Registers for reading the respective frequency band.

If the frequency band is configured to a velocity signal, this value is only calculated if bit EnableVelocityCalculation (configuration in register "[SensorConfig](#)" on page 29) of the corresponding channel is set; otherwise, 0 is output.

For more information, see "[Frequency bands](#)" on page 26.

Characteristic value in [Function model 0 - Standard](#)

Format	Resolution ¹⁾	Unit ¹⁾	Value on overflow
REAL	1	mg	16777215
	1/65.536	mg	256000
	0.001	mm/s	16777.21
	1/65536	mm/s	256

1) Depends on the configuration

Characteristic value in all other function models

Format	Resolution ¹⁾	Unit ¹⁾	Value on overflow
24-bit unsigned	1	mg	16777215
	1/65536	g	256000
	0.001	mm/s	16777.21
	1/65536	mm/s	256

1) Depends on the configuration

4.11 Characteristic values (minimum and maximum)

The module's characteristic values are recalculated every 300 ms. To prevent values from being lost, it is necessary to collect this data quickly enough. If this is not possible, the characteristic values on the module can be locked using data point RequestDataLock01 and then read acyclically in [Function model 2 - Slow master](#) and [Function model 254 - Bus controller](#). However, measurements are lost.

To prevent measurements from being lost, a special function has been implemented in the module that records the minimum and maximum values of all characteristic values calculated in the module. With each edge of bit MinMayUpdate01 in register "Control" on page 28, a new measurement can be started whereby the minimum and maximum values are reinitialized. Simultaneously, the current data is copied to the acyclic registers.

The number of collected measurements can then be read using acyclic register "MinMaxCounter" on page 52. The respective values are read using the acyclic minimum and maximum registers.

Example

For Iso10816, these are the Iso10816Min01-04 and Iso10816Max01-04 registers.

Information:

- If data is locked with RequestDataLock01, no further measured values are collected. This only affects [Function model 2 - Slow master](#) and [Function model 254 - Bus controller](#) since the characteristic values are not locked in the standard function model.
- If a characteristic value overflow or open circuit occurs, no new values are collected.
- The minimum and maximum registers are each initialized using the maximum and minimum of the respective data type. If there is no valid value on the characteristic value for the overall measurement, then the initial value is retained (e.g. on overflow, open circuit or locked data).
- If the data on the module is frozen (DataLockValid01 = 1), no new values are included in the minimum formation, but the measurement cycles are still counted.

The registers are only updated on an edge of "MinMaxUpdate01" on page 28 and only valid if "MinMaxCounter" on page 52 is not equal to 0. Register "MinMaxCounter" on page 52 specifies the number of collected measurement cycles for the minimum and maximum.

4.11.1 CrestFactorRawMax

Name:

CrestFactorRawMax01 to CrestFactorRawMax04

Maximum value of the "CrestFactorRaw" on page 46 characteristic value of the respective channel in 1/1000.

Since this register value is divided by the RMS value in the module ("RmsRaw" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.2 CrestFactorRawMin

Name:

CrestFactorRawMin01 to CrestFactorRawMin04

Minimum value of the "CrestFactorRaw" on page 46 characteristic value of the respective channel in 1/1000.

Since this register value is divided by the RMS value in the module ("RmsRaw" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.3 Iso10816Max

Name:

Iso10816Max01 to Iso10816Max04

Maximum value of the "Iso10816" on page 43 characteristic value of the respective channel in 1/1000 mm/s.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.4 Iso10816Min

Name:

Iso10816Min01 to Iso10816Min04

Minimum value of the "Iso10816" on page 43 characteristic value of the respective channel in 1/1000 mm/s.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.5 KurtosisRawMax

Name:

KurtosisRawMax01 to KurtosisRawMax04

Maximum value of the "KurtosisRaw" on page 48 characteristic value of the respective channel in 1/1000.

Since this register value is divided by the RMS value in the module ("RmsRaw" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
DINT	-2,147,483,648 to 2,147,483,647

4.11.6 KurtosisRawMin

Name:

KurtosisRawMin01 to KurtosisRawMin04

Minimum value of the "KurtosisRaw" on page 48 characteristic value of the respective channel in 1/1000.

Since this register value is divided by the RMS value in the module ("RmsRaw" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
DINT	-2,147,483,648 to 2,147,483,647

4.11.7 MinMaxCounter

Name:

MinMaxCounter01

This register specifies how many measurements were collected the last time the minimum and maximum were determined. It is only updated when an edge occurs for MinMaxUpdate01. If the MinMaxUpdate01 bit is not toggled after 65535 measurements, then the counter is limited to 65535. However, the minimum and maximum formation continues. All acyclic minimum and maximum values are only valid if "MinMaxCounter" on page 52 is not equal to 0.

Data type	Values
UINT	0 to 65,535

4.11.8 PeakHighFrequencyMax

Name:

PeakHighFrequencyMax01 to PeakHighFrequencyMax04

Maximum value of the "PeakHighFrequency" on page 42 characteristic value of the respective channel in 1/65536 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.9 PeakHighFrequencyMin

Name:

PeakHighFrequencyMin01 to PeakHighFrequencyMin04

Minimum value of the "PeakHighFrequency" on page 42 characteristic value of the respective channel in 1/65536 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.10 PeakRawMax

Name:

PeakRawMax01 to PeakRawMax04

Maximum value of the "PeakRaw" on page 42 characteristic value of the respective channel in 1/65536 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.11 PeakRawMin

Name:

PeakRawMin01 to PeakRawMin04

Minimum value of the "PeakRaw" on page 42 characteristic value of the respective channel in 1/65536 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.12 RmsAccEnvelopeMax

Name:

RmsAccEnvelopeMax01 to RmsAccEnvelopeMax04

Maximum value of the "RmsAccEnvelope" on page 44 characteristic value of the respective channel in 1/1000 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.13 RmsAccEnvelopeMin

Name:

RmsAccEnvelopeMin01 to RmsAccEnvelopeMin04

Minimum value of the "RmsAccEnvelope" on page 44 characteristic value of the respective channel in 1/1000 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.14 RmsAccRawMax

Name:

RmsAccRawMax01 to RmsAccRawMax04

Maximum value of the "RmsAccRaw" on page 44 characteristic value of the respective channel in 1/1000 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.15 RmsAccRawMin

Name:

RmsAccRawMin01 to RmsAccRawMin04

Minimum value of the "RmsAccRaw" on page 44 characteristic value of the respective channel in 1/1000 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.16 RmsHighFrequencyMax

Name:

RmsHighFrequencyMax01 to RmsHighFrequencyMax04

Maximum value of the "RmsHighFrequency" on page 44 characteristic value of the respective channel in 1/65536 g.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.17 RmsHighFrequencyMin

Name:

RmsHighFrequencyMin01 to RmsHighFrequencyMin04

Minimum value of the "[PeakHighFrequency](#)" on page 42 characteristic value of the respective channel in 1/65536 g.

See also "[Characteristic values \(minimum and maximum\)](#)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.18 RmsRawMax

Name:

RmsRawMax01 to RmsRawMax04

Maximum value of the "[RmsRaw](#)" on page 45 characteristic value of the respective channel in 1/65536 g.

See also "[Characteristic values \(minimum and maximum\)](#)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.19 RmsRawMin

Name:

RmsRawMin01 to RmsRawMin04

Minimum value of the "[RmsRaw](#)" on page 45 characteristic value of the respective channel in 1/65536 g.

See also "[Characteristic values \(minimum and maximum\)](#)" on page 50 for additional information.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.20 RmsVelEnvelopeMax

Name:

RmsVelEnvelopeMax01 to RmsVelEnvelopeMax04

Maximum value of the "[RmsVelEnvelope](#)" on page 45 characteristic value of the respective channel in 1/1000 mm/s.

See also "[Characteristic values \(minimum and maximum\)](#)" on page 50 for additional information.

This value is only calculated if the EnableVelocityCalculation bit (configuration in register "[SensorConfig](#)" on page 29) is set for the respective channel; otherwise, 0 is output.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.21 RmsVelEnvelopeMin

Name:

RmsVelEnvelopeMin01 to RmsVelEnvelopeMin04

Minimum value of the "[RmsVelEnvelope](#)" on page 45 characteristic value of the respective channel in 1/1000 mm/s.

See also "[Characteristic values \(minimum and maximum\)](#)" on page 50 for additional information.

This value is only calculated if the EnableVelocityCalculation bit (configuration in register "[SensorConfig](#)" on page 29) is set for the respective channel; otherwise, 0 is output.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.22 RmsVelRawMin

Name:

RmsVelRawMin01 to RmsVelRawMin04

Minimum value of the "RmsVelRaw" on page 45 characteristic value of the respective channel in 1/1000 mm/s.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

This value is only calculated if the EnableVelocityCalculation bit (configuration in register "SensorConfig" on page 29) is set for the respective channel; otherwise, 0 is output.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.23 RmsVelRawMax

Name:

RmsVelRawMax01 to RmsVelRawMax04

Maximum value of the "RmsVelRaw" on page 45 characteristic value of the respective channel in 1/1000 mm/s.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

This value is only calculated if the EnableVelocityCalculation bit (configuration in register "SensorConfig" on page 29) is set for the respective channel; otherwise, 0 is output.

Data type	Values
UDINT	0 to 4,294,967,295

4.11.24 SkewnessRawMax

Name:

SkewnessRawMax01 to SkewnessRawMax04

Maximum value of the "SkewnessRaw" on page 49 characteristic value of the respective channel in 1/1000.

Since this register value is divided by the RMS value in the module ("RmsRaw" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
DINT	-2,147,483,648 to 2,147,483,647

4.11.25 SkewnessRawMin

Name:

SkewnessRawMin01 to SkewnessRawMin04

Minimum value of the "SkewnessRaw" on page 49 characteristic value of the respective channel in 1/1000.

Since this register value is divided by the RMS value in the module ("RmsRaw" on page 45), there may be an overflow if the RMS value is very small. To prevent this, the module has an internal lower limit of 1 mg for the RMS value.

See also "Characteristic values (minimum and maximum)" on page 50 for additional information.

Data type	Values
DINT	-2,147,483,648 to 2,147,483,647

4.12 Frequency band configuration registers

4.12.1 HighFrequencyConfig

Name:

HighFrequencyConfig01

Register for defining the envelope signal high-pass and characteristic values "[PeakHighFrequency](#)" on page 42 and "[RmsHighFrequency](#)" on page 44. This setting applies to the entire module.

Data type	Values	Bus controller default setting
UINT	See bit structure.	1

Bit structure:

Bit	Description	Values	Information
0 - 3	High-pass configuration for the entire module	0	2000 Hz
		1	1000 Hz (bus controller default setting)
		2	500 Hz
		3 to 15	Invalid
4 - 15	Reserved	-	

4.12.2 HighFrequencyConfigRead

Name:

HighFrequencyConfig01Read

Register for reading the envelope signal high-pass and characteristic values "[PeakHighFrequency](#)" on page 42 and "[RmsHighFrequency](#)" on page 44.

Data type	Values
UINT	0 to 65,535

4.12.3 MaxFrequencyEnvelope

Name:

MaxFrequencyEnvelope01

Registers for setting the maximum frequency of the channel. Reducing the maximum frequency allows the frequency resolution in the spectrum to be increased.

Data type	Values	Bus controller default setting
UINT	See the bit structure.	0

Bit structure:

Bit	Description	Value	Information
0 - 3	Maximum frequency for channel 1	0	2000 Hz (bus controller default setting)
		1	1000 Hz
		2	500 Hz
		3	200 Hz
		4 to 15	Invalid
4 - 7	Maximum frequency for channel 2	x	For possible values, see channel 1.
8 - 11	Maximum frequency for channel 3	x	For possible values, see channel 1.
12 - 15	Maximum frequency for channel 4	x	For possible values, see channel 1.

Overview of the maximum frequency of the envelope signal

Maximum frequency	Sampling frequency	Duration of measurement	Frequency resolution in the frequency spectrum
2000 Hz	5156 Hz	1.5888 s	0.6294 Hz
1000 Hz	2578 Hz	3.1775 s	0.3147 Hz
500 Hz	1289 Hz	6.3550 s	0.1574 Hz
200 Hz	516 Hz	15.8875 s	0.0629 Hz

Important information for configuring the maximum frequency

- The frequency range must be larger than the damage frequency but should otherwise be kept as small as possible in order to achieve a good resolution.
- When using broadband values (e.g. PeakRaw), only the largest peak value is returned during a measurement. Using a longer measurement time at a lower frequency can lead to the measured value being overlooked in some applications.
- The maximum frequency influences the size of the sampling rate and can be configured using register "[AnalogInput](#)" on page 36.

4.12.4 MaxFrequencyEnvelopeRead

Name:

MaxFrequencyEnvelope01Read

This register is used to read the configured maximum frequency for the individual channels' envelope signal.

Data type	Values
UINT	0 to 65,535

4.12.5 MaxFrequencyRaw

Name:

MaxFrequencyRaw01

Registers for setting the maximum frequency of the channel. Reducing the maximum frequency allows the frequency resolution in the spectrum to be increased.

Data type	Values	Bus controller default setting
UINT	See bit structure.	8738

Bit structure:

Bit	Description	Value	Information
0 - 3	Maximum frequency for channel 1	0	10000 Hz
		1	5000 Hz
		2	2000 Hz (bus controller default setting)
		3	1000 Hz
		4	500 Hz
		5	200 Hz
		6 to 15	Invalid
4 - 7	Maximum frequency for channel 2	x	For possible values, see channel 1.
8 - 11	Maximum frequency for channel 3	x	For possible values, see channel 1.
12 - 15	Maximum frequency for channel 4	x	For possible values, see channel 1.

Overview of the maximum frequency of the raw signal

Maximum frequency	Sampling frequency	Duration of measurement	Frequency resolution in the frequency spectrum
10000 Hz	25781 Hz	0.3178 s	3.1471 Hz
5000 Hz	12891 Hz	0.6355 s	1.5736 Hz
2000 Hz	5156 Hz	1.5888 s	0.6294 Hz
1000 Hz	2578 Hz	3.1775 s	0.3147 Hz
500 Hz	1289 Hz	6.3550 s	0.1574 Hz
200 Hz	516 Hz	15.8875 s	0.0629 Hz

Important information for configuring the maximum frequency

- The frequency range must be larger than the damage frequency but should otherwise be kept as small as possible in order to achieve a good resolution.
- When using broadband values (e.g. PeakRaw), only the largest peak value is returned during a measurement. Using a longer measurement time at a lower frequency can lead to the measured value being overlooked in some applications.
- The maximum frequency influences the size of the sampling rate and can be configured using register ["AnalogInput"](#) on page 36.

4.12.6 MaxFrequencyRawRead

Name:

MaxFrequencyRaw01Read

This register is used to read the configured maximum frequency for the raw signal of individual channels.

Data type	Values
UINT	0 to 65,535

4.12.7 MinFrequencyEnvelope

Name:

MinFrequencyEnvelope01

This register is used to configure the lowest frequency of the envelope signal to be evaluated for the individual channels.

This configuration only needs to be set for channels whose EnableVelocityCalculation bit has been set (configured in the "SensorConfig" on page 29 register).

Bit EnableVelocityCalculation = 0

The following minimum values based on the maximum frequency ("MaxFrequencyEnvelope" on page 56) are used: This table shows the minimum MinFrequencyEnvelope based on MaxFrequencyEnvelope:

Maximum frequency	Frequency resolution in the frequency spectrum	Minimum frequency
2000 Hz	0.6294 Hz	1.888 Hz
1000 Hz	0.3147 Hz	0.944 Hz
500 Hz	0.1574 Hz	0.472 Hz
200 Hz	0.0629 Hz	0.188 Hz

Bit EnableVelocityCalculation = 1

Data type	Values	Bus controller default setting
UINT	See the bit structure.	0

Bit structure:

Bit	Description	Value	Information
0 - 3	Lowest frequency for channel 1	0	10 Hz (bus controller default setting)
		1	5 Hz
		2	2 Hz
		3	1 Hz
		4	0.5 Hz
		5 to 15	Invalid
4 - 7	Lowest frequency for channel 2	x	For possible values, see channel 1.
8 - 11	Lowest frequency for channel 3	x	For possible values, see channel 1.
12 - 15	Lowest frequency for channel 4	x	For possible values, see channel 1.

Information:

If a channel's frequency is set lower than the minimum frequency, then the channel will be limited to this lower frequency.

4.12.8 MinFrequencyEnvelopeRead

Name:

MinFrequencyEnvelope01Read

This register is used to read the lowest frequency of the envelope signal to be evaluated for the individual channels.

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Lowest frequency for channel 1	0	10 Hz
		1	5 Hz
		2	2 Hz
		3	1 Hz
		4	0.5 Hz
		5 to 14	Invalid
		15	Limited to lowest frequency
4 - 7	Lowest frequency for channel 2	x	For possible values, see channel 1.
8 - 11	Lowest frequency for channel 3	x	For possible values, see channel 1.
12 - 15	Lowest frequency for channel 4	x	For possible values, see channel 1.

4.12.9 MinFrequencyRaw

Name:

MinFrequencyRaw01

This register is used to configure the lowest frequency of the raw signal to be evaluated for the individual channels.

This configuration only needs to be set for channels whose EnableVelocityCalculation bit has been set (configured in the "SensorConfig" on page 29 register).

Bit EnableVelocityCalculation = 0

The following minimum values based on the maximum frequency ("MaxFrequencyRaw" on page 57) are used: This table shows the minimum MinFrequencyRaw based on MaxFrequencyRaw:

Maximum frequency	Frequency resolution in the frequency spectrum	Minimum frequency
10000 Hz	3.1471 Hz	9.441 Hz
5000 Hz	1.5736 Hz	4.720 Hz
2000 Hz	0.6294 Hz	1.888 Hz
1000 Hz	0.3147 Hz	0.944 Hz
500 Hz	0.1574 Hz	0.472 Hz
200 Hz	0.0629 Hz	0.188 Hz

Bit EnableVelocityCalculation = 1

Data type	Values	Bus controller default setting
UINT	See the bit structure.	0

Bit structure:

Bit	Description	Value	Information
0 - 3	Lowest frequency for channel 1	0	10 Hz (bus controller default setting)
		1	5 Hz
		2	2 Hz
		3	1 Hz
		4	0.5 Hz
		5 to 15	Invalid
4 - 7	Lowest frequency for channel 2	x	For possible values, see channel 1.
8 - 11	Lowest frequency for channel 3	x	For possible values, see channel 1.
12 - 15	Lowest frequency for channel 4	x	For possible values, see channel 1.

Information:

If a channel's frequency is set lower than the minimum frequency, then the channel will be limited to this lower frequency.

4.12.10 MinFrequencyRawRead

Name: MinFrequencyRaw01Read

This register is used to configure the lowest frequency of the raw signal to be evaluated for the individual channels.

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Lowest frequency for channel 1	0	10 Hz
		1	5 Hz
		2	2 Hz
		3	1 Hz
		4	0.5 Hz
		5 to 14	Invalid
15	Limited to lowest frequency		
4 - 7	Lowest frequency for channel 2	x	For possible values, see channel 1.
8 - 11	Lowest frequency for channel 3	x	For possible values, see channel 1.
12 - 15	Lowest frequency for channel 4	x	For possible values, see channel 1.

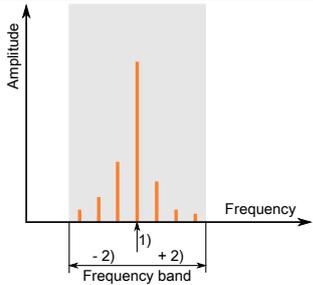
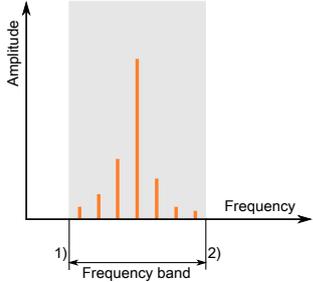
4.13 Frequency bands

For the early detection of damage and problems, it is often useful to monitor individual frequency bands. A selective RMS value can be used as the characteristic value for these frequency bands.

Possibilities:

- "Broadband RMS value" on page 61
- "Speed-dependent RMS value" on page 62
- "Noise" on page 62

Up to 32 different frequency bands can be defined. It is important to note that the format varies slightly depending on the configuration.

Speed-dependent RMS value	
	<p>Frequency broadband is formed from the speed frequency \pm the frequency interval.</p> <ol style="list-style-type: none"> 1) Speed frequency: <code>FrequencyBandxxDmgFreq60Rpm * ActSpeed</code> 2) Frequency interval: <code>FrequencyBandTolerance</code>
Speed-independent RMS value	
	<p>Frequency bandwidth is formed of the lower and upper frequency band limits.</p> <ol style="list-style-type: none"> 1) Lower frequency band limit: <code>FrequencyBandLowerFrequency</code> 2) Upper frequency band limit: <code>FrequencyBandUpperFrequency</code>

Restricting the frequency range allows certain errors to be identified easily.

One example of this is imbalance. This is clearly shown by an increase in the frequency line for speed. Forming a selective characteristic value for the speed frequency can thus improve classification.

Selective characteristic values can also be formed for RMS values formed from the envelope.

Damage to the outer ring is shown more clearly in the increase of the frequency known as the bearing damage frequency. Damage to the outer ring is shown in the increase of the component of the outer ring damage frequency.

This bearing damage frequency is generally available from the bearing manufacturer.

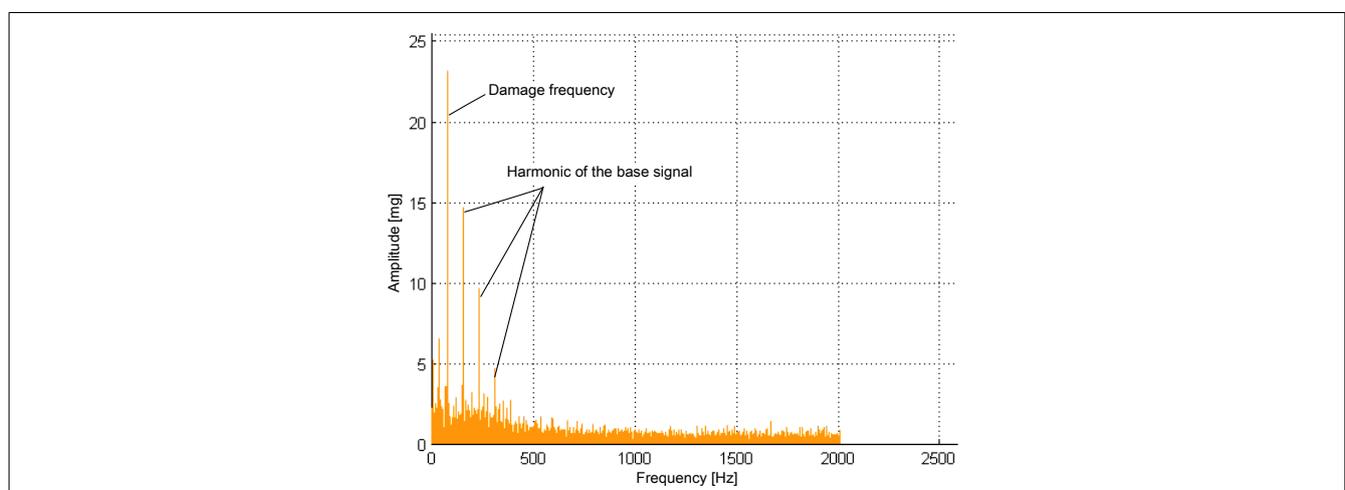


Figure 4: Selective characteristic value for outer ring damage

By configuring a characteristic value with a frequency range around the damage frequency, rolling bearing damage to the outer ring can be detected at an early stage.

If frequencies are entered that are outside the minimum and maximum signal frequency for the selected channel, then only the domains between the minimum and maximum frequency will be analyzed.

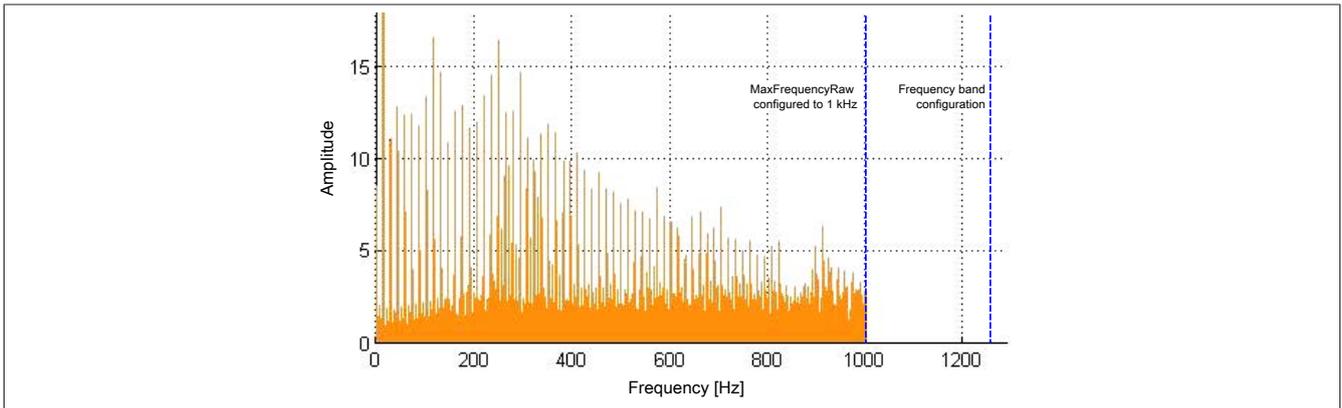


Figure 5: Restricting the frequency band evaluation

The two neighboring lines (samples) in the spectrum that are already outside the set window (one above and one below the window) will be partially included in the calculation depending on their distance from the window.

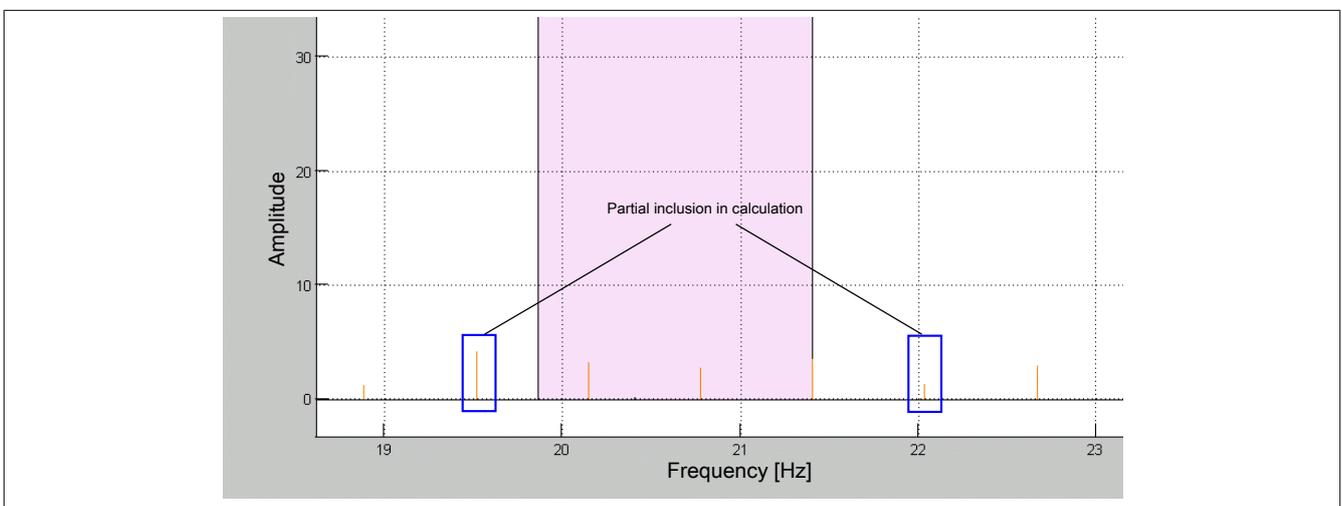


Figure 6: Partial inclusion of marginal lines in the calculation

4.13.1 Broadband RMS value

In this configuration, the RMS value of the configured signal and channel in the frequency band is calculated. The value is calculated from the configured minimum frequency ("[FrequencyBandXXLowerFrequency](#)" on page 65) to the configured maximum frequency ("[FrequencyBandXXUpperFrequency](#)" on page 66). The minimum and maximum frequency can be entered here in increments of 0.25 Hz.

Any channel for any frequency band can be selected. The following signals can be selected for each channel:

- Raw acceleration signal
- Raw velocity signal. Equal to 0 if the speed calculation is disabled.
- Enveloped acceleration signal
- Enveloped velocity signal. Equal to 0 if the speed calculation is disabled.

The harmonic frequencies (integer multiples) of the window can also be included in the calculation. Here, the width of the window is simply retained and the mean frequency of the window is multiplied (by 1, 2, 3, etc.) until the maximum frequency of the configured signal and channel is reached.

4.13.2 Speed-dependent RMS value

In this configuration, the RMS value is calculated in a movable window. There are 4 speed inputs available for this (see "ActSpeed" on page 27 register). One of the 4 speeds can be selected for each of the 32 frequency bands. In addition, the standardized damage frequency at 60 rpm ("FrequencyBandXXDmgFreq60rpm" on page 65) and a tolerance ("FrequencyBandXXTolerance" on page 65) must be configured. These can be configured separately for each frequency band.

The window in which the RMS value is calculated is determined as follows:

Minimum frequency = (speed * standardized damage frequency at 60 rpm) - tolerance

Maximum frequency = (speed * standardized damage frequency at 60 rpm) + tolerance

The standardized damage frequency and tolerance can be entered here in increments of 0.01 Hz.

The following signals can be selected for each channel:

- Raw acceleration signal
- Raw velocity signal. Equal to 0 if the speed calculation is disabled.
- Enveloped acceleration signal
- Enveloped velocity signal. Equal to 0 if the speed calculation is disabled.

The harmonic frequencies (integer multiples) of the window can also be included in the calculation. Here, the width of the window is simply retained and the mean frequency of the window is multiplied (by *1, *2, *3, etc.) until the maximum frequency of the set signal and channel is reached.

Information:

If a fixed frequency band is needed in which the minimum frequency ("FrequencyBandXXLowerFrequency" on page 65) and maximum frequency ("FrequencyBandXXUpperFrequency" on page 66) must be set with a higher precision than 0.25 Hz, then a speed-dependent frequency band with a fixed speed can be used.

4.13.3 Noise

In this configuration, the noise from a quadrant of the respective signal on the selected channel that is within the frequency band is calculated.

To do this, the configured maximum frequency (registers "MaxFrequencyEnvelope" on page 56 and "MaxFrequencyRaw" on page 57) of the signal on the selected channel is divided by 4. This results in 4 quadrants. The configuration can then be used to select one of the 4 quadrants in which the noise should be determined.

The following signals can be selected for each channel:

- Raw acceleration signal
- Raw velocity signal. Equal to 0 if the speed calculation is disabled.
- Enveloped acceleration signal
- Enveloped velocity signal. Equal to 0 if the speed calculation is disabled.

This configuration allows slippage to be effectively measured, for example. The higher the friction, the more noise that is created.

4.13.4 Configuration

Each of the 32 frequency bands can be configured to one of the following characteristic values.

Characteristic value	Description
FrequencyBandRmsAccEnvelope	The RMS value formed from the envelope signal of the oscillation acceleration in a freely selectable frequency range.
FrequencyBandRmsVelEnvelope	The RMS value formed from the envelope signal of the oscillation velocity in a freely selectable frequency range. This frequency band is only calculated if the EnableVelocityCalculation bit (configuration in register "SensorConfig" on page 29) is set for the respective channel; otherwise, 0 is output.
FrequencyBandRmsAccRaw	The RMS value formed from the raw signal of the oscillation acceleration in a freely selectable frequency range.
FrequencyBandRmsVelRaw	The RMS value formed from the raw signal of the oscillation velocity in a freely selectable frequency range.
FrequencyBandNoiseAccEnvelope	The noise value formed from the envelope signal of the oscillation acceleration in a freely selectable frequency range.
FrequencyBandNoiseVelEnvelope	The noise value formed from the raw signal of the oscillation velocity in a freely selectable frequency range. This frequency band is only calculated if the EnableVelocityCalculation bit (configuration in register "SensorConfig" on page 29) is set for the respective channel; otherwise, 0 is output.
FrequencyBandNoiseAccRaw	The noise value formed from the raw signal of the oscillation acceleration in a freely selectable frequency range.
FrequencyBandNoiseVelRaw	The noise value formed from the raw signal of the oscillation velocity in a freely selectable frequency range. This frequency band is only calculated if the EnableVelocityCalculation bit (configuration in register "SensorConfig" on page 29) is set for the respective channel; otherwise, 0 is output.

Calculating the total width of the frequency band

The measured damage frequency may vary slightly from the expected frequency. It must therefore be ensured that the frequency band is wide enough to monitor the harmful frequency to compensate for this deviation.

The maximum permissible deviation has been defined as 1/2% of the maximum drive frequency. Since the deviation could occur in either the positive or negative direction, it must be counted twice. This results in the following formula:

$$\text{Total range of frequency band} = 2 * 1/2\% * \text{max. drive frequency [Hz]}$$

Depending on the MaxFrequency selected, a least 3 lines should be configured within each frequency band.

The first lines to the left and right of the configured frequency band are included proportionally based on their distance from the frequency band (see figure in "Frequency bands" on page 60).

Example

At a maximum drive frequency of 50 Hz, a frequency band should be set at 10 Hz.

10 Hz ± (0.005 * 50 Hz) = At least 9.75 Hz lower and 10.25 Hz upper frequency band limits

4.13.5 FrequencyBandMax

Name:

FrequencyBandMax01 to FrequencyBandMax32

Maximum value of the respective frequency band in 1/1000 mm/s or g depending on the configuration. See also "Characteristic values (minimum and maximum)" on page 50 and "Frequency bands" on page 26 for additional information.

If the frequency band is configured to a velocity signal, this value is only calculated if bit EnableVelocityCalculation (configuration in register "SensorConfig" on page 29) of the corresponding channel is set; otherwise, 0 is output.

Data type	Values
UDINT	0 to 4,294,967,295

4.13.6 FrequencyBandMin

Name:

FrequencyBandMin01 to FrequencyBandMin32

Minimum value of the respective frequency band in 1/1000 mm/s or g depending on the configuration. See also "Characteristic values (minimum and maximum)" on page 50 and "Frequency bands" on page 26 for additional information.

If the frequency band is configured to a velocity signal, this value is only calculated if bit EnableVelocityCalculation (configuration in register "SensorConfig" on page 29) of the corresponding channel is set; otherwise, 0 is output.

Data type	Values
UDINT	0 to 4,294,967,295

4.13.7 FrequencyBandConfig

Name:

FrequencyBand01Config to FrequencyBand32Config

General configuration of individual frequency bands.

Each frequency band can be calculated on any channel with any of the 4 speed data points ("[ActSpeed](#)" on page 27).

Data type	Values	Bus controller default setting
UINT	See the bit structure.	0

Bit structure:

Bit	Description	Value	Information
0 - 3	Input channel for calculating the frequency band	0	Channel 1 (bus controller default setting)
		1	Channel 2
		2	Channel 3
		3	Channel 4
		4 to 15	Invalid
4 - 5	Signal for calculating the frequency band	0	Raw signal speed ¹⁾ (bus controller default setting)
		1	Enveloped velocity signal ¹⁾
		2	Raw acceleration signal
		3	Enveloped acceleration signal
6	Speed-dependant calculation of the frequency band of the configured ActSpeed data point	0	Off Uses " FrequencyBandLowerFrequency " on page 65 and " FrequencyBandUpperFrequency " on page 66 for the calculation (bus controller default setting)
		1	On " FrequencyBandDmgFreq60rpm " on page 65 and " FrequencyBandTolerance " on page 65 are used for the calculation.
7	Harmonic frequencies also calculated	0	Off (bus controller default setting)
		1	On All harmonic frequencies up to the maximum frequency are always calculated. The harmonic frequencies can also include frequency portions that are the result of damage at other positions. This can lead to a misinterpretation of the measurement.
8 - 11	ActSpeed data point used for the calculation if the frequency band is speed-dependent	0	ActSpeed01 (bus controller default setting)
		1	ActSpeed02
		2	ActSpeed03
		3	ActSpeed04
		4 to 15	Invalid
12	Enables noise calculation instead of RMS	0	Off (bus controller default setting)
		1	On The speed-dependent and harmonic settings are ignored.
13 - 14	Selects the quadrant to calculate the noise	0	1st quadrant from MinFrequency to 1/4 MaxFrequency ²⁾ (bus controller default setting)
		1	2nd quadrant from 1/4 MaxFrequency to 1/2 MaxFrequency ²⁾
		2	3rd quadrant from 1/2 MaxFrequency to 3/4 MaxFrequency ²⁾
		3	4th quadrant from 3/4 MaxFrequency to MaxFrequency ²⁾
15	Reserved	0	

1) This is only calculated if the EnableVelocityCalculation bit (configuration in register "[SensorConfig](#)" on page 29) is set for the respective channel; otherwise, 0 is output.

2) Refers to the respective signal (raw signal or envelope signal) on the channel

4.13.8 FrequencyBandConfigRead

Name:

FrequencyBand01ConfigRead to FrequencyBand32ConfigRead

Registers for reading the configuration of individual frequency bands.

Data type	Values
UINT	0 to 65,535

4.13.9 FrequencyBandDmgFreq60rpm

Name:

FrequencyBand01DmgFreq60rpm to FrequencyBand32DmgFreq60rpm

Standardized damage frequency at 60 rpm if the frequency band is configured as speed-dependent.

This is multiplied with the configured velocity data point in the module to calculate the frequency band.

The standardized damage frequency must be specified in 1/100.

Data type	Values	Information
UINT	1 to 65,535	Bus controller default setting: 0

4.13.10 FrequencyBandDmgFreq60rpmRead

Name:

FrequencyBand01DmgFreq60rpmRead to FrequencyBand32DmgFreq60rpmRead

Register to read the standardized damage frequency at 60 rpm for the individual frequency bands.

The standardized damage frequency is specified in 1/100.

Data type	Values
UINT	1 to 65,535

4.13.11 FrequencyBandTolerance

Name:

FrequencyBand01Tolerance to FrequencyBand32Tolerance

If the frequency band is configured as speed-dependent, then this data point can be used to specify the frequency band's width.

The [FrequencyBandTolerance](#) is subtracted once from the damage frequency calculated from [ActSpeed](#) and [FrequencyBandDmgFreq60rpm](#) to get the frequency band's lower frequency and added once to get its higher frequency.

The tolerance must be specified in 1/100 Hz.

Data type	Values	Information
UINT	1 to 65,535	Bus controller default setting: 0

4.13.12 FrequencyBandToleranceRead

Name:

FrequencyBand01ToleranceRead to FrequencyBand32ToleranceRead

Registers for reading the tolerance of individual frequency bands.

The tolerance is specified in 1/100 Hz.

Data type	Values
UINT	1 to 65,535

4.13.13 FrequencyBandLowerFrequency

Name:

FrequencyBand01LowerFrequency to FrequencyBand32LowerFrequency

Minimum frequency for calculating the frequency band if it is not speed-dependent.

The minimum frequency must be specified in 1/4 Hz.

Data type	Values	Information
UINT	1 to 40,000	Bus controller default setting: 0

4.13.14 FrequencyBandLowerFrequencyRead

Name:

FrequencyBand01LowerFrequencyRead to FrequencyBand32LowerFrequencyRead

Registers for reading the minimum frequency of individual frequency bands.

The minimum frequency is specified in 1/4 Hz.

Data type	Values
UINT	1 to 40,000

4.13.15 FrequencyBandUpperFrequency

Name:

FrequencyBand01UpperFrequency to FrequencyBand32UpperFrequency

Maximum frequency for calculating the frequency band if it is not speed-dependent.

The maximum frequency must be specified in 1/4 Hz.

Data type	Values	Information
UINT	1 to 40,000	Bus controller default setting: 0

4.13.16 FrequencyBandUpperFrequencyRead

Name:

FrequencyBand01UpperFrequencyRead to FrequencyBand32UpperFrequencyRead

Registers for reading the maximum frequency of individual frequency bands.

The maximum frequency is specified in 1/4 Hz.

Data type	Values
UINT	1 to 40,000

4.14 Flatstream

4.14.1 Transferring characteristic values via Flatstream

In function model "Standard", Automation Runtime takes over Flatstream communication for transferring characteristic values. The user is automatically provided with all characteristic values, already properly scaled. As a result, the registers for the characteristic value Flatstream are not shown in the I/O mapping in [Function model 0 - Standard](#).

When using Flatstream in [Function model 0 - Standard](#) and [Function model 1 - Fast master](#), the following should be noted:

- The maximum bus cycle time should not exceed 10 ms.
- The task cycle should either be the same speed and synchronous to the module's bus or faster.

Bus cycle time

Since the characteristic values are transferred over the Flatstream, a maximum bus cycle time of 10 ms should not be exceeded. Otherwise, the data calculated by the module every 300 ms cannot be fully transferred on the bus. If the current transfer is not completed when new characteristic values become available, then they will be discarded. A cycle time of ≤ 10 ms is therefore required in order to ensure seamless measurements.

Task cycle time

It is important to note that the module normally changes the values in the Flatstream in every X2X Link cycle. To optimize performance and avoid missing any values in the task, it is important to select a task cycle that is either the same speed and synchronous to the module's bus or faster. This also applies to fieldbus connections between the CPU and bus controller. If this is not possible for some reason, it is also possible to change the ForwardDelay of the Flatstream (see ["The Forward function on the X20CM4810" on page 69](#)).

4.14.1.1 Registers for the characteristic value Flatstream

The following registers are needed to transfer characteristic values:

["ParameterInputSequence01" on page 76](#)

["ParameterRxByte01-13" on page 74](#)

["ParameterOutputSequence01" on page 75](#)

This results in an [InputMTU](#) of 13 and an [OutputMTU](#) of 0 bytes for the characteristic value Flatstream.

4.14.1.2 Structure of the characteristic value Flatstream

Each characteristic value transferred from the module via Flatstream has a length of 3 bytes with the high byte first. For the exact format, see ["Characteristic values" on page 41](#). The stream has a total length of 240 bytes. First, all characteristic values from channel 1 are transferred, then those from channels 2, 3 and 4. The 32 frequency bands then follow next.

The RmsVelRaw and RmsVelEnvelope characteristic values and FrequencyBandxx are only transferred if they are set to a velocity signal and the EnableVelocityCalculation bit for the respective channel is set. Otherwise, 0 is output. EnableVelocityCalculation is configured in the ["SensorConfig" on page 29](#) register.

Byte offset in the stream				Characteristic value
Channel 1	Channel 2	Channel 3	Channel 4	
Channel parameters				
0	36	72	108	RmsAccRaw
3	39	75	111	RmsVelRaw
6	42	78	114	RmsAccEnvelope
9	45	81	117	RmsVelEnvelope
12	48	84	120	PeakHighFrequency
15	51	87	123	RmsHighFrequency
18	54	90	126	Iso10816
21	57	93	129	CrestFactorRaw
24	60	96	132	KurtosisRaw
27	63	99	135	PeakRaw
30	66	102	138	SkewnessRaw
33	69	105	141	RmsRaw
Frequency band 1 - 32				
144				FrequencyBand01
:				:
237				FrequencyBand32

4.14.2 Uploading buffers via Flatstream

For information about Flatstream, see ["Flatstream communication" on page 70](#).

Uploading buffers via Flatstream is available on the module in [Function model 0 - Standard](#) and [Function model 1 - Fast master](#).

Library **AsIOVib** is available for uploading the buffers from the module. For a description of the library, see ["Automation Help - Programming - Libraries - Direct I/O access - AsIOVib"](#).

The raw data buffers (raw signal and FFT) from the module are transferred on request (buffer Flatstream in the Tx direction) from the module via the buffer Flatstream (Rx direction) but only if the buffers were previously frozen on the module with settings `RequestBufferLock01 = 1` and `BufferLockValid01 = 1`. It is important to note that the module refreshes the data points of the buffer Flatstream every X2X Link cycle. Once the transmission is complete, the buffer from the stream needs to be "reassembled" on the PLC.

4.14.2.1 Registers for the buffer Flatstream

The following registers are needed to upload buffers:

["BufferInputSequence01" on page 76](#)

["BufferRxByte01-05" on page 74](#)

["BufferOutputSequence01" on page 75](#)

["BufferTxByte01-04" on page 74](#)

This results in an `InputMTU` of 5 and an `OutputMTU` of 4 bytes for the buffer Flatstream.

4.14.2.2 Buffer upload procedure

Before a buffer can be uploaded from the module, the buffers on the module must first be locked by setting `RequestBufferLock01 = 1`. The upload can only begin once the buffers have been locked by setting `BufferLockValid01 = 1`.

If a velocity buffer is uploaded, it only contains values if bit `EnableVelocityCalculation` of the corresponding channel is set; otherwise, 0 is output. Bit `EnableVelocityCalculation` can be configured in register ["SensorConfig" on page 29](#).

A buffer first needs to be requested from the module via the buffer Flatstream (Tx direction). Meaning:

- `BufferTxByte01`: Value 0x83 (frame end and 3 bytes are valid)
- `BufferTxByte02`: The requested buffer from the module
- `BufferTxByte03`: High byte of the number of the values to be read (per 4-byte value)
- `BufferTxByte04`: Low byte of the number of the values to be read (per 4-byte value)

The sequence is subsequently increased by 1. As soon as the sequence has been confirmed by the module, it is important to set the number of valid bytes in `BufferTxByte` to 0; otherwise, the module interprets this as a new request. The module can temporarily store up to 2 consecutive requests.

As soon as the module receives the request, it begins streaming the requested number of values from the specified buffer via the buffer Flatstream (Rx direction). Values are always transferred in 16.16 fixed data point format (1/65536) with the high byte first. A scaling factor is transferred first. All other values are then multiplied by this factor. With time signal buffers, the first value according to the scaling factor is always the oldest. With FFT buffers, the first value according to the scaling factor is always 0 Hz. FFT buffers are only valid from the configured `MinFrequency` to the configured `MaxFrequency` of the respective signal (raw or envelope) and the respective channel.

For the time and frequency intervals of individual values, see register ["MinFrequencyRaw" on page 59](#) or ["MinFrequencyEnvelope" on page 58](#).

`RequestBufferLock01` must be set to 1 throughout the entire uploading process. Once the requested buffers have been uploaded from the module, the lock can be reset. In the module, the buffers are then once again filled with new values. After some time has passed, they can be locked and uploaded again (see register ["Control" on page 28](#)).

