GSV COMMUNICATION PROTOCOL SPECIFICATION

GSV-6 / GSV-8

Updated: 11/03/2022

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf

Phone: +49 3302 89824 60
Fax: +49 3302 89824 69

Mail: info@me-systeme.de
Web: www.me-systeme.de

Change Log		
Version	**Changes**	**Editor**
00.00.01	initiated	F&S, OlD
00.00.02	created	F&S, SvS
00.01.00	revised	F&S, SöP
00.01.01	revised	F&S, OlD
00.02.00	checked	F&S, SvS
01.00.00	released	F&S, SöP
01.00.01	revised	HK
01.00.02	revised	HK
01.00.03	revised	F&S, OlD
01.00.04	revised	TMS
01.00.05	revised & commented until chap. 1.11.5	SW
01.00.06	revised	TMS
01.00.07	revised	SW
01.00.08	revised, corrected	SW
01.00.09	revised, Chap. 1.9 complemented	SW
01.00.10	revised, Chap. 1.9 complemented	SW
01.00.11	revised, Chap. 1.5	TMS
01.00.12	Formatted to ME-template Ooo	SW
01.00.13	Minor changes, Ooo versioning introduced	HK/SW
01.00.14	Read / Write Interface Setting added.	SW
01.00.15	TEDS commands added, minor corrections	SW
01.00.16	Updated, annex organized	SW
01.00.17	Updated, corrected	SW
01.00.18	Updated	SW
01.00.19	Updated, corrected	SW
01.00.20	Updated, complemented	SW
01.00.21	Updated, complemented	SW
01.00.22	Updated, corrected, complemented, translated to EN	SW
01.00.23	Updated, complemented	SW
01.00.24	Updated, complemented	SW
01.00.25	Updated, complemented	SW
01.00.26	Updated, complemented	SW
01.00.27	Updated, complemented	SW
01.01.01	Updated, new annex F	SW
01.01.02	Added new cmd, updated,	SW
01.01.03	Added new functionalities, revised	SW
01.01.04	Minor corrections, res. Cmds deleted	SW

File name: PK01079 TB03 01.01.04 GSV-ProtocolDefinitionEN.odt
pdf-file: ba-gsvcom_en.pdf

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Content

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Introduction

This document specifies the application protocol of two different interfaces: The serial / UART interface and the CANbus interface with the ME proprietary protocol. The serial protocol is implemented for both GSV-6 and GSV-8 devices, while the CANbus application protocol is so far been used by GSV-6 only.
Some GSV-8 models offer a CANopen option, which is **not** subject to this document, but described in the CANopen Manual. All GSV-8 devices have an USB interface that uses this serial protocol, specifically via the CDC device class, for which no driver has to be installed when using Windows® 10. The same applies to most Linux distributions. Both operating systems create a virtual COM port from the USB-CDC.

The GSV-6BT has an additional logical interface that follows the structure of this protocol, but is not subject to this document (described in an separate manual).

For both interfaces (Serial/USB-CDC and GSV-6 CAN), two different Windows® DLLs are available, to make device access more convenient for Windows® programmers. Separate DLL programming and reference manuals are available.

Purpose

This document describes the communication protocol specification for the serial and proprietary CANbus interfaces of the devices GSV-6 and GSV-8 and its model variants. The basic structure of the protocol and all particular commands are explained.

Audience

This document is intended to be used by developers.

Conventions / Abbreviations

The following describing elements are used:

<num>	Single bit with position ‚num‘ counted from 0 (LSBit) on
<high:low>	Bit field from position ‚low‘ to ‚high‘ (inclusive)
[low-high]	Value range (including borders)
name[num]	Byte/Char ‚num‘ out of an array ‚name‘
FWver	Firmware / Device Software Version
i.e.	that is
e.g.	for example

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

GSV-Protocol

Basics

The devices GSV-6 and GSV-8 use an almost identical protocol for their serial communication.

The basic structure of the protocol frames is identical, but may differ in particular commands in their parameters and device behaviour, since both measuring amplifiers are technically different.

The differences are highlighted in the particular command descriptions.

Numerical values are transmitted in Big-endian order inside a frame, i.e. all data types consisting of more than one byte are transmitted from MSB to LSB, or inserted into the structure, respectively. They also are described in that order in the command descriptions.

The representation of bits in the structure tables is always from MSBit to LSBit, until a byte is completed.

Basic layout of the Protocol

The basic layout of the protocol frames is as follows:

Length [bits]	Denotation	Description
(8)	Prefix	For any non-block-orientated transmission (serially), the frame always starts with the prefix 0xAA.
2	Frame-Type	Description of the frame: 0b00: Measuring Value 0b01: Command: response 0b10: Command: request 0b11: <reserved>
2	Interface	Description of the interface, from that the frame is transmitted: 0b00: CAN 0b01: Serial (RS232, USB ...) 0b10: Additional logical interface (GSV-8BT: "BGscript")[1] 0b11: Serial with CRC checksum
4	Length	This bit field is interpreted differently, depending on the fields Frame-Type and Interface. The corresponding description can be found in the chapters of the frame types.
8	Control / Status Byte	This bit field is interpreted differently, depending on the field Frame-Type. The corresponding description can be found in the chapters of the frame types.
0-480	Data	The data payload field is of various length and is interpreted

1 Not subject to this document, described in a separate manual

Length [bits]	Denotation	Description
or. 0-120		frame-type-specific. With CAN, a full CAN frame is transmitted and the payload is limited to 48 Bits.
(8)	Suffix	For any non-block-orientated transmission (serial), the frame always ends with the suffix 0x85.

Table: Protocol basic layout

Different interfaces have major differences in the layout of the protocol frames.

With the CAN interface, a maximum of 8 bytes can be transmitted in a frame.

With serial interfaces (UART, USB), the number of transmitted bytes in a frame can be larger, because it's a data steam. In order to mark clear frame boundaries, a constant prefix and a suffix is present at every frame.

Request-Frame (0b10)

The request frame is used to send commands to the device. Every request is answered by the device, normally with the response frame (with 2 exceptions: GetValue and ResetDevice). Various requests may read parameters („Read...“, „Get...“), write them („Write...“, „Set...“) or trigger actions („Set...“). The requests are distinguished by the command numbers (IDs); parameter list an response values and their meanings have a fixed assignment to the command numbers, i.e. are defined by them.

Advice: It is recommended to wait for the response after issuing a request, before issuing the next one.

CAN Request Frame

Length [bits]	Denotation	Description
2	Frame-Type	0b10: Command: Request
2	Interface	0b00: CAN
4	Length	Number of valid data bytes [0-6]
8	Control byte	Command number (ID)
48	Data	Command parameters

Table: CAN Request protocol frame

Information: The size of CAN request frames is always 8 bytes = 64 bits.

Serial Request Frame

Length [bits]	Denotation	Description
8	Prefix	Prefix Byte 0xAA
2	Frame Type	0b10: Command: Request

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Length [bits]	Denotation	Description
2	Interface	0b01: Serial (RS232, USB ...) without CRC. 0b11: Serial with CRC
4	Length	Number of data bytes [0-15]
8	Control byte	Command number (ID)
0-120	Data	Command parameters
(8)	Check sum	CRC-8 checksum, only present if Interface= 0b11
8	Suffix	Suffix Byte 0x85

Table: Serial Request protocol frame

Response Frame (0b01)

Response Frames have a status byte that contains an error code. A response frame signalling no error may contain return values (mostly with read requests), i.e. the status byte then is always ERR_OK =0 (Exception: see below "Serial'). But, if an error is signalled (Status byte >0), this response frame doesn't contain further data.

CAN Response Frame

Length [bits]	Denotation	Description
2	Frame Type	0b01: Command response
2	Interface	0b00: CAN
4	Length	Number of valid data bytes [0-6]
8	Status byte	Error Code (see odes odes)
48	Data	Response value(s)

Table: CAN Response protocol frame

Information: The size of CAN response frames is always 8 bytes = 64 bits.

Serial Response Frame

Length [bits]	Denotation	Description
8	Prefix	Prefix Byte 0xAA
2	Frame Type	0b01: Command response
2	Interface	0b01: Serial (RS232, USB ...) without CRC. 0b11: Serial with CRC
4	Length	Number of data bytes in this frame [0-15] =15: Number of data bytes in this frame = <Status byte> + 15
8	Status byte	Error Code (see *Error Codes)*
0-120	Data	Response value(s)
(8)	Check sum	CRC-8 checksum, only present if Interface= 0b11
8	Suffix	Suffix Byte 0x85

Table: Serial Response protocol frame

If the field 'Length' is =15, the field 'Status byte' contains an additional length qualifier, i.e. the number of data bytes in the frame is then:
<Status byte> +15.
That's how the GSV can return up to 270 bytes in a response frame, instead of up to 14, if <Length> is smaller than 15.
If the field <Length> is smaller than 15, the 'Status byte' contains an error code.
The long response frame was introduced for GSV-6BT with FW-version 3.20, used for the command *Read File Extended* and others.