Buffer number (dec.)				Buffer	Maximum number of values (1 values = 4 bytes)
Channel 1	Channel 2	Channel 3	Channel 4		
9	11	13	15	Raw signal (filter value: "MaxFrequencyRaw" on page 57)	8193 or 65535 ¹⁾
25	27	29	31	Envelope signal (filter value: "MinFrequencyEnvelope" on page 58)	8193 or 65535 ¹⁾
66	70	74	78	FFT amplitude spectrum raw velocity signal (filter value: "MaxFrequencyRaw" on page 57)	4097 ¹⁾
67	71	75	79	FFT amplitude spectrum raw acceleration signal (filter value: "MaxFrequencyRaw" on page 57)	4097 ¹⁾
82	86	90	94	FFT amplitude spectrum envelope signal velocity (filter value: "MinFrequencyEnvelope" on page 58)	4097 ¹⁾
83	87	91	95	FFT amplitude spectrum envelope signal acceleration (filter value: "MinFrequencyEnvelope" on page 58)	4097 ¹⁾

1) The first value in the buffer is the scaling factor.

In special applications it may be necessary to upload larger buffers.

In register "SensorConfig" on page 29, a buffer length of 8193 or 65535 values can be configured using bit 14. This makes it possible to read the raw signal and envelope signal (buffer number 9, 11, 13, 15, 25, 27, 29 and 31) using 65535 values, including the scaling factor, from the module. However, the FFT buffers still contain 4097 values including scaling factor and refer to the last 8193 values of the 65535 values of the raw or envelope signal.

After locking the buffer for the upload (RequestBufferLock01) until the next locking, it is necessary to wait until the longest buffer is filled again. If the buffer tries to lock again before this time elapses, it is prevented by the module until after the buffer is filled.

4.14.3 The Forward function on the X20CM4810

The ForwardDelay for the buffer Flatstream can be configured acyclically in [Function model 0 - Standard](#) and [Function model 1 - Fast master](#) in the I/O configuration of the module (BufferForwardDelay01). When using the "Fast master" function model, the ForwardDelay can also be configured acyclically for the parameter Flatstream in the I/O configuration (ParameterForwardDelay01).

Forwarding for the parameter and buffer Flatstreams can be adjusted acyclically (see registers "[ParameterForward01](#)" on page 94 and "[BufferForward01](#)" on page 94). However, it should only be adjusted after the forward delay for the respective Flatstream has already been set.

When using an SG4 CPU, the ForwardDelay cannot be configured manually for the parameter Flatstream. Starting with J3.09 and J4.01 of Automation Runtime, it is automatically configured and in earlier versions, it is ForwardDelay 0.

4.14.4 Flatstream communication

4.14.4.1 Introduction

B&R offers an additional communication method for some modules. "Flatstream" was designed for X2X and POWERLINK networks and allows data transmission to be adapted to individual demands. Although this method is not 100% real-time capable, it still allows data transfer to be handled more efficiently than with standard cyclic polling.

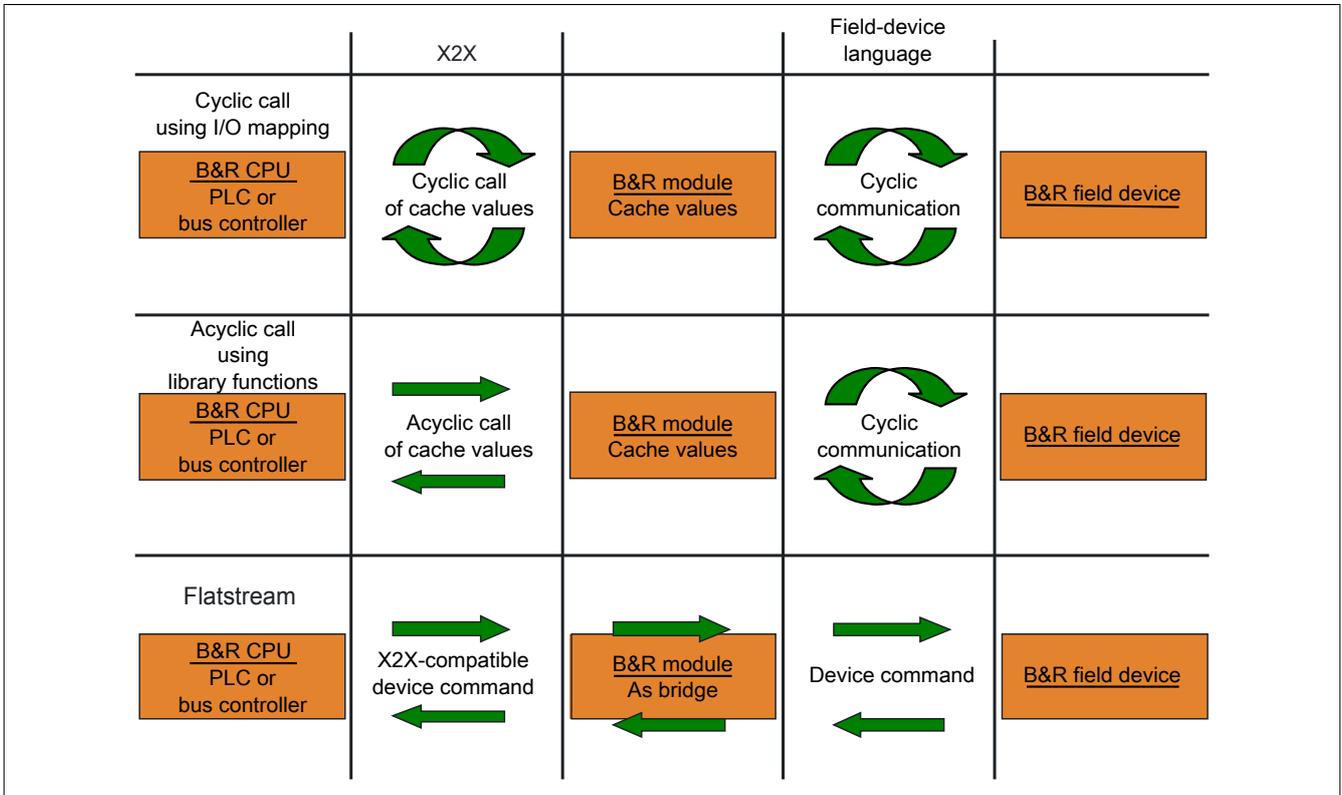


Figure 7: 3 types of communication

Flatstream extends cyclic and acyclic data queries. With Flatstream communication, the module acts as a bridge. The module is used to pass CPU queries directly on to the field device.

4.14.4.2 Message, segment, sequence, MTU

The physical properties of the bus system limit the amount of data that can be transmitted during one bus cycle. With Flatstream communication, all messages are viewed as part of a continuous data stream. Long data streams must be broken down into several fragments that are sent one after the other. To understand how the receiver puts these fragments back together to get the original information, it is important to understand the difference between a message, a segment, a sequence and an MTU.

Message

A message refers to information exchanged between 2 communicating partner stations. The length of a message is not restricted by the Flatstream communication method. Nevertheless, module-specific limitations must be considered.

Segment (logical division of a message):

A segment has a finite size and can be understood as a section of a message. The number of segments per message is arbitrary. So that the recipient can correctly reassemble the transferred segments, each segment is preceded by a byte with additional information. This control byte contains information such as the length of a segment and whether the approaching segment completes the message. This makes it possible for the receiving station to interpret the incoming data stream correctly.

Sequence (how a segment must be arranged physically):

The maximum size of a sequence corresponds to the number of enabled Rx or Tx bytes (later: "MTU"). The transmitting station splits the transmit array into valid sequences. These sequences are then written successively to the MTU and transferred to the receiving station where they are put back together again. The receiver stores the incoming sequences in a receive array, obtaining an image of the data stream in the process.

With Flatstream communication, the number of sequences sent are counted. Successfully transferred sequences must be acknowledged by the receiving station to ensure the integrity of the transfer.

MTU (Maximum Transmission Unit) - Physical transport:

MTU refers to the enabled USINT registers used with Flatstream. These registers can accept at least one sequence and transfer it to the receiving station. A separate MTU is defined for each direction of communication. OutputMTU defines the number of Flatstream Tx bytes, and InputMTU specifies the number of Flatstream Rx bytes. The MTUs are transported cyclically via the X2X Link network, increasing the load with each additional enabled USINT register.

Properties

Flatstream messages are not transferred cyclically or in 100% real time. Many bus cycles may be needed to transfer a particular message. Although the Rx and Tx registers are exchanged between the transmitter and the receiver cyclically, they are only processed further if explicitly accepted by register "InputSequence" or "OutputSequence".

Behavior in the event of an error (brief summary)

The protocol for X2X and POWERLINK networks specifies that the last valid values should be retained when disturbances occur. With conventional communication (cyclic/acyclic data queries), this type of error can generally be ignored.

In order for communication to also take place without errors using Flatstream, all of the sequences issued by the receiver must be acknowledged. If Forward functionality is not used, then subsequent communication is delayed for the length of the disturbance.

If Forward functionality is being used, the receiving station receives a transmission counter that is incremented twice. The receiver stops, i.e. it no longer returns any acknowledgments. The transmitting station uses SequenceAck to determine that the transmission was faulty and that all affected sequences must be repeated.

4.14.4.3 The Flatstream principle

Requirement

Before Flatstream can be used, the respective communication direction must be synchronized, i.e. both communication partners cyclically query the sequence counter on the opposite station. This checks to see if there is new data that should be accepted.

Communication

If a communication partner wants to transmit a message to its opposite station, it should first create a transmit array that corresponds to Flatstream conventions. This allows the Flatstream data to be organized very efficiently without having to block other important resources.

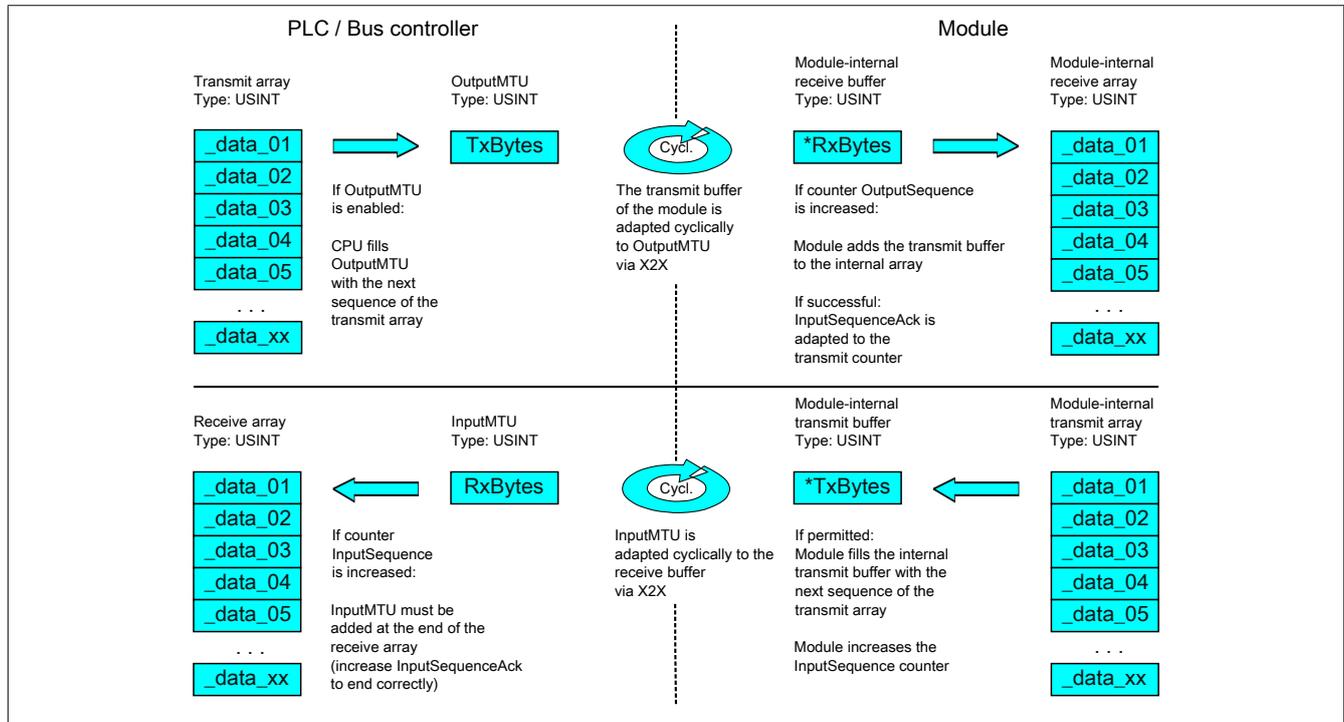


Figure 8: Flatstream communication

Procedure

The first thing that happens is that the message is broken into valid segments of up to 63 bytes, and the corresponding control bytes are created. The data is formed into a data stream made up of one control bytes per associated segment. This data stream can be written to the transmit array. The maximum size of each array element matches that of the enabled MTU so that one element corresponds to one sequence.

If the array has been completely created, the transmitter checks whether the MTU is permitted to be refilled. It then copies the first element of the array or the first sequence to the Tx byte registers. The MTU is transported to the receiver station via X2X Link and stored in the corresponding Rx byte registers. To signal that the data should be accepted by the receiver, the transmitter increases its SequenceCounter.

If the communication direction is synchronized, the opposite station detects the incremented SequenceCounter. The current sequence is appended to the receive array and acknowledged by SequenceAck. This acknowledgment signals to the transmitter that the MTU can now be refilled.

If the transfer is successful, the data in the receive array will correspond 100% to the data in the transmit array. During the transfer, the receiving station must detect and evaluate the incoming control bytes. A separate receive array should be created for each message. This allows the receiver to immediately begin further processing of messages once they have been completely transferred.

4.14.4.4 Registers for Flatstream mode

5 registers are available for configuring Flatstream. The default configuration can be used to transmit small amounts of data relatively easily.

Information:

The CPU communicates directly with the field device via registers "OutputSequence" and "InputSequence" as well as the enabled Tx and Rx bytes. For this reason, the user needs to have sufficient knowledge of the communication protocol being used on the field device.

4.14.4.4.1 Flatstream configuration

To use Flatstream, the program sequence must first be expanded. The cycle time of the Flatstream routines must be set to a multiple of the bus cycle. Other program routines should be implemented in Cyclic #1 to ensure data consistency.

At the absolute minimum, registers "InputMTU" and "OutputMTU" must be set. All other registers are filled in with default values at the beginning and can be used immediately. These registers are used for additional options, e.g. to transfer data in a more compact way or to increase the efficiency of the general procedure.

The Forward registers extend the functionality of the Flatstream protocol. This functionality is useful for substantially increasing the Flatstream data rate, but it also requires quite a bit of extra work when creating the program sequence.

4.14.4.4.1.1 Number of enabled Tx and Rx bytes

Name:

OutputMTU

InputMTU

These registers define the number of enabled Tx or Rx bytes and thus also the maximum size of a sequence. The user must consider that the more bytes made available also means a higher load on the bus system.

Information:

In the rest of this description, the names "OutputMTU" and "InputMTU" do not refer to the registers explained here. Instead, they are used as synonyms for the currently enabled Tx or Rx bytes.

Data type	Values
USINT	See the module-specific register overview (theoretically: 3 to 27).

4.14.4.4.2 Flatstream operation

When using Flatstream, the communication direction is very important. For transmitting data to a module (output direction), Tx bytes are used. For receiving data from a module (input direction), Rx bytes are used.

Registers "OutputSequence" and "InputSequence" are used to control and ensure that communication is taking place properly, i.e. the transmitter issues the directive that the data should be accepted and the receiver acknowledges that a sequence has been transferred successfully.

4.14.4.4.2.1 Format of input and output bytes

Name:

"Format of Flatstream" in Automation Studio

On some modules, this function can be used to set how the Flatstream input and output bytes (Tx or Rx bytes) are transferred.

- **Packed:** Data is transferred as an array.
- **Byte-by-byte:** Data is transferred as individual bytes.

4.14.4.4.2.2 Transport of payload data and control bytes

Name:

TxByte1 to TxByteN

RxByte1 to RxByteN

(The value the number N is different depending on the bus controller model used.)

The Tx and Rx bytes are cyclic registers used to transport the payload data and the necessary control bytes. The number of active Tx and Rx bytes is taken from the configuration of registers "OutputMTU" and "InputMTU", respectively.

In the user program, only the Tx and Rx bytes from the CPU can be used. The corresponding counterparts are located in the module and are not accessible to the user. For this reason, the names were chosen from the point of view of the CPU.

- "T" - "Transmit" →CPU *transmits* data to the module.
- "R" - "Receive" →CPU *receives* data from the module.

Data type	Values
USINT	0 to 255

4.14.4.4.2.3 Control bytes

In addition to the payload data, the Tx and Rx bytes also transfer the necessary control bytes. These control bytes contain additional information about the data stream so that the receiver can reconstruct the original message from the transferred segments.

Bit structure of a control byte

Bit	Description	Value	Information
0 - 5	SegmentLength	0 - 63	Size of the subsequent segment in bytes (default: Max. MTU size - 1)
6	nextCBPos	0	Next control byte at the beginning of the next MTU
		1	Next control byte directly after the end of the current segment
7	MessageEndBit	0	Message continues after the subsequent segment
		1	Message ended by the subsequent segment

SegmentLength

The segment length lets the receiver know the length of the coming segment. If the set segment length is insufficient for a message, then the information must be distributed over several segments. In these cases, the actual end of the message is detected using bit 7 of the control byte.

Information:

The control byte is not included in the calculation to determine the segment length. The segment length is only derived from the bytes of payload data.

nextCBPos

This bit indicates the position where the next control byte is to be expected. This information is especially important when using option "MultiSegmentMTU".

When using Flatstream communication with multi-segment MTUs, the next control byte is no longer expected in the first Rx byte of the subsequent MTU, but transferred directly after the current segment.

MessageEndBit

"MessageEndBit" is set if the subsequent segment completes a message. The message has then been completely transferred and is ready for further processing.

Information:

In the output direction, this bit must also be set if one individual segment is enough to hold the entire message. The module will only process a message internally if this identifier is detected. The size of the message being transferred can be calculated by adding all of the message's segment lengths together.

Flatstream formula for calculating message length:

Message [bytes] = Segment lengths (all CBs without ME) + Segment length (of the first CB with ME)	CB	Control byte
	ME	MessageEndBit

4.14.4.4.2.4 Communication status of the CPU

Name:

OutputSequence

Register "OutputSequence" contains information about the communication status of the CPU. It is written by the CPU and read by the module.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 2	OutputSequenceCounter	0 - 7	Counter for the sequences issued in the output direction
3	OutputSyncBit	0	Output direction disabled
		1	Output direction enabled
4 - 6	InputSequenceAck	0 - 7	Mirrors InputSequenceCounter
7	InputSyncAck	0	Input direction not ready (disabled)
		1	Input direction ready (enabled)

OutputSequenceCounter

The OutputSequenceCounter is a continuous counter of sequences that have been issued by the CPU. The CPU uses OutputSequenceCounter to direct the module to accept a sequence (the output direction must be synchronized when this happens).

OutputSyncBit

The CPU uses OutputSyncBit to attempt to synchronize the output channel.

InputSequenceAck

InputSequenceAck is used for acknowledgment. The value of InputSequenceCounter is mirrored if the CPU has received a sequence successfully.

InputSyncAck

The InputSyncAck bit acknowledges the synchronization of the input channel for the module. This indicates that the CPU is ready to receive data.

4.14.4.4.2.5 Communication status of the module

Name:

InputSequence

Register "InputSequence" contains information about the communication status of the module. It is written by the module and should only be read by the CPU.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 2	InputSequenceCounter	0 - 7	Counter for sequences issued in the input direction
3	InputSyncBit	0	Not ready (disabled)
		1	Ready (enabled)
4 - 6	OutputSequenceAck	0 - 7	Mirrors OutputSequenceCounter
7	OutputSyncAck	0	Not ready (disabled)
		1	Ready (enabled)

InputSequenceCounter

The InputSequenceCounter is a continuous counter of sequences that have been issued by the module. The module uses InputSequenceCounter to direct the CPU to accept a sequence (the input direction must be synchronized when this happens).

InputSyncBit

The module uses InputSyncBit to attempt to synchronize the input channel.

OutputSequenceAck

OutputSequenceAck is used for acknowledgment. The value of OutputSequenceCounter is mirrored if the module has received a sequence successfully.

OutputSyncAck

The OutputSyncAck bit acknowledges the synchronization of the output channel for the CPU. This indicates that the module is ready to receive data.

4.14.4.4.2.6 Relationship between OutputSequence and InputSequence

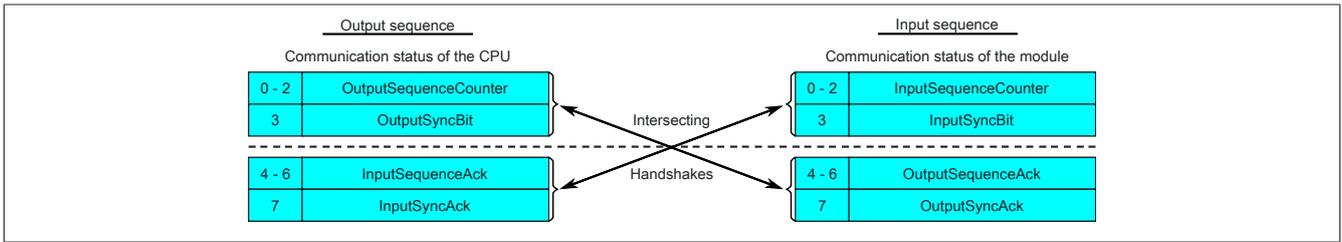


Figure 9: Relationship between OutputSequence and InputSequence

Registers "OutputSequence" and "InputSequence" are logically composed of 2 half-bytes. The low part signals to the opposite station whether a channel should be opened or if data should be accepted. The high part is to acknowledge that the requested action was carried out.

SyncBit and SyncAck

If SyncBit and SyncAck are set in one communication direction, then the channel is considered "synchronized", i.e. it is possible to send messages in this direction. The status bit of the opposite station must be checked cyclically. If SyncAck has been reset, then SyncBit on that station must be adjusted. Before new data can be transferred, the channel must be resynchronized.

SequenceCounter and SequenceAck

The communication partners cyclically check whether the low nibble on the opposite station changes. When one of the communication partners finishes writing a new sequence to the MTU, it increments its SequenceCounter. The current sequence is then transmitted to the receiver, which acknowledges its receipt with SequenceAck. In this way, a "handshake" is initiated.

Information:

If communication is interrupted, segments from the unfinished message are discarded. All messages that were transferred completely are processed.

4.14.4.4.3 Synchronization

During synchronization, a communication channel is opened. It is important to make sure that a module is present and that the current value of SequenceCounter is stored on the station receiving the message.

Flatstream can handle full-duplex communication. This means that both channels / communication directions can be handled separately. They must be synchronized independently so that simplex communication can theoretically be carried out as well.

Synchronization in the output direction (CPU as the transmitter):

The corresponding synchronization bits (OutputSyncBit and OutputSyncAck) are reset. Because of this, Flatstream cannot be used at this point in time to transfer messages from the CPU to the module.

Algorithm

1) The CPU must write 000 to OutputSequenceCounter and reset OutputSyncBit. The CPU must cyclically query the high nibble of register "InputSequence" (checks for 000 in OutputSequenceAck and 0 in OutputSyncAck).
<i>The module does not accept the current contents of InputMTU since the channel is not yet synchronized.</i> <i>The module matches OutputSequenceAck and OutputSyncAck to the values of OutputSequenceCounter and OutputSyncBit.</i>
2) If the CPU registers the expected values in OutputSequenceAck and OutputSyncAck, it is permitted to increment OutputSequenceCounter. The CPU continues cyclically querying the high nibble of register "OutputSequence" (checks for 001 in OutputSequenceAck and 0 in InputSyncAck).
<i>The module does not accept the current contents of InputMTU since the channel is not yet synchronized.</i> <i>The module matches OutputSequenceAck and OutputSyncAck to the values of OutputSequenceCounter and OutputSyncBit.</i>
3) If the CPU registers the expected values in OutputSequenceAck and OutputSyncAck, it is permitted to increment OutputSequenceCounter. The CPU continues cyclically querying the high nibble of register "OutputSequence" (checks for 001 in OutputSequenceAck and 1 in InputSyncAck).
Note: Theoretically, data can be transferred from this point forward. However, it is still recommended to wait until the output direction is completely synchronized before transferring data.
<i>The module sets OutputSyncAck.</i>
The output direction is synchronized, and the CPU can transmit data to the module.

Synchronization in the input direction (CPU as the receiver):

The corresponding synchronization bits (InputSyncBit and InputSyncAck) are reset. Because of this, Flatstream cannot be used at this point in time to transfer messages from the module to the CPU.

Algorithm

<i>The module writes 000 to InputSequenceCounter and resets InputSyncBit.</i> <i>The module monitors the high nibble of register "OutputSequence" and expects 000 in InputSequenceAck and 0 in InputSyncAck.</i>
1) The CPU is not permitted to accept the current contents of InputMTU since the channel is not yet synchronized. The CPU has to match InputSequenceAck and InputSyncAck to the values of InputSequenceCounter and InputSyncBit.
<i>If the module registers the expected values in InputSequenceAck and InputSyncAck, it increments InputSequenceCounter.</i> <i>The module monitors the high nibble of register "OutputSequence" and expects 001 in InputSequenceAck and 0 in InputSyncAck.</i>
2) The CPU is not permitted to accept the current contents of InputMTU since the channel is not yet synchronized. The CPU has to match InputSequenceAck and InputSyncAck to the values of InputSequenceCounter and InputSyncBit.
<i>If the module registers the expected values in InputSequenceAck and InputSyncAck, it sets InputSyncBit.</i> <i>The module monitors the high nibble of register "OutputSequence" and expects 1 in InputSyncAck.</i>
3) The CPU is permitted to set InputSyncAck.
Note: Theoretically, data could already be transferred in this cycle. If InputSyncBit is set and InputSequenceCounter has been increased by 1, the values in the enabled Rx bytes must be accepted and acknowledged (see also "Communication in the input direction").
The input direction is synchronized, and the module can transmit data to the CPU.

4.14.4.4.4 Transmitting and receiving

If a channel is synchronized, then the opposite station is ready to receive messages from the transmitter. Before the transmitter can send data, it needs to first create a transmit array in order to meet Flatstream requirements.

The transmitting station must also generate a control byte for each segment created. This control byte contains information about how the subsequent part of the data being transferred should be processed. The position of the next control byte in the data stream can vary. For this reason, it must be clearly defined at all times when a new control byte is being transmitted. The first control byte is always in the first byte of the first sequence. All subsequent positions are determined recursively.

Flatstream formula for calculating the position of the next control byte:

$$\text{Position (of the next control byte)} = \text{Current position} + 1 + \text{Segment length}$$

Example

3 autonomous messages (7 bytes, 2 bytes and 9 bytes) are being transmitted using an MTU with a width of 7 bytes. The rest of the configuration corresponds to the default settings.

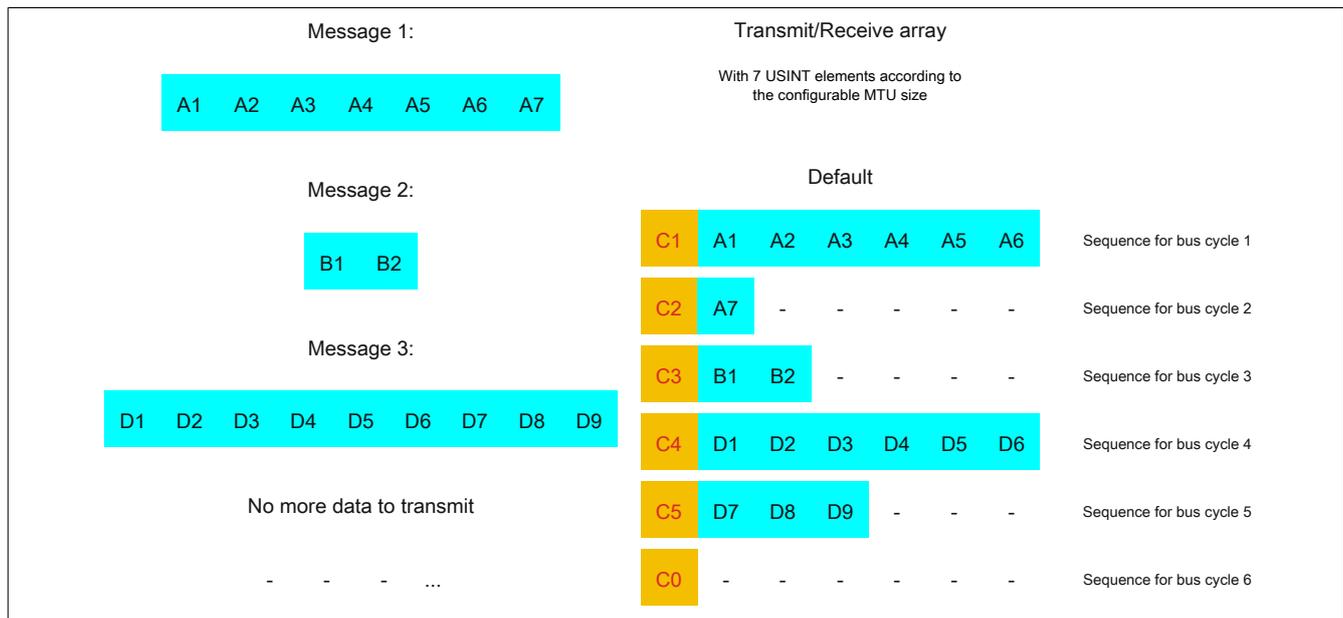


Figure 10: Transmit/Receive array (default)

First, the messages must be split into segments. In the default configuration, it is important to ensure that each sequence can hold an entire segment, including the associated control byte. The sequence is limited to the size of the enable MTU. In other words, a segment must be at least 1 byte smaller than the MTU.

MTU = 7 bytes → Max. segment length = 6 bytes

- Message 1 (7 bytes)
 - ⇒ First segment = Control byte + 6 bytes of data
 - ⇒ Second segment = Control byte + 1 data byte
- Message 2 (2 bytes)
 - ⇒ First segment = Control byte + 2 bytes of data
- Message 3 (9 bytes)
 - ⇒ First segment = Control byte + 6 bytes of data
 - ⇒ Second segment = Control byte + 3 data bytes
- No more messages
 - ⇒ C0 control byte

A unique control byte must be generated for each segment. In addition, the C0 control byte is generated to keep communication on standby.

C0 (control byte 0)		C1 (control byte 1)		C2 (control byte 2)	
- SegmentLength (0)	= 0	- SegmentLength (6)	= 6	- SegmentLength (1)	= 1
- nextCBPos (0)	= 0	- nextCBPos (0)	= 0	- nextCBPos (0)	= 0
- MessageEndBit (0)	= 0	- MessageEndBit (0)	= 0	- MessageEndBit (1)	= 128
Control byte	Σ 0	Control byte	Σ 6	Control byte	Σ 129

Table 3: Flatstream determination of the control bytes for the default configuration example (part 1)

C3 (control byte 3)		C4 (control byte 4)		C5 (control byte 5)	
- SegmentLength (2)	= 2	- SegmentLength (6)	= 6	- SegmentLength (3)	= 3
- nextCBPos (0)	= 0	- nextCBPos (0)	= 0	- nextCBPos (0)	= 0
- MessageEndBit (1)	= 128	- MessageEndBit (0)	= 0	- MessageEndBit (1)	= 128
Control byte	Σ 130	Control byte	Σ 6	Control byte	Σ 131

Table 4: Flatstream determination of the control bytes for the default configuration example (part 2)

4.14.4.4.5 Transmitting data to a module (output)

When transmitting data, the transmit array must be generated in the application program. Sequences are then transferred one by one using Flatstream and received by the module.

Information:

Although all B&R modules with Flatstream communication always support the most compact transfers in the output direction, it is recommended to use the same design for the transfer arrays in both communication directions.

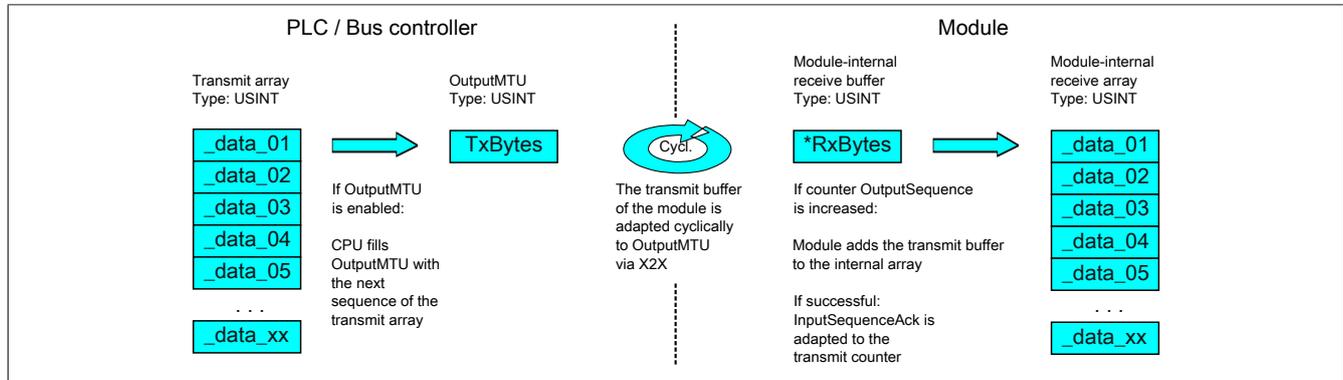


Figure 11: Flatstream communication (output)

Message smaller than OutputMTU

The length of the message is initially smaller than OutputMTU. In this case, one sequence would be sufficient to transfer the entire message and the necessary control byte.

Algorithm

<p><i>Cyclic status query:</i></p> <ul style="list-style-type: none"> - The module monitors <code>OutputSequenceCounter</code>.
<p>0) Cyclic checks:</p> <ul style="list-style-type: none"> - The CPU must check <code>OutputSyncAck</code>. → If <code>OutputSyncAck = 0</code>: Reset <code>OutputSyncBit</code> and resynchronize the channel. - The CPU must check whether <code>OutputMTU</code> is enabled. → If <code>OutputSequenceCounter > InputSequenceAck</code>: MTU is not enabled because the last sequence has not yet been acknowledged.
<p>1) Preparation (create transmit array):</p> <ul style="list-style-type: none"> - The CPU must split up the message into valid segments and create the necessary control bytes. - The CPU must add the segments and control bytes to the transmit array.
<p>2) Transmit:</p> <ul style="list-style-type: none"> - The CPU transfers the current element of the transmit array to <code>OutputMTU</code>. → The <code>OutputMTU</code> is transferred cyclically to the module's transmit buffer but not processed further. - The CPU must increase <code>OutputSequenceCounter</code>.
<p><i>Reaction:</i></p> <ul style="list-style-type: none"> - The module accepts the bytes from the internal receive buffer and adds them to the internal receive array. - The module transmits acknowledgment and writes the value of <code>OutputSequenceCounter</code> to <code>OutputSequenceAck</code>.
<p>3) Completion:</p> <ul style="list-style-type: none"> - The CPU must monitor <code>OutputSequenceAck</code>. → A sequence is only considered to have been transferred successfully if it has been acknowledged via <code>OutputSequenceAck</code>. In order to detect potential transfer errors in the last sequence as well, it is important to make sure that the length of the <i>Completion</i> phase is run through long enough.
<p>Note:</p> <p>To monitor communication times exactly, the task cycles that have passed since the last increase of <code>OutputSequenceCounter</code> should be counted. In this way, the number of previous bus cycles necessary for the transfer can be measured. If the monitoring counter exceeds a predefined threshold, then the sequence can be considered lost.</p> <p>(The relationship of bus to task cycle can be influenced by the user so that the threshold value must be determined individually.)</p> <ul style="list-style-type: none"> - Subsequent sequences are only permitted to be transmitted in the next bus cycle after the completion check has been carried out successfully.

Message larger than OutputMTU

The transmit array, which must be created in the program sequence, consists of several elements. The user has to arrange the control and data bytes correctly and transfer the array elements one after the other. The transfer algorithm remains the same and is repeated starting at the point *Cyclic checks*.

General flow chart

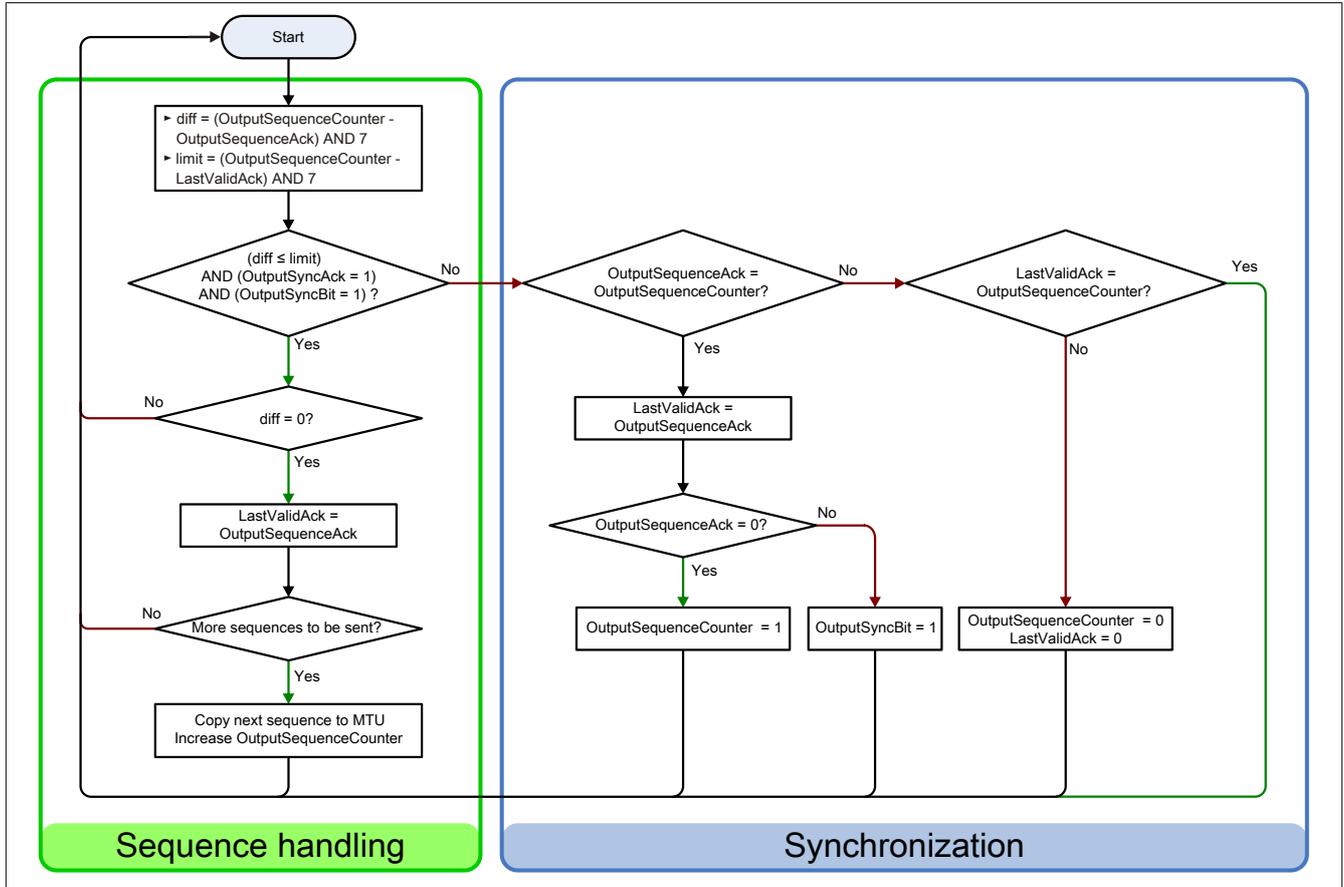


Figure 12: Flow chart for the output direction

4.14.4.4.6 Receiving data from a module (input)

When receiving data, the transmit array is generated by the module, transferred via Flatstream and must then be reproduced in the receive array. The structure of the incoming data stream can be set with the mode register. The algorithm for receiving the data does not change in this regard.

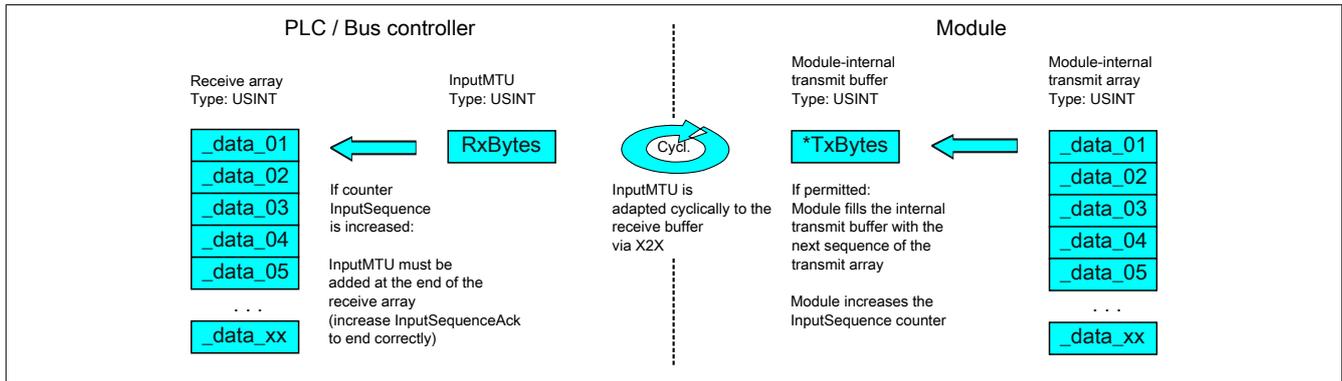


Figure 13: Flatstream communication (input)

Algorithm

<p>0) Cyclic status query:</p> <ul style="list-style-type: none"> - The CPU must monitor <code>InputSequenceCounter</code>.
<p>Cyclic checks:</p> <ul style="list-style-type: none"> - The module checks <code>InputSyncAck</code>. - The module checks <code>InputSequenceAck</code>.
<p>Preparation:</p> <ul style="list-style-type: none"> - The module forms the segments and control bytes and creates the transmit array.
<p>Action:</p> <ul style="list-style-type: none"> - The module transfers the current element of the internal transmit array to the internal transmit buffer. - The module increases <code>InputSequenceCounter</code>.
<p>1) Receiving (as soon as <code>InputSequenceCounter</code> is increased):</p> <ul style="list-style-type: none"> - The CPU must apply data from <code>InputMTU</code> and append it to the end of the receive array. - The CPU must match <code>InputSequenceAck</code> to <code>InputSequenceCounter</code> of the sequence currently being processed.
<p>Completion:</p> <ul style="list-style-type: none"> - The module monitors <code>InputSequenceAck</code>. → A sequence is only considered to have been transferred successfully if it has been acknowledged via <code>InputSequenceAck</code>. - Subsequent sequences are only transmitted in the next bus cycle after the completion check has been carried out successfully.

General flow chart

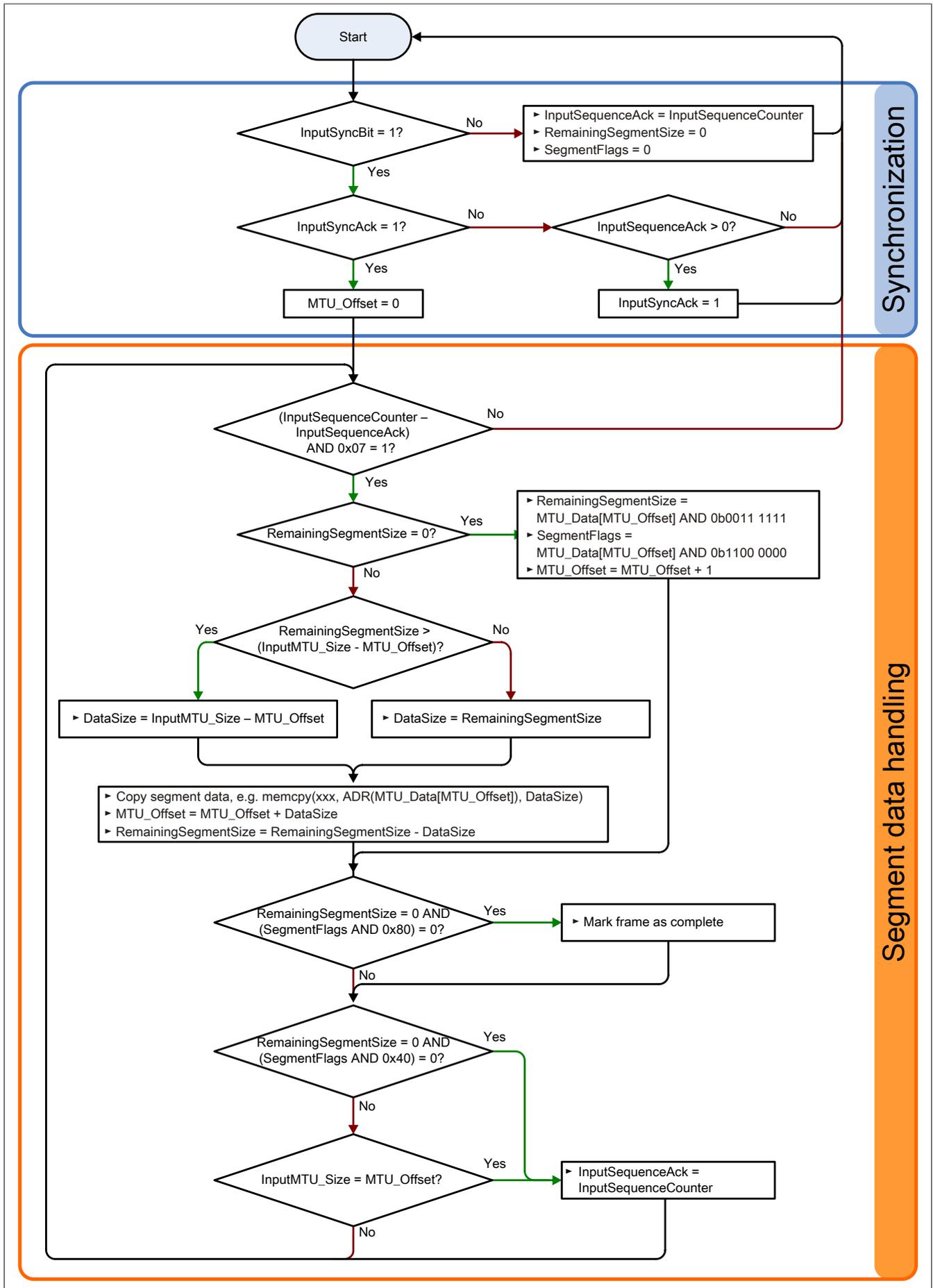


Figure 14: Flow chart for the input direction

4.14.4.4.7 Details

It is recommended to store transferred messages in separate receive arrays.

After a set MessageEndBit is transmitted, the subsequent segment should be added to the receive array. The message is then complete and can be passed on internally for further processing. A new/separate array should be created for the next message.

Information:

When transferring with MultiSegmentMTUs, it is possible for several small messages to be part of one sequence. In the program, it is important to make sure that a sufficient number of receive arrays can be managed. The acknowledge register is only permitted to be adjusted after the entire sequence has been applied.

If SequenceCounter is incremented by more than one counter, an error is present.

Note: This situation is very unlikely when operating without "Forward" functionality.

In this case, the receiver stops. All additional incoming sequences are ignored until the transmission with the correct SequenceCounter is retried. This response prevents the transmitter from receiving any more acknowledgments for transmitted sequences. The transmitter can identify the last successfully transferred sequence from the opposite station's SequenceAck and continue the transfer from this point.

Acknowledgments must be checked for validity.

If the receiver has successfully accepted a sequence, it must be acknowledged. The receiver takes on the value of SequenceCounter sent along with the transmission and matches SequenceAck to it. The transmitter reads SequenceAck and registers the successful transmission. If the transmitter acknowledges a sequence that has not yet been dispatched, then the transfer must be interrupted and the channel resynchronized. The synchronization bits are reset and the current/incomplete message is discarded. It must be sent again once the channel has been resynchronized.

4.14.4.4.8 Flatstream mode

Name:

FlatstreamMode

In the input direction, the transmit array is generated automatically. This register offers 2 options to the user that allow an incoming data stream to have a more compact arrangement. Once enabled, the program code for evaluation must be adapted accordingly.

Information:

All B&R modules that offer Flatstream mode support options "Large segments" and "MultiSegmentMTUs" in the output direction. Compact transfer must be explicitly allowed only in the input direction.

Bit structure:

Bit	Description	Value	Information
0	MultiSegmentMTU	0	Not allowed (default)
		1	Permitted
1	Large segments	0	Not allowed (default)
		1	Permitted
2 - 7	Reserved		

Standard

By default, both options relating to compact transfer in the input direction are disabled.

1. The module only forms segments that are at least one byte smaller than the enabled MTU. Each sequence begins with a control byte so that the data stream is clearly structured and relatively easy to evaluate.
2. Since a Flatstream message is permitted to be any length, the last segment of the message frequently does not fill up all of the MTU's space. By default, the remaining bytes during this type of transfer cycle are not used.

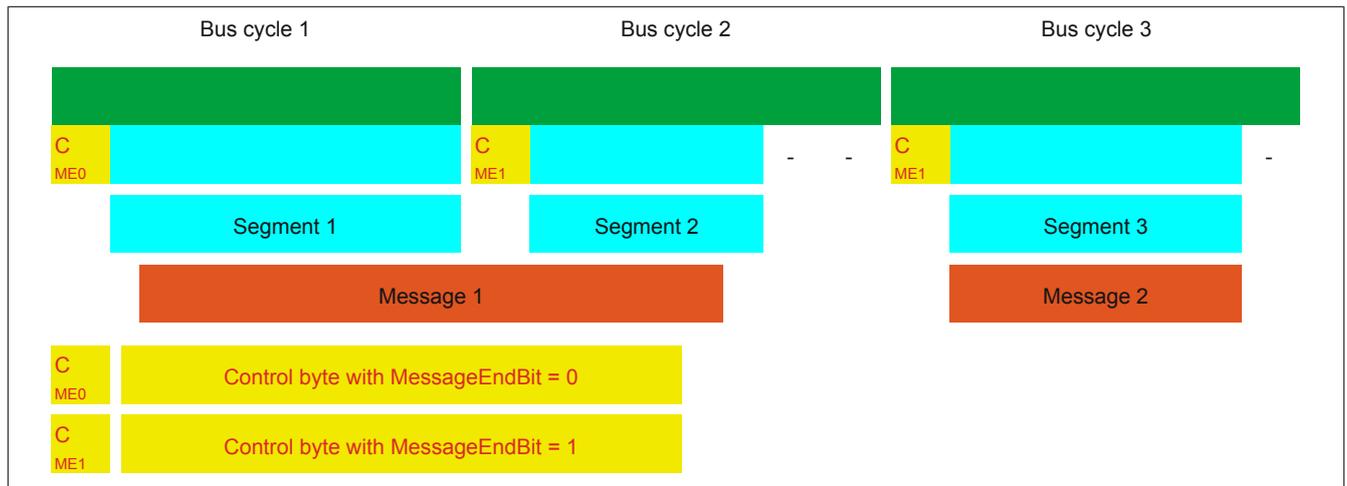


Figure 15: Message arrangement in the MTU (default)

MultiSegmentMTUs allowed

With this option, InputMTU is completely filled (if enough data is pending). The previously unfilled Rx bytes transfer the next control bytes and their segments. This allows the enabled Rx bytes to be used more efficiently.

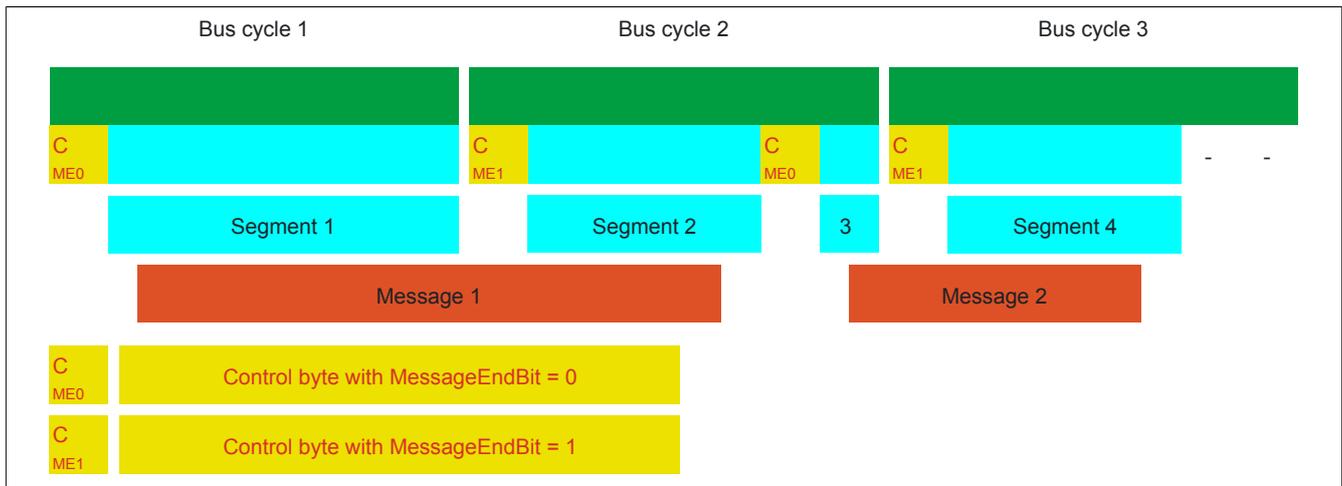


Figure 16: Arrangement of messages in the MTU (MultiSegmentMTUs)

Large segments allowed:

When transferring very long messages or when enabling only very few Rx bytes, then a great many segments must be created by default. The bus system is more stressed than necessary since an additional control byte must be created and transferred for each segment. With option "Large segments", the segment length is limited to 63 bytes independently of InputMTU. One segment is permitted to stretch across several sequences, i.e. it is possible for "pure" sequences to occur without a control byte.

Information:

It is still possible to split up a message into several segments, however. If this option is used and messages with more than 63 bytes occur, for example, then messages can still be split up among several segments.

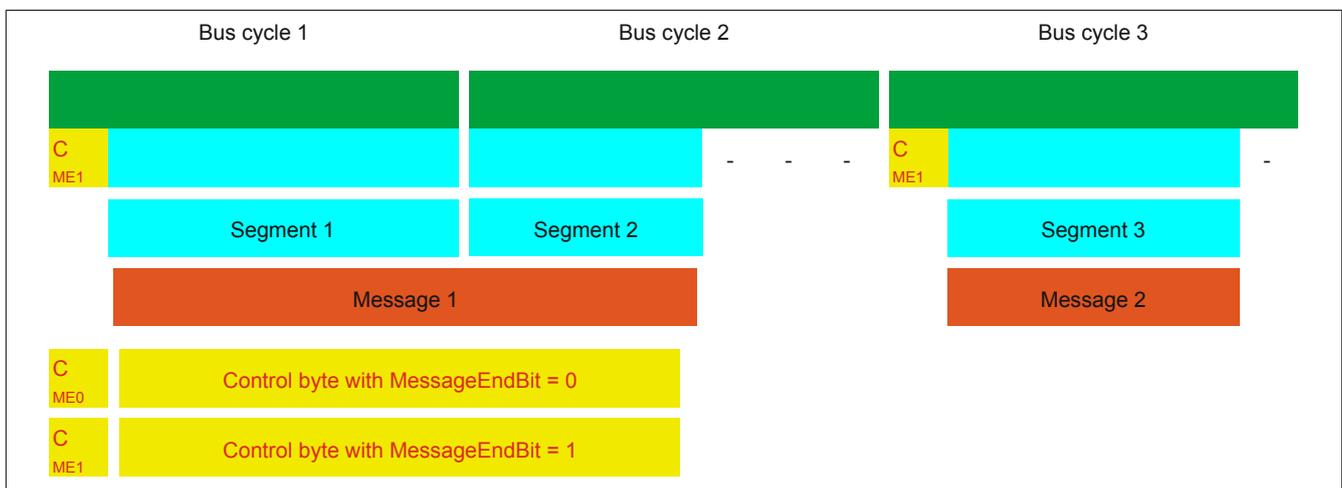


Figure 17: Arrangement of messages in the MTU (large segments)

Using both options

Using both options at the same time is also permitted.

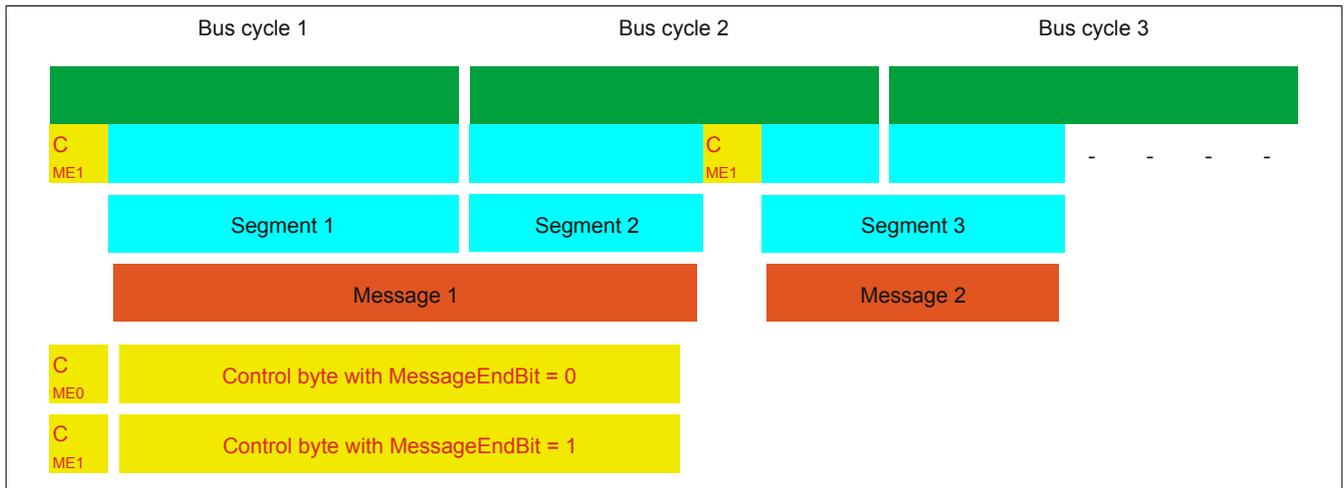


Figure 18: Arrangement of messages in the MTU (large segments and MultiSegmentMTUs)

4.14.4.9 Adjusting the Flatstream

If the way messages are structured is changed, then the way data in the transmit/receive array is arranged is also different. The following changes apply to the example given earlier.

MultiSegmentMTU

If MultiSegmentMTUs are allowed, then "open positions" in an MTU can be used. These "open positions" occur if the last segment in a message does not fully use the entire MTU. MultiSegmentMTUs allow these bits to be used to transfer the subsequent control bytes and segments. In the program sequence, the "nextCBPos" bit in the control byte is set so that the receiver can correctly identify the next control byte.

Example

3 autonomous messages (7 bytes, 2 bytes and 9 bytes) are being transmitted using an MTU with a width of 7 bytes. The configuration allows the transfer of MultiSegmentMTUs.

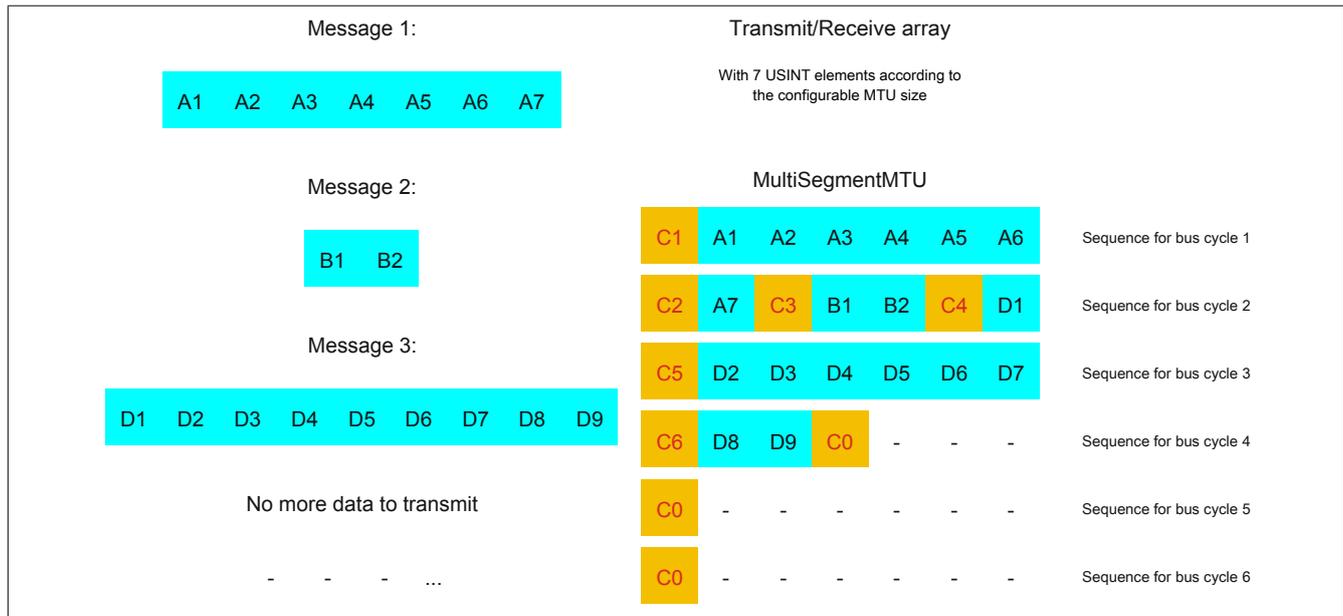


Figure 19: Transmit/receive array (MultiSegmentMTUs)

First, the messages must be split into segments. As in the default configuration, it is important for each sequence to begin with a control byte. The free bits in the MTU at the end of a message are filled with data from the following message, however. With this option, the "nextCBPos" bit is always set if payload data is transferred after the control byte.

MTU = 7 bytes → Max. segment length = 6 bytes

- Message 1 (7 bytes)
 - ⇒ First segment = Control byte + 6 bytes of data (MTU full)
 - ⇒ Second segment = Control byte + 1 byte of data (MTU still has 5 open bytes)
- Message 2 (2 bytes)
 - ⇒ First segment = Control byte + 2 bytes of data (MTU still has 2 open bytes)
- Message 3 (9 bytes)
 - ⇒ First segment = Control byte + 1 byte of data (MTU full)
 - ⇒ Second segment = Control byte + 6 bytes of data (MTU full)
 - ⇒ Third segment = Control byte + 2 bytes of data (MTU still has 4 open bytes)
- No more messages
 - ⇒ C0 control byte

A unique control byte must be generated for each segment. In addition, the C0 control byte is generated to keep communication on standby.

C1 (control byte 1)		C2 (control byte 2)		C3 (control byte 3)	
- SegmentLength (6)	= 6	- SegmentLength (1)	= 1	- SegmentLength (2)	= 2
- nextCBPos (1)	= 64	- nextCBPos (1)	= 64	- nextCBPos (1)	= 64
- MessageEndBit (0)	= 0	- MessageEndBit (1)	= 128	- MessageEndBit (1)	= 128
Control byte	Σ 70	Control byte	Σ 193	Control byte	Σ 194

Table 5: Flatstream determination of the control bytes for the MultiSegmentMTU example (part 1)

Warning!

The second sequence is only permitted to be acknowledged via SequenceAck if it has been completely processed. In this example, there are 3 different segments within the second sequence, i.e. the program must include enough receive arrays to handle this situation.

C4 (control byte 4)		C5 (control byte 5)		C6 (control byte 6)	
- SegmentLength (1)	= 1	- SegmentLength (6)	= 6	- SegmentLength (2)	= 2
- nextCBPos (6)	= 6	- nextCBPos (1)	= 64	- nextCBPos (1)	= 64
- MessageEndBit (0)	= 0	- MessageEndBit (1)	= 0	- MessageEndBit (1)	= 128
Control byte	Σ 7	Control byte	Σ 70	Control byte	Σ 194

Table 6: Flatstream determination of the control bytes for the MultiSegmentMTU example (part 2)

Large segments

Segments are limited to a maximum of 63 bytes. This means they can be larger than the active MTU. These large segments are divided among several sequences when transferred. It is possible for sequences to be completely filled with payload data and not have a control byte.

Information:

It is still possible to subdivide a message into several segments so that the size of a data packet does not also have to be limited to 63 bytes.

Example

3 autonomous messages (7 bytes, 2 bytes and 9 bytes) are being transmitted using an MTU with a width of 7 bytes. The configuration allows the transfer of large segments.

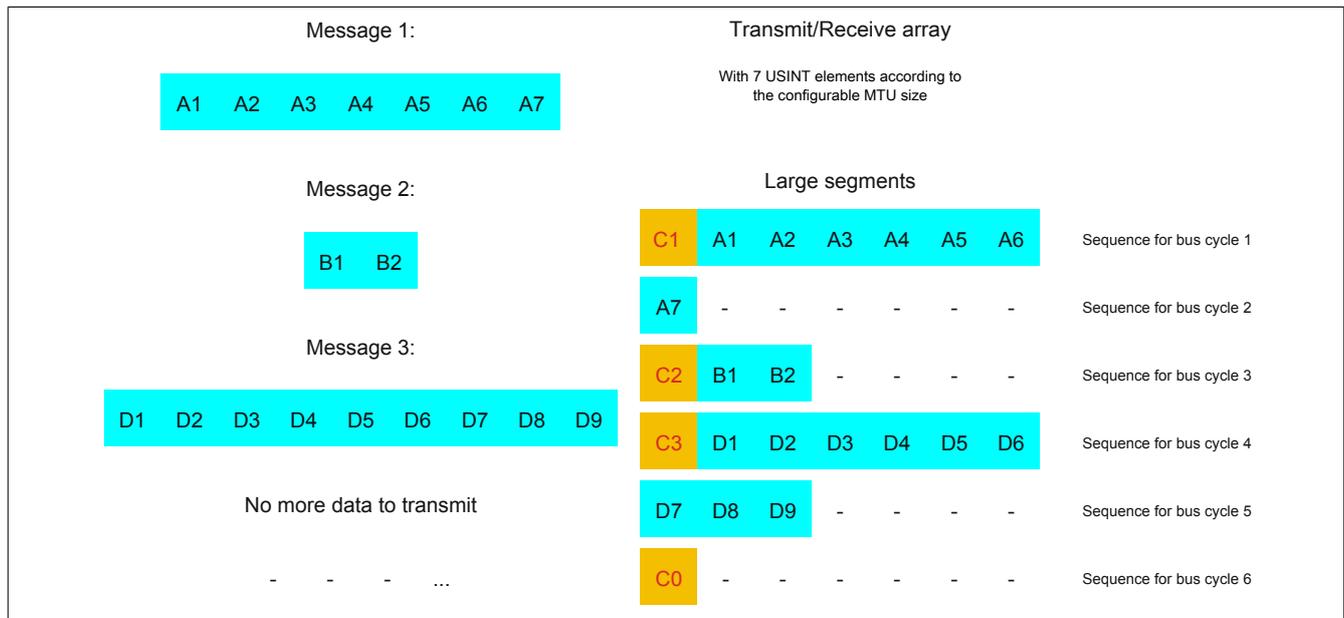


Figure 20: Transmit/receive array (large segments)

First, the messages must be split into segments. The ability to form large segments means that messages are split up less frequently, which results in fewer control bytes generated.

Large segments allowed → Max. segment length = 63 bytes

- Message 1 (7 bytes)
 - ⇒ First segment = Control byte + 7 bytes of data
- Message 2 (2 bytes)
 - ⇒ First segment = Control byte + 2 bytes of data
- Message 3 (9 bytes)
 - ⇒ First segment = Control byte + 9 bytes of data
- No more messages
 - ⇒ C0 control byte

A unique control byte must be generated for each segment. In addition, the C0 control byte is generated to keep communication on standby.

C1 (control byte 1)		C2 (control byte 2)		C3 (control byte 3)	
- SegmentLength (7)	= 7	- SegmentLength (2)	= 2	- SegmentLength (9)	= 9
- nextCBPos (0)	= 0	- nextCBPos (0)	= 0	- nextCBPos (0)	= 0
- MessageEndBit (1)	= 128	- MessageEndBit (1)	= 128	- MessageEndBit (1)	= 128
Control byte	Σ 135	Control byte	Σ 130	Control byte	Σ 137

Table 7: Flatstream determination of the control bytes for the large segment example

Large segments and MultiSegmentMTU

Example

3 autonomous messages (7 bytes, 2 bytes and 9 bytes) are being transmitted using an MTU with a width of 7 bytes. The configuration allows transfer of large segments as well as MultiSegmentMTUs.

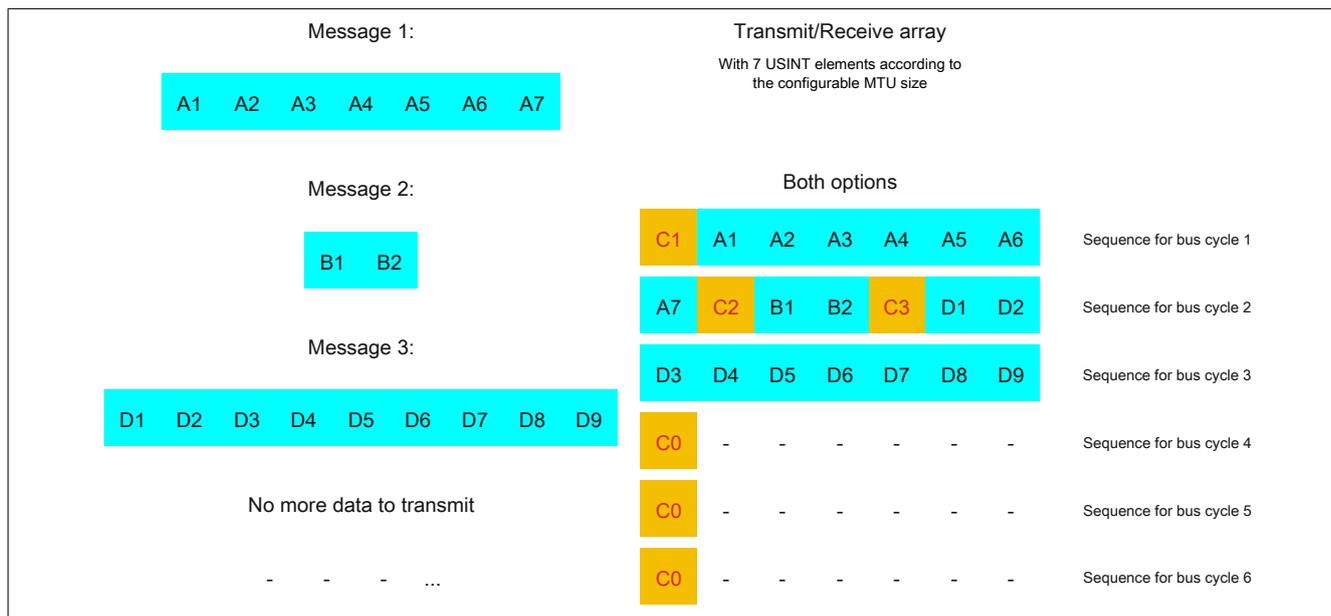


Figure 21: Transmit/receive array (large segments and MultiSegmentMTUs)

First, the messages must be split into segments. If the last segment of a message does not completely fill the MTU, it is permitted to be used for other data in the data stream. Bit "nextCBPos" must always be set if the control byte belongs to a segment with payload data.

The ability to form large segments means that messages are split up less frequently, which results in fewer control bytes generated. Control bytes are generated in the same way as with option "Large segments".

Large segments allowed → Max. segment length = 63 bytes

- Message 1 (7 bytes)
 - ⇒ First segment = Control byte + 7 bytes of data
- Message 2 (2 bytes)
 - ⇒ First segment = Control byte + 2 bytes of data
- Message 3 (9 bytes)
 - ⇒ First segment = Control byte + 9 bytes of data
- No more messages
 - ⇒ C0 control byte

A unique control byte must be generated for each segment. In addition, the C0 control byte is generated to keep communication on standby.

C1 (control byte 1)		C2 (control byte 2)		C3 (control byte 3)	
- SegmentLength (7)	= 7	- SegmentLength (2)	= 2	- SegmentLength (9)	= 9
- nextCBPos (0)	= 0	- nextCBPos (0)	= 0	- nextCBPos (0)	= 0
- MessageEndBit (1)	= 128	- MessageEndBit (1)	= 128	- MessageEndBit (1)	= 128
Control byte	Σ 135	Control byte	Σ 130	Control byte	Σ 137

Table 8: Flatstream determination of the control bytes for the large segment and MultiSegmentMTU example

4.14.4.5 Example of Forward functionality on X2X Link

Forward functionality is a method that can be used to substantially increase the Flatstream data rate. The basic principle is also used in other technical areas such as "pipelining" for microprocessors.

4.14.4.5.1 Function principle

X2X Link communication cycles through 5 different steps to transfer a Flatstream sequence. At least 5 bus cycles are therefore required to successfully transfer the sequence.

	Step I	Step II	Step III	Step IV	Step V
Actions	Transfer sequence from transmit array, increase SequenceCounter	Cyclic matching of MTU and module buffer	Append sequence to receive array Adjust SequenceAck	Cyclic matching of MTU and module buffer	Check SequenceAck
Resource	Sender (task to transmit)	Bus system (direction 1)	Recipient (task to receive)	Bus system (direction 2)	Sender (task for Ack checking)

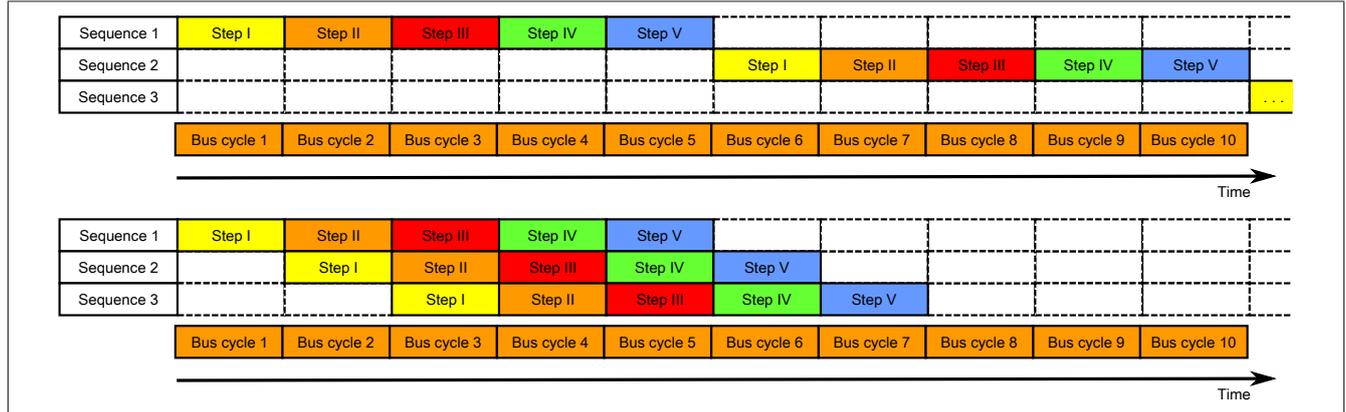


Figure 22: Comparison of transfer without/with Forward

Each of the 5 steps (tasks) requires different resources. If Forward functionality is not used, the sequences are executed one after the other. Each resource is then only active if it is needed for the current sub-action.

With Forward, a resource that has executed its task can already be used for the next message. The condition for enabling the MTU is changed to allow for this. Sequences are then passed to the MTU according to the timing. The transmitting station no longer waits for an acknowledgment from SequenceAck, which means that the available bandwidth can be used much more efficiently.

In the most ideal situation, all resources are working during each bus cycle. The receiver still has to acknowledge every sequence received. Only when SequenceAck has been changed and checked by the transmitter is the sequence considered as having been transferred successfully.

4.14.4.5.2 Configuration

The Forward function must only be enabled for the input direction. 2 additional configuration registers are available for doing so. Flatstream modules have been optimized in such a way that they support this function. In the output direction, the Forward function can be used as soon as the size of OutputMTU is specified.

4.14.4.5.2.1 Number of unacknowledged sequences

Name:
Forward

With register "Forward", the user specifies how many unacknowledged sequences the module is permitted to transmit.

Recommendation:

X2X Link: Max. 5

POWERLINK: Max. 7

Data type	Values
USINT	1 to 7 Default: 1

4.14.4.5.2.2 Delay time

Name:
ForwardDelay

Register "ForwardDelay" is used to specify the delay time in μs . This is the amount of time the module has to wait after sending a sequence until it is permitted to write new data to the MTU in the following bus cycle. The program routine for receiving sequences from a module can therefore be run in a task class whose cycle time is slower than the bus cycle.

Data type	Values
UINT	0 to 65535 [μs] Default: 0

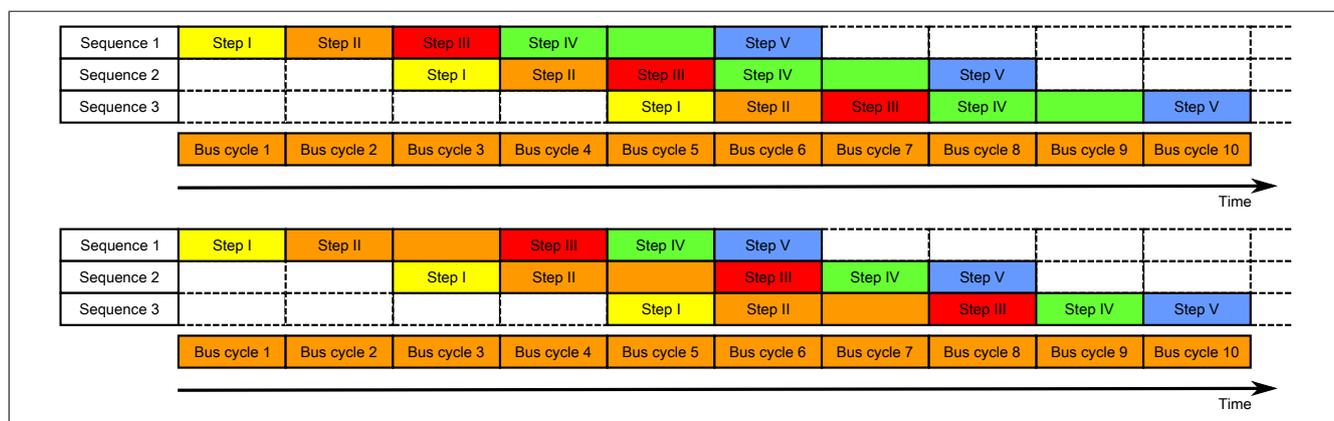


Figure 23: Effect of ForwardDelay when using Flatstream communication with the Forward function

In the program, it is important to make sure that the CPU is processing all of the incoming InputSequences and InputMTUs. The ForwardDelay value causes delayed acknowledgment in the output direction and delayed reception in the input direction. In this way, the CPU has more time to process the incoming InputSequence or InputMTU.

4.14.4.5.3 Transmitting and receiving with Forward

The basic algorithm for transmitting and receiving data remains the same. With the Forward function, up to 7 unacknowledged sequences can be transmitted. Sequences can be transmitted without having to wait for the previous message to be acknowledged. Since the delay between writing and response is eliminated, a considerable amount of additional data can be transferred in the same time window.

Algorithm for transmitting

<p><i>Cyclic status query:</i></p> <ul style="list-style-type: none"> - The module monitors <i>OutputSequenceCounter</i>.
<p>0) Cyclic checks:</p> <ul style="list-style-type: none"> - The CPU must check <i>OutputSyncAck</i>. → If <i>OutputSyncAck</i> = 0: Reset <i>OutputSyncBit</i> and resynchronize the channel. - The CPU must check whether <i>OutputMTU</i> is enabled. → If <i>OutputSequenceCounter</i> > <i>OutputSequenceAck</i> + 7, then it is not enabled because the last sequence has not yet been acknowledged.
<p>1) Preparation (create transmit array):</p> <ul style="list-style-type: none"> - The CPU must split up the message into valid segments and create the necessary control bytes. - The CPU must add the segments and control bytes to the transmit array.
<p>2) Transmit:</p> <ul style="list-style-type: none"> - The CPU must transfer the current part of the transmit array to <i>OutputMTU</i>. - The CPU must increase <i>OutputSequenceCounter</i> for the sequence to be accepted by the module. - The CPU is then permitted to <i>transmit</i> in the next bus cycle if the MTU has been enabled.
<p><i>The module responds since $OutputSequenceCounter > OutputSequenceAck$:</i></p> <ul style="list-style-type: none"> - The module accepts data from the internal receive buffer and appends it to the end of the internal receive array. - The module is acknowledged and the currently received value of <i>OutputSequenceCounter</i> is transferred to <i>OutputSequenceAck</i>. - The module queries the status cyclically again.
<p>3) Completion (acknowledgment):</p> <ul style="list-style-type: none"> - The CPU must check <i>OutputSequenceAck</i> cyclically. → A sequence is only considered to have been transferred successfully if it has been acknowledged via <i>OutputSequenceAck</i>. In order to detect potential transfer errors in the last sequence as well, it is important to make sure that the algorithm is run through long enough. <p>Note:</p> <p>To monitor communication times exactly, the task cycles that have passed since the last increase of <i>OutputSequenceCounter</i> should be counted. In this way, the number of previous bus cycles necessary for the transfer can be measured. If the monitoring counter exceeds a predefined threshold, then the sequence can be considered lost (the relationship of bus to task cycle can be influenced by the user so that the threshold value must be determined individually).</p>

Algorithm for receiving

<p>0) Cyclic status query:</p> <ul style="list-style-type: none"> - The CPU must monitor <i>InputSequenceCounter</i>.
<p><i>Cyclic checks:</i></p> <ul style="list-style-type: none"> - The module checks <i>InputSyncAck</i>. - The module checks if <i>InputMTU</i> for enabling. → Enabling criteria: $InputSequenceCounter > InputSequenceAck + Forward$
<p><i>Preparation:</i></p> <ul style="list-style-type: none"> - The module forms the control bytes / segments and creates the transmit array.
<p><i>Action:</i></p> <ul style="list-style-type: none"> - The module transfers the current part of the transmit array to the receive buffer. - The module increases <i>InputSequenceCounter</i>. - The module waits for a new bus cycle after time from <i>ForwardDelay</i> has expired. - The module repeats the action if <i>InputMTU</i> is enabled.
<p>1) Receiving ($InputSequenceCounter > InputSequenceAck$):</p> <ul style="list-style-type: none"> - The CPU must apply data from <i>InputMTU</i> and append it to the end of the receive array. - The CPU must match <i>InputSequenceAck</i> to <i>InputSequenceCounter</i> of the sequence currently being processed.
<p><i>Completion:</i></p> <ul style="list-style-type: none"> - The module monitors <i>InputSequenceAck</i>. → A sequence is only considered to have been transferred successfully if it has been acknowledged via <i>InputSequenceAck</i>.

Details/Background

1. Illegal SequenceCounter size (counter offset)

Error situation: MTU not enabled

If the difference between SequenceCounter and SequenceAck during transmission is larger than permitted, a transfer error occurs. In this case, all unacknowledged sequences must be repeated with the old SequenceCounter value.

2. Checking an acknowledgment

After an acknowledgment has been received, a check must verify whether the acknowledged sequence has been transmitted and had not yet been unacknowledged. If a sequence is acknowledged multiple times, a severe error occurs. The channel must be closed and resynchronized (same behavior as when not using Forward).

Information:

In exceptional cases, the module can increment OutputSequenceAck by more than 1 when using Forward.

An error does not occur in this case. The CPU is permitted to consider all sequences up to the one being acknowledged as having been transferred successfully.

3. Transmit and receive arrays

The Forward function has no effect on the structure of the transmit and receive arrays. They are created and must be evaluated in the same way.

4.14.4.5.4 Errors when using Forward

In industrial environments, it is often the case that many different devices from various manufacturers are being used side by side. The electrical and/or electromagnetic properties of these technical devices can sometimes cause them to interfere with one another. These kinds of situations can be reproduced and protected against in laboratory conditions only to a certain point.

Precautions have been taken for X2X Link transfers if this type of interference occurs. For example, if an invalid checksum occurs, the I/O system will ignore the data from this bus cycle and the receiver receives the last valid data once more. With conventional (cyclic) data points, this error can often be ignored. In the following cycle, the same data point is again retrieved, adjusted and transferred.

Using Forward functionality with Flatstream communication makes this situation more complex. The receiver receives the old data again in this situation as well, i.e. the previous values for SequenceAck/SequenceCounter and the old MTU.

Loss of acknowledgment (SequenceAck)

If a SequenceAck value is lost, then the MTU was already transferred properly. For this reason, the receiver is permitted to continue processing with the next sequence. The SequenceAck is aligned with the associated SequenceCounter and sent back to the transmitter. Checking the incoming acknowledgments shows that all sequences up to the last one acknowledged have been transferred successfully (see sequences 1 and 2 in the image).

Loss of transmission (SequenceCounter, MTU):

If a bus cycle drops out and causes the value of SequenceCounter and/or the filled MTU to be lost, then no data reaches the receiver. At this point, the transmission routine is not yet affected by the error. The time-controlled MTU is released again and can be rewritten to.

The receiver receives SequenceCounter values that have been incremented several times. For the receive array to be put together correctly, the receiver is only permitted to process transmissions whose SequenceCounter has been increased by one. The incoming sequences must be ignored, i.e. the receiver stops and no longer transmits back any acknowledgments.

If the maximum number of unacknowledged sequences has been sent and no acknowledgments are returned, the transmitter must repeat the affected SequenceCounter and associated MTUs (see sequence 3 and 4 in the image).

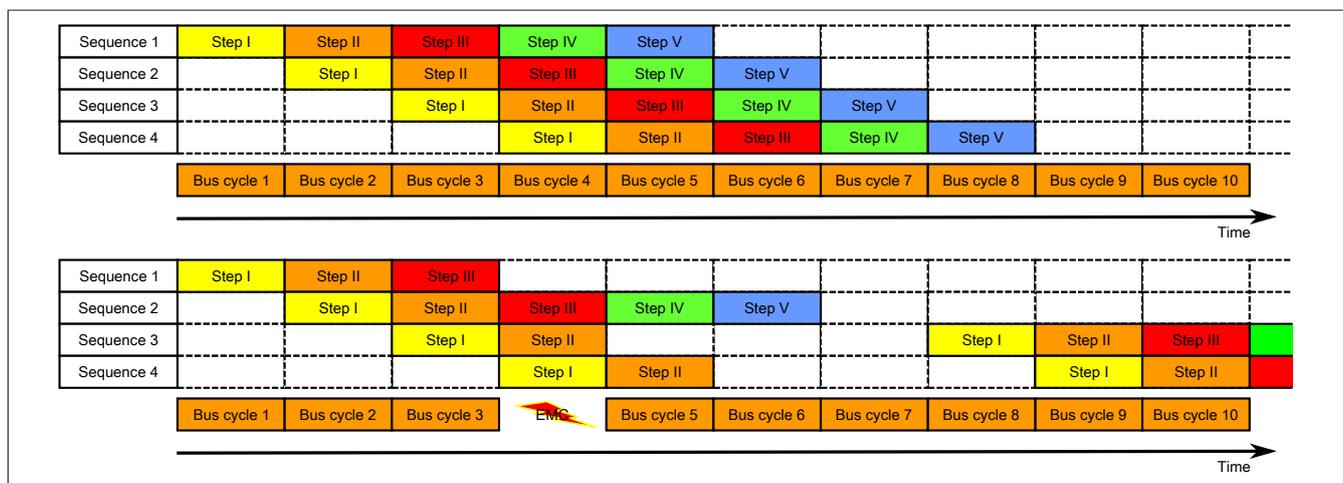


Figure 24: Effect of a lost bus cycle

Loss of acknowledgment

In sequence 1, the acknowledgment is lost due to disturbance. Sequences 1 and 2 are therefore acknowledged in Step V of sequence 2.

Loss of transmission

In sequence 3, the entire transmission is lost due to disturbance. The receiver stops and no longer sends back any acknowledgments.

The transmitting station continues transmitting until it has issued the maximum permissible number of unacknowledged transmissions.

5 bus cycles later at the earliest (depending on the configuration), it begins resending the unsuccessfully sent transmissions.

4.15 Using the module with on the fieldbus

4.15.1 Bus controller with FieldbusDESIGNER support

Only [Function model 1 - Fast master](#) and [Function model 2 - Slow master](#) are available for this.

The module is configured using FieldbusDESIGNER in Automation Studio. Implementation must be handled on the master.

4.15.2 Bus controller without FieldbusDESIGNER support

Only [Function model 2 - Slow master](#) is available for this. Configuration and implementation must be handled on the master.

4.15.3 B&R SG4 CPU with an interface module

This combination offers the following advantages:

- Modular condition monitoring solution possible
- All characteristic values prepared by Automation Runtime (no extra work required for implementation)
- Communication with the master via the interface card

4.15.4 CANIO bus controller

Only [Function model 254 - Bus controller](#) is available for this. Configuration and implementation must be handled on the master.

4.16 Assignment of the task class

Name:

Cycle time

By assigning the cycle time to a task class, the data exchange between the CPU and module can be prioritized and adapted to the requirements of the application. Faster task classes result in increased data throughput, but this increases the system load.

Information:

Using a task class that is too slow can result in new data from the module not being available within a measurement cycle.

The task class must be less than or equal to to the [maximum cycle time](#).

Information:

This configuration involves a driver setting that cannot be changed during runtime!

Data type	Values	Information
-	1 to 8	Task class used

4.17 Maximum cycle time

The maximum cycle time defines how far the cycle time of the X2X Link can be increased without causing a communication error or impaired functionality.

Maximum cycle time	
Function model 0 - Standard Function model 1 - Fast master	10 ms
Function model 2 - Slow master Function model 254 - Bus controller	-

4.18 Minimum cycle time

The minimum cycle time specifies the time up to which the bus cycle can be reduced without communication errors occurring. It is important to note that very fast cycles reduce the idle time available for handling monitoring, diagnostics and acyclic commands.

Minimum cycle time	
All function models	400 µs

5 Condition monitoring / Oscillation analyses

5.1 Basic information

System availability is one of the most important criteria in machine operation. For this reason, increasing and ensuring this availability over the long term is a primary objective of the system operator.

Unplanned stoppages and the resulting loss of production can lead to substantial costs. The implementation of condition monitoring has proven to be a very good method for supporting anticipatory maintenance.

5.1.1 What is condition monitoring?

Condition monitoring involves recording the status of the machines at regular intervals by measuring important values in order to recognize impending problems in the system. The goal is to recognize imminent damage early enough so that a faulty machine part can be repaired or replaced before it leads to consequential damage or a partial or complete breakdown of the system.

The specific purpose of condition monitoring is to collect and process sensor data (e.g. oscillations, temperatures, lubricant conditions, pressures, flows) that can be used to assess the overall condition of the system.

Deviations from normal process or system conditions are caused by defects, which can occur for a variety of reasons. If corresponding countermeasures are not taken, this can quickly lead to a malfunction and breakdown of the system. Monitoring the sources of defects through the analysis of machine parameters can make it possible to recognize malfunctions as early as possible in order to take preventive action. Possible responses can include, for example, an error message or warning to the operator or an automated action for fault clearance and prevention of damage all the way up to automatic shutdown.

The integration and systematic implementation of condition monitoring provides many advantages:

- System components are only repaired or replaced when actually necessary. Potential defects are recognized early on in operation.
- Reliability can be significantly increased by integrating condition monitoring into the process.

5.1.1.1 Bathtub / Deterioration curve

The operating performance of every mechanical component changes over the course of operation, with each component becoming defective at some point. It is crucial to recognize such a change before the component can no longer fulfill its function. This behavior can be illustrated very well by the so-called "bathtub curve". This indicates the probable failure rate over time.

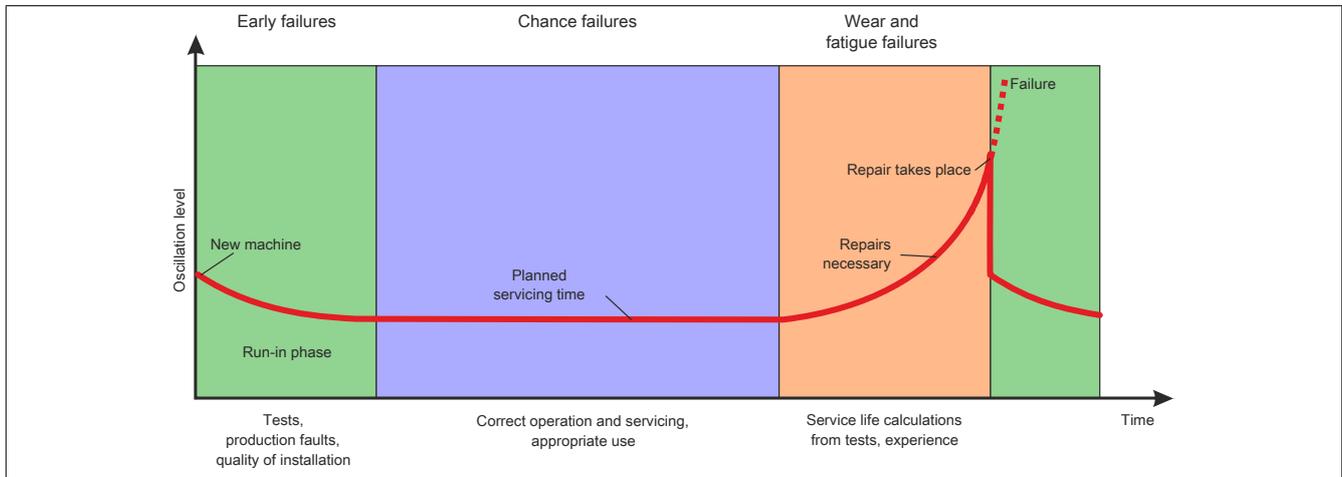


Figure 25: Bathtub curve illustrating the 3 typical phases

Each component is subject to the regularities of this curve, with a typical trend emerging as a result.