Checksum for serial command and response frames

If the "Interface" bit field is 0b11 for command requests, i.e. if bit 5 of the second byte in the serial request frame is =1, the response will also contain the checksum and this will also be signaled by interface bits= 0b11.
This CRC-8 checksum of the request and response frames is calculated over the protocol bytes <frame type><interface><length> and status byte as well as all data bytes, i.e. over all bytes in the frame without a prefix and without a suffix (and of course without the checksum itself). Details in the chapter "Checksum Calculation", p.21. The checksum is supported by the GSV-8 from firmware version 1.56 and by the GSV-6 from version 3.35.

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Measuring Value Frame (0b00)

Acquired measuring values are transmitted in measuring value frames. Measuring value frame may be transmitted autonomously and permanently by the device.

CAN Measuring Value Frame

Length [bits]	Denotation		Description	
2	Frame Type		0b00: Measuring value frame	
2	Interface		0b00: CAN	
4	Channel		Channel number beginning with 0	
8	Control / Status byte		<7> Indicator = 0 <6:4> Data type see Data typ <3:0> Error Bits <3:2> reserved <1> Error with multi-axis sensor <0> Saturation / clipping of the analogue input	
48	Data	16 Bits	Time Stamp	Counter. Identical for values that were acquired simultaneously and increments by 1 at every cycle. There's no correlation to a time base.
		32 Bits*	Measuring value	The measuring value of the transmitted channel. * In case the data type is smaller than 32 bits, Bits=0 are padded.

Table: CAN Measuring value frame

Information: The size of CAN measuring value frames is always 8 bytes = 64 bits.

Serial Measuring Value Frame

Length [bits]	Denotation	Description
8	Prefix	Prefix-Byte 0xAA
2	Frame-Type	0b00: Measuring value frame
2	Interface	0b01: Serial (RS232, ...) without CRC. 0b11: Serial with CRC
4	Number of value objects	Number of transmitted measured value objects (e.g. channels) minus 1 GSV-6: 1-6 GSV-8: 2-9 (high-speed mode: up to 15)
8	Control / Status byte	<7> Indicator = 1 <6:4> Data type see Data typ <3:0> Error Bits <3:2> reserved <1> Error with multi-axis sensor <0> Saturation / clipping of the analogue input
16-512	Data	One measuring value per channel with data type indicated High-speed mode: Up to 8 channel sequences, see below
(16)	Checksum	CRC-16 checksum, only present if Interface=0b11
8	Suffix	Suffix Byte 0x85

Table: Serial Measuring value frame

Data types

For measuring value frames, the following data types are defined:

Bits<6:4> Status byte	Enum Cmd 0x80.1 / 0x81.1	Type name
0b000	-	reserved
0b001	1	int16
0b010	2	int24 [GSV-8 only]
0b011	3	float32
0b100	-	reserved
0b101	-	reserved
0b110	-	reserved
0b111	-	reserved

Table Measuring value data types definitions

Evaluation of the Measuring value frame

Normal mode:

In measuring data frames, the measuring value data is always transmitted beginning with the lowest input channel number and ending with the greatest channel number. The byte order of the measuring values is big-endian, i.e. the most significant byte (MSB) comes first. The header field "Number of value objects" indicates the number of measuring values corresponding to channels -1.

High-speed mode:

With the GSV-8 from firmware version 1.54, several channel sequences can be transmitted at the USB interface within one measurement data frame if the measurement data frequency is> = 12000 frames / s. This reduces the protocol overhead and improves data throughput.

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

This frame mode is used only if allowed by the host, i.e. the communicating entity. Dis is done by setting Bit 2 of the parameter of the command GetInterface (see p.27). The header field "Number of objects" indicates the total number of measurement data objects in data frame -1. The channel sequences are transmitted first with the oldest values, channel 1 comes first within a sequence. The configured number of channels should be known for evaluation; this can be read at index 0 with the GetTxMapping command, see p.50. The channel and sequence configuration can only change if the number of channels or the measurement data rate is changed, or by calling Get Interface.

Checksum

The measurement data frame in normal mode (see above) can be configured to contain a CRC-16 checksum. When delivered, the measurement data frame has no checksum. It can be activated with the command GetInterface (No. 0x01) or SetTXmode (No. 0x81). This state is only saved non-volatile when set by SetTXmode, namely for USB and UART, if available. When calling GetInterface, on the other hand, bit 3 of the call parameter must be set correctly, regardless of the status of the TXmode value; this state is volatile and only applies to the interface with which the call was made. The checksum is supported by the GSV-8 from firmware version 1.56 and GSV-3 from 3.35. With this CRC-16 checksum, the low-order byte is transmitted first (little endian). Details in the chapter "Checksum Calculation", p.21.

Interpretation of Measured Data

The correct interpretation of the measured values, i.e. their evaluation to physical values depends on the data type and the right parametrization for the usage of the sensor connected to the correspondent input channel. With data type Float32, readily scaled values are transmitted. If the measuring amplifier is configured correctly (e.g. calibration matrix loaded for multi-axis sensors), these values already represent physical values.

With Integer data types, the values have to be converted in the following way:

1. With GSV-8, the raw values are transmitted in the binary offset sign format. So, this offset has to be subtracted first to yield Signed-Int values:

int16: IntermediateValue_1 = raw value - 0x8000
int24: IntermediateValue_1 = raw value - 0x800000
With GSV-6, this step is omitted: IntermediateValue_1 = raw value

2. This intermediate result is to be converted to a real number (e.g. floating point), then multiplied by the range overhead of 1.05 and divided by the binary (unipolar) numeric magnitude:

int16: IntermediateValue_2 = IntermediateValue_1 * 1,05 / 2^15
int24: IntermediateValue_2 = IntermediateValue_1 * 1,05 / 2^23

3. This yields measuring values that are scaled to a range of ±1.0. The value of 1.0 corresponds to the nominal input range. If that one, for example, is 2mV/V, the IntermediateValue_2 must be multiplied by 2 in order to display the bridge deviation directly in mV/V. If the input range is Single-Ended 10V,

IntermediateValue_2 must be multiplied by 10 to display the input voltage directly in volts (provided the zero point was not changed by SetZero or another offset manipulation to a value unequal to 0).

This calculation doesn't take the transfer function of the sensor between its physical quantity and electrical output into account. That can be done additionally, by calculating a scaling factor, that IntermediateValue_2 must be multiplied with to get values scaled in that physical quantity. This scaling factor can be stored in the device by using the command WriteUserScale (no. 0x15). Application software like GSVmultichannel uses that factor to display values scaled in the particular physical unit.

Example: The following table shows raw values and **IntermediateValue_2 for a nominal input measuring range of 2mV/V:**

Bridge sensor deviation in mV/V	GSV-8		GSV-6	Read value MEGSVxx.dll:: GSVread and other reading functions for measuring values (IntermediateValue_2)
	Integer raw value, 16-Bit (Uint16) Hex	Integer raw value, 24-Bit (Uint24) Hex	Integer raw value, 16-Bit (Sint16) Hex	
<= -2.1	0x0000	0x000000	0x8000	-1.05
-2.0	0x0618	0x061862	0x8618	-1.0
0	0x8000	0x800000	0x0000	0.0
2.00	0xF9E7	0xF9E79E	0x79E7	1.0
>= 2.1	0xFFFF	0xFFFFFF	0x7FFF	1.05

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Error Codes

All defined error codes are listed in this chapter. One of these values is put in the Status Byte field of the response frame, depending on the situation.

Some error codes are reserved for future use.

Error byte in the command response frame:

Designator	**Value** (Hex)	**Description**
ERR_OK	0x00	Command executed without errors.
ERR_OK_CHANGED	0x01	No error, but further device parameters changed
ERR_CMD_NOTKNOWN	0x40	Command number unknown
ERR_CMD_NOTIMPL	0x41	Command not implemented. This may be a command that is not supported by the device type connected.
ERR_FRAME_ERROR	0x42	Frame error: Wrong suffix
ERR_CMD_CRC	0x43	CRC checksum error in command request
ERR_PAR	0x50	Parameter wrong
ERR_PAR_ADR	0x51	Wrong index or address parameter
ERR_PAR_DAT	0x52	Wrong data parameter
ERR_PAR_BITS	0x53	Wrong bits inside parameter
ERR_PAR_ABSBIG	0x54	Parameter absolutely too big
ERR_PAR_ABSMALL	0x55	Parameter absolutely too small
ERR_PAR_COMBI	0x56	Wrong parameter / setting combination
ERR_PAR_RELBIG	0x57	Parameter too big in relation to other parameters / settings
ERR_PAR_RELSMALL	0x58	Parameter too small in relation to other parameters / settings
ERR_PAR_NOTIMPL	0x59	Function invoked by parameter is not implemented
ERR_PAR_TIMEOUT	0x5A	Command parameters have not been send within timeout (normally 200ms)
ERR_WRONG_PAR_NUM	0x5B	Wrong number of parameters in frame or parameter number descriptor in frame wrong.
ERR_PAR_NOFIT_SETTINGS	0x5C	Parameter improper with respect to device's settings
ERR_PAR_HW_COLLISION	0x5D	Function leads to hardware (connection) collision, e.g. short-circuit
ERR_NO_DATA_AVAIL	0x60	Data requested not available (e.g. not initialized)
ERR_DATA_INCONSISTENT	0x61	Data stored not consistent in itself or with parameters
ERR_WRONG_MOD_STATE	0x62	Command could not be executed, because device or functionality in improper state
ERR_NOT_SUPPORTED_D	0x63	Denied, because requested functionality not supported
ERR_FDATA_TOO_HIGH	0x64	Denied, because data rate too high for requested setting
ERR_MEMORY_WRONG_COND	0x6E	Memory write denied, because condition(s) not satisfied
ERR_MEMORY_ACCESS_DENIED	0x6F	Memory write: Access denied
ERR_ACC_DEN	0x70	Access denied