- **Area 1 (early failures)** is characterized by a decreasing failure rate. Early failures are caused almost exclusively by errors in assembly or construction. However, special care and quality during production and commissioning significantly reduces the failure rate at the beginning. This area also explains the increased failure performance after an intervention in a system that functions well.
- During continuous operation in **Area 2 (chance failures)**, the failure rate is almost consistent. These chance failures are fundamentally difficult to ascertain and, most importantly, not easily influenced. Operating and maintenance errors contribute towards an increase in the failure rate in this region.
- The failure rate increases dramatically in **Area 3 (wear and failure due to fatigue)**. Wear and fatigue failures are primarily characterized by damage that progresses slowly over time.

The trend provides valuable information about the failure probability of the parameters measured by condition monitoring. This usually behaves like the bathtub curve, i.e. an increase in the characteristic value signifies a change in the system. The characteristic values can be synchronized with the operating parameters through the integration of automation technology. Assessing curve performance and incorporating operating parameters makes it possible to identify the best time to perform service work depending on condition.

In addition, oscillations are often representative of the condition of a machine or component. They are a good indication of wear or damage. One example of this is roller bearings. Surface damage (pitting) leads to increased oscillations in the bearing housing. These can be measured and evaluated. An increase in oscillations during operation indicates damage and/or increased wear. By constantly observing this condition, deviations from normal operation can be recognized immediately.

5.1.1.2 Damage development and the damage chain

Explaining damage development with a roller bearing example

The majority of bearing damage develops slowly and usually without being noticed. In many cases, it is only once the damage is more advanced that irregular running and unusual operating noises indicate bearing damage. This points to material fatigue, e.g. chipping or altered radial clearance through wear.

If the damage is so advanced that it can be detected without measuring equipment, spontaneous failures such as blockages and breakage of the bearing housing components often occur.

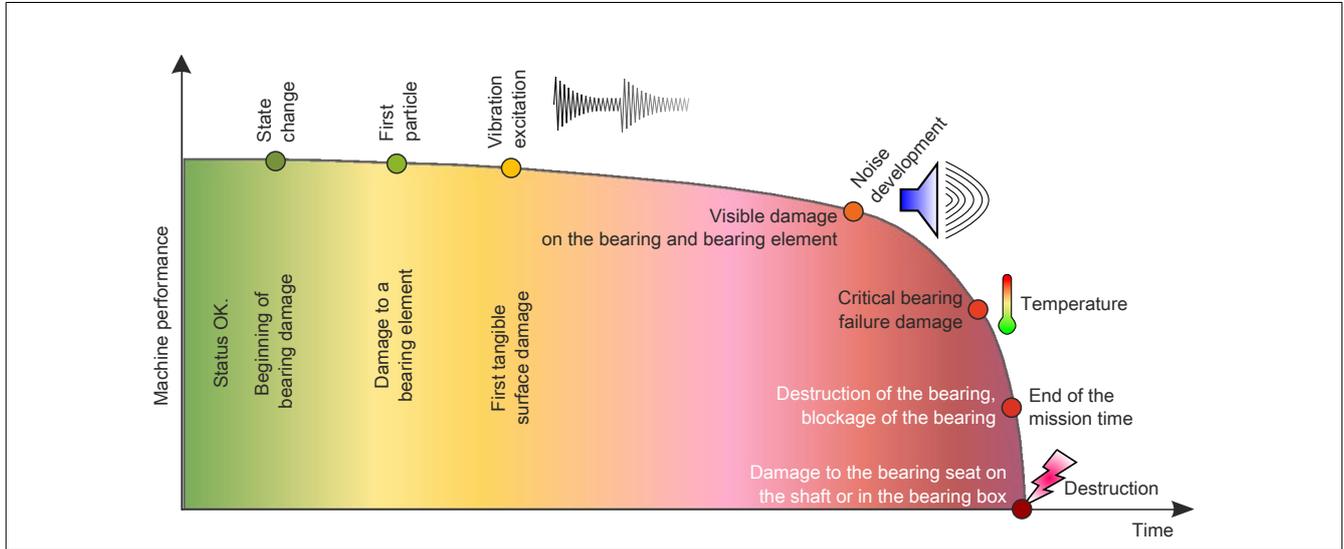


Figure 26: Illustration of the damage chain and the development of damage over time

The damage chain illustrated in the image above presents condition monitoring as an important tool in the condition-based operation and maintenance of a system.

It is possible to draw conclusions about the condition of the machine or its components from the parameters recorded by the sensors. Deterioration of the condition of components or system parts becomes apparent through the detection and observation of their condition, the consideration of the trend and, if necessary, through the detailed analysis of the measurement data obtained. Based on this, targeted measures can be put in place for maintenance.

Condition monitoring is a suitable option under the following conditions:

- Measurable parameters that correlate with a failure need to be identified and selected.
- The failure cannot be prevented by redesign or altered usage.
- Events lead to failures that occur randomly.
- Sufficient warning time must be given in advance before a function fails.

The consideration of condition monitoring resources is not permitted to be limited to the consideration of the recording of operating conditions alone but should be integrated into this as part of a general asset management strategy. In order to manage this, all types of condition monitoring and industrial diagnostics should be combined into an overall strategy.

For a failure-oriented operational mode ("reactive maintenance"), components are only replaced if they can no longer fulfill their function. For planned operation ("preventive maintenance"), components are replaced at a certain point in time – regardless of their current condition.

For condition-based maintenance, the area where maintenance is planned can be significantly isolated while reducing the risk of failure at the same time.

	Advantages	Disadvantages
Reactive maintenance	- Utilization of the wear reserve - No costs during the period of use	- Unexpected failure - Consequential damage - High downtime costs - Low operational safety
Preventive maintenance	- Can be planned well	- No utilization of the wear reserve - Increased risk of failure after maintenance - Repair costs
Condition-based maintenance	- Early recognition of problems - Downtime can be planned - Utilization of the wear reserve - High operational safety - Avoidance of consequential damage	- Dealing with the issue - Investment costs - Additional costs for equipment to measure condition

5.1.2 Conventional condition monitoring

- There exists a lot of measurement and process data that is not used for condition monitoring (CM). Links and correlations with process parameters and further CM parameters can often be made possible only with considerable expenditure.
- CM systems are implemented as "isolated applications". In such cases condition monitoring is performed by standalone sensor and measurement systems with standalone hardware and software
- The range of different systems can lead to significant problems in the on-site operation of the system. Different software solutions make it more difficult since there is separate hardware and a separate user interface for each CM method and each CM tool.
- In many cases, necessary expert knowledge specific to a system is not available. The complexity of some configuration tools exceeds the user's existing expertise.

Using the X20CM4810 together with B&R standard modules has the following advantages:

- Easy exchange of process data and condition monitoring data
- Easy integration of parameters in the overall process
- Modular construction

5.1.3 Overview of condition monitoring methods

Method	Brief description of operation	Available signals/interfaces
Determination of the condition of coolants and lubricants, visual assessment, filtering, ferrography, magnetic detection, spectroscopy, radioactive trace analysis	Quantitative examination of wear products (filtering, magnetic catches, ferrography, spectral oil analysis, particle counting) that are obtained from the lubricant oil or coolant fluid Regular taking of samples according to a defined time schedule or according to operating hours Quantitative comparison of the samples	Using suitable measurement systems, these condition monitoring methods can be processed in the B&R system using analog/digital inputs or bus controller modules.
Thermal diagnostics Temperature sensors, thermometry, infrared measurement technology	Recording of temperatures through sensors with different physical modes of action Recording of temperature distribution through detection of infrared radiation	
Acoustic emission analysis Airborne sound measurement, pulse analysis, pulse density fluctuation analysis, sound pressure measurement, sound emission location	Airborne sound measurement from infrasound to ultrasound using a microphone Multidimensional microphone structure for emission source location Recognition of microdamage (cracks), etc. through the measurement of acoustic emissions of transient waves with high frequencies	
Vibration measurement Structure-borne sound measurement, FFT analyses, order analyses, modal analyses	Vibroacoustic diagnostics Measurement of the structure-borne sound at bearing positions or structures with accelerometers, analysis and evaluation	
Electrical parameters, motor current analysis, Isolation resistance measurement	Compilation of electro-technical parameters and analysis with regard to condition monitoring	

5.2 Oscillation measurement technology

5.2.1 Sensor technology

Oscillation sensors convert the mechanical oscillations of the machines being monitored into electrical signals.

For the most part, structure-borne sound, i.e. the sound that spreads through a solid object, is measured.

The acceleration measured within the scope of the condition monitoring is typically measured with piezoelectric sensors. The oscillation sensors being used utilize the piezoelectric properties of quartz or certain ceramics. The actual measured value is a force that is proportional to the acceleration.

The piezoelectric effect involves a charge separation when a force acts on the piezoelectric material. This is in proportion to the force and is consequently proportional to the acceleration. Piezoelectric quartz or piezoelectric ceramic is used as the piezoelectric material. The output signal is an electrical charge that is specified in pC (picocoulombs). A charging amplifier is necessary in order to convert the charge into a voltage signal.

The Integrated Electronics Piezo Electric (IEPE) technology used in B&R sensors strengthens the signal directly in the sensor and emits it as a low-resistance voltage signal. Sensor sensitivity is specified in mV/g.

1 g = 9.81 m/s² (gravitational acceleration)

Information:

Piezoelectric sensors cannot measure static magnitudes.

5.2.1.1 Basic design

In a compression sensor, the piezoelectric crystal is held between a seismic mass and the base of the sensor. Through the acceleration that occurs, the force acting upon the crystal increases or decreases. The larger the seismic mass, the larger the output signal.

Sensors of this design can be built with a very high stiffness and thus have a correspondingly high resonance frequency.

5.2.1.2 Influencing variables on the sensor

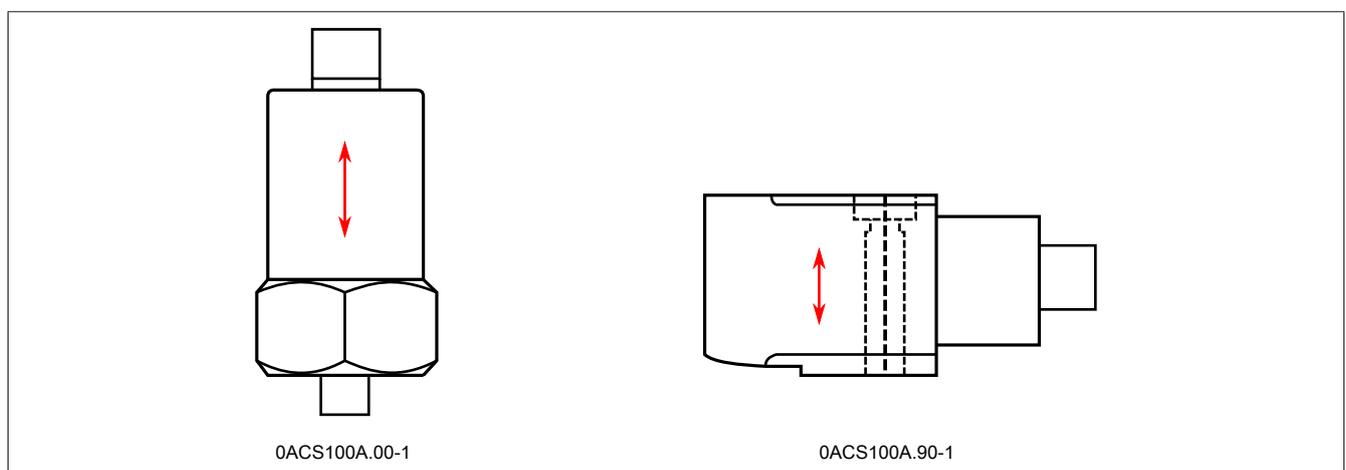
5.2.1.2.1 Mounting direction - Preferred direction

Oscillation sensors can be fitted at any location. The installation position usually results from the measuring task itself. Nevertheless, vibration sensors have a preferred measuring direction. This is normally marked on the sensor housing.

Oscillations perpendicular to the installation position also act upon the sensor. These can be compensated as much as possible through appropriate constructive measures and suitable selection of the piezoelectric crystal.

Installation position

B&R vibration sensor 0ACS100A.00-1 is designed for measurements in the longitudinal axis; sensor 0ACS100A.90-1 is designed for measurements in the transverse axis.



5.2.1.2.2 Cross-sensitivity

Oscillations in all directions act upon the sensor. The sensor preferentially transmits vibrations in its main direction, i.e. in the direction indicated on the sensor. Vibrations deviating from this main direction, however, are also registered by the sensor and transmitted differently in the overall signal depending on the design.

5.2.1.2.3 Cables

When transferring a signal over a connector, errors such as noise, ground loops and contortions may occur. This influence is particularly important in the transfer of charges since system noise is a function of the cable's capacity. When using IEPE technology, the sensor produces a high voltage signal with a low source impedance due to its internal electronics.

As a result, this technology is well suited for signal transmission over long lines.

The power supply unit provides constant current to supply the IEPE electronics in the sensor. The maximum frequency that can be transferred via the test lead without considerable loss depends on the length of the cable, the cable capacity and the ratio between the output amplitude and the constant current.

Information:

Maximum cable length when using B&R cable 0ACCxxx0.01-1: 100 m

5.2.1.2.4 Temperature influence

All piezoelectric materials also have a distinct pyroelectric effect. This describes the change in the electric polarization of ferroelectric materials as a result of a change in temperature. This effect is undesirable since it often leads to charge separations in the oscillation measurement. These arise from the change in temperature and not through the mechanical vibrations occurring in the measured object.

However, this effect can be limited due to the design. The interferences are grouped together in the interference transfer factor for temperature changes.

Temperature drift

The values specified in the sensor's technical data have been determined as a step response to a change in temperature for the lower electrical limit frequency $f_u = 1$ Hz.

Changes in temperature, in particular, cause interference in the low frequency domain below $f = 10$ Hz since this is generally a low frequency event.

5.2.1.2.5 Interference

In the vicinity of electromechanical machines such as motors and generators, electromagnetic alternating fields and the associated induction and magnetostriction cause a reaction on the sensor. Due to the shielding concept used and the use of IEPE technology, however, this retroactive effect is very low.

This effect is specified in the interference transfer factor. It is determined by a magnetic flow density of 0.01 T and a frequency of 50 Hz.

Information:

B&R sensors have an isolated base. When using sensors from other manufacturers, it is important to pay attention to isolation/screening in order to minimize interference. This allows interference frequencies with single and double mains frequency.

5.2.1.2.6 Linearity

The piezoelectric sensor is linear over long distances of the frequency response.

5.2.1.2.7 Frequency response

The frequency response of the sensor is determined by its mechanical construction. The seismic mass and the rigidity of the inner structure is crucial here, as is its construction.

The sensitivity is linear in other areas of the frequency response. The frequency response only increases significantly when near the resonant frequency. Since misinterpretations can occur in proximity to the resonant frequency, the resonant frequency must be correspondingly high.

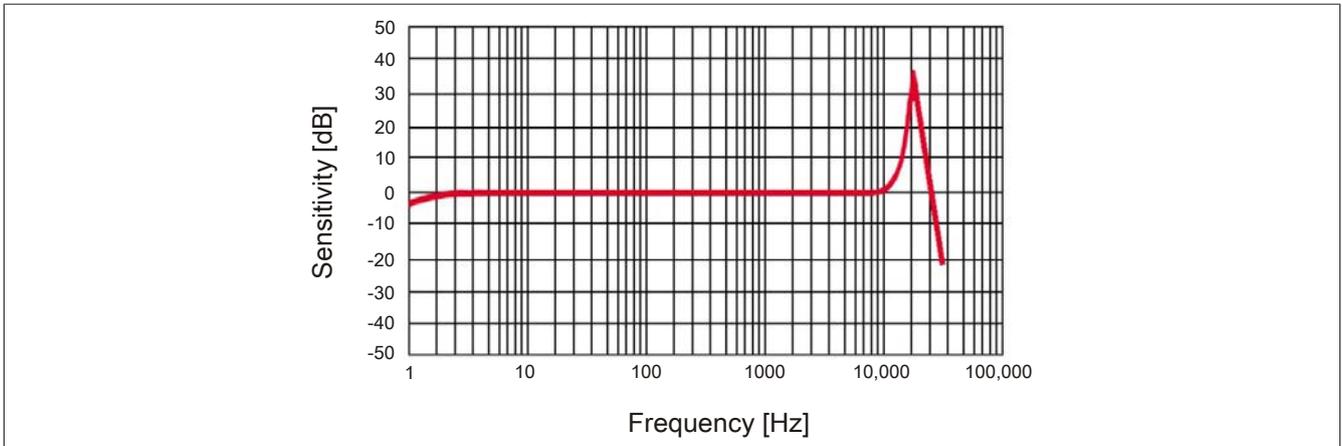


Figure 27: Frequency response of B&R sensors 0ACS100A.00-1 and 0ACS100A.90-1

5.2.1.3 Installing sensors

Sensors can be connected to object to be measured using various methods. How the sensors are installed on the measurement object is crucial for the quality of the overall measurement.

In order to transfer all frequency portions to be measured to the sensor accordingly, very good coupling of the sensor to the mechanical component is necessary.

Machine parts may be subject to temperature-related expansion and deformation, with the result that the sensor no longer has its whole measurement surface available. This affects the quality of the measurement.



Figure 28: Temperature deformation of a machine part (excessive deformation shown)

Information:

For optimal measurement, the mounting surface of the sensor must be flat and the sensor must be fully supported.

In order to increase the quality of the measurement accordingly, a fixed connection to the measurement object is necessary. Cover plates and plastic parts are therefore not suitable for securing sensors.

The following methods are available for installing sensors:

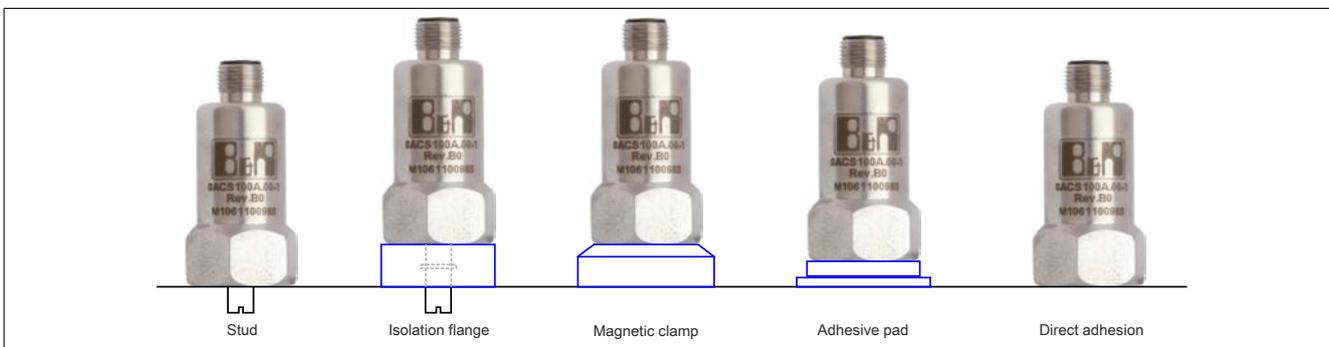


Figure 29: Overview of installation methods

Of all of the mounting methods, fastening by means of a screw is preferable due to the low attenuation between the sensor and the measurement surface.

Typical sensor installation is performed by screwing the sensor to the measurement object with what is called a stud (supplied with the sensor). Studs are specially-made setscrews made from special materials that facilitate the optimal transfer of vibrations.

If the mounting location is well prepared and the sensor is screwed on correctly, frequencies of up to approximately 10 kHz can be transferred without significant loss.

Information:

To ensure measurements of sufficient quality, B&R recommends mounting the sensor with a screw.

Ground loops can occur when the module and sensor are mounted far apart. If necessary, isolated mounting must be performed using an isolated adhesive mounting plate.

The resonant frequency is reduced by additional elements such as an isolation flange, collar screw, magnetic clamp and probe tip positioned between the coupling surfaces. The coupling becomes softer using these elements. These differences can be clearly seen in the frequency response diagram

Different frequency responses of the relative voltage transfer factor are illustrated in the image below.

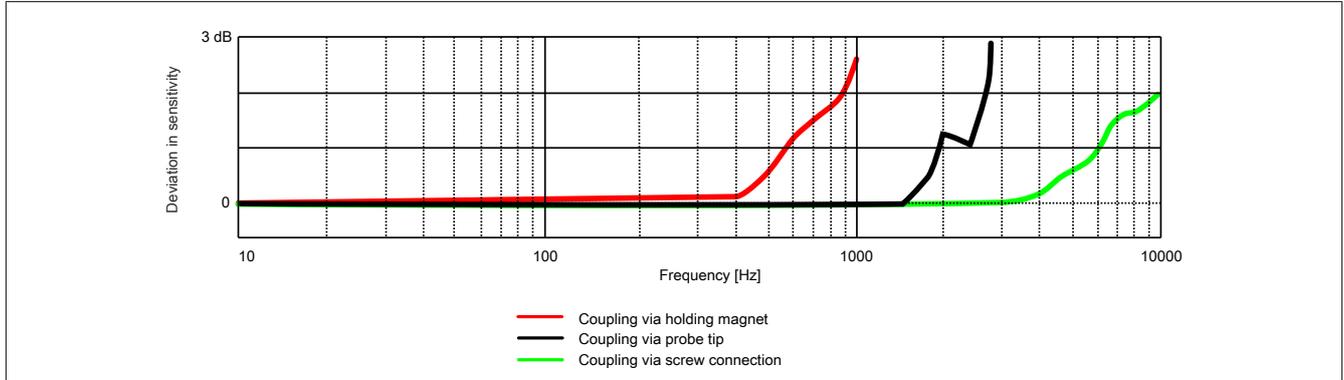


Figure 30: Attenuation of the different types of sensor mounting

The choice of mounting is influenced by the resonance frequency and temperature. The following table shows how strongly these influences impact on the different mounting methods.

	Resonant frequency	Temperature
Stud	●	●
Instant adhesive	●	●
Beeswax	◐	○
Double-sided adhesive tape	○	◐
Magnetic clamp	◐	●
Probe tip	○	○

Effects on the installation methods:



5.2.1.3.1 Installation procedure

Depending on the conditions of the location, sensors can be screwed directly onto the surface of the object to be measured.

STEP 1: As smooth and flat a surface as possible is needed to mount the sensors. The required size depends on the sensor; see the data sheet.

For a vertical installation position: See ["Dimensions" on page 165](#).

For a horizontal installation position: See ["Dimensions" on page 168](#).

STEP 2: An M8 blind hole is needed to attach the B&R sensors.

STEP 3: To further improve the transfer performance, a thin layer of silicon grease can be applied between the object surface and the sensor mounting surface. This is not necessary in most cases and is only useful when measuring particularly high-frequency vibrations.

STEP 4: The sensor should be tightened by approx. 8 Nm when using the M8x1 screw thread. If necessary, the sensor can additionally be secured with adhesive.

5.2.1.3.2 Sensor positioning

In order to optimally measure and detect the sound propagation of damage, the position of the sensor must be carefully selected. Although it is often not possible to mount a sensor at the absolute best position, this is not always necessary. Since sound waves propagate throughout the entire structure, the damage frequencies can be measured at various locations with varying intensity or amplitude (green arrow). If a flexible connection is used, a valid measurement is no longer possible (red arrow).

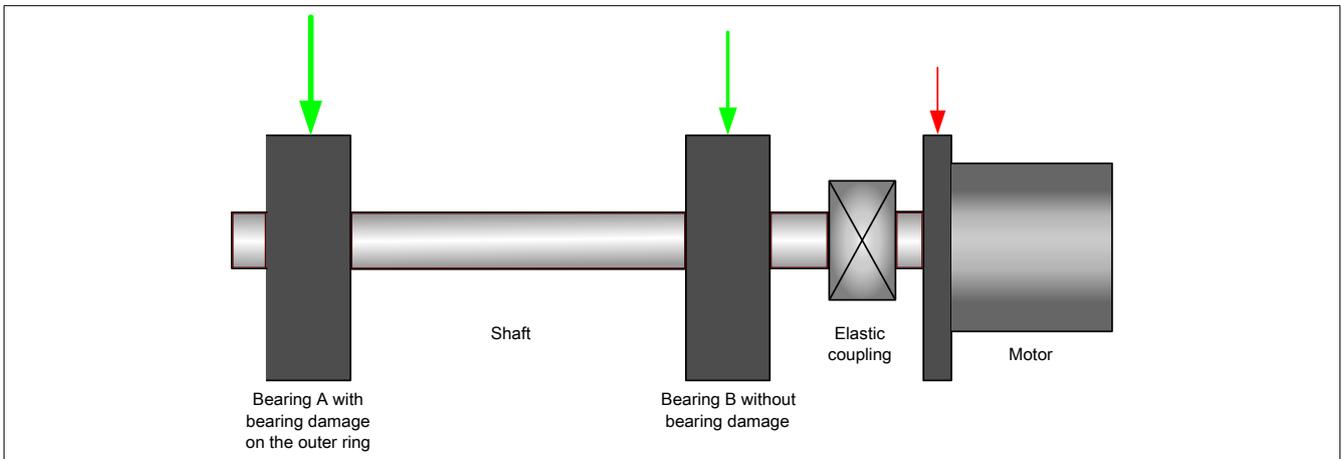


Figure 31: Suitable and unsuitable sensor attachments

5.2.2 Oscillations - Overview of measuring structure-borne sound

5.2.2.1 Oscillations

Oscillations are forms of movement that occur very frequently in nature. Harmonic oscillation is the third basic type of form of movement alongside uniform movement and uniform accelerating movement. An oscillation or vibration is a cyclic, i.e. repetitive simultaneous movement of a structure in its rest or equilibrium position.

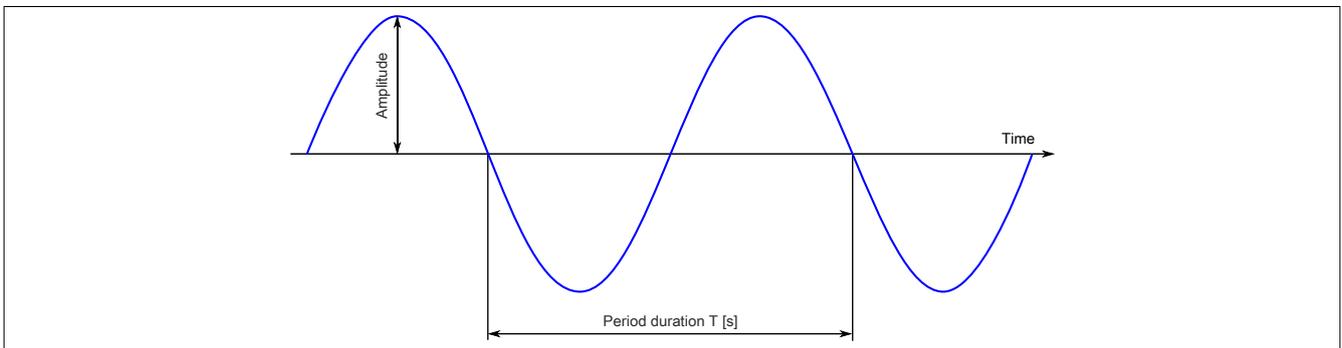


Figure 32: Illustration of a basic oscillation

If a fixed medium is stimulated by an impact, structure-borne sound spreads throughout it. This consists of additional frequencies which are determined by the shape of the structure and the material it is made from (e.g. gong or concrete block).

A portion of the structure-borne sound energy is converted into airborne sound through the surrounding atmosphere.

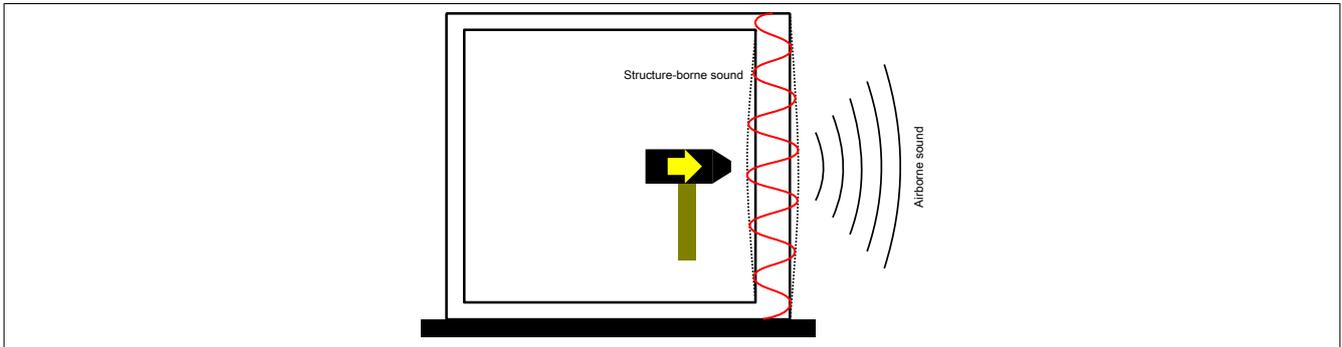


Figure 33: Propagation of structure-borne sound

The measurement and subsequent analysis of the mechanical oscillations on stationary and rotating parts of machines, support structures and pipelines has become accepted as technically possible with practically applicable monitoring procedures.

The absolute bearing oscillation is measured on the housing of the machine and involves the movements of the housing in relation to a fixed reference point in the room.

Mechanical oscillations are a good parameter for detecting initial defects and damage and can be used for machine diagnostics.

There are numerous overlapping causes for oscillations. The size (amplitude) of the oscillation depends on several factors such as attenuation through joints or grease, the rigidity of the component, the housing and foundations and much more.

Damage is always a consequence of stress. If cyclic stress occur, it can be identified by its excitation frequency and intensity.

Resonant frequency

Every machine has what are known as resonant frequencies. These must be observed during operation since with these frequencies the amplitude of the oscillation increases dramatically, putting strain on the mechanical components. If harmonic oscillations occur for a long time in proximity to the resonant frequency, this can lead to a so-called "resonance catastrophe", which causes the destruction of the affected part.

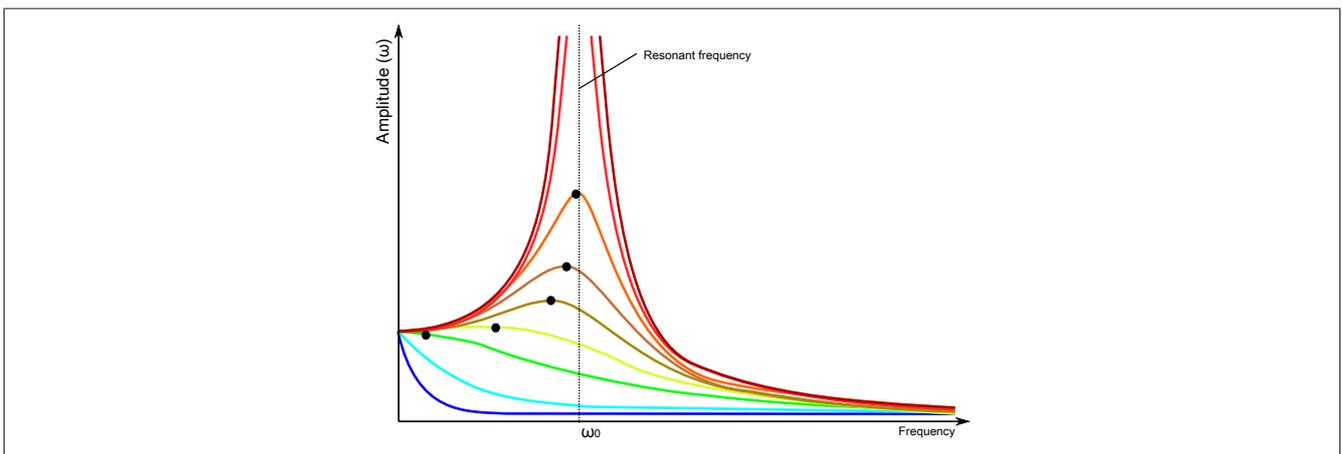


Figure 34: Increase of the amplitude in proximity to the resonant frequency

5.2.2.1.1 Causes of oscillations:

Imbalance

Per DIN ISO¹⁾, an imbalance is present in a rotating system if forces or oscillation movements are transferred to the bearing as a consequence of imbalanced centrifuges.

Imbalances on a rotating structure not only cause forces on the bearing and foundations, but also oscillations in the machine. These oscillations have a harmonic nature, with the excitation frequency corresponding to the rotary frequency of the imbalanced rotor.

Alignment errors

The main function of coupling is to connect 2 shafts in order to achieve a statically determinate complete system. In addition to transferring torque, couplings also offset misalignment (radial, axial, angular) to a certain extent. However, if the misalignment exceeds the compensation capability of the coupling, additional loads such as increased bearing forces, forced shaft deformation and axial forces occur for the machine elements involved.

Oscillations have a harmonic nature and are bound to the rotary frequency of the misaligned shafts and the multiples of this frequency. Misalignment can be measured using the rotary frequency of the misaligned part or its harmonic oscillations.

Impacts

Foreign objects as well as loose or colliding parts can cause shocks between rotating and stationary parts. These shocks repeat periodically once or several times each time the shaft revolves.

The frequency of these shock repetitions corresponds to the rotary frequency of the shaft or its harmonic frequency.

Roller bearing damage

Most bearing damage results from changes on the surface (pitting). By rolling over the damaged area on the inner ring, outer ring, cage or rolling element, pulse-shaped shocks occur that make the bearing structure and its components vibrate.

Each of these shocks appears in the oscillation signal through the typical course of a shock sequence. Characteristic values can be obtained from these measurements that give an indication of the condition of the bearing.

The excitation frequency on the inner ring, outer ring, cage and roller bearing damage is specified by the bearing manufacturer.

Magnetic induction

A rotating magnetic field causes counterforces in the stator of the machine. This electric magnetic stimulation often causes oscillations on the electric motors that are hard to detect.

Inverters also often lead to an inference of oscillations, the cause of which is of an electrical magnetic nature.

5.2.2.1.2 Effects

Machines and systems with moving parts cause mechanical oscillations. The effects on the immediate location and surrounding area include tremors and structure-borne sound, often creating a disturbing noise development.

Increased vibrations can lead to malfunctions in the machine, particularly in measurement and control devices. If this causes the measuring equipment to resonate as well, incorrect measurements will result and manufacturing quality will suffer.

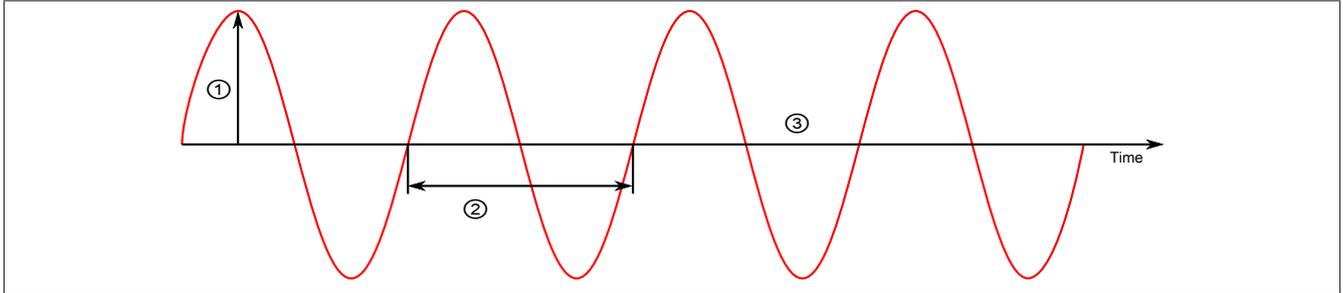
In addition, stress will develop on the components of the machine. Unwanted vibrations lead to increased wear with partly plastic distortion of components and increased crack formation all the way up to failure.

Noticeable oscillations are felt through the equilibrium organ and sense of touch. Long-term exposure can impair working performance and well-being, even leading to health damage.

¹⁾ 1925 DIN ISO 1925: Issue: 1996-11 Mechanical oscillations - Balancing technology - Terms (ISO 1925:1990 + AMD 1:1995)

5.2.2.1.3 Oscillation parameters

Parameter	Explanation	Symbol	Formula
① Amplitude	The greatest displacement from the rest position	A	
② Period duration	Minimal time span for a complete oscillation after which a structure has reached its initial position and initial speed again	T	
③ Rest position	The structure's undisplaced position	s_0	
Frequency	Number of oscillations per unit of time	f	$f = 1 / T$
Angular velocity	Change of angle in the radian measure over time	ω	$\omega = 2 * \pi * f$



5.2.2.1.4 Large mechanical oscillations

The amplitude of an oscillation can be defined by its path (s), velocity (v) and acceleration (a) values.

These 3 values have a consistent relationship with one another and can be converted from one to the other using simple calculations.

The B&R sensors measure oscillation acceleration. The unit is m/s^2 . The acceleration is also often specified with the unit g ($1\text{ g} = 9.81\text{ m/s}^2$) for the acceleration due to gravity.

For some diagnoses, however, the oscillation velocity or oscillation path is more meaningful. The acceleration can then be passed converted to the oscillation velocity through integration. The path can be calculated from the acceleration by using integration twice.

Mathematical relationship

Displacement

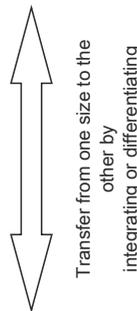
$$s = \hat{s} \cdot \sin(\omega t + \varphi)$$

Oscillation velocity

$$v = \frac{ds}{dt} = \hat{s} \cdot \omega \cdot \cos(\omega t + \varphi)$$

Oscillation acceleration

$$a = \frac{d^2s}{dt^2} = \frac{dv}{dt} = -\hat{s} \cdot \omega^2 \cdot \sin(\omega t + \varphi)$$



Information:

Displacement is not calculated by the module.

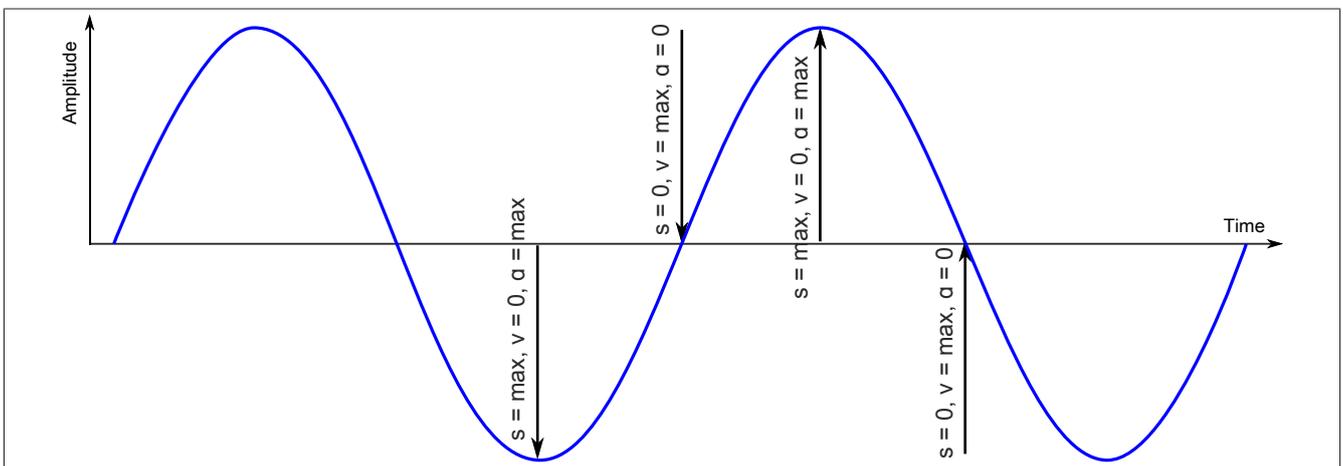


Figure 35: The s-v-a relationship

A harmonic oscillation can be clearly described by its amplitude, frequency and phase angle.

- The amplitude in the path, velocity or acceleration indicates the instantaneous value.
- The frequency describes how often an oscillation changes within one second. In oscillation diagnostics, this plays an important role since many frequencies can be assigned to one cause.
- The phase angle refers to the starting point of the oscillation. This is usually not that important since in most cases several oscillations are present.

5.2.2.2 Fast Fourier transforms (FFT)

Oscillation signals generally consist of a number of oscillations that occur simultaneously and overlap. Individual frequencies are not directly evident from a timing diagram.

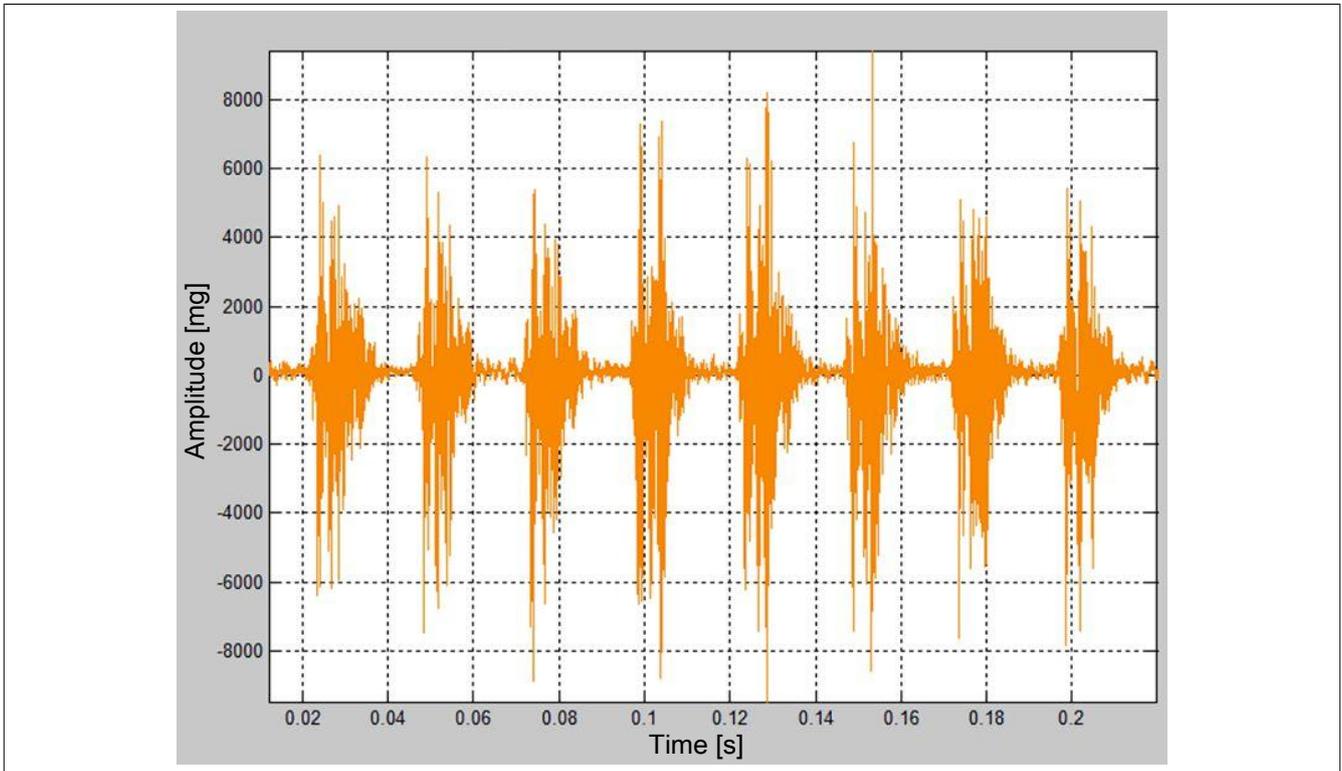


Figure 36: Timing diagram of an oscillation

Calculating a line spectrum is suitable for analyzing a mixture of oscillations of different frequencies where each participating oscillation and their frequencies and amplitudes are represented by a single line.

Within the scope of condition monitoring, spectra are a valuable aid in finding the cause of a failure. Many frequencies can be traced back to individual components so that damaged components can be identified.

5.2.2.2.1 Adding sinusoidal oscillations to generate a signal

The following figures show how a square wave signal is created by overlapping sinusoidal oscillations.

Sinusoidal oscillation with 1 Hz and an amplitude of 1.

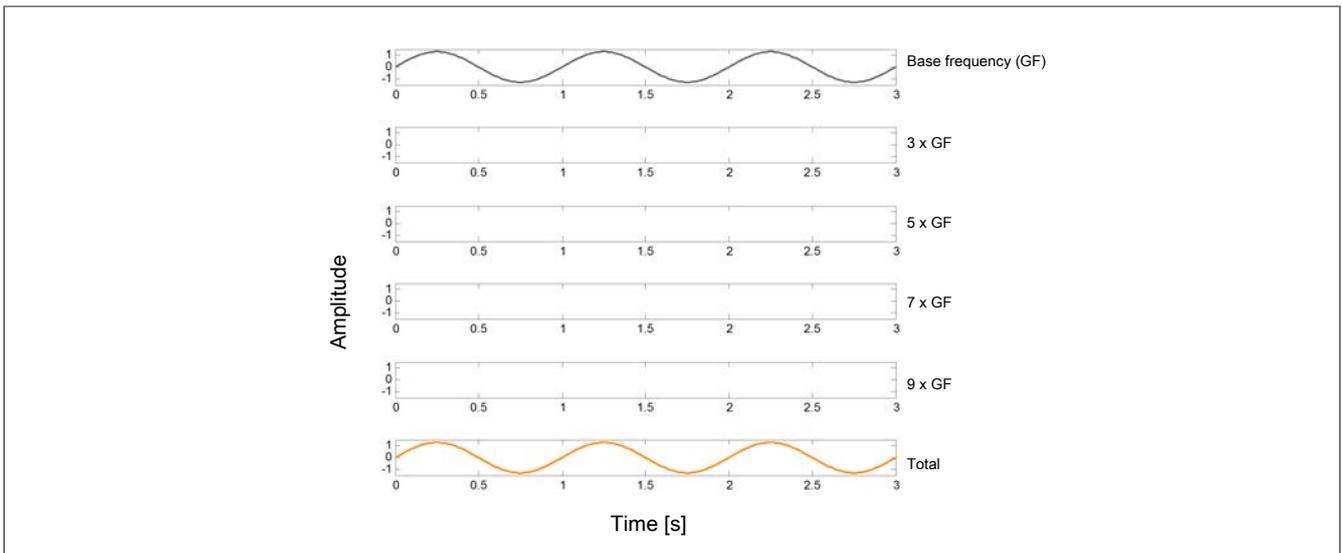


Figure 37: Pure sinusoidal oscillation

Sinusoidal oscillation with 1 Hz and an amplitude of 1 and sinusoidal oscillation with 3x the base frequency, i.e. 3 Hz and a lower amplitude.

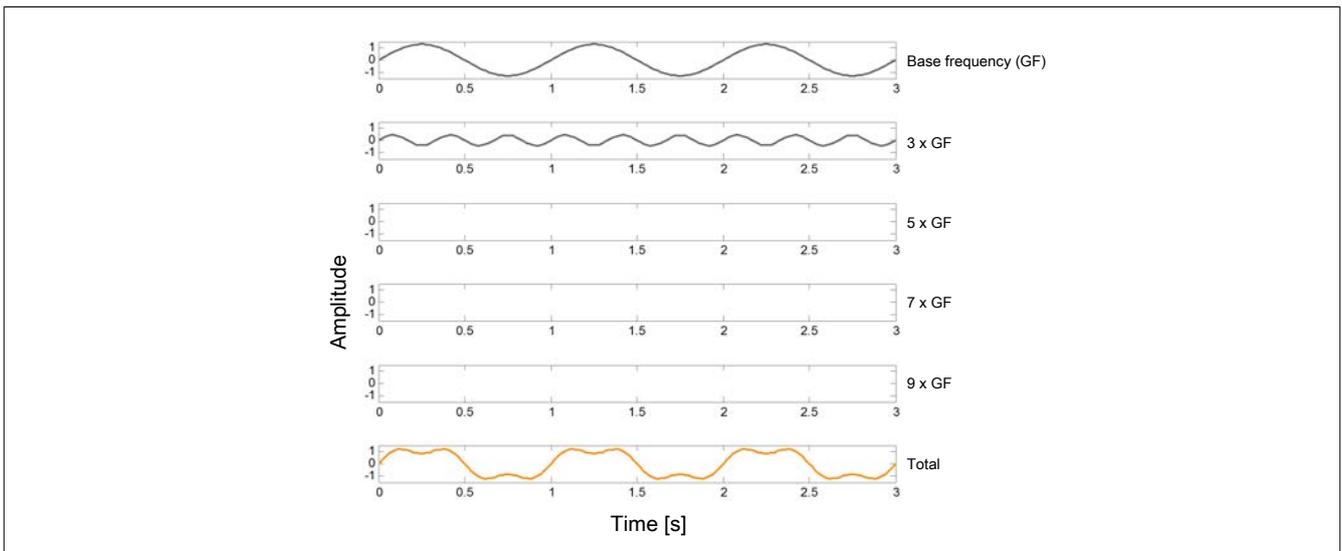


Figure 38: Sinusoidal oscillation with one harmonic

Sinusoidal oscillation with 1 Hz and an amplitude of 1 and sinusoidal oscillations with 3x, 5x, 7x and 9x the base frequency and a lower amplitude.

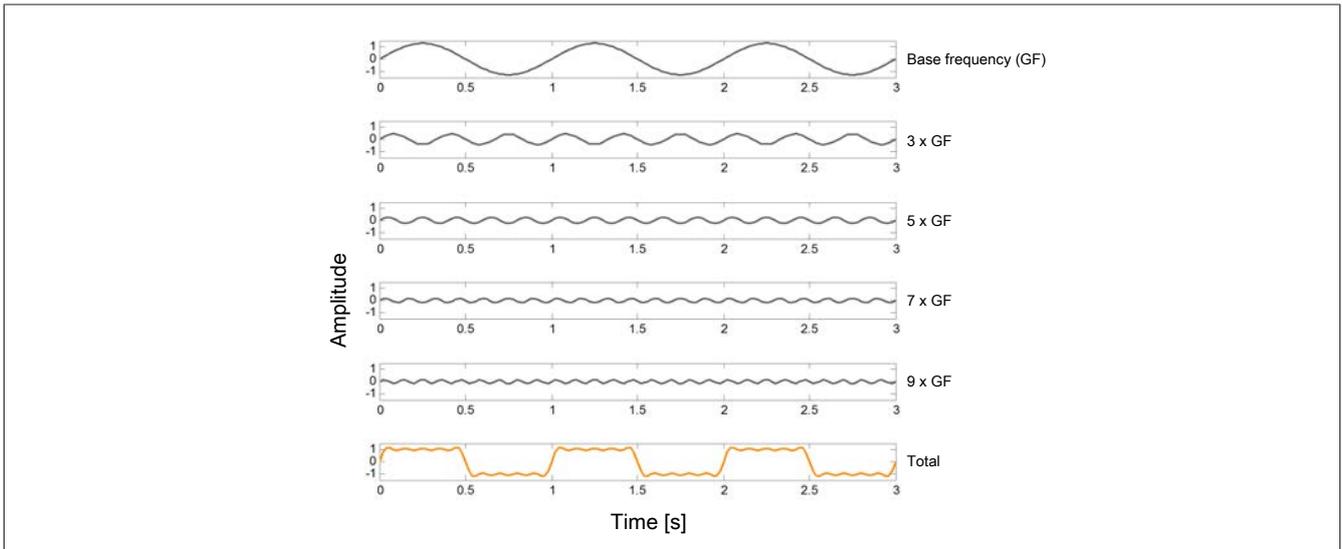


Figure 39: Sinusoidal oscillation with several harmonics

5.2.2.2.2 General description

The Fourier transform is the basic principle of frequency analysis. It assumes that each harmonic oscillation can be broken down into any number of sinusoidal and cosinusoidal waves, the sum of which reproduces the original oscillation. Linked individual waves are "broken down" again accordingly.

Probably the most well-known concept in connection with signal processing and frequency analysis is the fast Fourier transform, or FFT.

In order to be able to evaluate single partial oscillations into amplitude and frequency, the digitized time signal is converted into a frequency spectrum. In addition, a small extract is taken from the signal; this is known as the time window. Using the FFT algorithm, the frequency spectrum is calculated from this so that each involved oscillation and its associated frequencies and amplitudes is shown as a single line in the line spectrum.

Example

For a single sine signal with a constant frequency, a single line is shown in the frequency spectrum.

5.2.2.3 Information about using FFT

5.2.2.3.1 Window functions

Depending on the signal structure and boundary conditions, discontinuities may occur at the time window limits of the extract taken. These reflect partial oscillations that do not exist at all in reality.

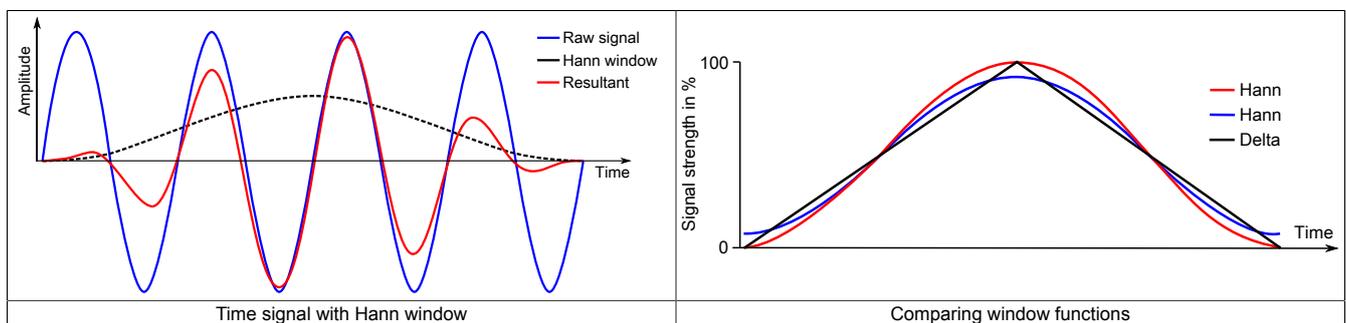
These discontinuities arise if the period of the sampling does not correspond to an integral multiple of the period of the time signal. This occurs with practically every measured signal since this is composed of a number of signals with different period durations.

Window functions are used to suppress these discontinuities. This is done by multiplying the input signal with the window function and supplying the Fourier transform with the resulting signal.

Common window functions are:

- Triangular window
- Hann window
- Hamming window

All of these functions share the fact that they approach zero at the edges, so the periodic continuation now no longer has any jump discontinuities.



Information:

The Hann window is used in the X20CM4810 module.

5.2.2.3.2 Sampling

Scanning or sampling refers to the recording of an analog value at certain intervals.

At defined times, the precise voltage level of the signal is recorded and stored. The distance Δt (Delta t) between the recording points is called the sampling interval.

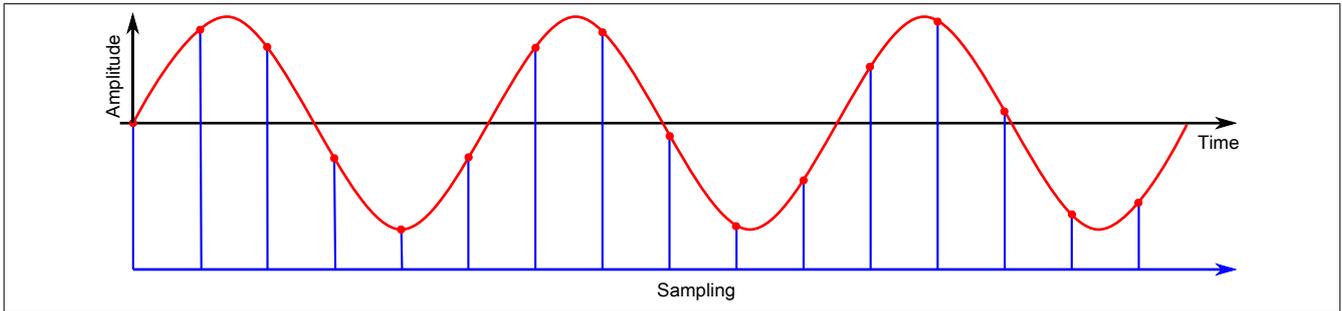


Figure 40: Sampling of a curve

If the actual sampling rate is many times higher than the theoretically required sampling frequency, it is called "oversampling". The reduction of the sampling rate to the required rate is called "downsampling".

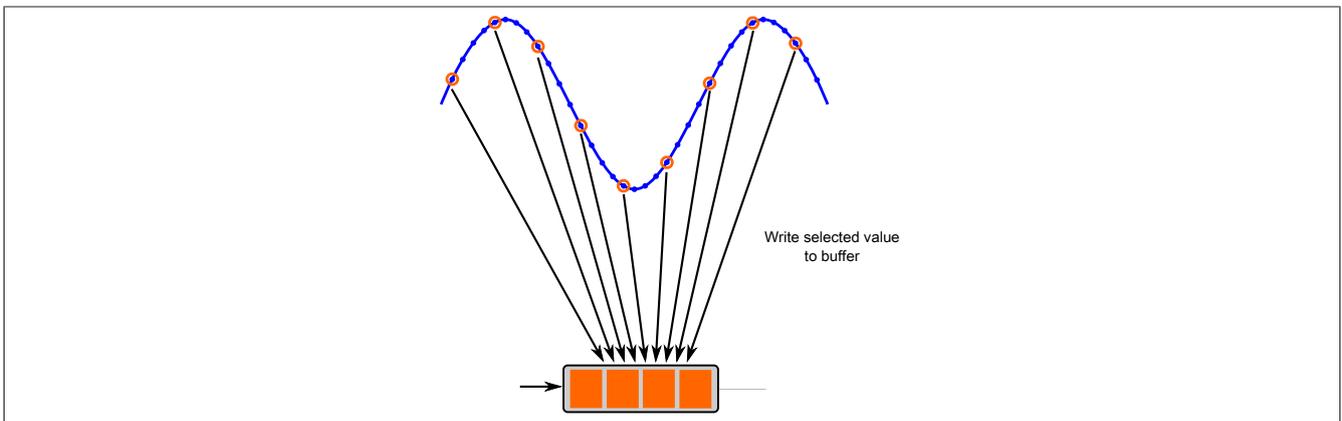
If the Fourier transform is used in accordance with the sampling theorem on a purely sinusoidal signal whose frequency corresponds exactly with a node in the frequency spectrum (whole number multiple of the frequency resolution), then this signal appears as a single line in the spectrum. If the frequency of the sinusoidal oscillation does not fall on a node in the frequency spectrum, then it will appear as 2 neighboring lines whose vertical relationship to each other is indirectly proportional to the frequency deviation of the respective node.

In practice, a pure sinusoidal signal will rarely occur. Rather, a signal will consist of a multitude of sinusoidal oscillations of different frequencies. As a result, the resulting frequency spectrum also consists of a large number of lines.

Depending on the resolution selected, the lines manifest themselves in different ways on the spectrum.

Buffer storage

The values that have been sampled are stored for 300 ms in the module's internal buffer and must be transferred during this time. (See "Transferring characteristic values via Flatstream" on page 67.)



The size of the buffer is constant and can store 8192 measured values. This results in the following relationship between the sampling rate and the duration of measurement.

Measurement duration = Buffer size / Samples per second

Since the number of values stored depends on the configured sampling rate and not the hardware-based sampling rate, not all values that are measured are stored. At a measurement duration of 318 ms, every second value is stored; at a duration of 15.9 ms, every hundredth value is stored.

The number of stored values can be set indirectly through registers "MaxFrequencyEnvelope" on page 56 and "MaxFrequencyRaw" on page 57

5.2.2.3.3 Aliasing effect

The input signal is sampled at regular intervals. If the sampling rate used is too low, the input signal will be sampled incorrectly and a flawed image of the oscillation will occur. This undesirable phenomenon is called the aliasing effect.

To avoid such false results, the requirements of the theorem known as Nyquist's sampling theorem must be fulfilled in the sampling. This sampling theorem describes the necessary frequency ratio between the sampling and the signal and states that the sampling rate must be more than double the maximum frequency of the measured signal.

Example of incorrect sampling

Sine wave with 4 kHz sampled at 6 kHz. The red wave shows the sine wave measuring 2 kHz, which is a result of the sampling being too low.

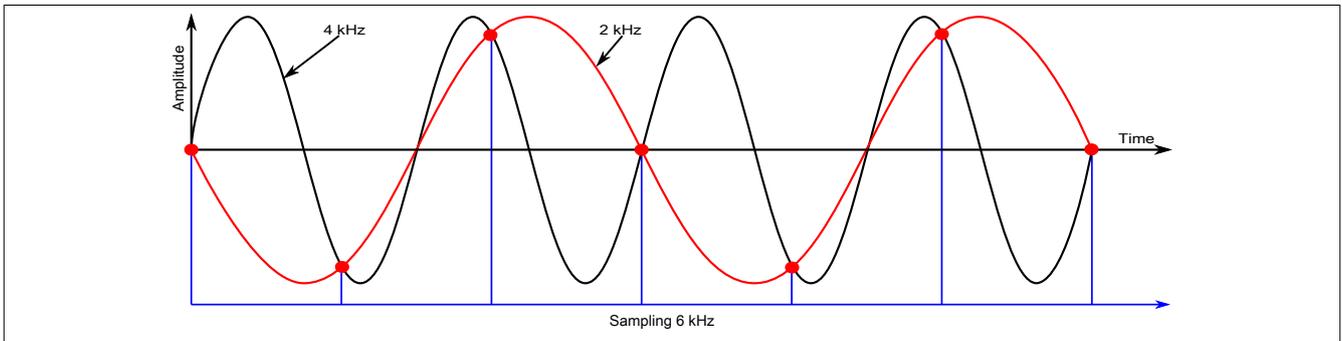


Figure 41: Incorrect sampling with 6 kHz and 4 kHz

Information:

The module ensures that Nyquist's sampling theorem is always fulfilled.

With a wanted signal of 10 kHz, a reduced sampling frequency of 25.7812 kHz is used!

5.2.2.3.4 Resolution

The time-continuous input signal is converted into a series of discrete digital output values. The A/D converter resolution determines the maximum possible number of digital values. Since this is always a discrete number, digitization always results in a deviation from the actual input signal, which is known as quantization error.

The sampling rate determines the interval between 2 conversions and must be selected to suit the type of processing planned for the signal. For many signal processing tasks, it is fundamentally important to adhere to the sampling theorem. The module performs appropriate filtering and decimation steps to ensure that the sampling theorem is adhered to at all times, regardless of the configured sampling rate.

In the time domain, a higher sampling rate allows for a more detailed description of the signal but also consumes a larger amount of memory. In the frequency domain, a higher sampling rate increases the maximum frequency that can be evaluated. Since a defined number of values is used for the FFT, however, the resolution decreases and thus the possibility of distinguishing frequency ranges close to each other.

Information:

On the X20CM4810, a MaxFrequency of 200 Hz results in a distance of 0.0629 Hz between 2 lines.

Time signal

The resolution can be identified in the time signal from the distance between 2 adjacent measurement points.

Example

MaxFrequencyRaw is 2 kHz in this example.

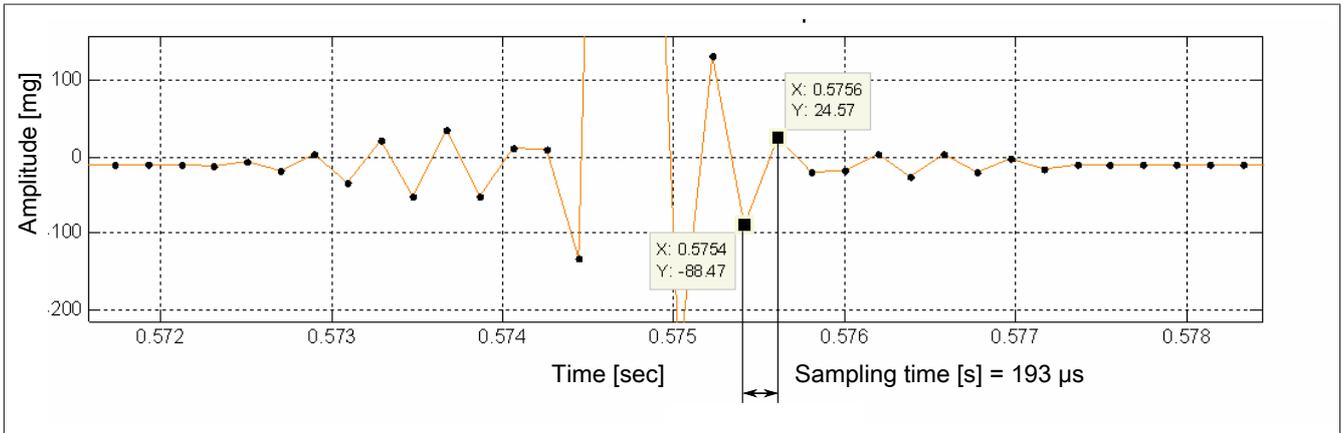


Figure 42: Time signal with corresponding resolution

Frequency spectrum

The resolution in the frequency spectrum indicates the distance between the individual frequency lines (spectral lines) that can still be evaluated.

Example

Frequency spectrum at a MaxFrequency of 2 kHz.

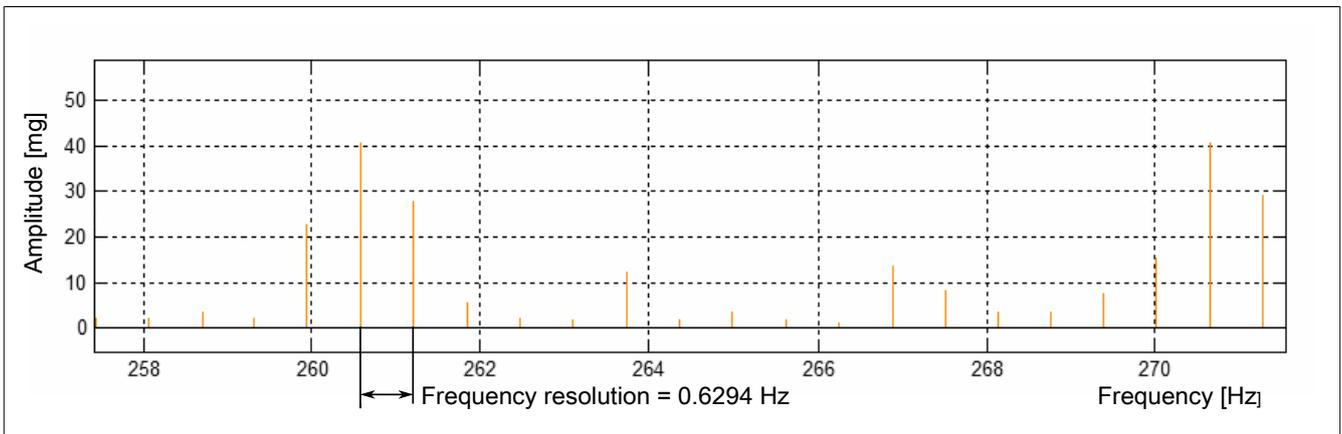


Figure 43: Frequency spectrum

Quantization

To process analog signals digitally, an analog-to-digital converter (AD converter) is required. However, only voltages in steps can be measured by an A/D converter. This is known as quantum voltage. The range to be measured is thus quantized.

In an AD converter, the digital resolution depicts the number of levels there are in the quantization. This determines the accuracy and sensitivity with which a previously analog level value is depicted. The more available levels there are, the more precise the discrete signal received and the higher the sensitivity of the measurement.

The resolution indicates how many varying digital values an analog signal can be converted into. The resolution is expressed in bits.

8-bit resolution	256 level values
16-bit resolution	65,536 level values
24-bit resolution	16,777,216 level values

Example

A 24-bit resolution, a sensor measurement range of ± 10 V and a sensor sensitivity of 100 mV/g result in:

$$20\text{V} / 2^{24} = 1.192 \mu\text{V} \rightarrow \text{Corresponds to } 11.92 \mu\text{g}$$

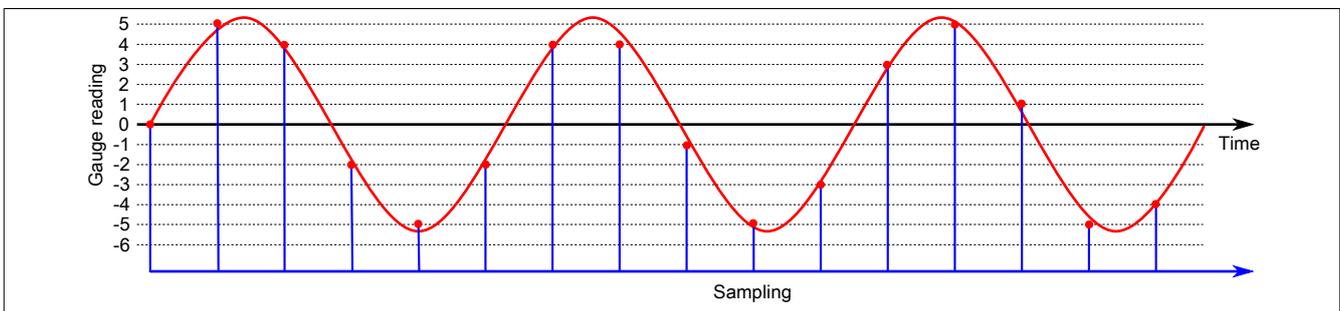


Figure 44: Quantization error at low resolution

Information:

The module has 24-bit resolution.

5.2.2.2.3.5 Duration of measurement

The duration of measurement depends on the configured maximum frequency. Depending on whether the envelope or raw values are to be measured, the following registers are used for configuration:

- "MaxFrequencyEnvelope" on page 56 for envelope measurement
- "MaxFrequencyRaw" on page 57 for raw value measurement

Maximum frequency	Sampling frequency	Duration of measurement
10000 Hz	25781 Hz	0.3178 s
5000 Hz	12891 Hz	0.6355 s
2000 Hz	5156 Hz	1.5888 s
1000 Hz	2578 Hz	3.1775 s
500 Hz	1289 Hz	6.3550 s
200 Hz	516 Hz	15.8875 s

5.2.2.3 Envelope

When you look at the various causes of oscillations, it becomes clear that they primarily come from 2 sources:

1. Imbalance and misalignment

Imbalance and problems with alignment lead to predominantly harmonic, sinusoidal oscillations.

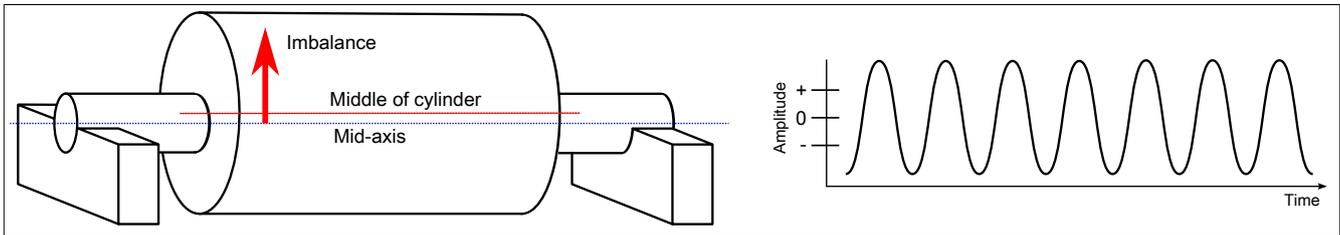


Figure 45: Imbalance - Harmonic oscillation

2. Impacts

Many types of machine damage can result in oscillations that cause the structure of the machine or adjacent machine parts to vibrate at their natural frequency. Impact-related causes of this are a result of corrosion, a rotor brushing up against the machine housing or roller bearing/gearing damage to the gears.

In the case of roller bearing damage, impact occurs when either the rolling elements roll over damage on the inner or outer track or when one of the rolling elements themselves is damaged.

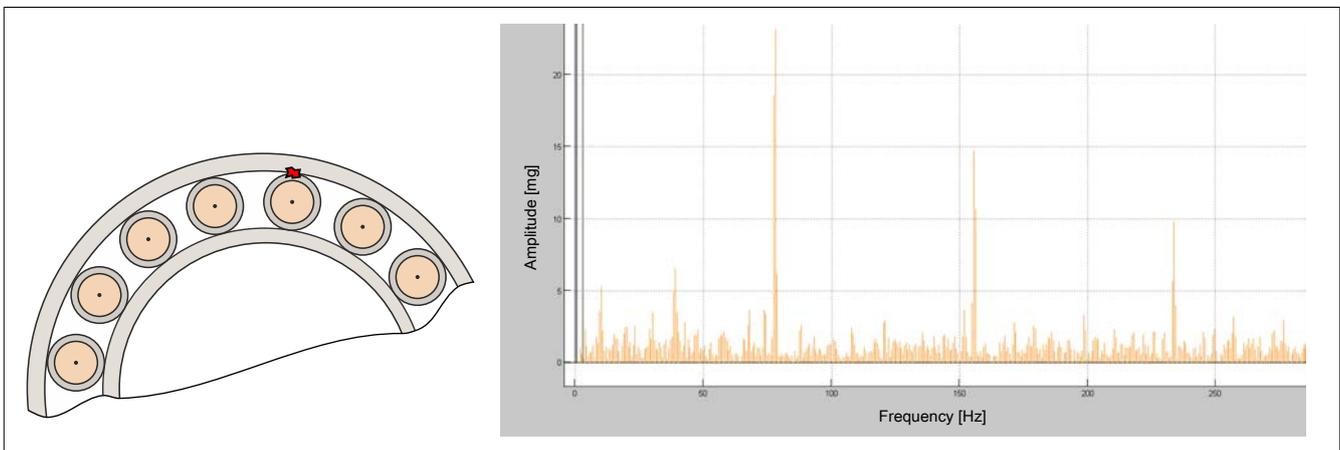


Figure 46: Bearing damage - FFT of envelope signal

This kind of impact can be compared to a clapper hitting a bell. If the bell is struck 2x per second, it vibrates at its natural frequency and not 2 Hz. The natural frequency is determined by the constructive design and material properties. As with any sound boxes, mounting also factors into this.

If the damage is to the roller bearing track, every shock pulse will lead to a corresponding reaction in the vibrating system. It is only practical to measure the sum of all pulses, i.e. the total signal.

In the case of damage to roller bearings, the bearing rings are the first to start vibrating.

Analyzing oscillations

Oscillations are transmitted in the machine as waves and can be measured on the machine's surface. A prerequisite, of course, is that a path is available for the sound transport, i.e. there are no sound-absorbing border crossings such as air, rubber, etc. between the roller bearing and sensor.

This signal can be measured on the surfaces of the machine by means of an accelerometer. The signal is made up of a variety of oscillations and impacts that overlap each other. When considering the time signal in this way, it is easy to see that allocating individual frequencies is difficult, if not impossible.

Using a Fast Fourier transform, this time signal can be converted to give the natural frequency of the system. These are in the high-frequency range. The rotor frequency and its harmonic frequency are clearly identifiable as the dominant frequency portion.

The rollover frequencies of the roller bearings are between 15 and 70 Hz at a speed of approximately 600 rpm, depending on the bearing.

In particular, shock pulses of a lower intensity, as they are when damage first begins, can barely be noticed or assessed. It is only when there is advanced damage to bearings that signal peaks can be clearly observed.

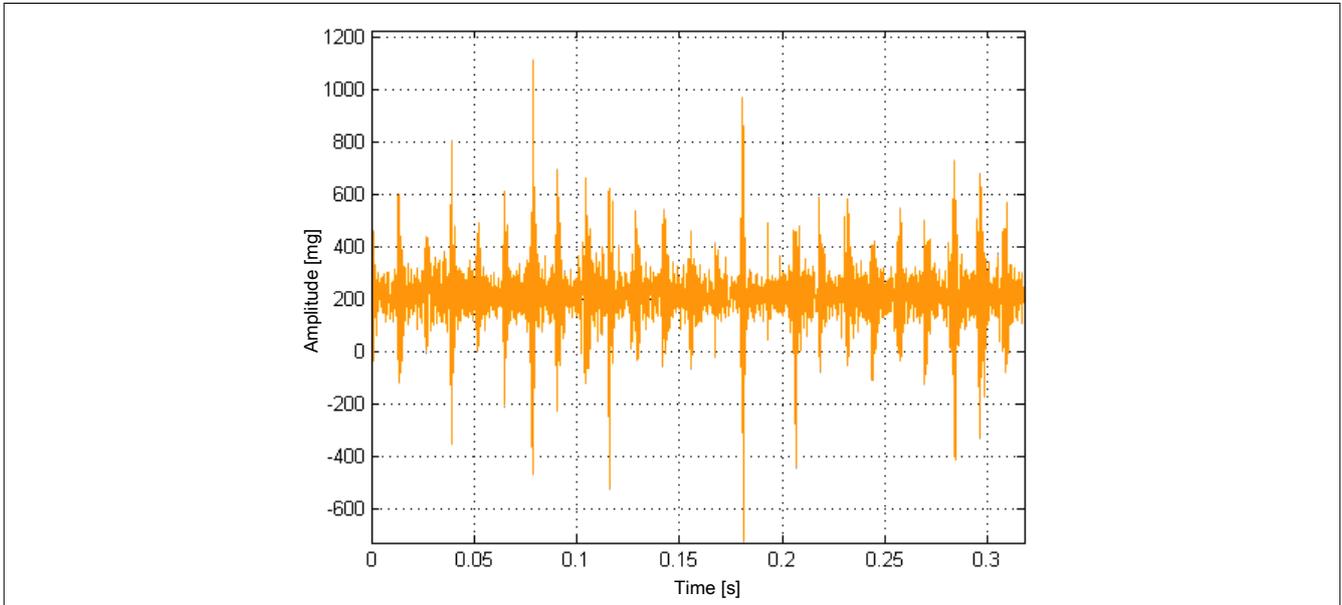


Figure 47: Advanced bearing damage at 600 rpm

To make the actual result, or shock sequence, clearly visible, it is obviously not enough to simply provide the amplitude spectrum. It is much more important that the process of convolution that took place when the signal occurred be appropriately reversed so as to separate the excitation function from the natural frequency. This is what envelope analysis provides.

An amplitude-modulated signal is made up of a high-frequency carrier signal and a low-frequency wanted signal. The amplitude of the carrier signal changes depending on the wanted signal. In the receiver, the wanted signal is extracted from the carrier signal by the formation of the envelope curve (demodulation).

In the case of machine resonances caused by periodic impacts, the machine resonances can be viewed as the carrier signal and the low-pass-filtered shock pulses can be viewed as the low-frequency modulation signal. Demodulation results in the shock pulses being extracted from the resonance frequencies.

5.2.2.3.1 Envelope analysis

Normally in an envelope analysis, the frequency spectrum of the envelope signal is evaluated. Suppressing the steady component gives a frequency spectrum that exhibits only one increased amplitude at the frequency of the low-frequency sinusoidal oscillation.

The envelope signal of the periodically peaked excited machine resonance shows mostly increased amplitudes in the shock pulse repetition rate and its multiples.

The envelope analysis is thus a method for differentiating between harmonic causes (imbalance, orientation) and impact-related causes (roller bearing damage, gearing damage, etc.).

Conversely, it must be stated that harmonic causes in an envelope spectrum cannot be determined accordingly.

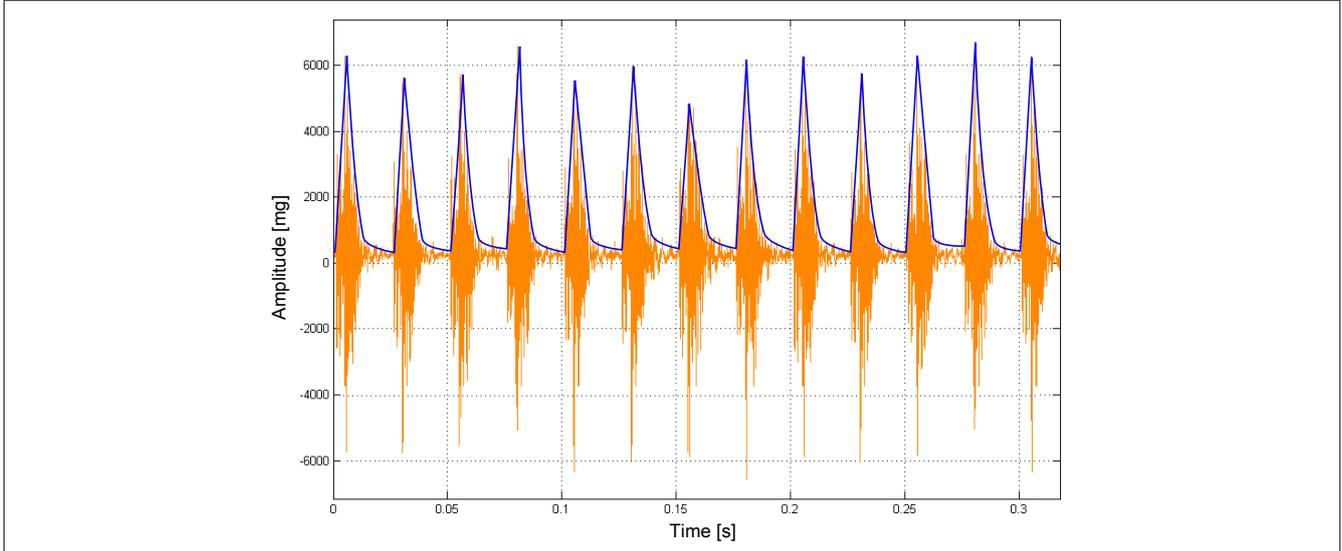


Figure 48: Time signal with envelope

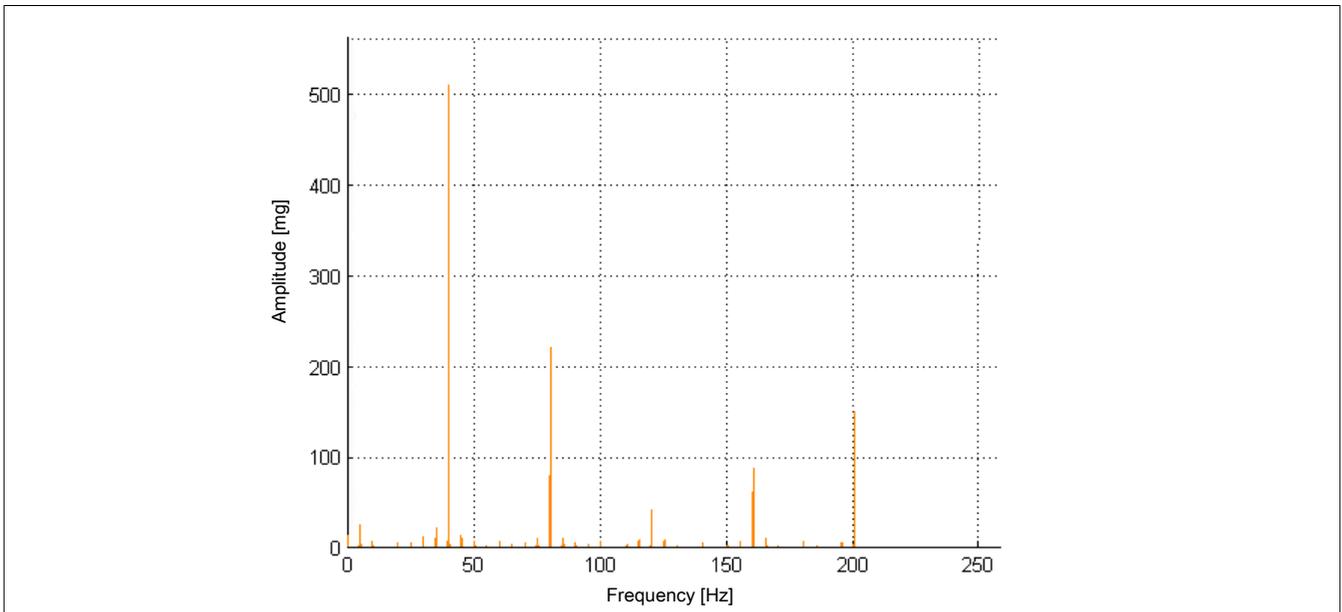


Figure 49: Resulting frequency spectrum of the envelope

5.2.2.4 Displacement, velocity and acceleration

Sensors can record oscillation acceleration, oscillation velocity or oscillation displacement. Regardless of the physical value that the sensor records, the oscillation can be represented as a combination of acceleration, velocity or displacement since:

$$s = \int v dt = \iint a dt^2, \text{ or } a = \frac{dv}{dt} = \frac{d^2s}{dt^2}$$

Legend:

- s - Oscillation path
- v - Oscillation velocity
- a - Oscillation acceleration
- t - Time

Since oscillation velocity is calculated through integration from the oscillation acceleration and oscillation displacement is calculated through integration from the oscillation velocity, it is possible to convert the sensor value whenever necessary.

Acceleration is proportional to the force. In contrast, the velocity is an indication of the energy.

What is significant from a practical standpoint is that, when one physical value is converted to another, the frequency is included. For a sinusoidal oscillation, the following applies:

$$s = \frac{a}{(2 \cdot \pi \cdot f)^2} \quad v = \frac{a}{2 \cdot \pi \cdot f}$$

The frequency is in the denominator. As a result, high-frequency signal portions are underestimated when using oscillation velocity in comparison with oscillation acceleration. This effect is squared with the application of oscillation displacement.

5.2.3 Determining limits and alarm limits

A general statement about the health of a machine can be made by comparing the measured characteristic values with limit values defined by standards (e.g. ISO 10816) or by the manufacturer of the machine.

Characteristic values are calculated from the measured signals, which are representative of the status of the system at the given measurement point.

Norms can be used in part to determine alarm limits. In addition, limits can be set based on the machine's history and the experience of the operator.

5.2.3.1 Comparison with references and norms

For certain machines and systems, limits are fixed by norms. With the exception of ISO 10816, these give little information for assessing the actual status of the machine.

5.2.3.2 Manufacturer's limits

A few machine manufacturers give limits for permissible oscillations and other relevant factors for assessing the status of the machine. These are based partly on calculations as well as the manufacturer's own knowledge and experience.

Whenever possible, these are the limits that should be used for assessing condition.

5.2.3.3 Operator's limits - Experience-based values

Operators can also draw on their own experience when assessing the status of the machine. Long-term observation of the characteristic values and the machine's history can provide relevant values based on experience.

The limits ascertained from this can vary significantly from limits set by norms and those prescribed by the manufacturer of the machines. This assessment is only possible when operators have considerable experience with oscillations in the machine and are in a position to differentiate between positive and negative characteristics.

When determining limit values, it is important to note that vibration measurements are influenced by factors such as:

- The location of the damage
- The location of the sensor
- The speed of the moving parts of the machine
- The load on the machine parts

5.2.3.4 Assessing the trend

In many cases, not enough is known about the actual behavior of a machine during operation, in particular when there is damage.

To make a reliable statement about its condition, the chronological sequence of the characteristic values (characteristic value trend) must also be used when making the assessment.

In the progression of the trend, the "Normal condition" is the starting point. The reference level (normal level) is the level of the characteristic value as set in the normal condition.

Changes in the trend are observed with the normal condition as the starting point. When there is damage, the relevant characteristic values in each case generally increase, although a decrease can also be an indication of a problem in the system.

In order to assess trends accurately, it is essential that the characteristic values of the oscillations are always recorded under the same operating conditions and always classified in the same way. In particular, the speed and load ratio have a strong effect on the characteristic values. Increases in trends under differing operating conditions cannot necessarily be a sign of a change in the actual condition. In many cases, trend observation can assist in monitoring the condition of the machine and bearings.

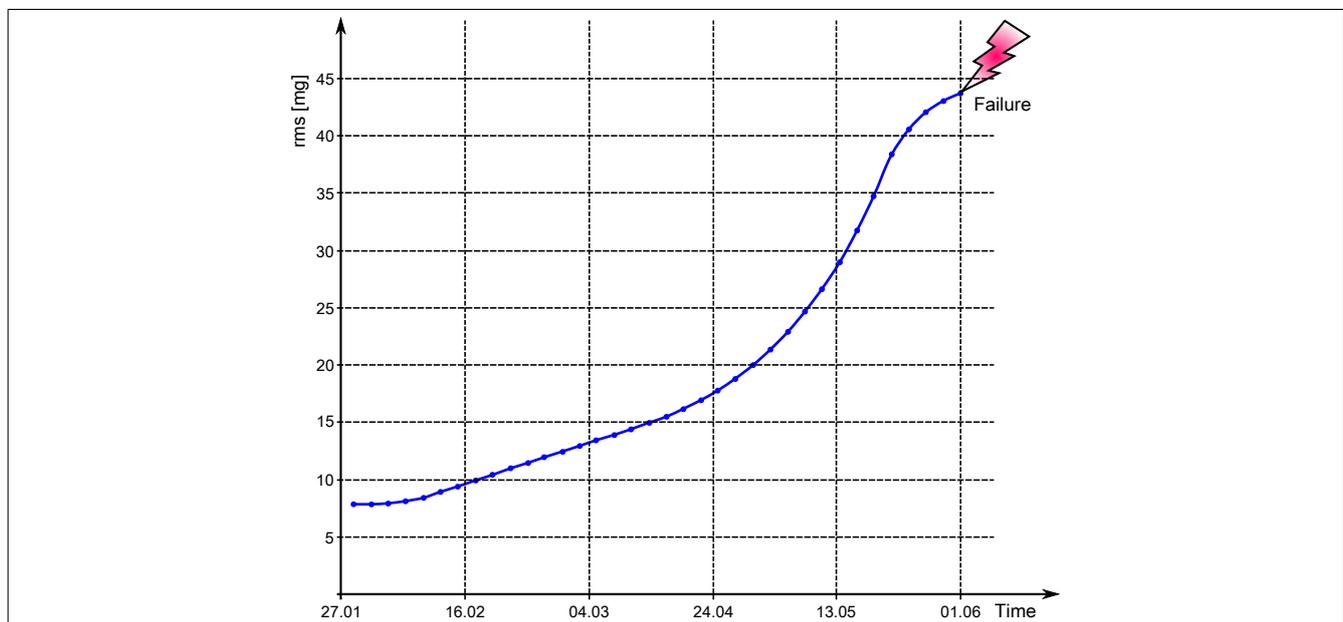


Figure 50: Typical trend progression

The first and second derivatives of the damage progression trend line can provide good information about how badly a component is damaged.

Example

This example will use the first and second derivative to determine the best time for repairs. The maximum service life will be taken into account when selecting a time to perform a replacement. The possibility of lowered production quality is not taken into consideration.

Various times for performing a replacement can be selected by referring to the trend progression.

- k1 = First increase. Very conservative, service life is wasted.
- k2 = Second increase. Good compromise between conservative and optimal.
- k3 = Drop in second derivative. Optimal utilization of service life.

Information:

This example of derivative usage is not permitted to be used as a rule in your own applications.