Designator	Value (Hex)	Description
ERR_ACC_BLK	0x71	Access denied, because write functions are blocked
ERR_ACC_PWD	0x72	Access denied: Missing password/PIN
ERR_ACC_MAXWR	0x74	Access denied: Maximum executions reached
ERR_ACC_PORT	0x75	Access from this port denied (other port seems to have write access)
ERR_ACC_RDONLY	0x76	Attempt to write a read-only parameter
ERR_INTERNAL	0x80	Internal exception in device. Please contact manufacturer
ERR_ARITH	0x81	Internal arithmetic exception in device. Please contact manufacturer
ERR_INTER_ADC	0x82	Erratic behaviour of AD converter. Please contact manufacturer
ERR_MWERT_ERR	0x83	Actual measuring value inappropriate to fulfil request
ERR_EEPROM	0x84	Erratic behaviour of EEPROM memory. Please contact manufacturer
ERR_EXT_HW	0x85	Required external hardware (e.g. SDcard) not present or faulty
ERR_FILE	0x86	SDcard: File system driver reports error
ERR_WRONG_DIR	0x87	SDcard: Wrong directory / dir. settings inappropriate
ERR_RET_TXBUF	0x91	Device transmission buffer full
ERR_RET_BUSY	0x92	Device too busy to execute request
ERR_RET_RXBUF	0x99	Device receive buffer full
GETTEDS_ERR_NOSENSOR	0xB0	TEDS: No sensor connected at all
GETTEDS_ERR_NOTEDSEE	0xB1	TEDS: No TEDS memory connected
GETTEDS_ERR_BASICONLY	0xB2	TEDS: Only Basic TEDS data found
GETTEDS_ERR_NOTEDSDAT	0xB3	TEDS: Data not in conformance with IEEE1541.4
GETTEDS_ERR_ENTRY_INVALID	0xB4	TEDS: Data entry not set
GETTEDS_ERR_TOUT	0xB5	TEDS: 1-wire EEPROM driver timed out
GETTEDS_ERR_CHKSUM	0xB6	TEDS: Data checksum error
GETTEDS_ERR_UNKNOWN_TEMPL	0xB7	TEDS: TEDS template not (yet) supported
GETTEDS_ERR_VERIFY_FAIL	0xB8	TEDS: Data write-verify failed
BT_CONFIG_ERR	0xC0	BGscript: BlueTooth application error

Table: Codes of the Error byte in the command response frame

Error Codes read by „GetLastValueError" (Cmd. No. 0x43) with GSV-8

The 48 Bits (6 Bytes) of the error structure in particular:

Bits<47:45>

Error-Types / Categories		
Value	**Denotation**	**Description**
1	VALERR_TYPE_SATURATED	Analogue value at sensor input saturated, i.e. input range exceeded
2	VALERR_TYPE_MAX_EXCEED	Maximum allowed physical value exceeded

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

		(especially with multi-axis sensor)
3	VALERR_TYPE_SENSOR_BROKEN	Bridge sensor or its connection defective
4	HWERR_TYPE_ANA_OUT	Analogue output configured as current output left unconnected or output driver overheated
5	HWERR_TYPE_DIO	Digital output configured as output has short-circuit or wrong level

Bits<44:16>

Error-Time: Device time when error occurred

Stored in minutes of device operation, i.e. this value / 60 are the absolute device hours when the error occurred.

Bits<15:0> Error-Flags: Type-dependent error code

Description of the error flags in particular:

1. ErrType = VALERR_TYPE_SATURATED:

Bits<15:8>: If Bit=1: Negative saturation occurred in one or several input channels, whereby bit 8 corresponds to channel 1, bit 9 to channel 2, and so on, until bit 15: channel 8.

Bits<7:0>: If Bit=1: Positive saturation occurred in one or several input channels, whereby bit 0 corresponds to channel 1, bit 1 to channel 2, and so on, until bit 7: channel 8.

Remarks:

1. If this fault is actually present, the red "FUNCTION"-LED of the device is lit permanently.

2. Then, also Bit 0 of the Status Byte in the measuring data frame is set.

2. ErrType = VALERR_TYPE_MAX_EXCEED:

2.1. With multi-axis sensor:

Bit 0: If Bit=1: In the Fx direction, a positive or negative exceedance of the maximum values (defined in the sensor calibration data) occurred.

Bit 1: If Bit=1: In the Fy direction, a positive or negative exceedance of the maximum values occurred.

Bit 2: If Bit=1: In the Fz direction, a positive or negative exceedance of the maximum values occurred.

Bit 3: If Bit=1: In the Mx direction, a positive or negative exceedance of the maximum values occurred.

Bit 4: If Bit=1: In the My direction, a positive or negative exceedance of the maximum values occurred.

Bit 5: If Bit=1: In the Mz direction, a positive or negative exceedance of the maximum values occurred.

2.2. With PT1000 temperature sensor:

Bits<15:0>: If the measuring value is above the maximum of 1500°C or below the minimum of -230°C, both bits [ChannelNo -1] and [ChannelNo +7] will be set, e.g. 0x0101 for channel 0, 0x0202 for channel 2 and so on, until 0x8080 for channel 8. If the maximum is exceeded, the communicated measuring value will be set 9999, if it's below the minimum, the communicated output value is set to -9999.

Remarks:

1. If this fault is actually present, the red "FUNCTION"-LED of the device is lit permanently.

2. Then, also Bit 1 of the Status Byte in the measuring data frame is set.

3. ErrType = VALERR_TYPE_SENSOR_BROKEN:

Bit0: If Bit=1: A fault occurred at the Ud+ line of channel 1

Bit1: If Bit=1: A fault occurred at the Ud- line of channel 1

Bit2: If Bit=1: A fault occurred at the Ud+ line of channel 2

Bit3: If Bit=1: A fault occurred at the Ud- line of channel 2

Bit4: If Bit=1: A fault occurred at the Ud+ line of channel 3

Bit5: If Bit=1: A fault occurred at the Ud- line of channel 3

Bit6: If Bit=1: A fault occurred at the Ud+ line of channel 4

Bit7: If Bit=1: A fault occurred at the Ud- line of channel 4

Bit8: If Bit=1: A fault occurred at the Ud+ line of channel 5

Bit9: If Bit=1: A fault occurred at the Ud- line of channel 5

Bit10: If Bit=1: A fault occurred at the Ud+ line of channel 6

Bit11: If Bit=1: A fault occurred at the Ud- line of channel 6

Bit12: If Bit=1: A fault occurred at the Ud+ line of channel 7

Bit12: If Bit=1: A fault occurred at the Ud- line of channel 7

Bit14: If Bit=1: A fault occurred at the Ud+ line of channel 8

Bit15: If Bit=1: A fault occurred at the Ud- line of channel 8

Remark:

If this fault is actually present, the red "FUNCTION"-LED of the device is lit permanently.

4. ErrType = HWERR_TYPE_ANA_OUT:

Constant value 0xFFFF: A fault occurred at any of the analogue outputs.

Else:

Bit0: If Bit=1: Current output at channel 1 is left unconnected

Bit1: If Bit=1: Current output at channel 2 is left unconnected

Bit2: If Bit=1: Current output at channel 3 is left unconnected

Bit3: If Bit=1: Current output at channel 4 is left unconnected

Bit4: If Bit=1: Current output at channel 5 is left unconnected

Bit5: If Bit=1: Current output at channel 6 is left unconnected

Bit6: If Bit=1: Current output at channel 7 is left unconnected

Bit7: If Bit=1: Current output at channel 8 is left unconnected

Bit8: If Bit=1: Voltage output driver at channel 1 overheated

Bit9: If Bit=1: Voltage output driver at channel 2 overheated

Bit10: If Bit=1: Voltage output driver at channel 3 overheated

Bit11: If Bit=1: Voltage output driver at channel 4 overheated

Bit12: If Bit=1: Voltage output driver at channel 5 overheated

Bit13: If Bit=1: Voltage output driver at channel 6 overheated

Bit14: If Bit=1: Voltage output driver at channel 7 overheated

Bit15: If Bit=1: Voltage output driver at channel 8 overheated

Remarks:

1. Overheating of the output driver may be due to a short-circuit.

2. If this fault is actually present, the red "FUNCTION"-LED of the device is blinking slowly (about 1x/sec).

5. ErrType = HWERR_TYPE_DIO:

Bits<15:0> If Bit=1: A short-circuit occurred at the correspondent DIO-No (=BitNo+1), i.e. the output is set to high, but carries a low level, or it's set to low, but carries a high level (voltage >=3V). Bit 0 corresponds to DIOno 1 (Group1: 1.1.), Bit 1 DIOno 2 (Group1: 1.2.), and so on, until Bit 15: DIOno 16 (Group4: 4.4.)