First derivative

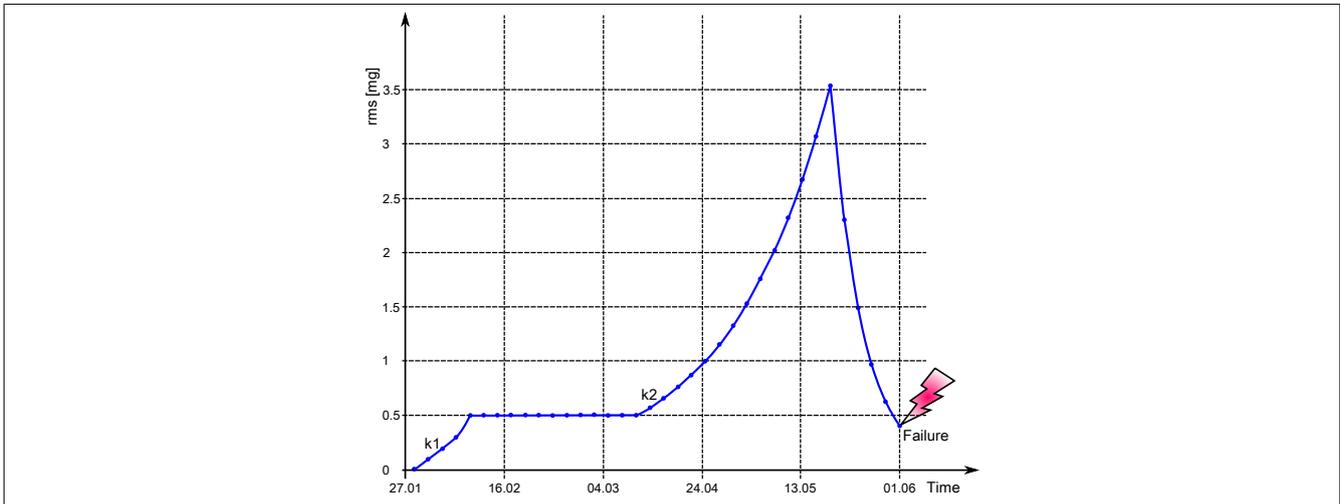


Figure 51: First derivative of the trend progression

Second derivative

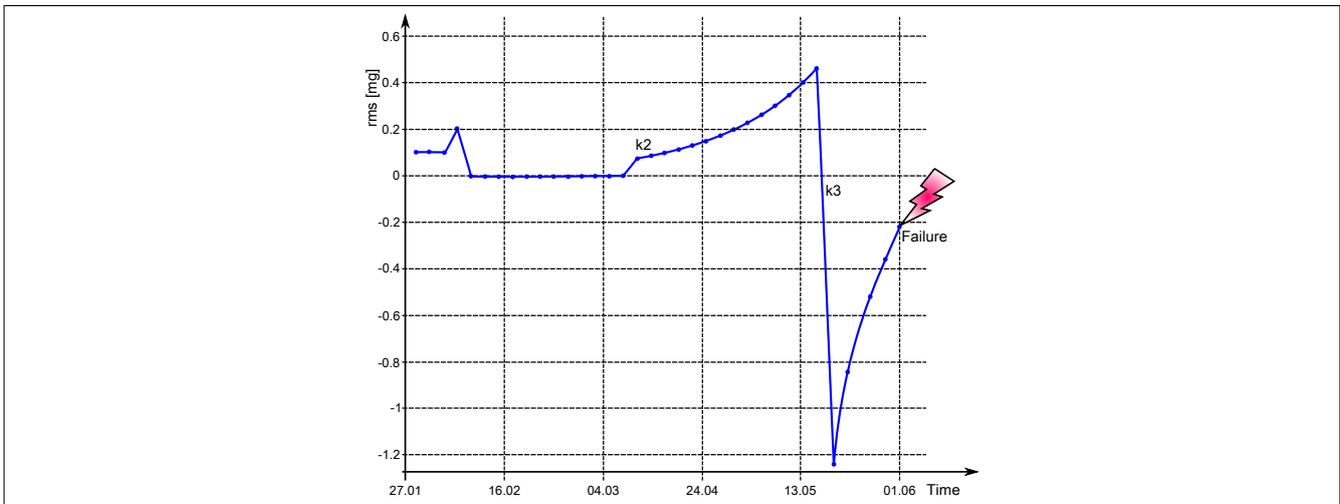


Figure 52: Second derivative of the trend progression

5.2.3.5 Dynamic speed change

With the FFT calculation in the module, it is assumed that the frequency spectrum and the resulting lines do not shift during the time it takes to calculate a data buffer. For shafts, this is only the case when they are running at a constant speed.

In the case of dynamic speed, narrow-band frequencies cannot be used to monitor or analyze damaged frequencies since these do not give valid results.

The following example shows the effect of variations in shaft speed on the resulting frequency spectrum during FFT calculation.

Example

The speed of a shaft changes from 100 Hz to 200 Hz.

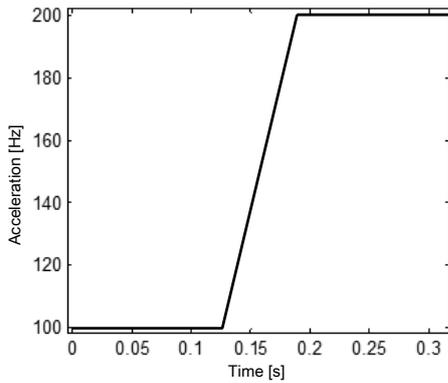


Figure 53: Speed profil

Changing the speed of a shaft
(Within one buffer measurement length.
Damage is simulated with the speed frequency.)

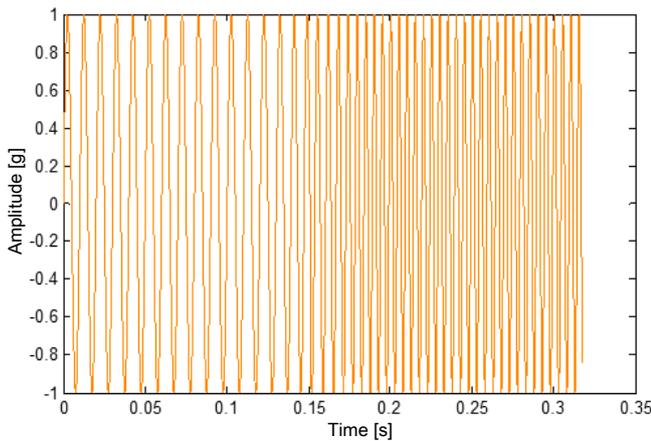


Figure 54: Time signal

Change in the time signal

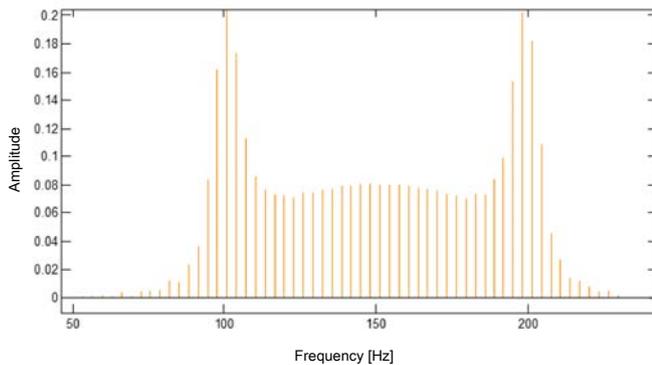


Figure 55: Invalid result in the frequency spectrum

Resulting spectrum
calculated from the time signal

Possible approaches to measuring in a dynamic system

- **Best solution:** Adopting separate measurements for trend analysis where the speed can be kept constant for the duration of measuring.
- If a measurement cannot be taken at constant speed, an assessment of the machine condition can be made using the characteristic values or a broadband frequency (in this example, this would be 80 to 220 Hz).

5.3 Practical applications of damage recognition

It is possible to gain good insight into the condition of a machine or system by measuring the parameters associated with errors. This parameter data is used by different algorithms in the module to calculate the characteristic values. This method of forming characteristic values occurs continually and automatically. It requires little technical knowledge and is easy to use and implement. It is an easy way to check a machine for damage and error states.

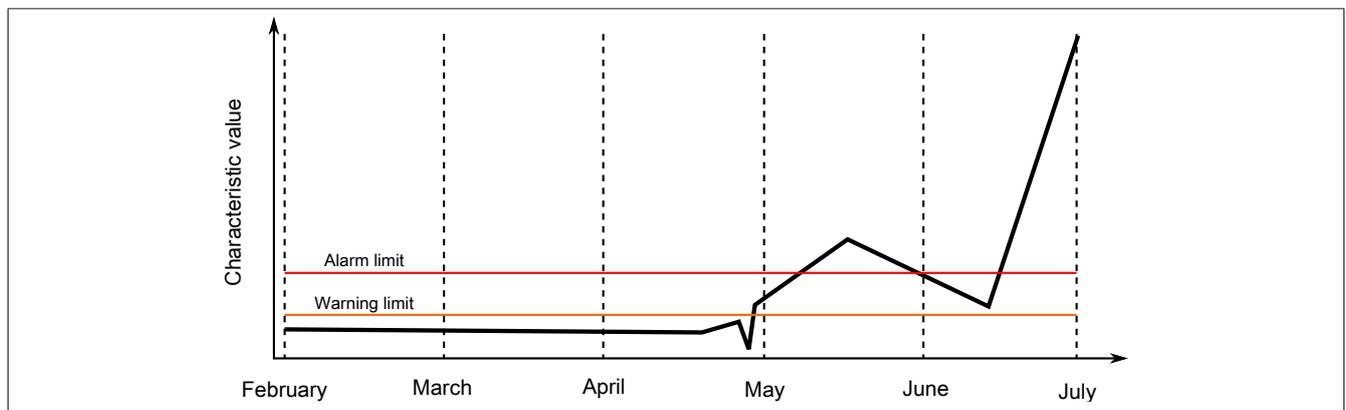
Selecting suitable characteristic values and assessing them over a longer time frame is the basis for effective and successful monitoring of a machine, a process known as "trending". It enables multiple aggregates to be monitored with a relatively low need for technical measuring and a low demand on staff.

Simply setting basic limits as a warning is not enough. Instead, logical correlations with other parameters such as load or speed, or even the shape of the trend curve, are also required.

Characteristic values are thus gathered in the trend curve and compared with norms or values taken from available experience over a long period of time. This trend progression can be used to make a good assessment of the machine's condition. The way these factors change over time indicates whether the condition is worsening, i.e. if damage is beginning to occur.

Recording the measured values in a diagram over a longer period of time shows the status of the characteristic values at the defined warning and alarm thresholds. If these thresholds are exceeded oscillation diagnostics gives the cause of the error so that appropriate maintenance measures can be taken.

Example: Trend curve for a characteristic value



5.3.1 Characteristic values

The following matrix shows the relationship between potential failures and the suitability of the characteristic values provided in the module for error analysis. The potential failures refer to the individual application possibilities.

This assessment is based on an estimation of typical applications.

Application possibilities		Potential failures												
Fan with rigid coupling														
Fan with countershaft														
Fan with belt drive														
Directly coupled pump														
Gearbox														
Gearbox with countershaft														
<p>Frequency of the cause of failure in these Potential failures:</p> <ul style="list-style-type: none"> ■ Frequent ■ Depends on where used ■ Rare to never ■ Not assessed <p>Suitability of the respective characteristic values to monitor the potential failures:</p> <ul style="list-style-type: none"> ■ Very good ■ Good ■ Less good ■ Not assessed 		Imbalance	Orientation	Loose components	Beginning of bearing damage	Advanced bearing damage	Suboptimal lubrication	Attachment of components	Coupling faults	Belt tension and alignment	Defects in flat belts, v-belts and timing belts	Gearbox damage	Electrical errors	
Characteristic values														
PeakHighFrequency														
CrestFactorHighFrequency														
Vdi3832KtHighFrequency														
PeakRaw														
CrestFactorRaw														
SkewnessRaw														
KurtosisRaw														
Vdi3832KtRaw														
RmsHighFrequency														
RmsAccRaw														
RmsVelRaw														
Iso10816														
RmsAccEnvelope														
RmsVelEnvelope														
FrequencyBandRmsVelEnvelope														
FrequencyBandRmsAccEnvelope														
FrequencyBandRmsAccRaw														
FrequencyBandRmsVelRaw														
FrequencyBandNoiseAccEnvelope														
FrequencyBandNoiseVelEnvelope														
FrequencyBandNoiseAccRaw														
FrequencyBandNoiseVelRaw														

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.1 Imbalance

The term imbalance refers to rotating bodies with a mass that is not rotationally symmetrical. In other words, the center of mass is not on the axis of rotation

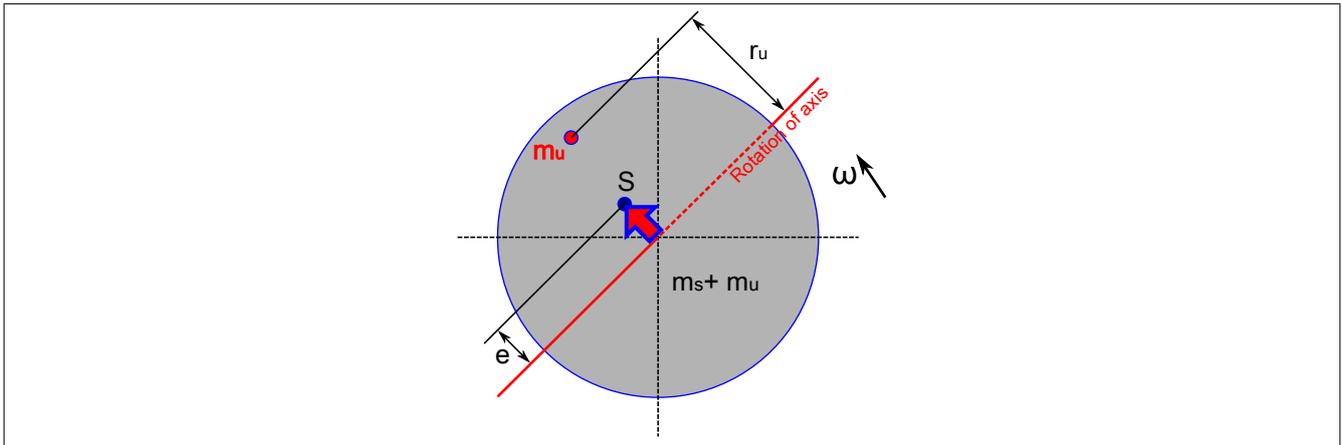


Figure 56: Representative sketch of an imbalance

Imbalance leads to vibrations and increased wear, particularly at high speeds, which is why counterweights are applied to compensate for this as counterbalance. In practice, it is very rarely possible to fully compensate for this, meaning every rotating body has some residual imbalance.

The centrifugal force caused by imbalance is dependent on the square speed and thus has a more significant effect at higher speeds (higher speed frequency). On a spectrum, therefore, the frequency line denoting speed is significantly higher.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Imbalance	1 x f_n	-	Only severe imbalance causes peaks in the speed that show up in the envelope spectrum.

f_n ... Nominal speed

Information:

The module can only measure the intensity of the imbalance, not its position on the shaft. For this reason, it cannot be used for balancing.

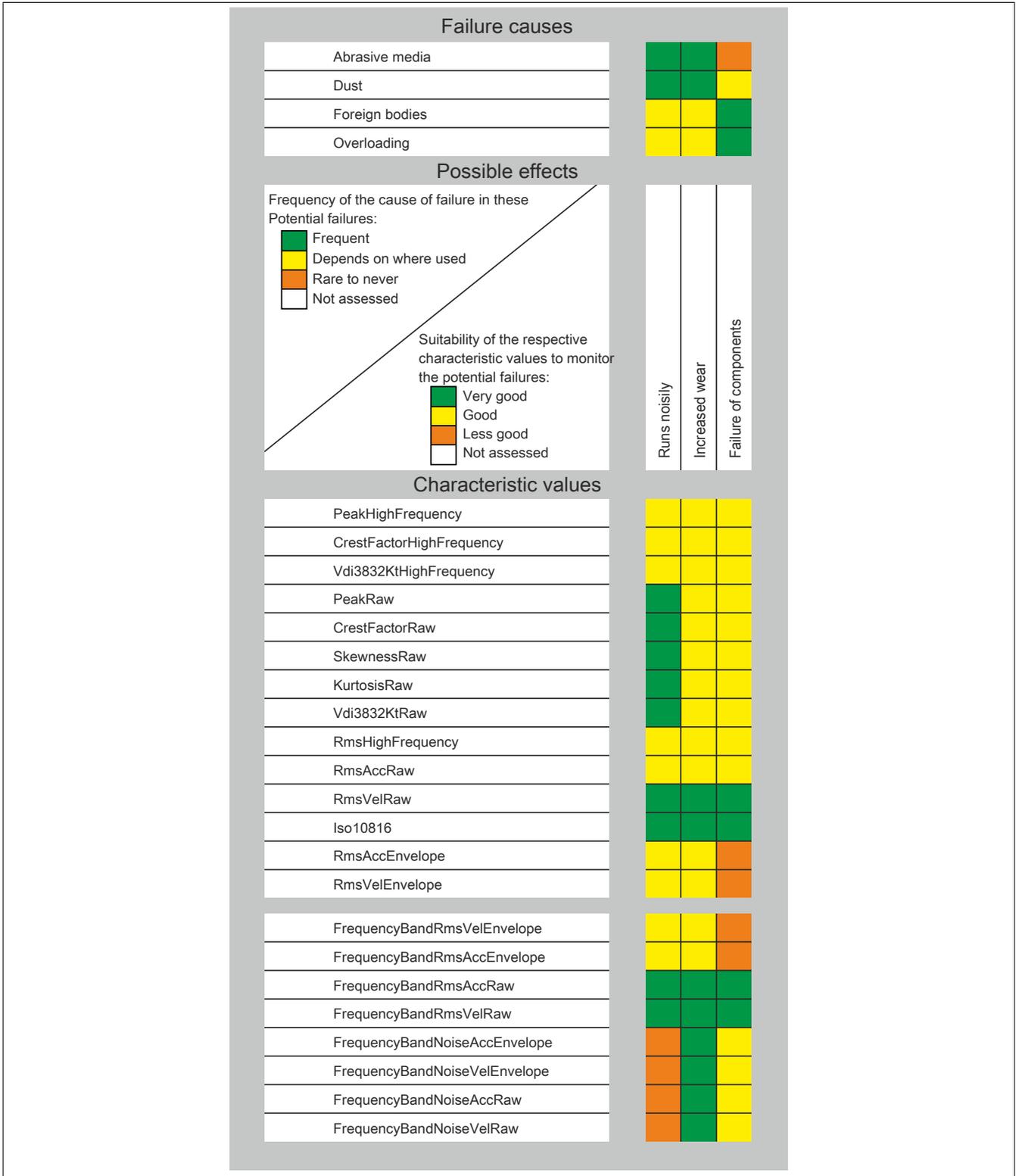


Figure 57: Failure causes and effects of imbalance

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.2 Misalignment

During operation, a number of different factors can cause machine parts to fail to align or stop aligning with each other.

Shafts should rotate around a linear axis at the coupling positions, so that the restoring forces at the coupling position and the loading forces in the bearings are kept as low as possible. Misalignment causes increased vibrations and wear.

Misalignment usually consists of a parallel and an angular misalignment. In the event of a severe parallel misalignment, strongly increased values can be observed in the area of double speed.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Misalignment in coupling	1 x fn, 2 x fn (sometimes 3 x fn, 4 x fn)	-	With parallel misalignment, usually only "1 x fn" occurs.

fn ... Nominal speed

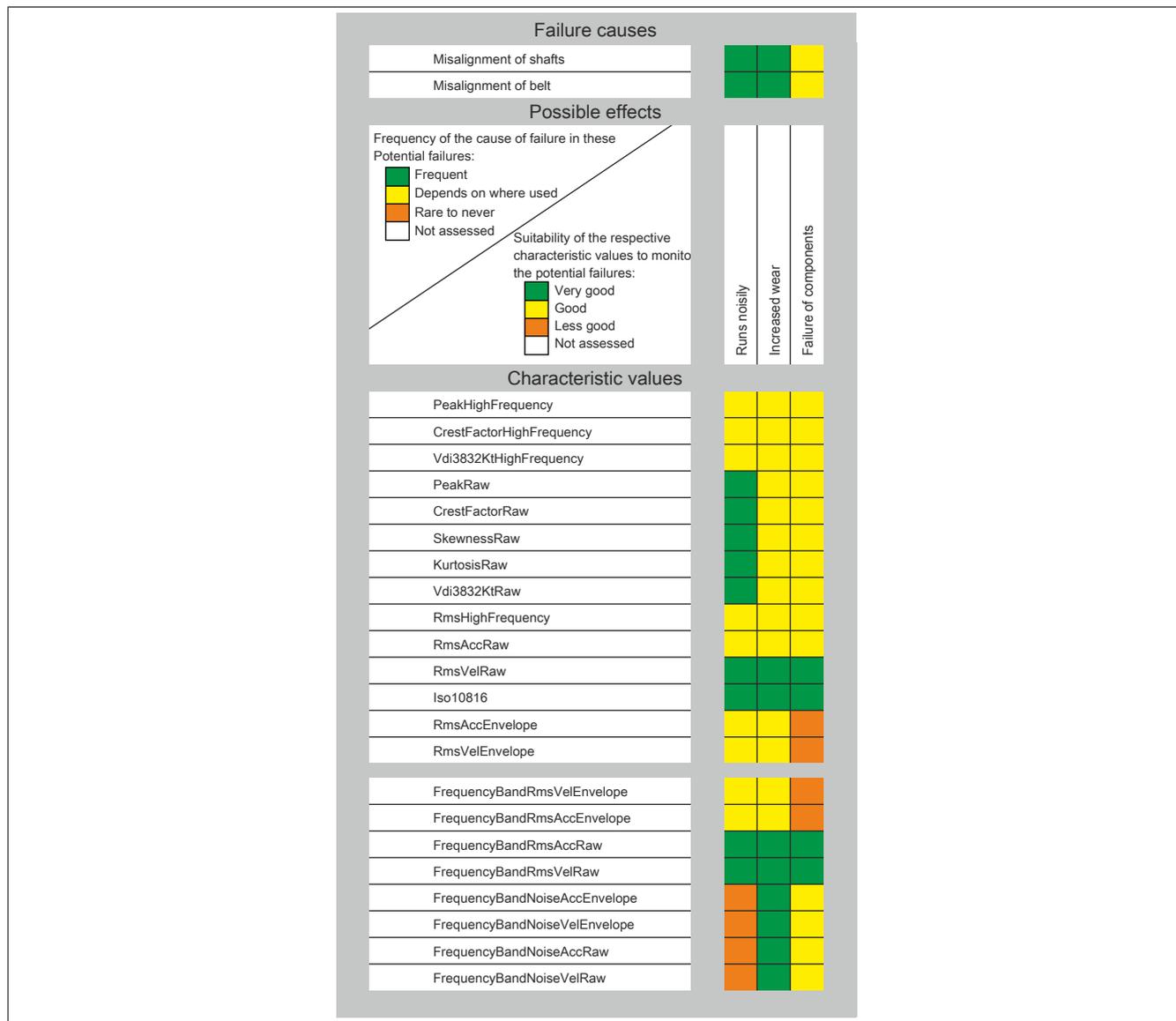


Figure 58: Failure causes and effects of an alignment failure

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.3 Belt damage

Belts often cause various problems during operation. A belt can be damaged if the tension is too low or too high or if the belt is incorrectly aligned. If the damaged belt part rolls over the pulley, this causes impacts that can be measured.

5.3.2.3.1 Flat belts and V-belts

With flat belts and V-belts, torque is transferred between the belt and the pulley through the contact surface. Belt drives are not very sensitive with regard to alignment but nevertheless lead to vibration development and above all to increased wear and energy consumption if the deviation is too high.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Defective belt	1 x fr, 2 x fr, 3 x fr ...	1 x fn1, 1 x fn2, 1 x fr	Belt frequencies usually occur in both spectra.

fr ... Belt speed

fn1 ... Nominal speed of shaft 1

fn2 ... Nominal speed of shaft 2

Poorly aligned or incorrectly tensioned belts can also cause severe stress on the bearing points. The increased vibration also causes heavier wear on belts and pulleys.

The vibrations are clearly identifiable at the bearing points of the individual shafts.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Misalignment of belt	1 x fn, 2 x fn 1 x fr	1 x fn	If the belts strike against the pulley laterally, impacts can also occur with speed and belt frequency.

fn ... Nominal speed

fr ... Belt speed

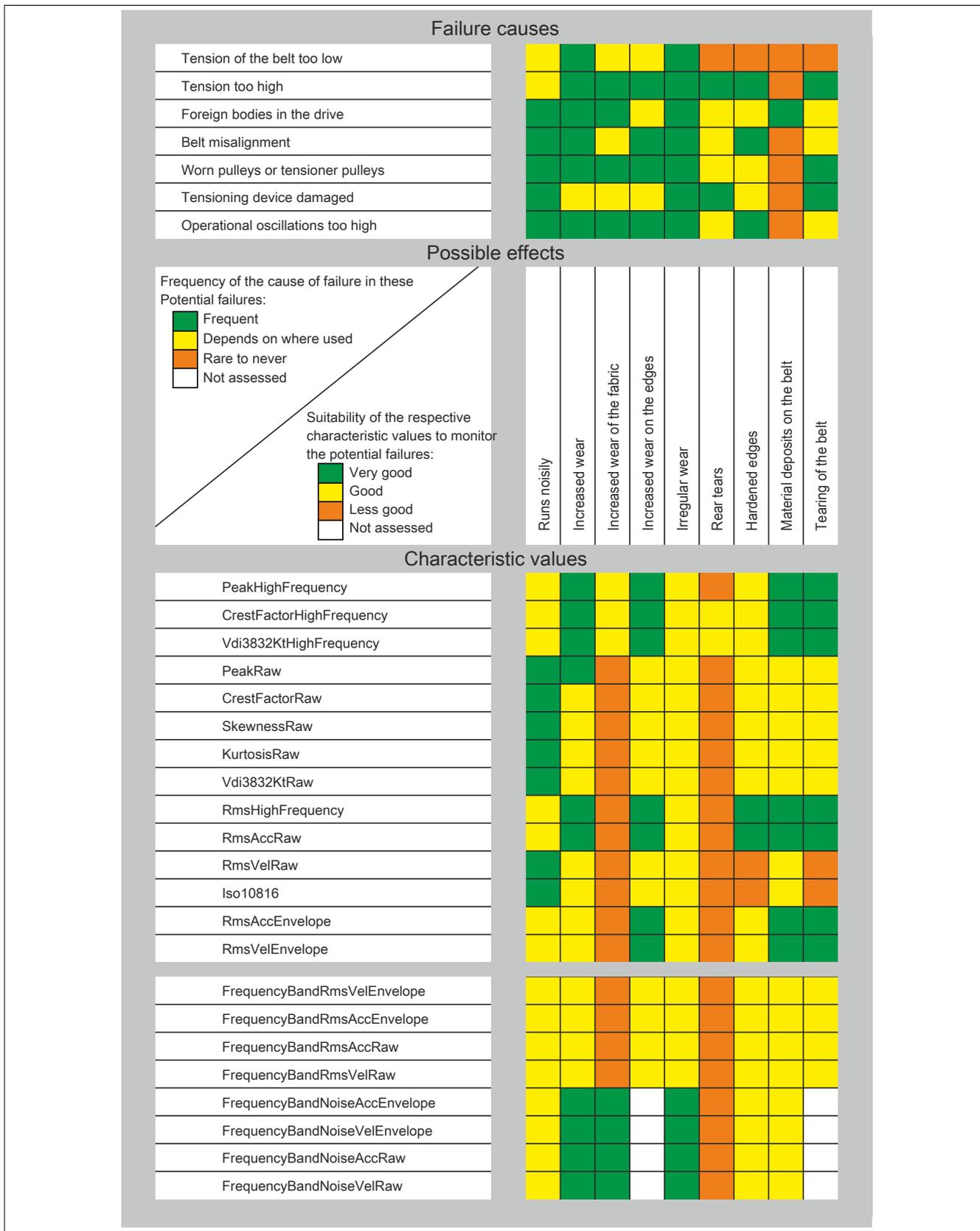


Figure 59: Failure causes and effects of belt damage in flat belts and V-belts

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.3.2 Toothed belt

With toothed belts, torque is transferred via the meshing of the teeth. In addition to the already known failure causes, there are also the failures caused by the toothing.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Toothed belt damage	1 x fr	1 x fr, 1 x fzn	The tooth meshing frequencies in combination with the respective speed are clearly visible.

fr ... Belt speed
fzn ... Meshing frequency

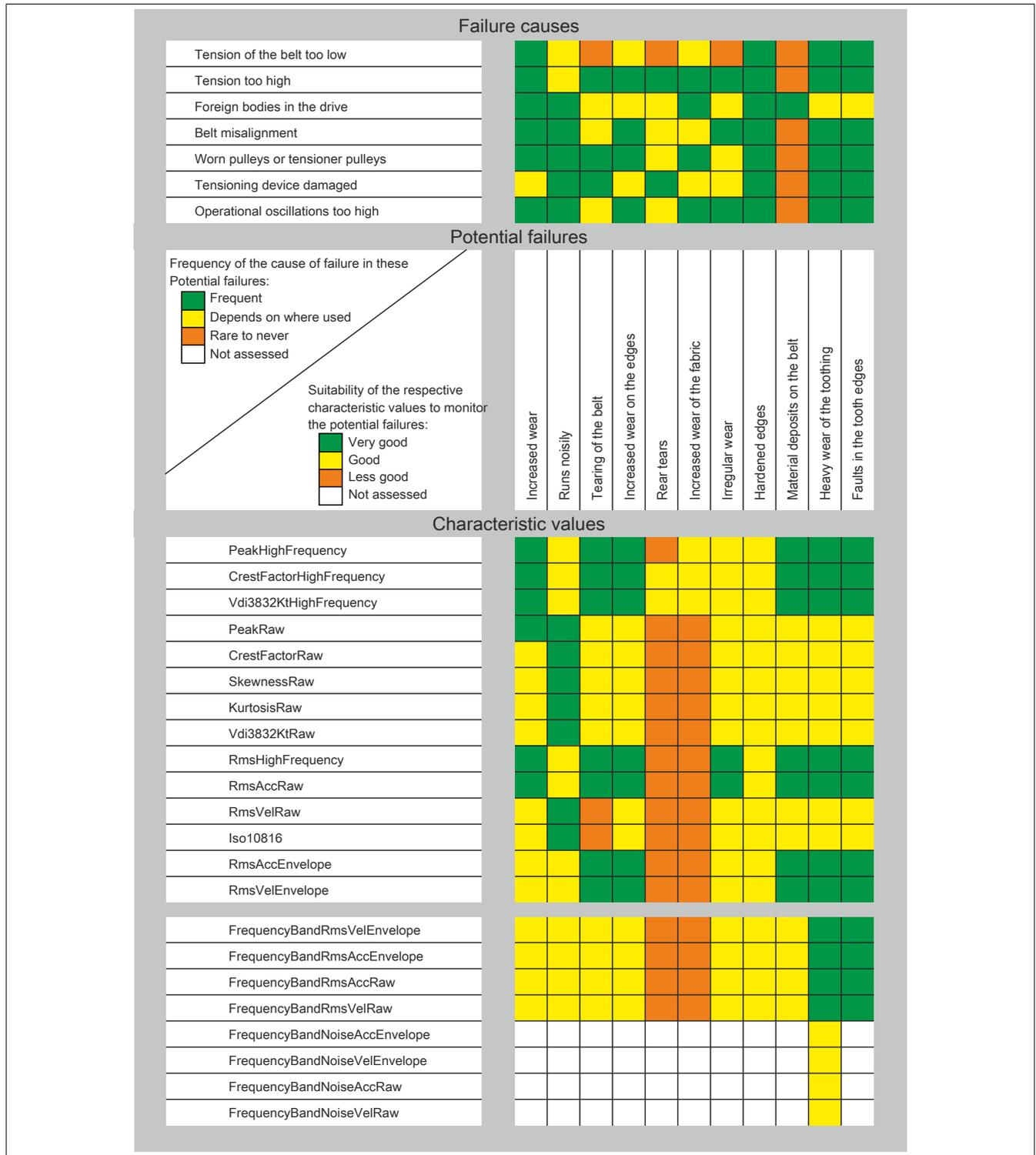


Figure 60: Frequency of failure indicators on toothed belts

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.4 Loose or striking parts

If individual parts in the machine are loose, they can cause unwanted oscillations. When individual parts strike housings or attachment parts, these appear very similar. As a result, the two causes of damage cannot be analyzed separately.

In addition, components strike their counterparts on each revolution. This in turn causes the attachment parts to vibrate at their natural frequency. Envelope analysis can be used to separate the causes of impact.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Loose parts, parts striking each other	(1 x fn)	1 x fn	There is usually one strike per revolution. If an envelope occurs for the load, a double frequency can be determined.

fn ... Nominal speed

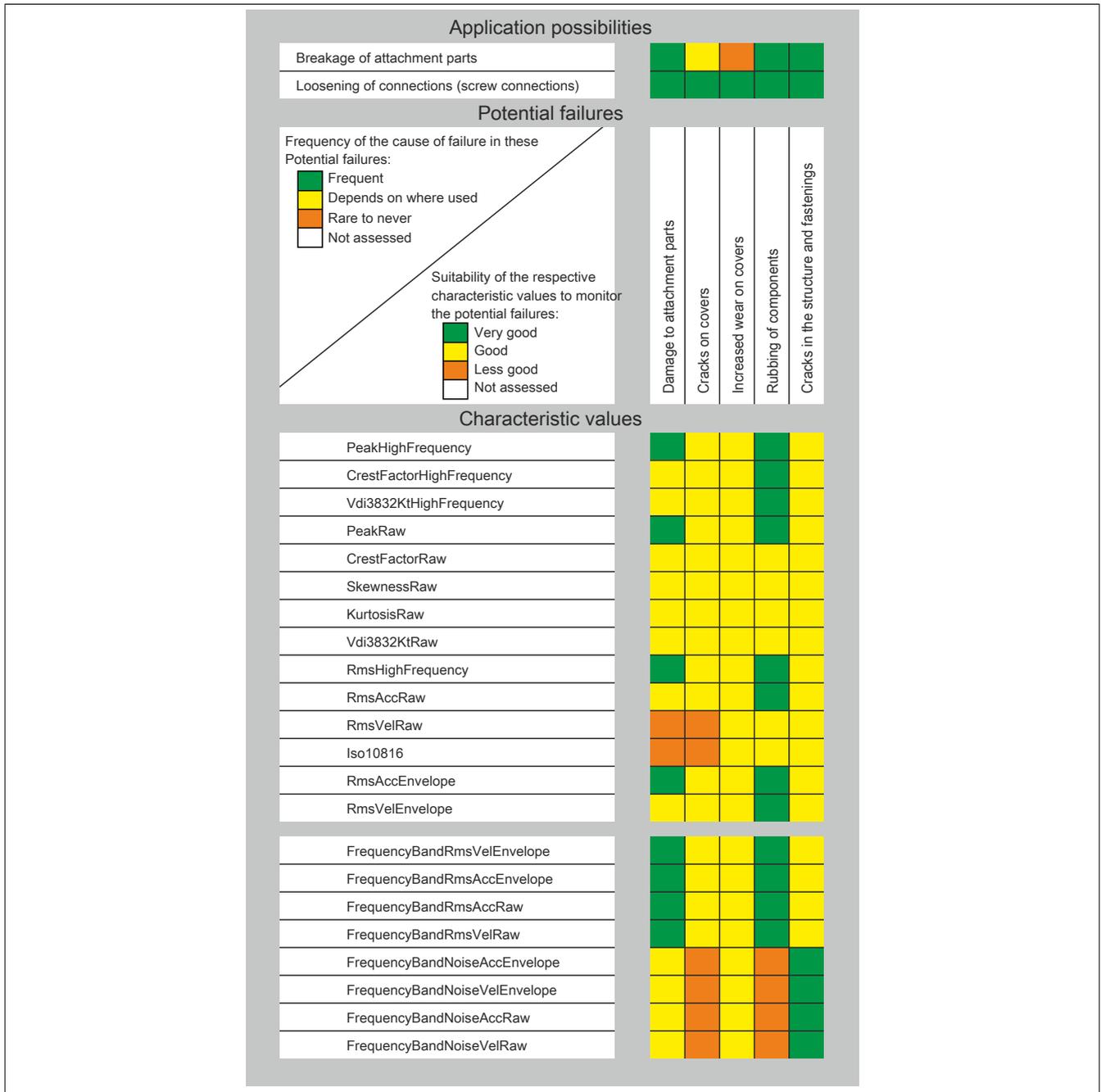


Figure 61: Causes of errors and symptoms of loose or striking parts

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.5 Slide bearing damage

In a structure-borne sound measurement, the typical damage frequencies for a slide bearing do not manifest themselves until a very late stage. For this reason, this method is less suitable for early detection.

5.3.2.6 Roller bearing damage

Many types of bearing damage are caused by imprecisions in the bearing surface such as material damage or micro-cracks. These pittings (i.e. material damage or micro-cracks) are rolled over by the roller elements cause impacts on the roller bearing and its attachment parts.

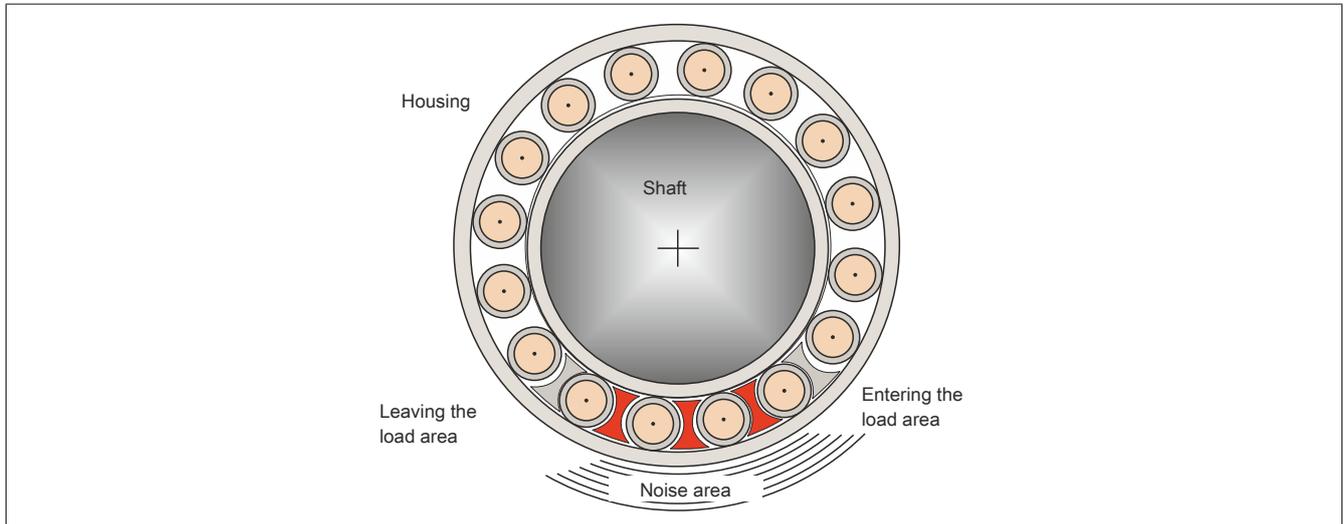


Figure 62: Roll-over processes in the bearing

The mechanism is very similar to the striking of a bell: The clapper strikes the body of the bell, and the bell starts vibrating at its natural frequency.

In the case of the bearing, each time the roller moves over the damaged area it is like the striking of the clapper, and the roller parts and attachment parts start to vibrate.

These very small oscillations can be measured as a modulation or superposition of the excitation frequency on the surface of the bearing.

Appropriate analysis methods such as formation of the envelope can separate the superposition to make the roll-over frequencies of the bearing clearly discernible.

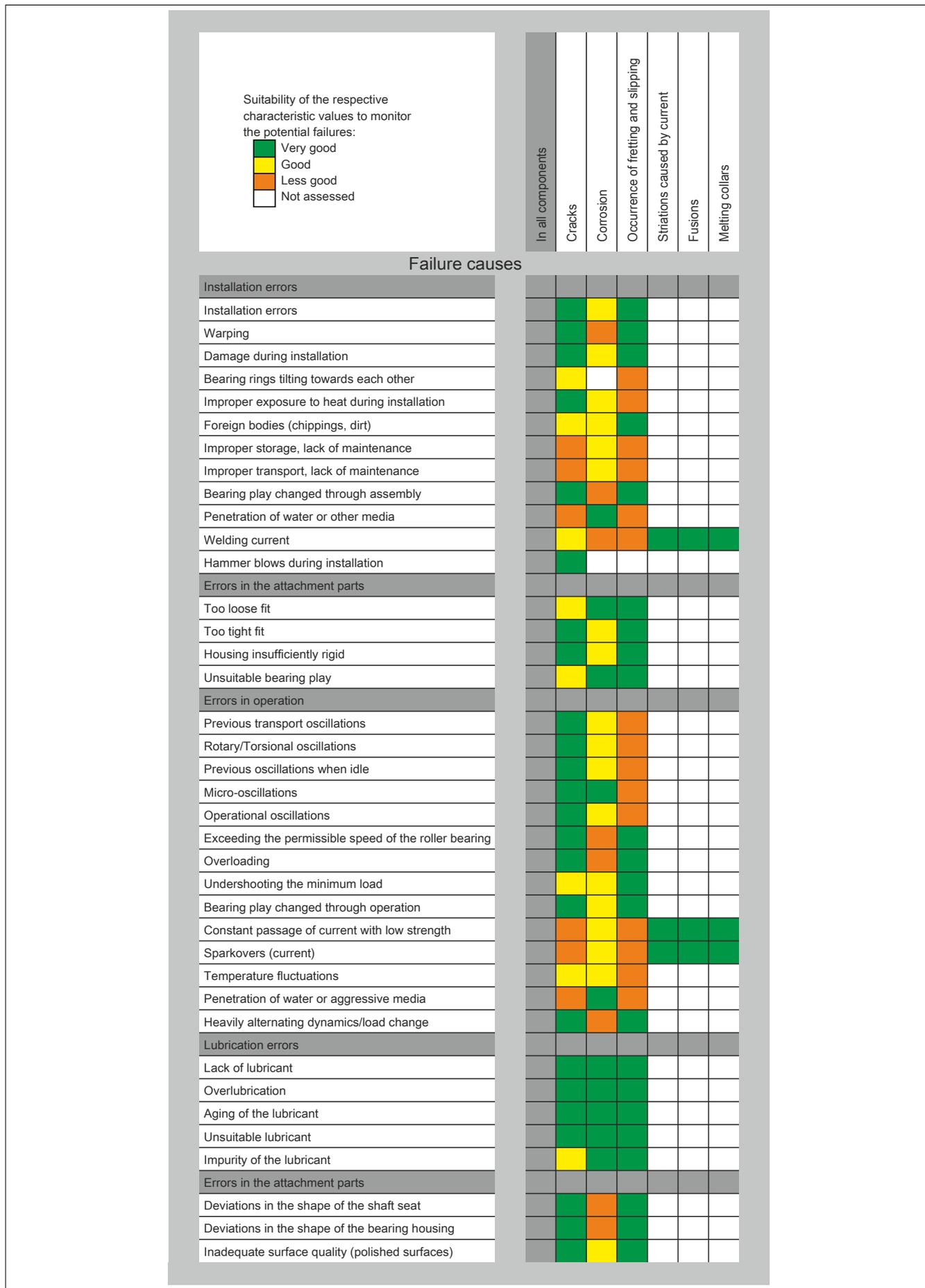


Figure 63: Frequency of failure indicators on roller bearings

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

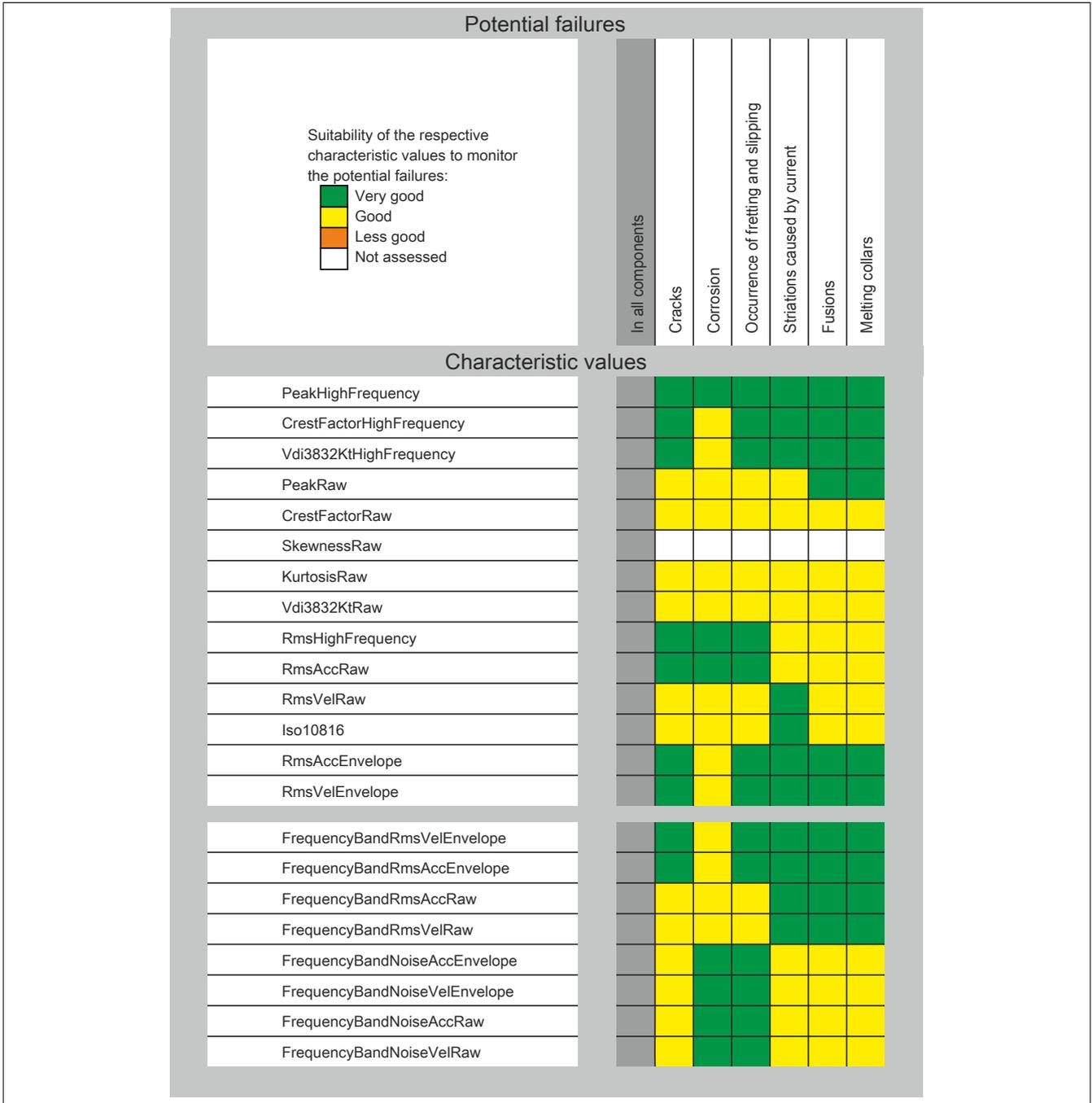


Figure 64: Frequency of failure indicators on roller bearings

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.6.1 Typical outer and inner ring damage

Outer ring damage

In most cases the outer ring remains stationary while the inner ring turns. This gives a clearly defined fixed load zone. Most damage occurs in this load zone. If pitting or other surface damage forms, vibrations occur when the roller element moves over it that can be measured on the housing components.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Outer ring damage	(1 x fa)	1 x fa, 2 x fa, 3 x fa ...	With advanced damage, the outer ring frequencies can also be seen in the raw signal spectrum

fa ... Frequency of the outer ring damage

Inner ring damage

Any inner ring damage that occurs travels with the rotating shaft. Due to the different rotary speeds of the revolving roller elements and the inner ring, pronounced modulations occur. As a result, inner ring damage frequencies are usually shown with sidebands in the spectrum.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Inner ring damage	(1 x fi)	i x fi ± i x fn	Inner ring damage generally appears based on modulation with pronounced sidebands for the speed.

fi ... Frequency of the inner ring damage

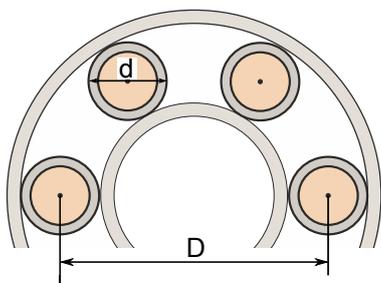
fn ... Nominal speed

Calculating damage frequencies

Details of bearing damage frequencies are normally provided by the manufacturer and can be taken from the data sheets for the bearings.

However, they can also be calculated easily. The following values are necessary for this.

N	Speed in rpm
n_b	Number of roller elements
d	Diameter of roller elements
β_c	Pressure angle
D	Roller ring diameter



Formula for calculating the inner ring damage frequency

$$f_i = \frac{n_b \cdot N}{2 \cdot 60} \cdot \left[1 + \frac{d}{D} \cdot \cos \beta_c \right]$$

Figure 65: Calculation of the inner ring damage frequency

fi ... Inner ring damage frequency

Formula for calculating the outer ring damage frequency

$$f_a = \frac{n_b \cdot N}{2 \cdot 60} \cdot \left[1 - \frac{d}{D} \cdot \cos \beta_c \right]$$

Figure 66: Calculation of the outer ring damage frequency

fa ... Outer ring damage frequency

Formula for calculating the roller element damage frequency

Damage impacting the individual pitch line:

$$f_{ew} = \frac{1}{2} \cdot \frac{D \cdot N}{d \cdot 60} \cdot \left[1 - \left(\frac{d}{D} \cdot \cos \beta_c \right)^2 \right]$$

Figure 67: Calculating the damage frequency on one roller element

f_{ew} ... Damage frequency of the individual roller element

Damage impacting both pitch lines:

$$f_w = \frac{D \cdot N}{d \cdot 60} \cdot \left[1 - \left(\frac{d}{D} \cdot \cos \beta_c \right)^2 \right]$$

Figure 68: Calculating the damage frequency on both roller elements

f_w ... Damage frequency of both roller elements

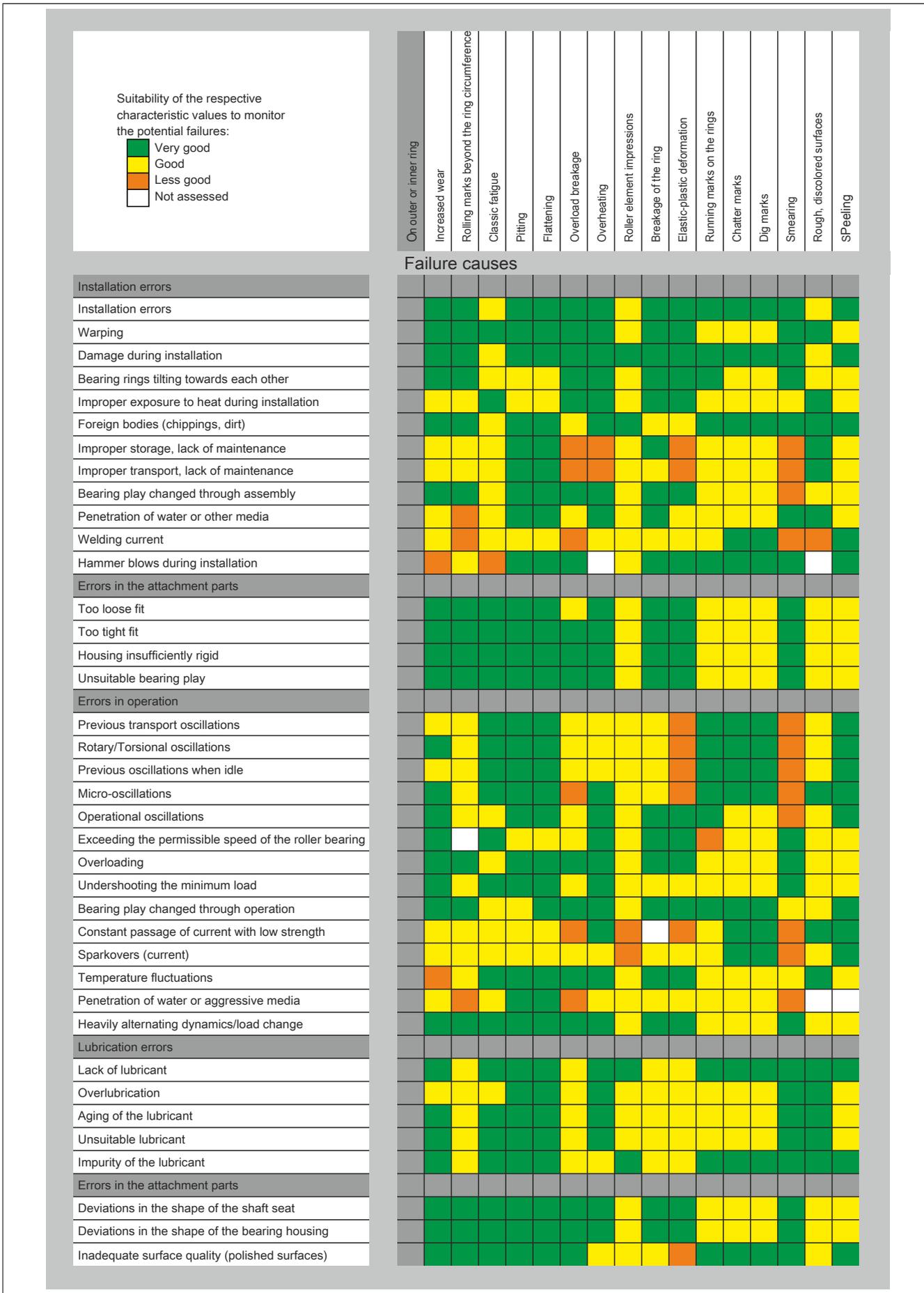


Figure 69: Frequency of failure indicators on roller bearings

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

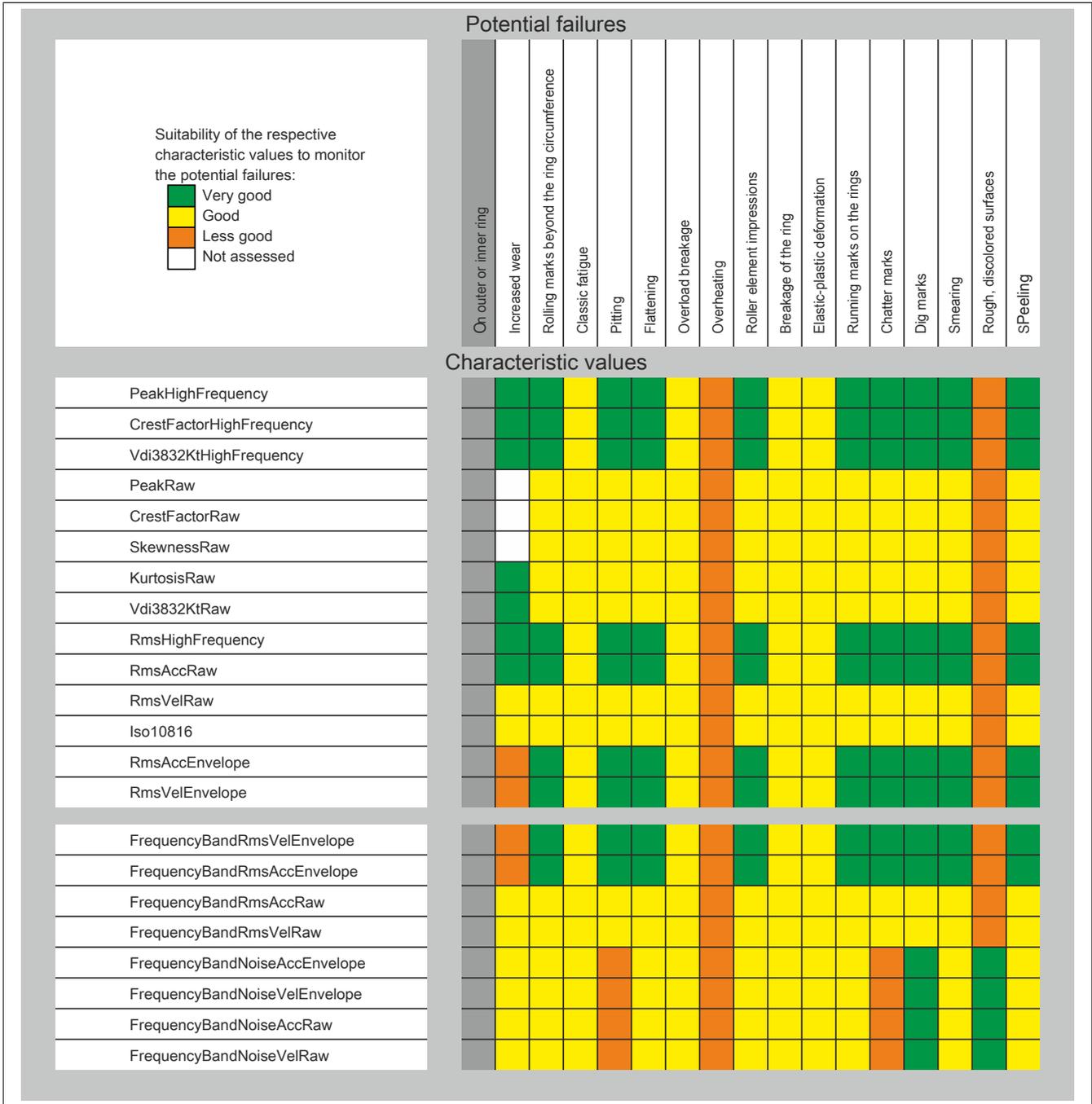


Figure 70: Frequency of failure indicators on roller bearings

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.6.2 Typical cage and roller element damage

Cage damage

Cage frequencies often occur in electric motors, especially where bearings with increased bearing play are used. If this is not used, oscillation of the cage often occurs. This is then reflected in increased operating noise.

Roller element damage

Damage to rolling elements without damage to the outer or inner ring occurs extremely rarely; the individual features are therefore only given as examples.

If the roller element is damaged, an impact occurs either on the inner ring, outer ring or both. This makes it possible to detect damage to rolling elements caused by the rollover frequency or twice the frequency. For this reason, the harmonic should be included in the characteristic value calculation where possible to detect roller element damage.

For the damage symptoms, see "Typical outer and inner ring damage" on page 140.

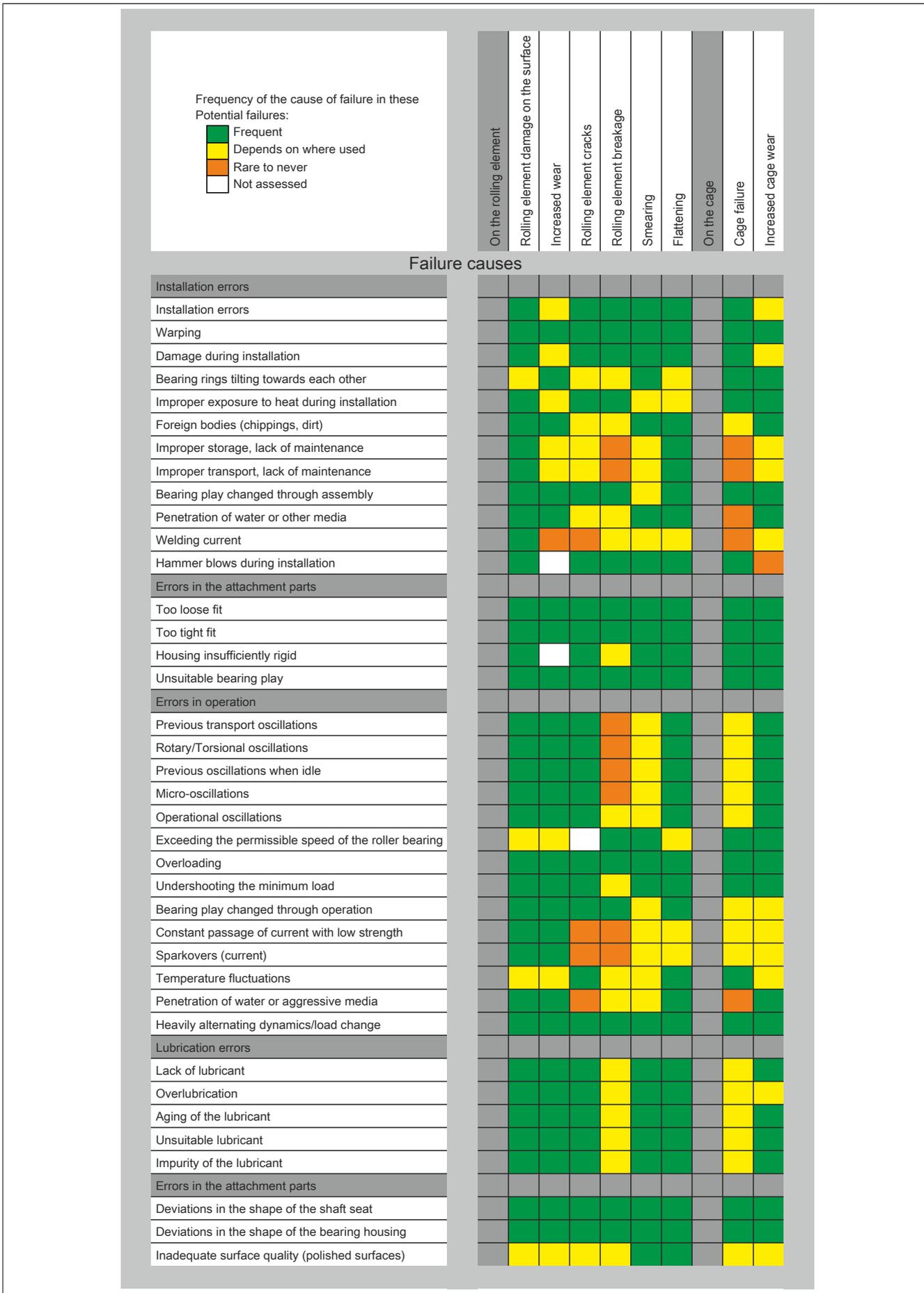


Figure 71: Frequency of failure indicators on roller bearings

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

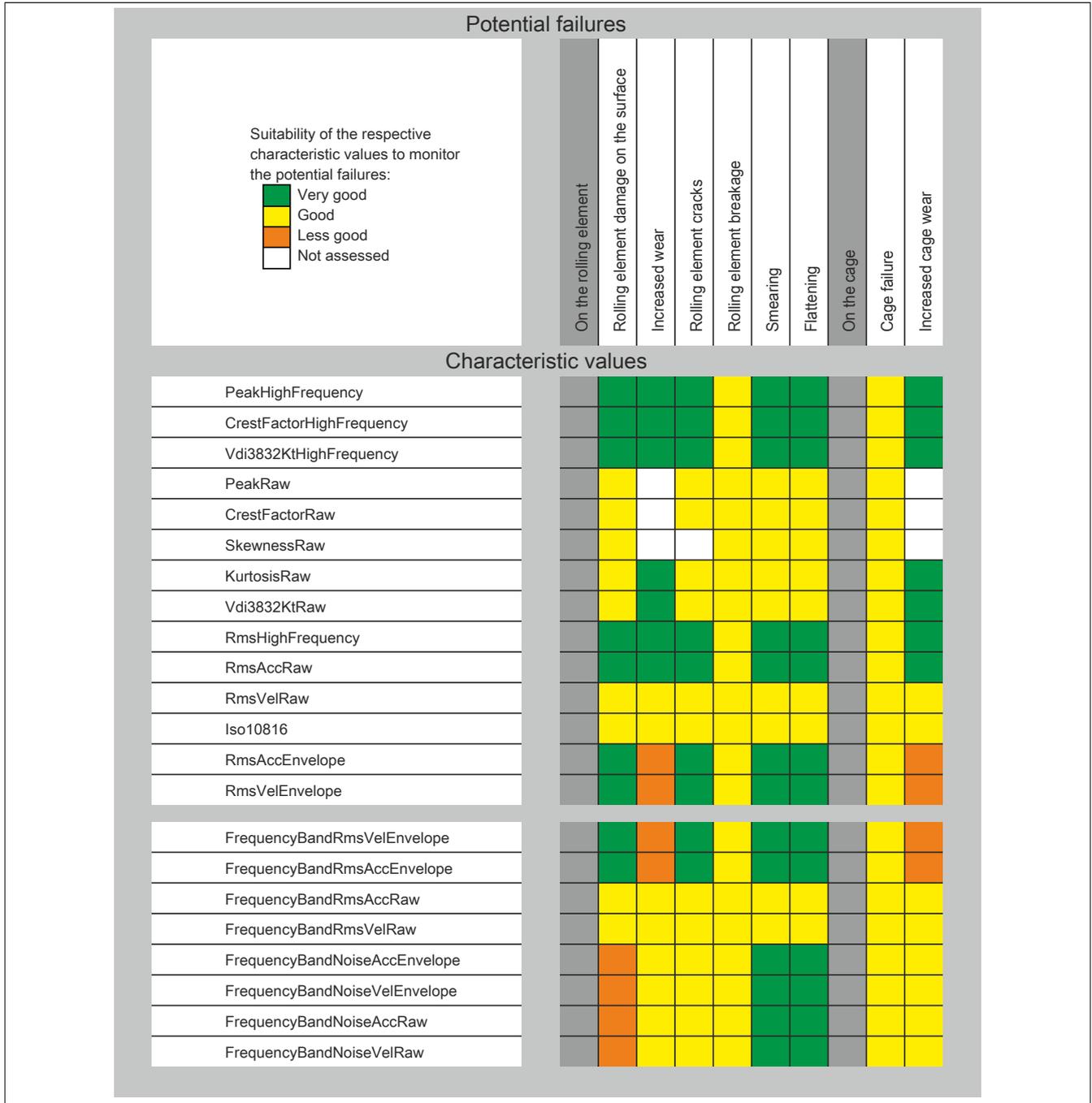


Figure 72: Frequency of failure indicators on roller bearings

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.7 Gear damage

DIN 3979 provides a description of gear errors and defects. The most common errors in individual gearboxes are described below.

The complexity of the machine dynamics must be taken into consideration in individual applications.

Gear ratio

The speed ratio must always be converted according to the gear ratio. The frequencies on the gearwheels are always based on the speed of the respective axis.

5.3.2.7.1 Manufacturing defects

Oscillations due to manufacturing always occur in a gearbox. Typical manufacturing defects are pitch errors, profile deviations, concentricity and spacing errors.

Depending on the gearwheel pairing, these individual defects can intensify or become less intense depending on how the defects affect each other. The interplay of the individual defects is also responsible for the overall oscillation behavior.

Pitch error is a frequently occurring defect and is used as an example here. Depending on whether the pitch error is positive or negative, it is intensified or compensated by an existing load. The effects on the oscillation behavior depend very much on the rigidity of the gearwheel. If there is a pitch error, a shock occurs that can be measured effectively.

In addition to pitch errors, all form and dimensional gear errors lead to oscillations.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Meshing frequencies	$1 \times fz$	$1 \times fz$	Gear frequencies depend on the respective geometric ratios of the wheels but can be precisely calculated in any case.

fz ... Meshing frequency

5.3.2.7.2 Defects caused by wear

If form and dimensional deviations occur during operation, observing the trend can lead to an appropriate level of confidence when performing diagnostics.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Deterioration	$1 \times fz$	$1 \times fz \pm i \times fn$	Geometric errors increasingly appear additionally with sidebands for the tooth meshing frequency.

fz ... Meshing frequency

fn ... Nominal speed

5.3.2.7.3 Wobbling

If the toothed belt axis and the rotation axis are not parallel, the phenomenon called wobbling occurs. This causes 2 flank errors per revolution. Depending on the position, there is one transmission on the inner edge and one on the outer edge of the toothed wheel.

The doubled speed frequency is clearly evident in the frequency spectrum.

Failure mode	Frequency in raw signal spectrum	Frequency in envelope spectrum	Comment
Wobbling	$1 \times fn, 2 \times fn$	$1 \times fz, 2 \times fz$	Wobbling movements manifest as doubled speed frequency and are usually accompanied by sidebands.

fn ... Nominal speed

fz ... Meshing frequency

5.3.2.7.4 Cyclic running errors and axial distance errors

In DIN 3960, axial distance errors are defined as a deviation between the target/actual value.

An error that changes the distance between two axes changes the way the gears mesh and has a negative effect on the overlap ratio.

Even small pitch errors lead to increased noise in the gearbox.

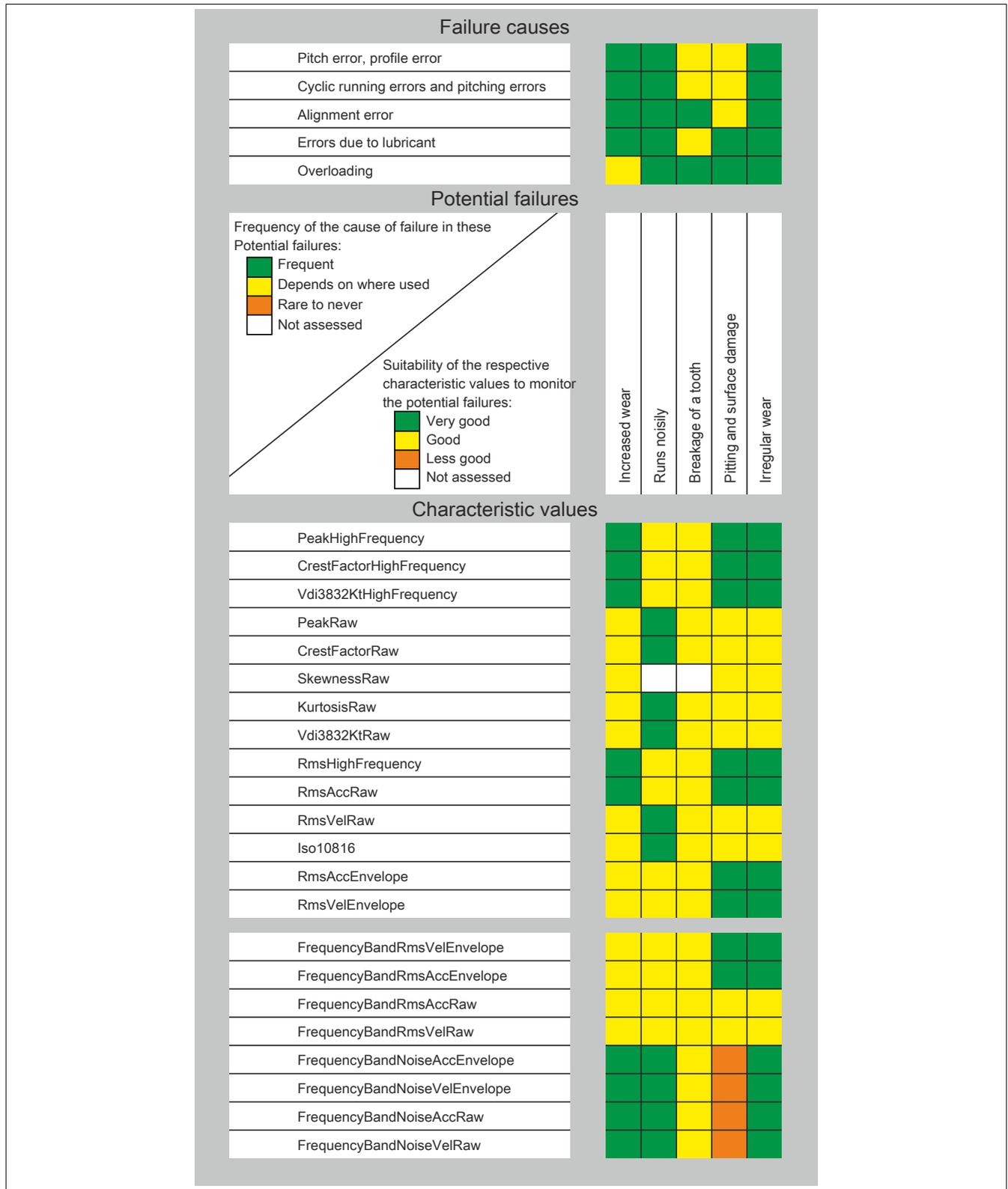


Figure 73: Frequency of failure indicators for gearbox damage

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.2.8 Electrical errors

Occasionally, bridges in the rotor bars or short circuit rings occur. They occur as a result of overloading as well as aging and the ongoing oscillation load. This leads to a very uneven distribution of the induction current in the rotor.

Failure causes				
Overloading	Green	Green	Green	Yellow
Aging	Green	Green	Green	Yellow
Dust entry	White	Green	Yellow	White
Foreign bodies	Yellow	Green	Yellow	White
Short circuits in coil	Yellow	Green	Yellow	Green

Potential failures				
Frequency of the cause of failure in these Potential failures: Green: Frequent Yellow: Depends on where used Orange: Rare to never White: Not assessed				
Suitability of the respective characteristic values to monitor the potential failures: Green: Very good Yellow: Good Orange: Less good White: Not assessed				
Loose rods, rod breakage	Green	Green	Green	Green
Increased running noise	Green	Green	Green	Green
Break in short circuit ring	Green	Green	Green	Green
Magnetic imbalance	Green	Green	Green	Green

Characteristic values				
PeakHighFrequency	Green	Yellow	Yellow	Yellow
CrestFactorHighFrequency	Green	Yellow	Yellow	Yellow
Vdi3832KtHighFrequency	Green	Yellow	Yellow	Yellow
PeakRaw	Yellow	Yellow	Yellow	Green
CrestFactorRaw	Yellow	Yellow	Yellow	Green
SkewnessRaw	White	White	White	Green
KurtosisRaw	Yellow	Yellow	Yellow	Green
Vdi3832KtRaw	Yellow	Yellow	Yellow	Green
RmsHighFrequency	Green	Green	Yellow	Yellow
RmsAccRaw	Green	Green	Yellow	Yellow
RmsVelRaw	Yellow	Green	Yellow	Green
Iso10816	Yellow	Green	Yellow	Green
RmsAccEnvelope	Green	Yellow	Yellow	Yellow
RmsVelEnvelope	Green	Yellow	Yellow	Yellow
FrequencyBandRmsVelEnvelope	Green	Yellow	Yellow	Yellow
FrequencyBandRmsAccEnvelope	Green	Yellow	Yellow	Yellow
FrequencyBandRmsAccRaw	Green	Yellow	Yellow	Green
FrequencyBandRmsVelRaw	Green	Yellow	Yellow	Green
FrequencyBandNoiseAccEnvelope	White	White	White	White
FrequencyBandNoiseVelEnvelope	White	White	White	White
FrequencyBandNoiseAccRaw	White	White	White	White
FrequencyBandNoiseVelRaw	White	White	White	White

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.3.1 Fan with rigid coupling

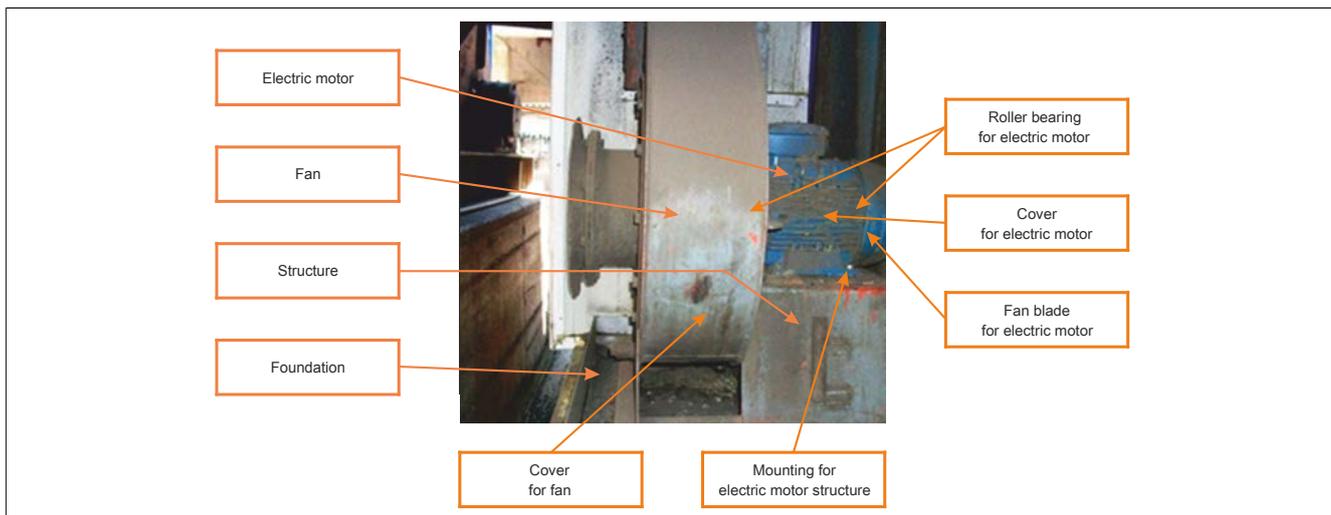


Figure 75: Drive unit with fan

Condition monitoring solution

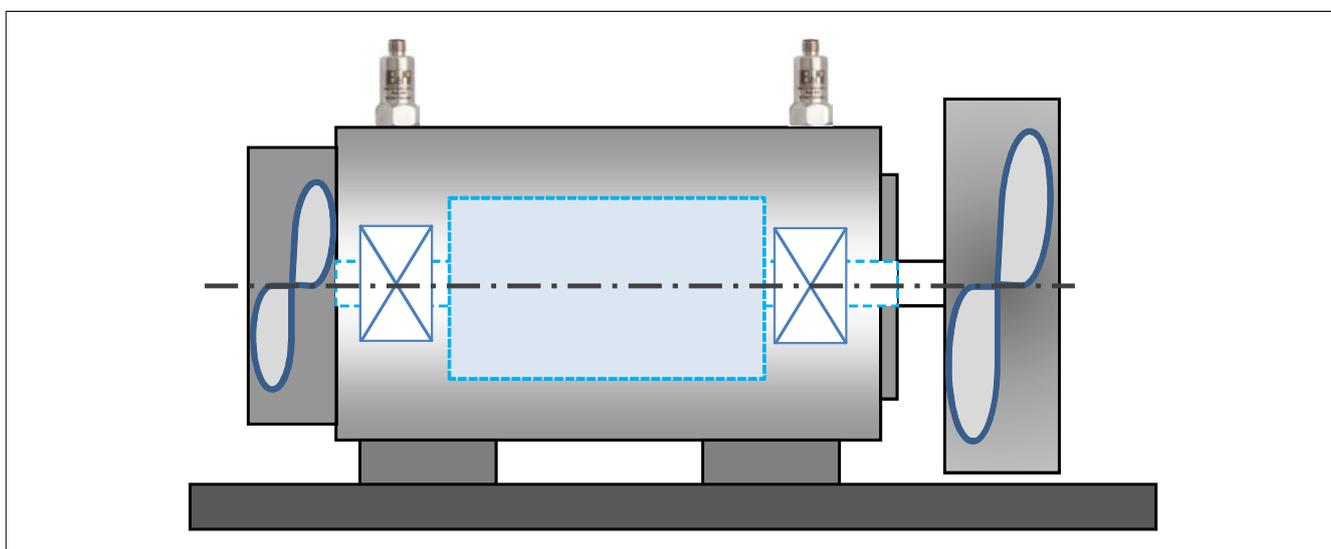
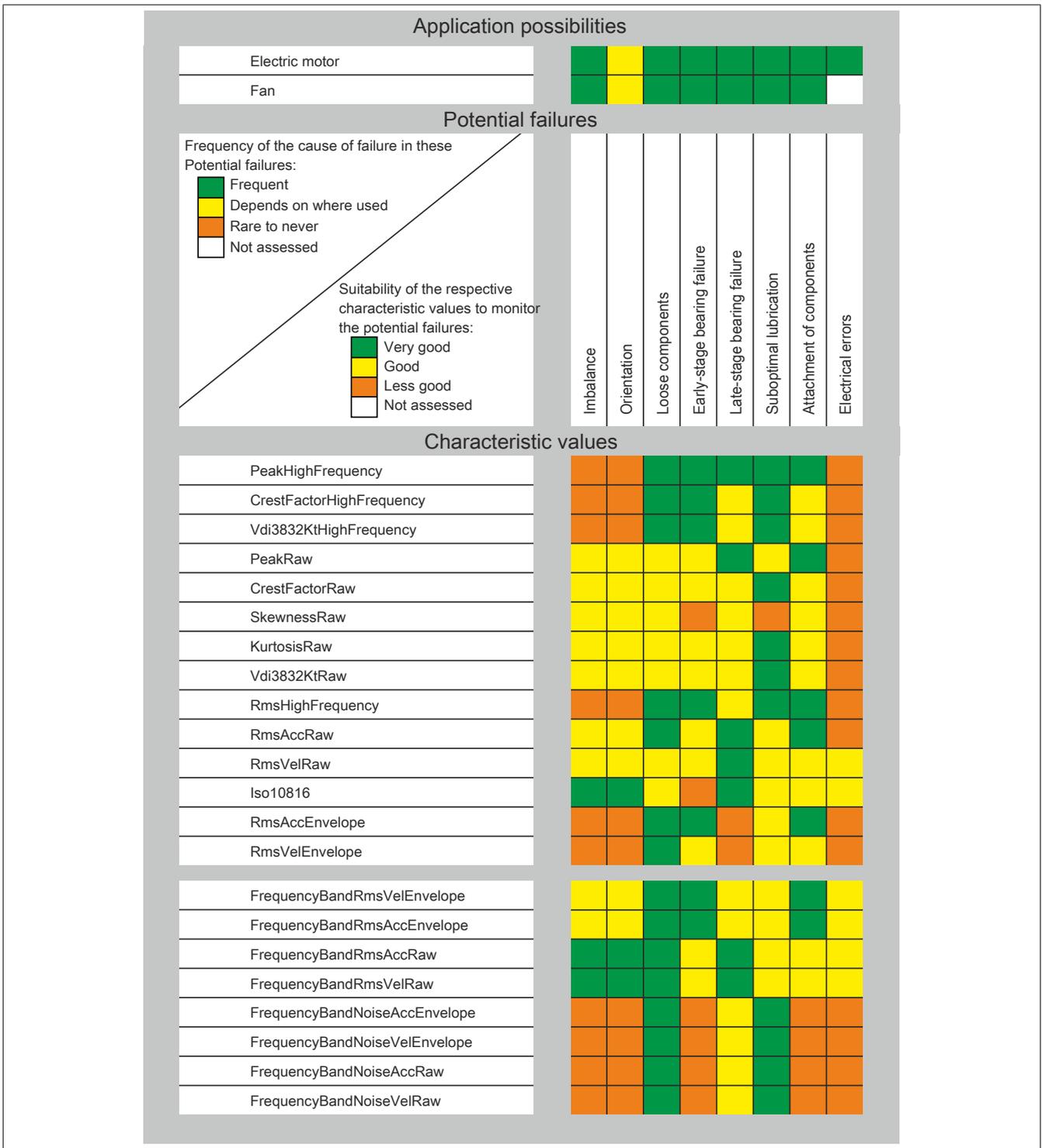


Figure 76: System diagram - Drive unit with fan

Sensor usage:

Number of sensors	Usually 2 sensors. One sensor is sufficient for smaller drive units.
Sensor installation	Preferably vertical. Horizontal installation is also possible, if necessary.

Fan with rigid coupling - Common problems:



For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.3.2 Fan with countershaft

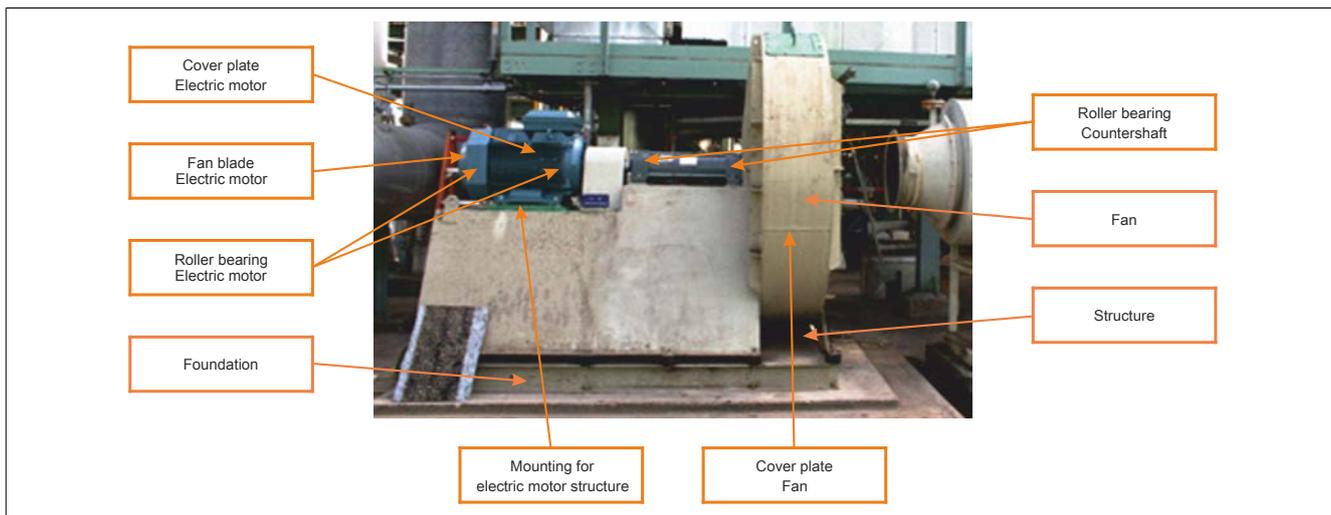


Figure 77: Drive unit with fan

Condition monitoring solution

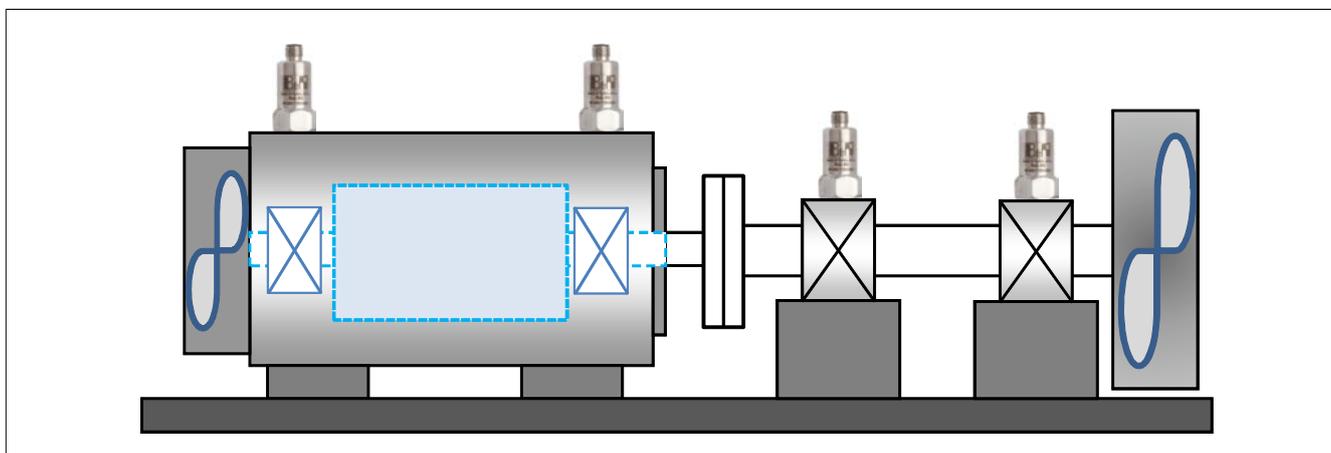
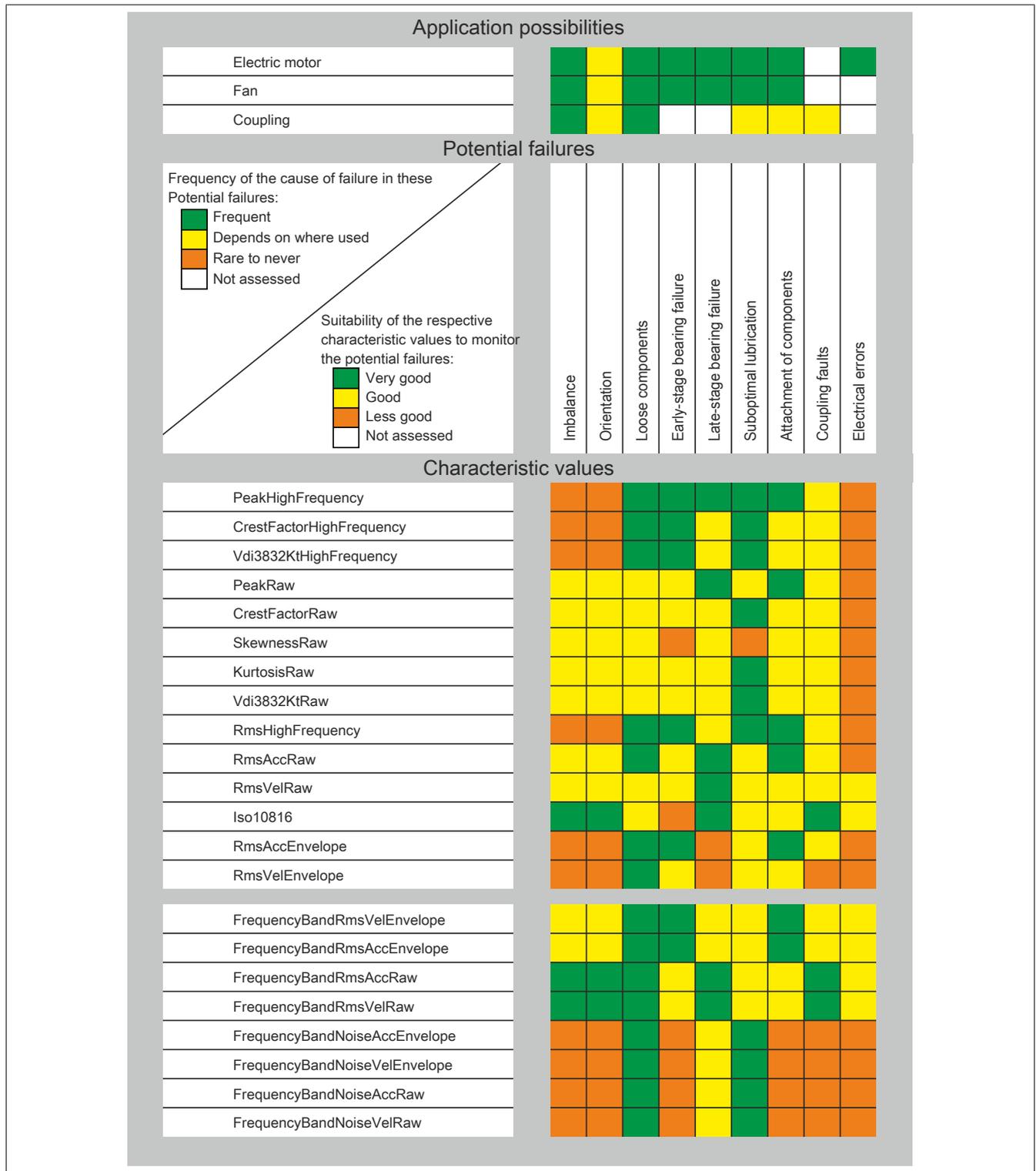


Figure 78: System diagram - Drive unit with countershaft and fan

Sensor usage:

Number of sensors	Usually 4 sensors. One sensor is sufficient for smaller drive units.
Sensor installation	Preferably vertical. Horizontal installation is also possible, if necessary.