Remark:

If this fault is actually present, the red "FUNCTION"-LED of the device is blinking fast (about 3x/sec).

ErrInfo of „GetLastValueError" (0x43) with GSV-6:

Bits <5:0> / Byte 0: Exceedance of the maximum value defined in the six-axis-sensors calibration data occurred at channel(s) defined by bit No: Bit 0: Channel 1 (Fx), bit 1: Channel 2 (Fy) ... bit 5: Ch6 (Mz)

Bits <13:8> / Byte 1: Saturation of input channel(s) occurred at channel(s) defined by bit No: Bit 8: Channel 1, bit 9: Channel 2 ... bit 13: Ch. 6

Checksum Calculation

In serial frames, if bits <5:4> of the second protocol byte are =0x3, the frame contains a CRC checksum. This is calculated using the information contained in protocol bytes 2 and 3 and possibly using all data bytes. It is in the frames after the payload, just before the suffix. The existence of the checksum is managed independently for the command interface and the measurement data frames. The following applies to commands: If the request has the CRC-8, the response will also be provided with it.

The following applies to the measurement data frames:
1. If the CRC-16 was switched on with the SetTXmode command, the GSV-8 will first send measurement data frames with CRC-16 to all serial interfaces from the next restart. This status is stored in non-volatile memory.
2. This status can be temporarily (i.e. in a non-volatile manner) overwritten by the GetInterface command. If bit 3 is set in the GetInterface calling parameter, frames are sent with CRC-16; if it is not set, frames are without CRC-16. The latter also applies if it is activated in TxMode. In this way, the GSV-8 is downwards-compatible with older (or existing) application software that uses the Windows DLL MEGSV86w32.dll, e.g. GSVmulti Version 1.48 or earlier (because this always sends GetInterface with Bit3=0 during initialization).

CRC-16 for measurement data frames

In the measurement data frame, the checksum is a 16-bit value in which, exceptionally, the

lower-order byte comes first (little-endian). It corresponds to the CRC16 of the Modbus protocol, i.e. the generator polynomial is:

$x^{16}+x^{15}+x^2+1$ or 0x8005. Input and output data are bit-reversed and the CRC start value is 0xFFFF.
Example (hex):
AA 37 B0 C1 C7 CD 38 3F E6 19 7E 3F C0 B6 0B BF 49 7E 95 40 22 DD 1D 3F B2 11 53 3E E6 C3 72 3F 92 65 3B E7 6E 85
The checksum is calculated here over the 34 bytes from 0x37 to 0x3B and the result is 0x6EE7. An example of a calculation routine in C is in annex G, p.99

CRC-8 bei Kommandos

For command requests and responses, the checksum is an 8-bit value that is calculated in the same way. The generator polynomial is:
x^8+x^2+x+1 or 0x07. The bit order of the input and output data is regular, i.e. not reversed, and the CRC start value is 0x00.
Examples (hex):
AA B1 01 08 AC 85 (Get Interface request with activation of the CRC-16 in the measured value frame)

AA 74 00 C8 73 00 02 B9 85 (Get Interface response on that)

AA B0 23 A6 85 (StopTX for halting the measred data stream)

AA 70 00 A2 85 (General response: OK)

An example of a calculation routine for this CRC-8 in C can be found in annex G, p. 99.

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Command Interface

In the following chapter, the structure of the command description is explained first. Then, the sub-chapter with the description of each command follows.

General structure:

The description of the particular commands follows this table structure:

No.	Direction (R / W) or A: Triggers action	Name	Device model(s)
Description			
Num. of Bytes	Num. of Parameters		
Offset P1	Parameter1 Type	Paramter1 Name	Parmeter1 description
Offset P2	Parameter2 Type	Paramter2 Name	Paramter2 description
Num. of Bytes	Num. of return values		
Offset R1	return value 1 Type	return value 1 Name	return value 1 description

Table: General Command description

If the entry in "Direction" is **W**, the value will be stored in non-volatile memory. If it's **A,** an action will be triggered only, without storing to non-volatile memory. "**W / A"** does both, it triggers an action and stores to non-volatile memory. "**W (RAM)**" means that the value is stored to an array/struct in RAM and to copy to non-volatile memory, another command has to be used.

The number of bytes and the offset in the command tables refer to the data payload field inside the request/response frame.

If commands for GSV-8 and GSV-6 differ fundamentally (e.g. by data types), separate descriptions are presented, as well as with different interfaces (CAN / serial).

The following device models may appear:

Device model or variant	Description
GSV-6	GSV-6 in general
GSV-6 (Reset)	GSV-6: Command requires a reset or power cycle to become effective
GSV-6BT	GSV-6 BlueTooth
GSV-6BTanalog	Bluetooth receiver with analogue outputs for GSV-6BT (reserved)
GSV-8	GSV-8 in general
GSV-8 CAN	GSV-8 with CANopen interface

Table: Device models and variants

The data types for parameters and return values are:

Designator	Bytes	Description
uint8_t	1	Unsigned 8-Bit Integer
uint16_t	2	Unsigned 16-Bit Integer
U24	3	Unsigned 24-Bit Integer
S24	3	Signed 24-Bit Integer
uint32_t	4	Unsigned 32-Bit Integer
int32_t	4	Signed 32-Bit Integer
S7.24	4	Signed fixed-point number (Sign-Bit, then 7 before and then 24 bits after radix point)
float	4	32-Bit floating point number, following IEEE 754
char[x]	x	String

Table: Data types for parameters and return values

Several denotations are common in the command descriptions; these logical types are described in the following table:

Denotation	Description
Channel	Channel numbers are generally beginning with 1, unless differently documented. The general model GSV-6 has 6 channels with No. 1-6; while GSV-6BT may have up to 7. GSV-8 has 8 analog input channels and up to 2 counter/frequency channels. With many write commands, the special value 0 for the channel parameter means that all channels are set to the same value passed.
Index, Sub Index	Mostly a parameter that accesses a value inside an array, or selects an element specified by channel/address.
Flags	Bit fields, whose bit position correspond to a configuration/function.

Table: Common parameter or return value designators

Command categories

Commands are always answered with a response frame[2], provided that the basic protocol structure is observed for the request (e.g. prefix 0xAA present), see above. The response frame contains one of the error codes listed above.

Parametrization, Storing behaviour

Generally, the devices store their configured parameters to non-volatile memory. So, a parametrization has to be done only once with desired values. The non-volatile memory is an EEPROM or a Flash memory with wear-levelling write management. However, the maximum number of write cycles to the memory device is limited, so it's not a good practise to write parameters at every program start, especially with GSV-6, because that one stores parameters immediately. Instead, it's better to read a value first, then write it only in case the value differs from the desired one.

Configure data flow

The autonomous permanent measuring data transmission can be started with *StartTransmission* and stopped with *StopTransmission*. The transmission of a single measuring value frame can be triggered with *GetValue,* which is useful if autonomous data transmission is stopped.

The state configured with *StartTransmission* or *StopTransmission* is volatile and discarded at a reset or power-on-cycle.

The devices behaviour on permanent value transmission after the power-on cycle can be configured by *SetTXmode,* which is stored in non-volatile memory. By default, permanent data transmission is on.

The number of measuring value frames, that are acquired and transmitted per second (if permanent data transmission on), can be configured with *WriteDataRate*.

Adjust zero-point

The current measuring value(s) can be tared to zero by applying the command *SetZero.* This function is also triggered by the control input "TARE".

With GSV-6, by using the command WriteZero, the level to that the measuring value will be adjusted after applying *SetZero,* can be configured. This level can be read with *ReadZero.*

2 With the exception of the ResetDevice (No. 0x78) and GetValue (0x3B) commands, see there.

The function SetZero (command or trigger input) operates directly on the raw values of the analogue-digital converter, i.e. before further processing, e.g. with user-scalings that are configurable by *SetUserOffset* and *SetUserScale*.

Set scaling

With the commands *WriteUserOffse*t and *WriteUserScale* the scaling of the measuring values may be set.

The overall scaling is also determined by the input scaling settings of the measuring amplifier. With GSV-6, the input type can be selected with the command *MEwriteInputRange*, while the range may be continuously adapted by WriteAoutScale (OverallScaling), so that the resulting input sensitivity is: Physical input range / OverallScaling.
With GSV-8, for selecting the input type, *SetInputType* is used.

The raw measuring values are scaled to a numeric range of ±1. the value of 1 corresponds to 100% of the configured input sensitivity / input range; e.g. +10V or 2mV/V.
For example, if the input sensitivity is 2mV/V bridge input, the user scaling factor must be configured to 2.0 by using *WriteUserScale* to get values scaled in mV/V.

This user-scale factor can be selected in a way that the device converts the raw values to physical values, because the raw value (scaled to ±1) is multiplied with this factor:

Physical measuring value [data type: Float] = raw value * user scaling factor

To scale the raw signals of multi-axis sensors (Force-/Torque Sensors, F/T Sensors), the device can perform a matrix multiplication. To enable this multiplication with the calibration matrix, a bit in the mode register is set, after having stored the calibration data in the device. If the mode for F/T sensors is active, the scaling by *WriteUserScale* and the offset set by *WriteUserOffset* are not used. Settings zero by *SetZero* is still effective, but should then be used for all channels. With GSV-8, the attempt to use this command on single channels No. 1 to 6 is rejected, if multi-axis mode is enabled.