Fan with countershaft - Common problems:



For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.3.3 Fan with belt drive

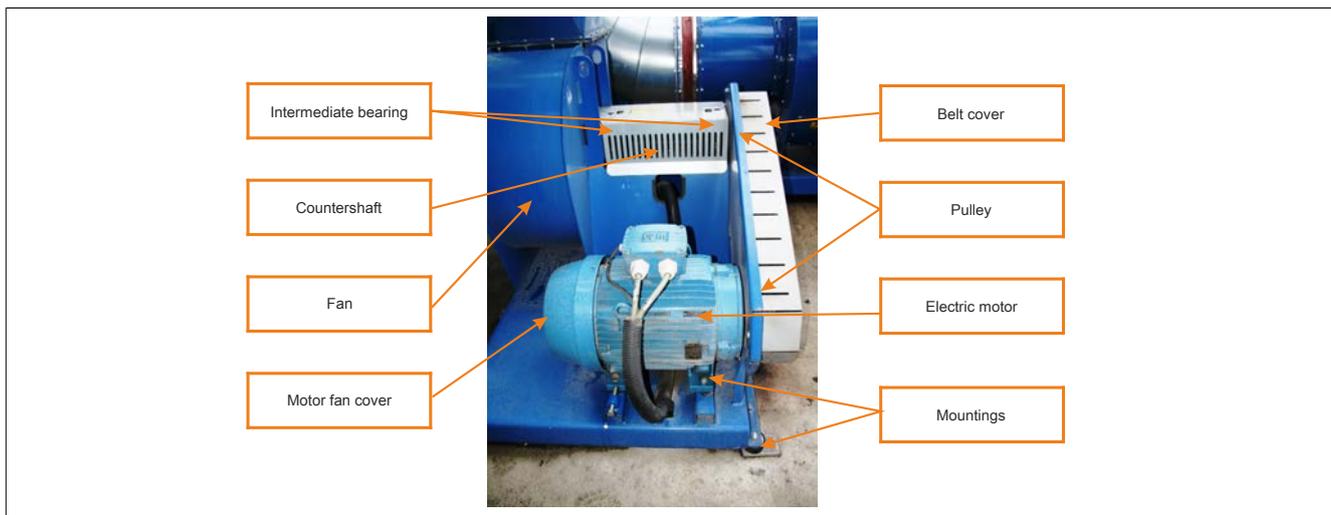


Figure 79: Structure of a fan with belt drive

Condition monitoring solutions

Assembly A

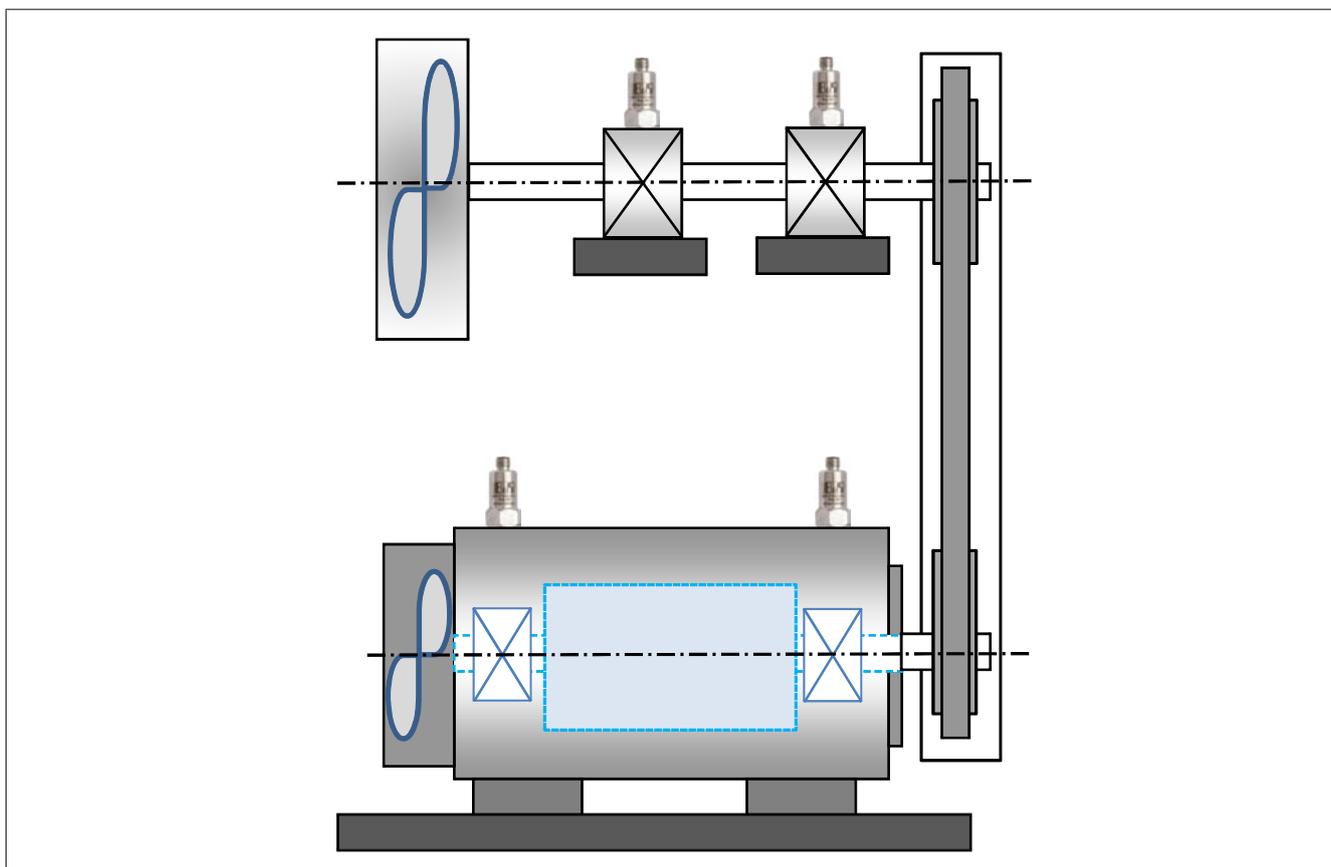


Figure 80: System diagram - Structure of a fan with belt drive

Assembly B

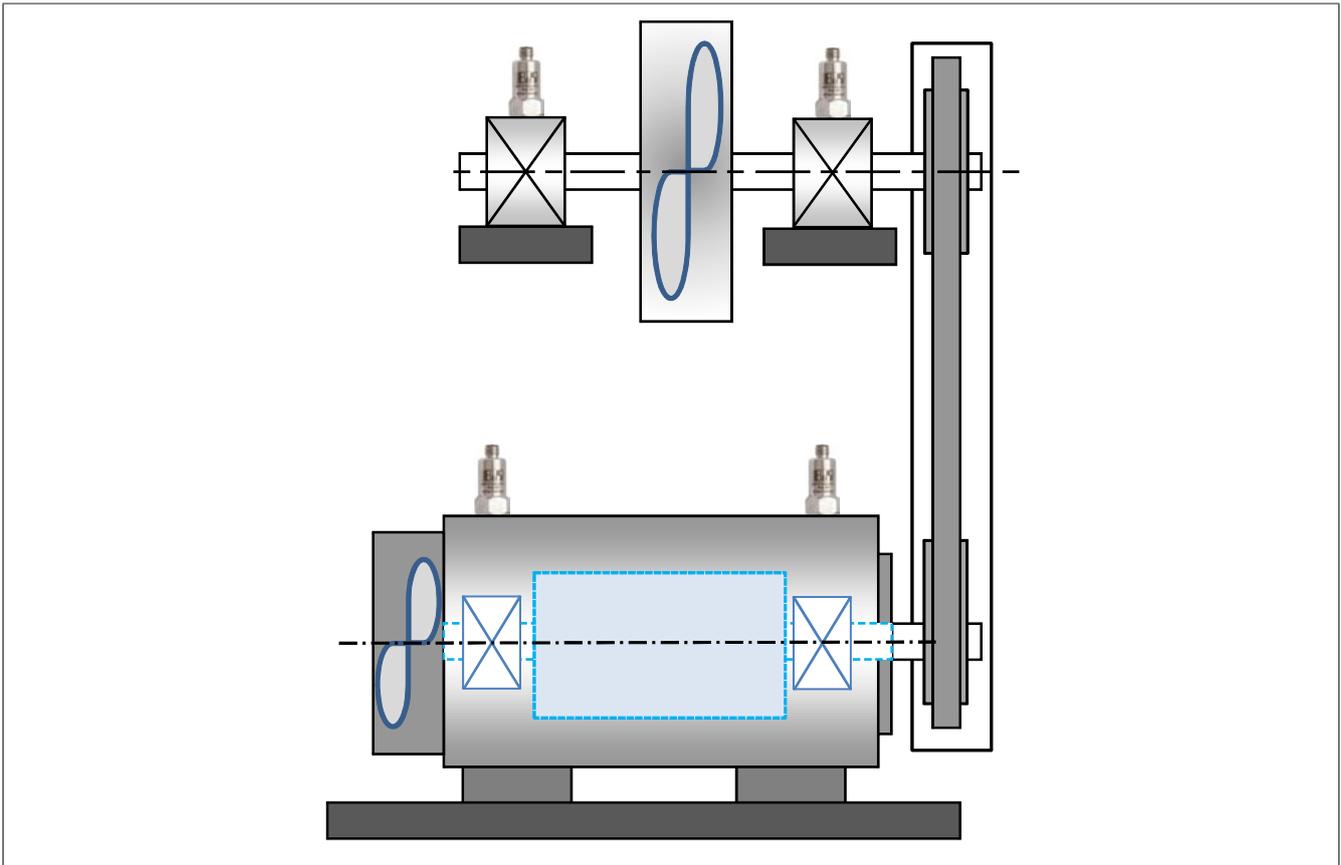


Figure 81: System diagram - Structure of a fan with belt drive - Alternative bearing

Sensor usage:

Number of sensors	Usually 4 sensors. One sensor is sufficient for smaller drive units.
Sensor installation	Preferably vertical. Horizontal installation is also possible, if necessary. Belt misalignment is particularly apparent in the axial direction.

Fan with belt drive - Common problems:

Application possibilities									
Electric motor	Green	Yellow	Green	Green	Green	Green	Green	White	Green
Fan	Green	Yellow	Green	Green	Green	Green	Green	White	White
Flat belts and V-belts	Green	Yellow	Green	White	White	White	Yellow	Yellow	White
Toothed belt	Green	Yellow	Green	White	White	White	White	White	White

Potential failures										
Frequency of the cause of failure in these Potential failures: Green Frequent Yellow Depends on where used Orange Rare to never White Not assessed	Suitability of the respective characteristic values to monitor the potential failures: Green Very good Yellow Good Orange Less good White Not assessed	Imbalance	Orientation	Loose components	Early-stage bearing failure	Late-stage bearing failure	Suboptimal lubrication	Attachment of components	Belt tension and alignment	Electrical errors
		Orange	White	White	Green	Green	Green	Green	Green	Green

Characteristic values									
PeakHighFrequency	Orange	Orange	Green	Green	Green	Green	Green	Yellow	Orange
CrestFactorHighFrequency	Orange	Orange	Green	Green	Yellow	Green	Yellow	White	Orange
Vdi3832KtHighFrequency	Orange	Orange	Green	Green	Yellow	Green	Yellow	White	Orange
PeakRaw	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Green	Yellow	Orange
CrestFactorRaw	Yellow	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Yellow	Orange
SkewnessRaw	Yellow	Yellow	Yellow	Orange	Yellow	Orange	Yellow	Green	Orange
KurtosisRaw	Yellow	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Yellow	Orange
Vdi3832KtRaw	Yellow	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Yellow	Orange
RmsHighFrequency	Orange	Orange	Green	Green	Yellow	Green	Green	Yellow	Orange
RmsAccRaw	Yellow	Yellow	Green	Yellow	Green	Yellow	Green	Yellow	Orange
RmsVelRaw	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Green	Yellow	Orange
Iso10816	Green	Green	Yellow	Orange	Green	Yellow	Yellow	Green	Yellow
RmsAccEnvelope	Orange	Orange	Green	Green	Orange	Yellow	Green	Yellow	Orange
RmsVelEnvelope	Orange	Orange	Green	Yellow	Orange	Yellow	Yellow	Orange	Orange
FrequencyBandRmsVelEnvelope	Yellow	Yellow	Green	Green	Yellow	Yellow	Green	Yellow	Yellow
FrequencyBandRmsAccEnvelope	Yellow	Yellow	Green	Green	Yellow	Yellow	Green	Yellow	Yellow
FrequencyBandRmsAccRaw	Green	Green	Green	Yellow	Green	Yellow	Green	Yellow	Orange
FrequencyBandRmsVelRaw	Green	Green	Green	Yellow	Green	Yellow	Green	Yellow	Orange
FrequencyBandNoiseAccEnvelope	Orange	Orange	Green	Orange	Yellow	Green	Orange	Yellow	Orange
FrequencyBandNoiseVelEnvelope	Orange	Orange	Green	Orange	Yellow	Green	Orange	Yellow	Orange
FrequencyBandNoiseAccRaw	Orange	Orange	Green	Orange	Yellow	Green	Orange	Yellow	Orange
FrequencyBandNoiseVelRaw	Orange	Orange	Green	Orange	Yellow	Green	Orange	Yellow	Orange

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.3.4 Directly coupled pump



Figure 82: Structure of a pump drive

Condition monitoring solution

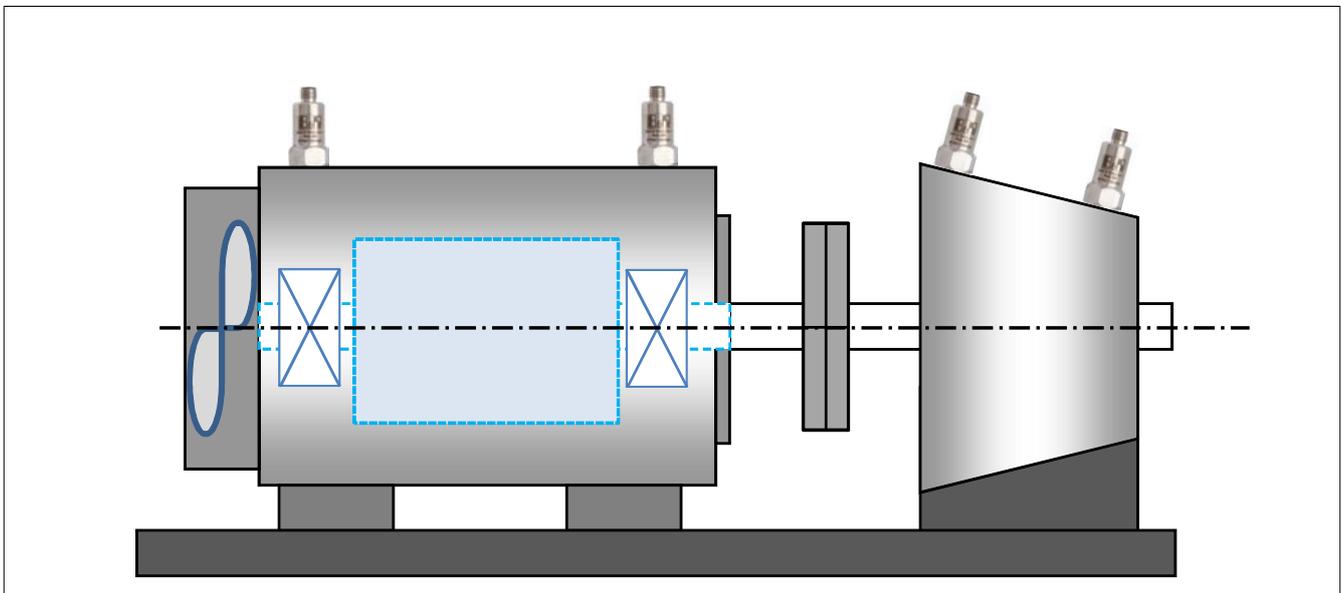


Figure 83: System diagram - Structure of a pump drive

Sensor usage:

Number of sensors	Usually 4 sensors. 2 sensors are sufficient for smaller drive units.
Sensor installation	Preferably vertical. Horizontal installation is also possible, if necessary.

5.3.3.5 Gearbox

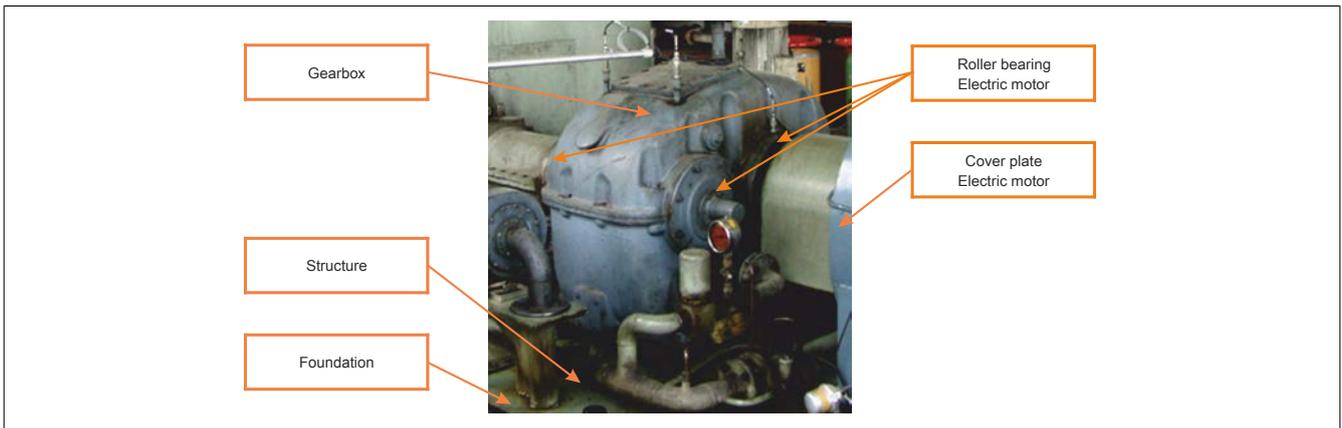


Figure 84: Structure of a gearbox

Condition monitoring solution

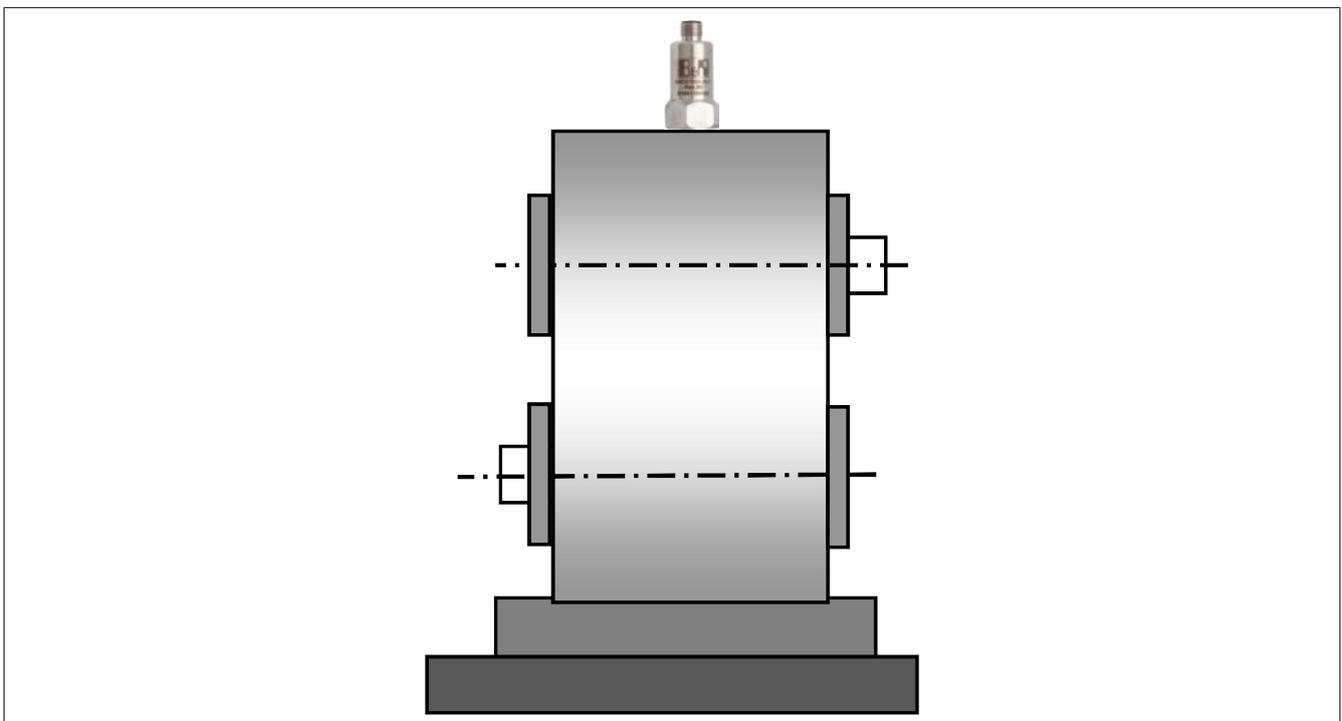


Figure 85: System diagram - Structure of a gearbox

Sensor usage:

Number of sensors	The number of sensors depends on the type and size of the gearbox.
Sensor installation	Preferably vertical. Horizontal installation is also possible, if necessary. The mounting direction depends greatly on the loading direction of the gearbox.

Gearbox - Common problems:

Application possibilities												
Gearbox	Yellow	Yellow	Green	Green	Green	Green	Yellow	White	White	Green		
Potential failures												
Frequency of the cause of failure in these Potential failures: Green: Frequent Yellow: Depends on where used Orange: Rare to never White: Not assessed Suitability of the respective characteristic values to monitor the potential failures: Green: Very good Yellow: Good Orange: Less good White: Not assessed	Imbalance	Orientation	Loose components	Early-stage bearing failure	Late-stage bearing failure	Suboptimal lubrication	Attachment of components	Coupling faults	Belt tension and alignment	Defects in flat belts, v-belts and timing belts	Gearbox damage	Electrical errors
	Characteristic values											
PeakHighFrequency	Orange	Orange	Green	Green	Green	Green	Yellow	Yellow	Green	Green	Orange	Orange
CrestFactorHighFrequency	Orange	Orange	Green	Green	Yellow	Green	Yellow	Yellow	Green	Green	Orange	Orange
Vdi3832KtHighFrequency	Orange	Orange	Green	Green	Yellow	Green	Yellow	Yellow	Green	Green	Orange	Orange
PeakRaw	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Green	Yellow	Yellow	Yellow	Orange	Orange
CrestFactorRaw	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Yellow	Yellow	Yellow	Yellow	Orange	Orange
SkewnessRaw	Yellow	Yellow	Orange	Orange	Orange	Yellow	Yellow	Yellow	Green	Orange	Orange	Orange
KurtosisRaw	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Yellow	Yellow	Yellow	Yellow	Orange	Orange
Vdi3832KtRaw	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Yellow	Yellow	Yellow	Yellow	Orange	Orange
RmsHighFrequency	Orange	Orange	Green	Green	Yellow	Green	Yellow	Yellow	Green	Green	Orange	Orange
RmsAccRaw	Yellow	Yellow	Green	Green	Yellow	Green	Yellow	Yellow	Yellow	Yellow	Orange	Orange
RmsVelRaw	Yellow	Yellow	Yellow	Yellow	Green	Yellow	Yellow	Yellow	Yellow	Yellow	Orange	Orange
Iso10816	Green	Green	Yellow	Orange	Yellow	Yellow	Yellow	Green	Green	Yellow	Orange	Orange
RmsAccEnvelope	Orange	Orange	Green	Green	Orange	Yellow	Green	Yellow	Green	Green	Orange	Orange
RmsVelEnvelope	Orange	Orange	Green	Yellow	Orange	Yellow	Yellow	Orange	Orange	Yellow	Orange	Orange
FrequencyBandRmsVelEnvelope	Yellow	Yellow	Green	Green	Yellow	Yellow	Green	Yellow	Green	Green	Orange	Orange
FrequencyBandRmsAccEnvelope	Yellow	Yellow	Green	Green	Yellow	Yellow	Green	Yellow	Green	Green	Orange	Orange
FrequencyBandRmsAccRaw	Green	Green	Green	Yellow	Green	Yellow	Green	Green	Yellow	Green	Orange	Orange
FrequencyBandRmsVelRaw	Green	Green	Green	Yellow	Green	Yellow	Green	Green	Yellow	Green	Orange	Orange
FrequencyBandNoiseAccEnvelope	Orange	Orange	Green	Orange	Yellow	Green	Orange	Orange	Yellow	Yellow	Orange	Orange
FrequencyBandNoiseVelEnvelope	Orange	Orange	Green	Orange	Yellow	Green	Orange	Orange	Yellow	Yellow	Orange	Orange
FrequencyBandNoiseAccRaw	Orange	Orange	Green	Orange	Yellow	Green	Orange	Orange	Yellow	Yellow	Orange	Orange
FrequencyBandNoiseVelRaw	Orange	Orange	Green	Orange	Yellow	Green	Orange	Orange	Yellow	Yellow	Orange	Orange

For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.3.3.6 Gearbox with countershaft

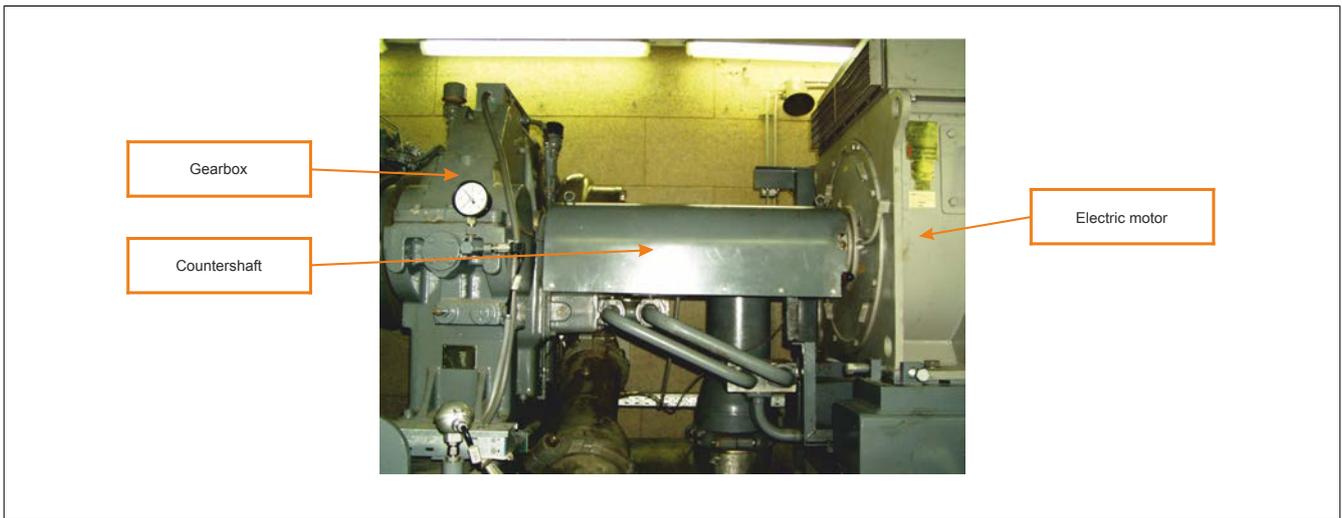


Figure 86: Structure of a gearbox with countershaft

Condition monitoring solution

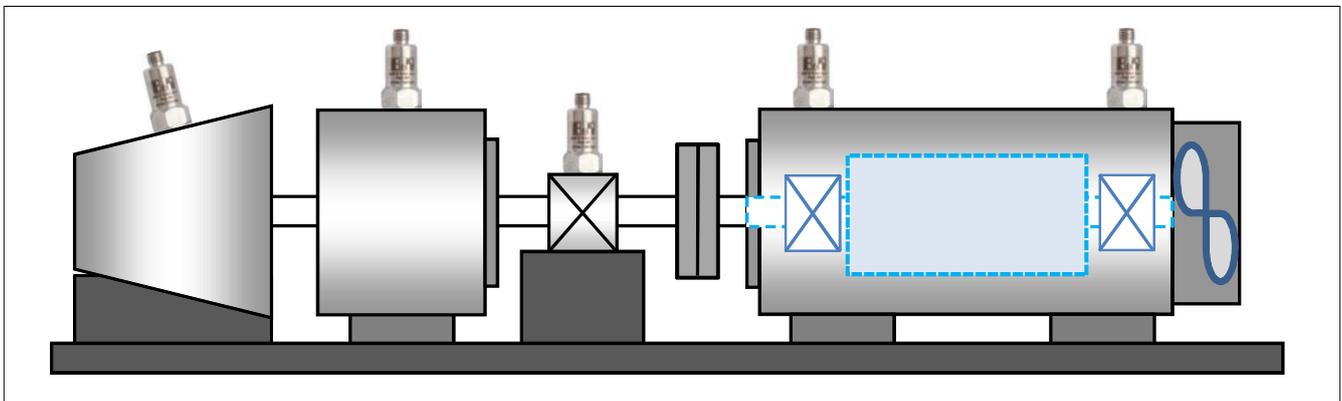
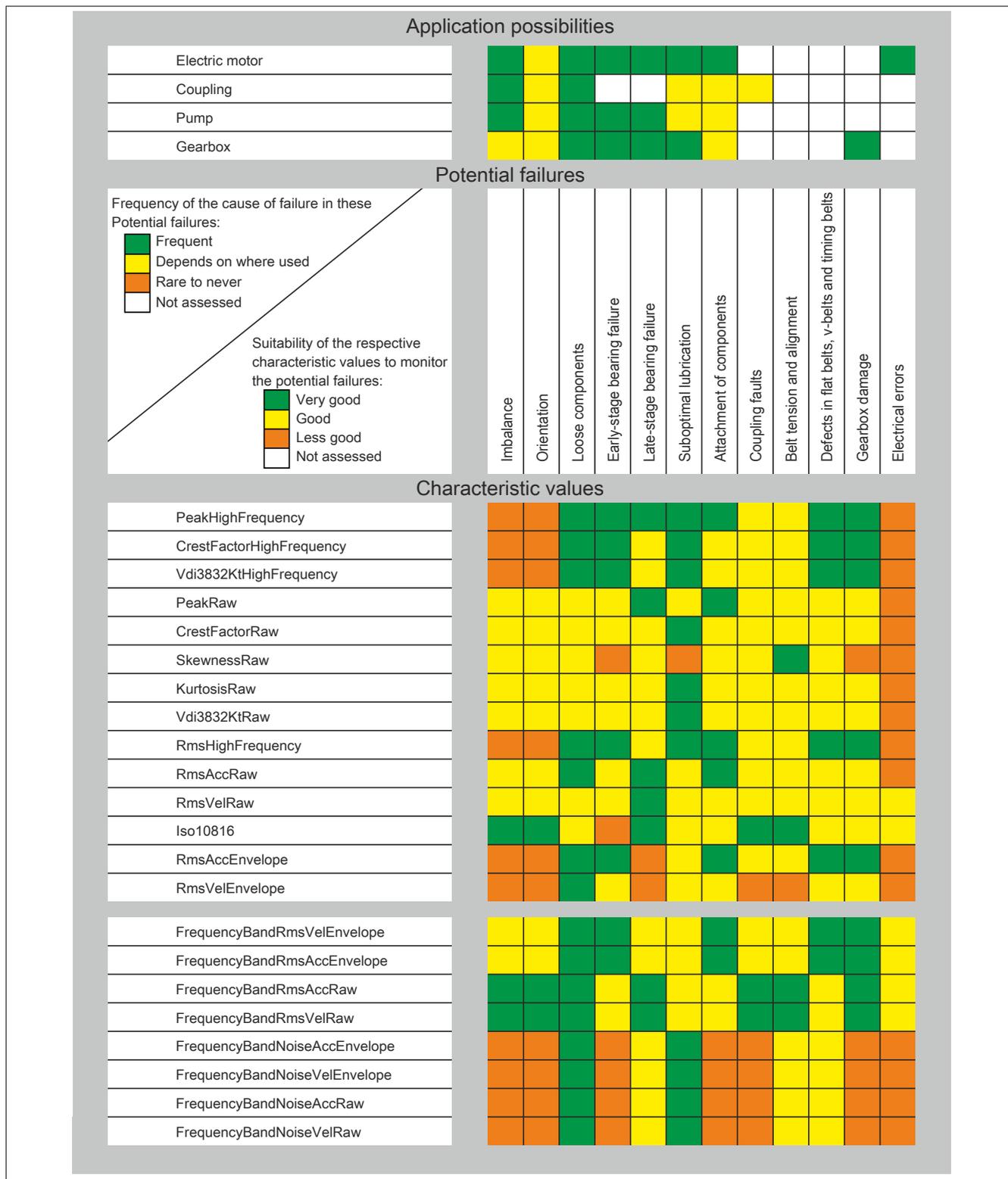


Figure 87: System diagram - Gearbox with countershaft

Sensor usage:

Number of sensors	Usually 5 sensors. 2 sensors are sufficient for smaller and inflexibly coupled drive units.
Sensor installation	Preferably vertical. Horizontal installation is also possible, if necessary.

Gearbox with countershaft - Common problems:



For the meaning of individual characteristic values, see "Characteristic values" on page 41 and "Configuration" on page 63.

5.4 Further reading

Due to its extensive nature, the subject of oscillation analysis can only be outlined in this user's manual.

The following book is well suited to beginners and recommended for those wishing to research this subject in greater detail.

Zustandsüberwachung von Maschinen [Condition Monitoring of Machines]

Publisher: Expert-Verlag GmbH
Author: Dr. Josef Kolerus and Prof. Dr. Johann Wassermann
Edition: 5th, newly revised 2011 edition
Language: German
Pages: 408
ISBN-13: 978-3-8169-3080-8

6 Accessories

6.1 Sensors

6.1.1 0ACS100A.00-1

6.1.1.1 Order data

Model number	Short description	Figure
	Sensors	
0ACS100A.00-1	Accelerometer, nominal sensitivity 100 mV/g, top exit	
	Required accessories	
	Sensor cable	
0ACC0020.01-1	Cable for accelerometer, length 2 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0050.01-1	Cable for accelerometer, length 5 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0100.01-1	Cable for accelerometer, length 10 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0150.01-1	Cable for accelerometer, length 15 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0200.01-1	Cable for accelerometer, length 20 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0500.01-1	Cable for accelerometer, length 50 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC1000.01-1	Cable for accelerometer, length 100 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	

Table 9: 0ACS100A.00-1 - Order data

6.1.1.2 Technical data

Model number	0ACS100A.00-1
Sensor properties	
Natural resonance (installed)	22 kHz (rated)
Sensitivity	100 mV/g $\pm 10\%$ nominal 80 Hz at 22°C
Frequency response	2 Hz to 10 kHz $\pm 5\%$ 0.8 Hz to 15 kHz ± 3 dB
Isolation	Isolated base
Measurement range	± 50 g
Cross-sensitivity	<5%
Electrical properties	
Electrical disturbances	Max. 0.1 mg
Broadband resolution	0.2 mg (200 μ g) over 1 Hz to 15 kHz
Spectral noise	10 Hz to 10 μ g/Hz 100 Hz to 4 μ g/Hz 1 kHz to 3 μ g/Hz
Current range	0.5 to 8 mA
Bias voltage	10 to 12 VDC
Settling time	2 s
Output impedance	Max. 200 Ω
Housing isolation	>10 ⁸ Ω at 500 V
Operating conditions	
Degree of protection per EN 60529	IP67
Ambient conditions	
Temperature	
Operation	-55 to 140°C
Max. shock resistance	5000 g
Emitted interferences	EN 61000-6-4:2001
Immunity to interference	EN 61000-6-2:1999
Mechanical properties	
Housing	
Material	Stainless steel
Installation	M8 x 1.25 x 6 mm bolt, pre-assembled on the sensor
Weight	110 g
Measurement element	PZT piezoelectric crystal (lead zirconate titanate)
Measurement execution	Compressed
Tightening torque	8 Nm
Connectors	M12

Table 10: 0ACS100A.00-1 - Technical data

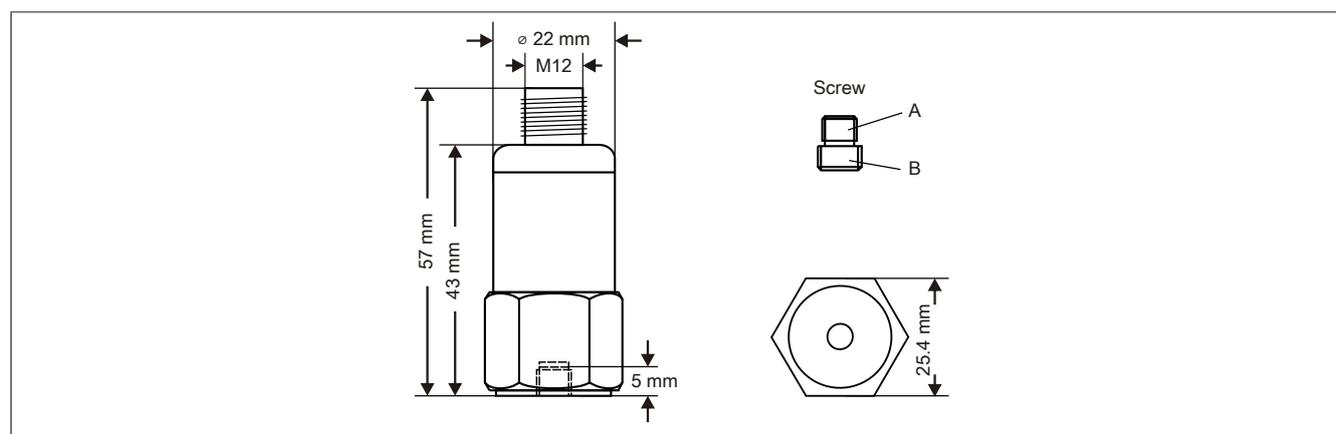
For applicable certifications for the sensor, see the manufacturer's website.



Certificates

<http://www.hansfordsensors.com/resources/certificates/>

6.1.1.3 Dimensions

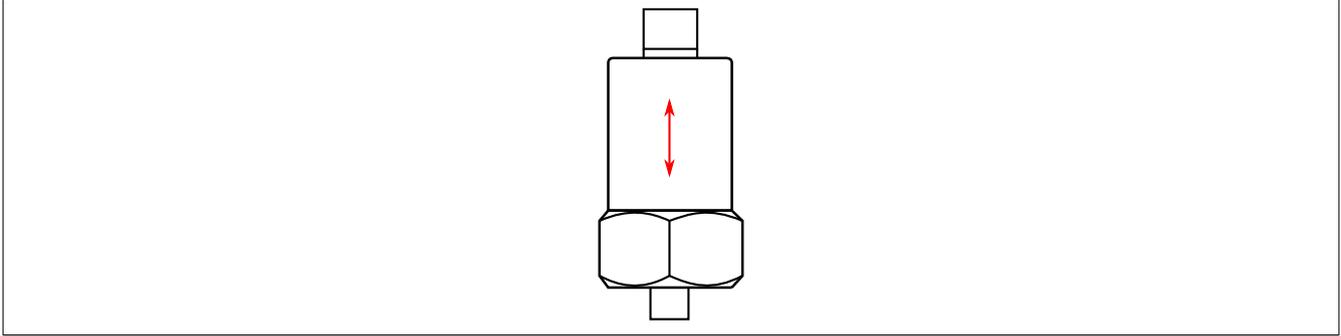


The screw is already installed when delivered.

- A 1/4" - 28 UNF (maximum thread length: 5 mm)
- B M8 x 6 x 1.25

6.1.1.4 Installation direction

B&R vibration sensor 0ACS100A.00-1 is designed for measurements in the longitudinal axis.



6.1.2 0ACS100A.90-1

6.1.2.1 Order data

Model number	Short description	Figure
	Sensors	
0ACS100A.90-1	Accelerometer, nominal sensitivity 100 mV/g, side exit	
	Required accessories	
	Sensor cable	
0ACC0020.01-1	Cable for accelerometer, length 2 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0050.01-1	Cable for accelerometer, length 5 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0100.01-1	Cable for accelerometer, length 10 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0150.01-1	Cable for accelerometer, length 15 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0200.01-1	Cable for accelerometer, length 20 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC0500.01-1	Cable for accelerometer, length 50 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	
0ACC1000.01-1	Cable for accelerometer, length 100 m, 2x 0.34 mm ² , female M12 connector on the sensor side, can be used in cable drag chains, UL listed	

Table 11: 0ACS100A.90-1 - Order data

6.1.2.2 Technical data

Model number	0ACS100A.90-1
Sensor properties	
Natural resonance (installed)	22 kHz (rated)
Sensitivity	100 mV/g \pm 10% nominal 80 Hz at 22°C
Frequency response	2 Hz to 10 kHz \pm 5% 0.8 Hz to 15 kHz \pm 3 dB
Isolation	Isolated base
Measurement range	\pm 50 g
Cross-sensitivity	<5%
Electrical properties	
Electrical disturbances	Max. 0.1 mg
Broadband resolution	0.2 mg (200 μ g) over 1 Hz to 15 kHz
Spectral noise	10 Hz to 10 μ g/Hz 100 Hz to 4 μ g/Hz 1 kHz to 3 μ g/Hz
Current range	0.5 to 8 mA
Bias voltage	10 to 12 VDC
Settling time	2 s
Output impedance	Max. 200 Ω
Housing isolation	>10 ⁸ Ω at 500 V
Operating conditions	
Degree of protection per EN 60529	IP67
Ambient conditions	
Temperature	
Operation	-55 to 140°C
Max. shock resistance	5000 g
Emitted interferences	EN 61000-6-4:2001
Immunity to interference	EN 61000-6-2:1999
Mechanical properties	
Housing	
Material	Stainless steel
Installation	M8 x 1.25 x 33 mm screw, included in delivery
Weight	170 g
Measurement element	PZT piezoelectric crystal (lead zirconate titanate)
Measurement execution	Compressed
Tightening torque	8 Nm
Connectors	M12

Table 12: 0ACS100A.90-1 - Technical data

For applicable certifications for the sensor, see the manufacturer's website.



Certificates

<http://www.hansfordsensors.com/resources/certificates/>

6.1.2.3 Dimensions

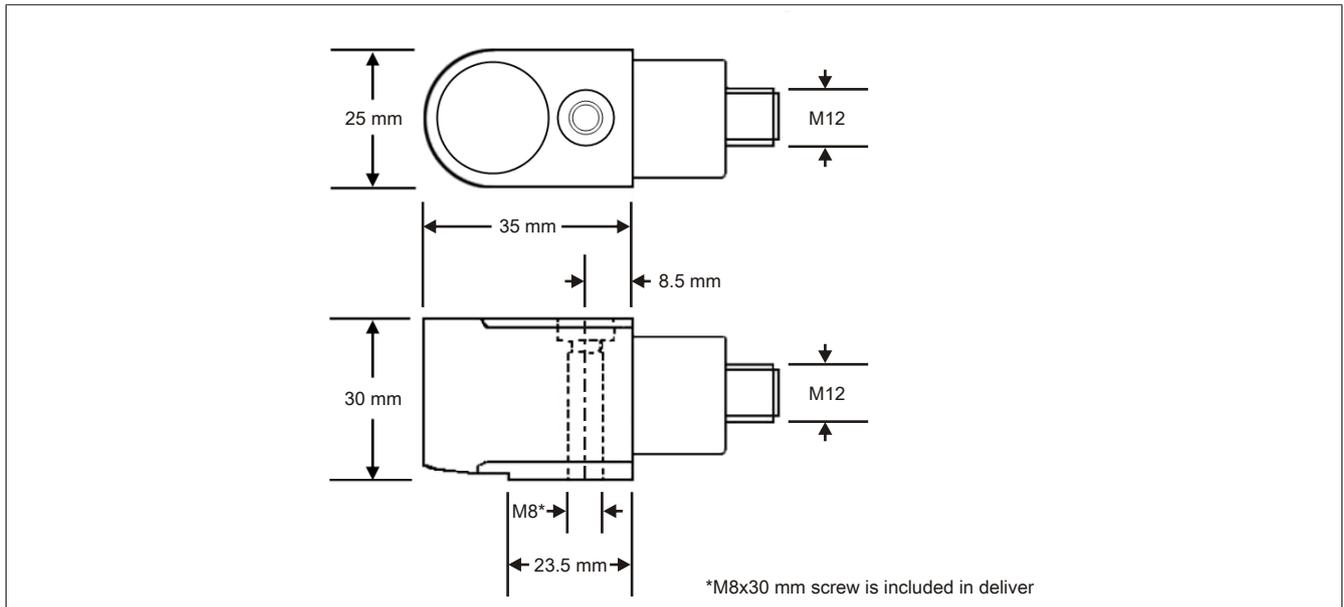
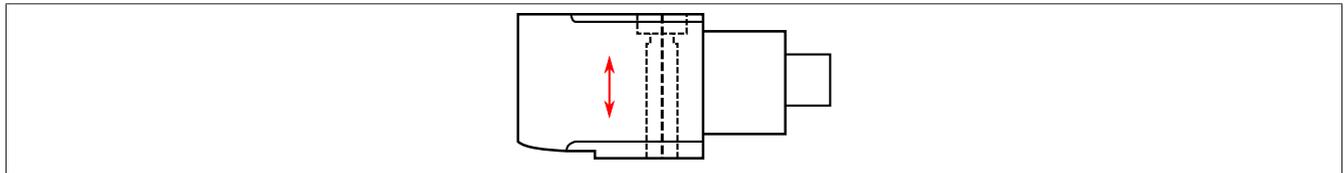


Figure 88: 0ACS100A.90-1 - Dimensions

6.1.2.4 Installation direction

B&R vibration sensor 0ACS100A.90-1 is designed for measurements in the transverse axis.



6.1.3 General information

6.1.3.1 Pinout

Pin	Description
1	Not assigned
2	18 to 30 V (brown)
3	Not assigned
4	0 V (blue)

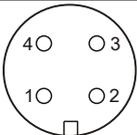


Table 13: 0ACS100A.x0-1 - Pinout

6.1.3.2 Frequency response

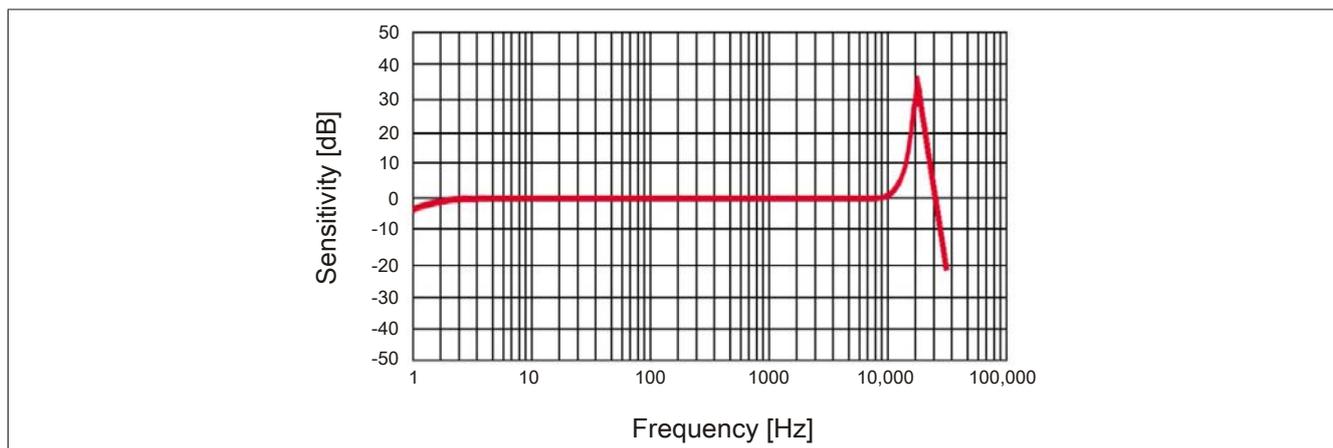


Figure 89: 0ACS100A.x0-1 - Frequency response

6.2 Sensor cables

6.2.1 Order data

Model number	Length	Short description
0ACC0020.01-1	2 m	Cable for accelerometer, 2x 0.34 mm ² , 1x 0.25 mm ² , M12 female connector on sensor side, 1x 25 mm ² shield connection, can be used in cable drag chains, UL/CSA listed
0ACC0050.01-1	5 m	
0ACC0100.01-1	10 m	
0ACC0150.01-1	15 m	
0ACC0200.01-1	20 m	
0ACC0500.01-1	50 m	
0ACC1000.01-1	100 m	

6.2.2 Technical data

Name	0ACC0xx0.01-1
General information	
Number of pins	3
Cable length	x
Characteristic values of the wire	
Cable type	PUR halogen-free black shielded
AWG signal lead	22
Conductor design signal lead	42x 0.10 mm
Wire diameter incl. insulation	1.27 mm ±0.02 mm
Wall thickness insulation	≥0.21 mm (wire insulation) Approx. 1.1 mm (outer jacket)
External diameter of cable	5.9 mm ±0.15 mm
Insulation resistance	≥100 GΩ*km (at 20°C)
Conductor resistance	Max. 58 Ω/km (at 20°C)
Shielding	Braided copper wires
Cable weight	44 kg/km
Smallest bend radius, fixed	29.5 mm
Smallest bend radius, movable	59 mm
Number of flex cycles	4000000
Bend radius	59 mm
Travel path	10 m
Movement speed	3 m/s
Acceleration	10 m/s ²
Ambient temperature (during operation)	-40°C ... 80°C (cable, inflexible installation) -25°C ... 80°C (cable, flexible installation)
Degree of protection per EN 60529	IP67

6.2.3 Sensor cables with female M12 connector

Pos.	Pieces	Description	Note
1	1	Sensor cable	2x 0.34 mm ² (1501702 3x 0.34)
2	1	Female M12 connector (axial)	Female M12 connector (M12x1 A-keyed)
3	2	Wire end sleeve (2x sensor cables)	3203066 AI 0.34-8 TQ
4	1	Heat shrink tubing	
5	1	Shield connection	1x 0.25 mm ² black
6	1	Wire end sleeve (1x shield connection)	3200632 AI 0.25-12 BU

6.2.4 Cable diagram