Setting the unit with *SetUnitNo* / *SetUnitText* hasn't any influence on the scaling; it's only a storage value to decorate physical values.

With GSV-8, the additional scaling of the analogue output is performed after the input scaling of the, measuring values, it can be set with *WriteAoutScale*.

Configure Interfaces

The CANbus can be configured with the command *SetCANSetting*.

Configuration of Special Functions

Many special functions are available, like configuring a digital IIR filter of 4th order, reading a fault memory, noise suppression around zero, reading maximum and minimum values and many more.

Access Control, Write Protection

For some commands, a user-ID must be set before, i.e. setting the correct password is a pre-condition. This is done by issuing *MEsetID (0x19).*

In the description, this condition is then mentioned ("User-ID required").
With GSV-8, the User-ID can be changed using the command *Set Password* (0x58).

Also, all write commands can be locked by the command *Switch Blocking* (0x92).[3] Unlocking is possible with user-ID setted before. With GSV-8, a write command may also be rejected, because another interface has the write access. For example, this is the case with devices equipped with a

3 With GSV-6, the write-protection is usable from firmware version 3.30

field bus (CANopen, EtherCAT): Whenever the field bus has write access according to its device state, e.g. from pre-operational state onwards, the serial interface has no write access.

Command Descriptions

ResetStatus (0x00)

0x00	W / A	ResetStatus	GSV-6 / GSV-8
Clear all protocol errors. GSV-8: Faults of the measuring application are cleared, too, provided that the reason of it doesn't exist anymore.			
0 Byte	0 Parameter		
0 Byte	0 Return values		

GetInterface (0x01)

0x01	R	GetInterface	GSV-6 / GSV-8
Request of an interface descriptor. With this command, information about the actual interface is requested and with a parameter, the desired measuring transmission behaviour can be controlled.			
1 Byte	1 Parameter		
0	uint8_t (1)	Request-Flags	<7:4> Reserved Bits (=0) <3>: =1: Measured data frames with CRC16 checksum. =0: No CRC 2: =1: Allow High-speed measuring value frames. =0: Normal frames only. See p.14 <1:0> Desired measuring data transmission - 0b00: No change • 0b01: Transmission OFF • 0b10: Transmission ON
4 Bytes	4 Return values		
0	uint8_t (1)	Protocol and device model	<7:6> Protocol type 0b01: Measured data frames without CRC 0b11: Measured data frames with CRC16 <5:0> Device model - 0x00: Unknown - 0x06: GSV-6 - 0x08: GSV-8
1	uint8_t (1)	Measuring value frame-Information	<7:4> Number of objects in value frame minus 1 <3> Actual setting of the measuring value transmission (ON/OFF) <2:0>Measuring value type (see Data type) - 0x01: int16 - 0x02: S24 (GSV-8 only) - 0x03: float
2	uint8_t (1)	Write protection	<7> =1: Interface-specific write protection enabled <6> =1: General write protection enabled <5:0> Number of the interface, over that

0x01	R	GetInterface	GSV-6 / GSV-8
			this request was done (0...N-1)
3	uint8_t (1)	Number of descriptors	Number of supported interfaces N. Also determines the index range of the "Basic settings" of the commands 0x7B/0x7C. If =0: These commands are not (yet) supported.

ReadZero (0x02)

0x02	R	ReadZero	GSV-6 / GSV-8
Read actual tare offset value of specified channel			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel no. GSV-6: [1-6] GSV-8: [1-8]
4 Bytes	1 Return value		
0	int32_t (4)	Tara value	Actual tare offset value Remark: The GSV-6 uses a 16-bit value internally, which is returned sign-extended to 32 bits; with GSV-8 it's sign-extended from 24 to 32 bits.

WriteZero (0x03)

0x02	W	WriteZero	GSV-6 / GSV-8
Set new Tare offset value for specified channel / all channels GSV-8: Pre-condition: USER_ID_LEVEL set			
5 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-6] GSV-8: [0-8] Channel=0: Set parameters of all channels to the same value.
1	int32_t (4)	Tara value	Set new Tare offset value Remark: GSV-6 uses the lower 16Bits only. GSV-8: 24 Bits.
0 Byte	0 return values		

ReadAoutOffset (0x04)

0x04	R	ReadAoutOffset	GSV-6 / GSV-8
Read the offset for the analogue output (in percent) of specified channel			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [1] GSV-8: [1-8]
4 Bytes	1 return value		
0	float (4)	Offset value	Analogue output offset in the range: -60.0 to 60.0 %

WriteAoutOffset (0x05)

0x04	W	WriteAoutOffset	GSV-6 (Reset) / GSV-8
Set the analogue output offset in percent for specified channel / all channels			

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x04	W	WriteAoutOffset	GSV-6 (Reset) / GSV-8
5 Bytes	2 Parameters		
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-1] GSV-8: [0-8] Channel=0: Set parameters of all channels to the same value.
1	float (4)	Offset value	Analogue output offset in the range: -60.0 to 60.0 %
0 Byte	0 return values		

ReadAoutScale (0x06)

0x06	R	ReadAoutScale	GSV-6
Read the scaling value, which is applied to the raw measuring value. It has effect on both digital and analog output measuring value.			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-6]
4 Bytes	1 return value		
0	float (4)	Scaling value	Actual scaling value

0x06	R	ReadAoutScale	GSV-8
Read the scaling value, which is applied to the analogue output only.			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-8]
4 Bytes	1 return value		
0	float (4)	Scale value	Present scaling for the analogue output.

WriteAoutScale (0x07)

0x07	W	WriteAoutScale	GSV-6
Set the scaling value, which is applied to the raw measuring value and has effect on both digital and analog output measuring value.			
5 Bytes	2 Parameters		
0	uint8_t (1)	Channel	Channel No. [0-6] Channel=0: Set parameters of all channels to the same value.
1	float (4)	Scale value	New scaling value, which is applied to the raw measuring value.
0 Byte	0 return values		

0x07	W	WriteAoutScale	GSV-8
Set the scaling value, which is applied to the analogue output only.			
5 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. [0-8] Channel=0: Set parameters of all channels to the same value.
1	float (4)	Scale value	New scaling value for the analogue output
0 Byte	0 return values		

WriteAoutDirect (0x08)

0x08	W	WriteAoutDirect	GSV-8
Set the analogue output to a value, independently from the input This command works only it the GSV-8 war put in the direct mode with SetAoutType (0x0E)			
3 Bytes	2 Parameters		
0	uint8_t (1)	Channel	Channel No. GSV-8: [1-8]
1	uint16_t (2)	DAC-Code	value written directly to the DAC-register.
0 Byte	0 return values		

LoadConfig (0x09)

0x09	W / A (GSV-6: A)	GetAll	GSV-6 / GSV-8
Read parameters from the data set specified and set them. ***Advice***: With GSV-6, the settings read are temporary and are not written automatically as actual parameters. To save them, issuing the command SaveAll with data set 0 is necessary!			
1 Byte	1 Parameter		
0	uint8_t (1)	Data Set No.	Data Set No. to read and set, respectively. GSV-6: [0-1] GSV-8: [0-7] [0]: The actual (GSV-8: lastly stored) Settings [1]: Default settings [2-7]: User data sets (GSV-8 only)
0 Byte	0 return values		

See annex F, p.96 for a list of parameters and their default values, that are loaded by using LoadConfig (No=1 for load default).

StoreConfig (0x0A)

0x0A	W	SaveAll	GSV-6 / GSV-8
Store parameters at the data set specified.			
1 Byte	1 Parameter		
0	uint8_t (1)	Data Set No.	Data Set No. to store to GSV-6: [0-1] GSV-8: [0-7] [0]: The actual Settings [1]: Default settings [2-7]: User data sets (GSV-8 only) The default settings are not storable for the user.
0 Byte	0 return values		

See annex F, p.96 for a list of parameters that are saved by using StoreConfig.

SetZero (0x0C)

0x0C	W / A	SetZero	GSV-6 / GSV-8
Set tare to the actual input value, so that the output value becomes zero.			
1 Byte	1 Parameter		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x0C	W / A	SetZero	GSV-6 / GSV-8
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-6/7] GSV-8: [0-8/10] Channel =0: Set ALL channels to zero
0 Byte	0 return values		

GetAoutType (0x0D)

0x0D	R	GetAoutType	GSV-8 / GSV-6
Request the type of an analogue output GSV-6 From firmware version 3.29			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-8: [1-8] GSV-6: =1
2 Bytes	2 return values		
0	uint8_t (1)	Type-Enum	Type enumerator 0: 0V – 10V 1: -10V – 10V 2: 0V – 5V 3: -5V – 5V 4: 4mA – 20mA 5: OFF 6: 0mA – 20mA 7: 0V - 2.5V (only GSV-6 CPU/DEV) Other values reserved
1	uint8_t (1)	Channel Mode	GSV-6: =0 GSV-8:<7:3> reserved <2> Alternate input (Ch. 7&8 only): linked to Counter / Frequency input <1> Output channel: inactive =1, active =0 <0> Direct mode active - =0: Follows measuring value - =1: Direct mode (see WriteAoutDirect (0x08))

SetAoutType (0x0E)

0x0E	W	SetAoutType	GSV-8 / GSV-6
Set the type of an analogue output GSV-6 From firmware version 3.29			
3 Bytes	3 Parameters		
0	uint8_t (1)	Channel	Channel No. GSV-8: [0-8] GSV-6: [0-1] Channel =0: Set ALL channels to the same value
1	uint8_t (1)	Type Enum	see GetAoutType (0x0D)
2	uint8_t (1)	Channel Mode	see GetAoutType (0x0D) (GSV-6: =0)

0x0E	W	SetAoutType	GSV-8 / GSV-6
0 Byte	0 return values		

GetUnitNo (0x0F)

0x0F	R	GetUnitNo	GSV-6 / GSV-8
Read the number of the unit stored			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel no. GSV-6: [1-6] / [1-7] (FW-ver >=3.11) GSV-8: [1-10]
1 Byte	1 return values		
0	uint8_t (1)	PhysUnit-Enum	Enumerator value of physical units (see (Codes), or 0xFE,FF: Free unit text 1,0 see GetUnitText (0x11)

SetUnitNo (0x10)

0x10	W	SetUnitNo	GSV-6 / GSV-8
Store the number of a physical unit			
2 Bytes	2 Parameters		
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-6] / [0-7] (FW-ver >=3.11) GSV-8: [0-10] Channel=0: Set parameters of all channels to the same value.
0	uint8_t (1)	PhysUnit-Enum	Enumerator value of physical units (see (Codes) or 0xFE,FF: Free unit text 1, 0 see SetUnitText (0x12)
0 Byte	0 return values		

GetUnitText (0x11)

0x11	R (Serial)	GetUnitText	GSV-6
Read a user-defined unit string Advice: This command has a interface-dependent difference.			
1 Byte	1 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
9 Bytes	2 return values		
0	uint8_t (1)	Indicator	With serial interface always =0
1	char[8]	Unit String	The user-defined unit string (NULL-terminated or exactly 8 chars long)

0x11	R (CAN)	GetUnitText	GSV-6
Read a user-defined unit string ***Advice***: On a request, **two** response frames are transmitted. And this command has a interface-dependent difference.			
1 Byte	1 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
5 Bytes	2 return values		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x11	R (CAN)	GetUnitText	GSV-6
0	uint8_t (1)	Indicator	=1 First half of the string =2 Second half of the string
1	char[4]	Unit String	The user-defined unit string (NULL-terminated or exactly 8 chars long)

0x11	R	GetUnitText	GSV-8
Read a user-defined unit string			
1 Byte	1 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
8 Bytes	1 return value		
1	char[8]	Unit String	The user-defined unit string (NULL-terminated or exactly 8 chars long)

SetUnitText (0x12)

0x12	W (Serial)	SetUnitText	GSV-6
Write a user-defined unit string (maximum of 8 chars) Advice: This command has a interface-dependent difference.			
10 Bytes	3 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
1	uint8_t (1)	Indicator	With serial always =0
2	char[8]	Unit String	The user-defined unit string (NULL-terminated or exactly 8 chars long).
0 Byte	0 return values		

0x12	W (CAN)	SetUnitText	GSV-6
Write a user-defined unit string (maximum of 8 chars) In order for the string gets stored, first the command with the 1st half, then the 2nd half always has to be sent to the device! If the 1st half will be set only, it remains in the RAM of the device. If only the 2nd half will be set, the device responds with an error message			
6 Bytes	3 Parameters		
0	uint8_t (1)	Slot	Text-Slot [0-1]
1	uint8_t (1)	Indicator	=1 First half of the string =2 Second half of the string
2	char[4]	Unit String	The user-defined unit string (NULL-terminated or exactly 8 chars long).
0 Byte	0 return values		

0x12	W (Serial)	SetUnitText	GSV-8
Write a user-defined unit string (maximum of 8 chars)			
9 Bytes	3 Parameters		
1	uint8_t (1)	Slot	Text-Slot [0-1]
2	char[8]	Unit String	The user-defined unit string (NULL-terminated or exactly 8 chars long).
0 Byte	0 return values		

ReadUserScale (0x14)

0x14	R	ReadUserScale	GSV-6 / GSV-8
Read the user-defined scaling value			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [1-6] / [1-7] (FW-ver >=3.11) GSV-8: [1-10]
4 Bytes	1 return value		
0	float (4)	UserScale Factor	User-Scale value, used for measuring values with float data type

WriteUserScale (0x15)

0x14	W	WriteUserScale	GSV-6 / GSV-8
Write the user-defined scaling value **Advice**: If the multi-axis sensor mode is enabled, this value has no effect.			
5 Bytes	2 Parameters		
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-6] / [0-7] (FW-ver >=3.11) GSV-8: [0-10] Channel=0: Set parameters of all channels to the same value.
1	float (4)	UserScale Factor	User-Scale value, used for measuring values with float data type
0 Byte	0 return values		

ReadCal (0x17)

0x17	R	MEreadCal	GSV-6
Read a manufacturer calibration value. The "Calibration Channel" is the input-Channel No. based to 0, with an offset into the corresponding calibration set. ***Advice***: This command requires the manufacturer ID set.			
2 Bytes	2 Parameter		
0	uint8_t (1)	Calibration Channel	0x00-0x05: Input factor 0x08-0x0D: Input offset 0x10: Output factor 0x18: Output offset 0x20-0x25: Scale value (Factor) 0x28-0x2D: Tare value (Offset) Other values reserved.

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x17	R	MEreadCal	GSV-6
1	uint8_t (1)	Sub-Index	Input factor /-offset: for Pre-amplification: - 0x00: Amplification 1x - 0x01: Amplification 2x - 0x02: Amplification 4x - 0x03: Amplification 8x Output factor /-offset for output type: - 0x00: 0V – 10V - 0x01: -10V – 10V - 0x02: 0V – 5V - 0x03: -5V – 5V - 0x04: 4mA – 20mA - 0x05: reserved - 0x06: 0mA – 20 mA - 0x07: 0V - 2.5V Scale- / Tara value - unused Other values reserved.
4 Bytes	1 return value		
0	float (4) / int32_t (4)	Factor Offset	

0x17	R	MEreadCal	GSV-8
Read a manufacturer calibration value.			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	[1-8] Input or output channel
1	uint8_t (1)	SubIndex	- 0x00: ScaleCal f. bridge input Us= 8,75V - 0x01: ScaleCal f. bridge input Us= 5V - 0x02: ScaleCal f. bridge input Us= 2,5V - 0x03: ScaleCal f. Single-ended Input +-10V - 0x04: ScaleCal f. PT1000-Input - 0x10: ScaleCal f. analogue output 10V - 0x11: OffsetCal f. analogue output 10V - 0x12: ScaleCal f. analogue output 5V - 0x13: OffsetCal f. analogue output 5V - 0x14: ScaleCal f. analogue output 4..20mA - 0x15: OffsetCal f. analogue output 4..20mA - 0x20: OffsetCal f. bridge input Us= 8,75V - 0x21: OffsetCal f. bridge input Us= 5V - 0x22: OffsetCal f. bridge input Us= 2,5V - 0x23: OffsetCal f. Single-ended Input +-10V - 0x24: OffsetCal f. PT1000 input Other values reserved.
4 Bytes	1 return values		
0	float (4)	Factor / Offset	

MEwriteCal (0x18)

0x18	W	MEwriteCal	GSV-6 (Reset)
Write a manufacturer calibration value. The "Calibration Channel" is the input-Channel No. based to 0, with an offset into the corresponding calibration set. ***Advice***: This command requires the manufacturer ID set.			
6 Bytes	3 Parameters		
0	uint8_t (1)	Calibration Channel	See ReadCal (0x17)
1	uint8_t (1)	Sub-Index	See ReadCal (0x17)
2	float (4) / int32_t (4)	Factor Offset	
0 Byte	0 return values		

0x18	W	MEwriteCal	GSV-8
Write a manufacturer calibration value. ***Advice***: This command requires the manufacturer ID set.			
6 Bytes	3 Parameter		
0	uint8_t (1)	Calibration Channel	Input or output channel of calibration value. Channel=0: Set parameters of all channels to the same value.
1	uint8_t (1)	Sub-Index	See ReadCal (0x17) (GSV-8)
2	float (4)	Factor / Offset	
0 Byte	0 return values		

MEsetID (0x19)

0x19	A	MEsetID	GSV-6 / GSV-8
Set an unlock code / set an ID level, that is required for some write commands. Only defined passwords are accepted. After three trials with wrong passwords, an error is returned, even if a correct password is given then. Unlocked state is active until reboot.			
4 Bytes	1 Parameter		
0	uint32_t (4)	Unlocking code	ID_EXT_MESYS ID_EXT_USER_CONF ID_EXT_USER_CONF_ALL (reserved)
0 Byte	0 return values		

the unlocking codes are defines as follows:

```
ID_EXT_MESYS
With GSV-8, the code of ID_EXT_USER_CONF can be changed with the command
SetPassword (0x58)
ID_EXT_USER_CONF      GSV-8: (0x42656C6E) = "Beln" (Default)
ID_EXT_USER_CONF      GSV-6: (0x55534331) = "USC1"
```

MEgetIDstate (0x1A)

0x1A	R	MEgetIDstate	GSV-6 / GSV-8
Request the ID state, as set by MEsetID			
0 Byte	0 Parameter		
1 Byte	1 return values		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x1A	R	MEgetIDstate	GSV-6 / GSV-8
0	uint8_t (1)	Unlocking state	0x00: Normal state 0x08: Privileged commands unlocked, i.e. ID_EXT_USER_CONF given 0x10: ME (manufacturer) commands unlocked

Read Sensor Model (0x1B)

0x1B	R	ReadSensorModel	GSV-8
Read the model name of the multi-axis sensor. Present with Firmware-version >= 1.39			
0 Byte	0 Parameter		
1..15 Bytes	1 return value		
0	String(1..15)	Model Name	Null-terminated string or exactly 15 bytes in length (in that case, no termination)

Write Sensor Model (0x1C)

0x1C	W	WriteSensorModel	GSV-8
Read the model name of the multi-axis sensor. Present with Firmware-version >= 1.39 Precondition: USER_ID_LEVEL set			
1..15 Bytes	1 Parameter		
0	String(1..15)	Model Name	Null-terminated string or exactly 15 bytes in length (in that case, no termination)
0 Byte	0 return values		

Read Calibration Operator Name (0x1D)

0x1D	R	ReadCalOpName	GSV-8
Read the name of the institution of the last device calibration and/or operatos's acronym. Present with Firmware-version >= 1.40			
0 Byte	0 Parameter		
8 Bytes	1 return value		
0	String(8)	Operator's Name	Exactly 8 bytes in length (in that case, no termination) or with 0-Bytes padded to 8 bytes in length

Write Calibration Operator Name (0x1E)

0x1E	W	WriteCalOpName	GSV-8
Write the name of the institution of the last device calibration and/or operatos's acronym. Present with Firmware-version >= 1.40. ***Remark***: This command requires a MESYS unlocking.			
8 Bytes	1 Parameter		
0	String(8)	Operator's Name	Exactly 8 bytes in length (in that case, no termination) or with 0-Bytes padded to 8 bytes in length
0 Byte	0 return values		

GetSerNo (0x1F)

0x1F	R	GetSerNo	GSV-6 / GSV-8
Request the device serial number			
0 Byte	0 Parameter		

0x1F	R	GetSerNo	GSV-6 / GSV-8
4 Bytes	1 return values		
0	uint32_t (4)	Serial number	[1-99999999]. GSV-6 gives 0xFFFFFFFF or 0 if no serial number is set yet, GSV-8 d01234567 in that case

MEsetSerNo (0x20)

0x20	W	MEsetSerNo	GSV-6 / GSV-8
Write the device serial number ***Remark***: This command requires a MESYS unlocking.			
4 Bytes	1 Parameter		
0	uint32_t (4)	Serial number	[1-99999999]
0 Byte	0 return values		

Read Calibration Date (0x21)

0x21	R	ReadCalDate	GSV-8
Read the date of the last device calibration. Present with Firmware-version >= 1.40			
0 Byte	0 Parameter		
4 Bytes	4 Return values		
0	uint8_t(1)	Year	Year of the last calibration since 2000, i.e. add 2000 to the value to obtain the real year.
1	uint8_t(1)	Month	Month of the last calibration (1..12)
2	uint8_t(1)	Day	Day of month of the last calibration (1..31)
3	uint8_t(1)	Hour	Hour of the day of the last calibration (0..23)

Write Calibration Date (0x22)

0x22	W	WriteCalDate	GSV-8
Write the date of the last device calibration. Present with Firmware-version >= 1.40 ***Remark***: This command requires a MESYS unlocking.			
4 Bytes	4 Parameter		
0	uint8_t(1)	Year	Year of the last calibration since 2000, i.e. add 2000 to the value to obtain the real year.
1	uint8_t(1)	Month	Month of the last calibration (1..12)
2	uint8_t(1)	Day	Day of month of the last calibration (1..31)
3	uint8_t(1)	Hour	Hour of the day of the last calibration (0..23)
0 Byte	0 Rückgaben		

StopTransmission (0x23)

0x23	A	StopTrasmission	GSV-6 / GSV-8
Stop the permanent, autonomous measuring value transmission. This state is volatile, i.e. it doesn't persist after reboot. For persisting stopping the permanent value transmission, please use the command *SetTXmode.*			
0 Byte	0 Parameter		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x23	A	StopTrasmission	GSV-6 / GSV-8
0 Byte	0 return values		

StartTransmission (0x24)

0x24	A	StartTrasmission	GSV-6 / GSV-8
Start the permanent, autonomous measuring value transmission. This state is volatile, i.e. it doesn't persist after reboot. For persisting starting the permanent value transmission, please use the command *SetTXmode* (if it was stopped using this method).			
0 Byte	0 Parameter		
0 Byte	0 return values		

ClearBufferAbortTX (0x25)

0x25	A	ClearBufferAbortTX	GSV-8 (USB)
Clear the transmission buffer. Measuring value frames residing in the USB send queue are deleted from it. But a "pending transmission" in the USB device stack won't be aborted, so that value frames, that are just being send or about to be sent, are preserved. No frame will be splitted.			
0 Byte	0 Parameter		
0 Byte	0 return values		

GetMode (0x26)

0x26	R	GetMode	GSV-6
Read Mode-Flags. With GSV-6, these are system flags.			
0 Byte	0 Parameter		
4 Bytes	1 return values		
0	uint32_t (4)	Mode-Flags	GSV-6: <2:0> Number of channels enabled 0b001: 1 Channel ... 0b111: 7 Channels <3> Save-Tara <4> User-Monitor <5> Load offset from TEDS <6> Scale-Negate <7> Peak display <8> reset peak and tare <9> Read TEDS at boot <10> Multi-axis sensor calculation active <11> ClickRClackR menu output 0-20mA, if analog out type mA[4] All other bits are reserved

0x26	R	GetMode	GSV-8
Read Mode-Flags.			
0 Byte	0 Parameter		
4 Bytes	1 return values		

4 Available from version 3.29

0x26	R	GetMode	GSV-8
0	uint32_t (4)	Mode-Flags	GSV-8: <**0**> Multi-axis sensor calculation active <**1**> Set analogue filter automatically <**2**> Acquire maximum & minimum value <**3**> Noise-Cut suppression enabled <**7**> Lock all write functions <**15:8**>: If a sensor with TEDS is connected to the input channel BitNo-7, their TEDS data shall be used for the UserScale value (if possible) <**16**>: If corresponding bit in <15:8> is set, the unit shall be set as defined in the TEDS data (if possible) <**17**>: If corresponding bit in <15:8> is set, the input sensitivity shall be set according to the TEDS data (if possible) <**18**>: If corresponding bit in <15:8> is set, the analog output shall be scaled according to the TEDS data (if possible) <**19**>: If corresponding bit in <15:8> is set, the zero offset shall be used according to the TEDS data (if possible) <**22**>: Auto-Zero enabled (see Write AutoZero Setting (0x97)

SetMode (0x27)

0x27	W	SetMode	GSV-6 (Reset)
Write Mode-Flags			
4 Bytes	1 Parameter		
0	uint32_t (4)	Mode-Flags	GSV-6: <5> Load offset from TEDS (Fw >=3.19) <6> Scale-Negate (from FW-ver. 3.11) <7> Peak display (FW-ver. >=3.11) <9> Read TEDS at boot (FW-ver.>= 3.10) <10> Multi-axis sensor calculation active <11> ClickRClackR menu output 0-20mA, if analog out type mA (FW-ver. >=3.29) All other bits are ignored.
0 Byte	0 return values		

0x27	W	SetMode	GSV-8
Write Mode-Flags			
4 Bytes	1 Parameter		
0	uint32_t (4)	Mode-Flags	see GetMode (0x26) Bit 7 is not changeable (read-only).
0 Byte	0 return values		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

GetSoftwareConfiguration (0x2A)

0x2A	R	GetSoftwareConfiguration	GSV-8
This value indicates the existence of certain hardware-dependent features that have to correspond to similar featured of the device software. That's why this value is also displayed in the firmware update software. If the value is different between existing device software and the hex file of the new software, an update is not recommended (unless it's done after a hardware upgrade in consultation with the manufacturer).			
0 Byte	0 Parameter		
4 Bytes	1 return value		see Table

Table: Meaning BUILD-Val

Bit	Mask	Name	Meaning
0	1	HAS_ADC	AD-Converter present
1	2	HAS_ETHERCAT	EtherCAT field Bus present
2	4	HAS_LCD	LC-Display present
3	8	HAS_TEDS	TEDS-Sensor support present
4	16	HAS_DIGI_IN_OUT	Digital In- and outputs present
5	32	HAS_ETH_TWOLEDS	With EtherCAT: Run/Err-LED separated
6	64	HAS_ANALOG_OUT	Analogue outputs present
7	128	HAS_SERIAL	Serial Interface present (e.g. via Ethernet)
8	256	HAS_FREQ_OUT	Frequency modulated analogue output present
9	512	HAS_AIN_MCU	Sensor supervisor at analogue inputs present
10	1024	HAS_SIXAXIS	Multi axis sensors are supported
11	2048	HAS_CANOPEN	CANopen field Bus present
12	4096	IS_HIGHSPEED	Is a 4-Channel High-Speed device with Fdata,max= 96000/s

0x2A	R	GetSoftwareConfiguration	GSV-6
This value indicates the presence of hard- and software-dependend features. Present from FW-ver 3.29			
0 Byte	0 Parameter		
4 Byte	1 return value		see table

Table: Meaning equipment value

Bit	Wert (hex)	Bedeutung
0..3		Number of thershold switch outputs
4	0x10	Signal-Conditioner f. Analogausgang present, i.e. ouput types 0..4 and 6 usable
5	0x20	Quadrature encoder present, i.e. channel 7 f. Counter/Frequency usable
6	0x40	Real-Time clock (RTC) present
8	0x100	SD-card slot present
9	0x200	CAN-Bus Transceiver present
10	0x400	GSV-6 BT device
11	0x800	Embedded Linux / hand-held device

Bit	Wert (hex)	Bedeutung
0..3		Number of thershold switch outputs
4	0x10	Signal-Conditioner f. Analogausgang present, i.e. ouput types 0..4 and 6 usable
5	0x20	Quadrature encoder present, i.e. channel 7 f. Counter/Frequency usable
6	0x40	Real-Time clock (RTC) present
16	0x10000	reserved (=1)
17	0x20000	Temperature sensor present
18	0x40000	I2C Bus present (CPU-Software feature)
19	0x80000	CAN Bus vorhanden (CPU-Software feature)
20	0x100000	Monitor-Interface switchable to RS232 Port (CPU-Software feature)
21	0x200000	GSV protocol present (CPU-Software feature, always =1)
22	0x400000	DA converter present (CPU-Software feature)
23	0x800000	SPI port present (CPU-Software feature)
27	0x8000000	Is Low-Power special device

FirmwareVersion (0x2B)

0x2B	R	FirmwareVersion	GSV-6 / GSV-8
Read Firmware Version			
0 Byte	0 Parameter		
4 Bytes	2 return values		
0	uint16_t (2)	Major version no.	VER_MAJOR
1	uint16_t (2)	Minor version no / revision	VER_MINOR

Get Prohibit Set Zero (0x32)

0x32	R	GetProhibutSetZero	GSV-8
Setting to zero with Cmd. 0x0C and by key or field bus is rejected if the flag corresponding to the input channel is set (Bit 0: Ch1 and so on to Bit 9: Counter 2). From FW-ver. 1.54			
0 Byte	0 Parameter		
2 Bytes	1 Return value		
0	uint16_t (2)	Zero Prohibit Flags	Bits 0-9: SetZero prohibited at channel 1-10

Write Prohibit Set Zero (0x33)

0x33	W	WriteProhibutSetZero	GSV-8
Setting to zero with Cmd. 0x0C and via key or field bus can be permanently disabled. Requirement: User ID set. From FW ver. 1.54			
2 Bytes	1 Parameter		
0	uint16_t (2)	Zero Prohibit Flags	Bits 0-9: SetZero prohibited at channel 1-10
0 Byte	0 return values		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

MEwriteInputRange (0x34)

0x34	W	MEwriteInputRange	GSV-6 (Reset)
Set input sensitivity ***Advice***: With GSV-6, this cannot be set for individual channels ***Advice***: With Firmware-versions < 3.11 this command requires a MESYS unlock!			
6 Bytes	3 Parameter		
0	uint8_t (1)	Channel	[0] Channel shall be =0: Set parameters of all channels to the same value.
1	uint8_t (1)	SensIndex	unused (pass =0)
2	uint32_t (4)	Input sensitivity	Input sensitivity in mV*100 Possible values are: 800: 8mV/V 400: 4mV/V 200: 2mV/V 100: 1mV/V
0 Byte	0 return values		

0x34	W	MEwriteInputRange	GSV-8
Manufacturer setting: Write communicated value for the input sensitivity (=Input measuring range). ***Advice***: This command requires a MESYS unlock!			
6 Bytes	3 Parameter		
0	uint8_t (1)	Channel	[0-8] Channel =0: Set parameters of all channels to the same value.
1	uint8_t (1)	SensIndex	[1-4] Input type index: - 0x01: Range for bridge input Us= 8,75V - 0x02: Range for bridge input Us= 5V - 0x03: Range for bridge input Us= 2,5V - 0x04: Range for bridge input +-10V
2	uint32_t	Input sensitivity	Input sensitivity in mV*100 Standard version: 2mV/V = 200, 3,5 ->350 7 -> 700. +-10V -> 1000000
0 Byte	0 return values		

SetInjectValOrOffset (0x35)

0x35	A	Set Inject Val or Offset	GSV-8
Emulate input measuring values or load zero signal offset values			
1 Byte	1 Parameter		

0x35	A	Set Inject Val or Offset	GSV-8
0	uint8_t (1)	Index	**0:** Reset all Inject Values or Offsets to normal mode **1:** Half of the nominal full scale value (raw value: 0x003CF3CF) is set at all channels as a pseudo measuring value and the measuring application is decoupled from the AD converter. Useful for testing purpose. **2:** The offset values that were determined at the last SetZero procedure aren't used; instead, the manufacturer offset calibration values valid for exactly 0mV/V at the inputs, are loaded. With sensor at zero physical load connected, its basic bridge deviation can be detected (AD converter remains active). **3:** Similar to 2., but the offset calibration values of the multi-axis sensor calibration data are loaded (if present). That way, at physical zero load of the sensor its basic bridge deviation difference from the calibration time can be observed, which can be taken as a degree of wear-and-tear of the sensor.
0 Byte	0 return values		

GetHardwareVersion (0x36)

0x36	R	GetHardwareVersion	GSV-6 / GSV-8
Read Hardware Options (GSV-6: always reads 0)			
0 Byte	0 Parameter		
4 Bytes	1 return value		
0	uint32_t (4)	Options (GSV-8)	Bits<31:16>: Hardware version of the main board. Hardware-Releases, that require a newer or different software-release, have a different number. Bits<15:0>: Lower byte of the hardware-version. Reserved for daughter-boards, that have changes, which are relevant for the device-software.

GetCustomSerNo (0x37)

0x37	R (Serial)	GetCustomSerNo	GSV-6
Request 2nd serial number. Advice: There is a distinction with this command that is interface-specific.			
0 Byte	0 Parameter		
9 Bytes	2 return values		
0	uint8_t (1)	Indicator	Serial: always =0
1	char[8]	Ser.No.-Text	The Ser.No. as ASCII text (NULL-terminated or exactly 8 characters long)

0x37	R (CAN)	GetCustomSerNo	GSV-6
Request 2nd serial number. Advice: There is a distinction with this command that is interface-specific. Warning: **Two** response packets are sent to one request.			
0 Byte	0 Parameter		
5 Bytes	2 return values		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x37	R (CAN)	GetCustomSerNo	GSV-6
0	uint8_t (1)	Indicator	=1 ‚First half‘ =2 ‚Second Half‘
1	char[4]	Ser.No.-Text	The Ser.No. as ASCII text (NULL-terminated or exactly 8 characters long)

The commands 0x37-0x39 are available from firmware version 3.33. If no second Ser.No is stored, an empty string is output (all text bytes =0). If one is saved, it is also output with the GetSerNo 0x1F command, provided that all bytes of the custom no. are stored as ASCII numbers (big endian). The first Ser.Nr is always read with GetManufacturerSerNo 0x39. Typical cases of use are "smart sensors" with built-in GSV-6, where the custom serial no. corresponds to the printed serial number of the product.

SetCustomSerNo (0x38)

0x38	W (Serial)	SetCustomSerNo	GSV-6
Set the 2nd serial number (maximum 8 chars). Requires a MESYS unlock. Similar to the request, there is also a difference when setting.			
9 Bytes	2 Parameter		
0	uint8_t (1)	Indicator	Serial: always =0
1	char[8]	Ser.No.-Text	The Ser.No. as ASCII text (NULL-terminated or exactly 8 characters long)
0 Byte	0 return values		

0x38	W (CAN)	SetCustomSerNo	GSV-6
Write a user-defined unit string (maximum of 8 chars) In order for the string gets stored, first the command with the 1st half, then the 2nd half always has to be sent to the device! If the 1st half will be set only, it remains in the RAM of the device. If only the 2nd half will be set, the device responds with an error message			
5 Bytes	2 Parameter		
0	uint8_t (1)	Indicator	=1 ‚First half‘ =2 ‚Second Half‘
1	char[4]	Ser.No.-Text	The Ser.No. as ASCII text (NULL-terminated or exactly 8 characters long)
0 Byte	0 Rückgaben		

This command can only be executed by the manufacturer and is only available from firmware version 3.33.

GetManufacturerSerNo (0x39)

0x39	R	GetManuf.SerNo	GSV-6
Request for the first serial number. The command is used to distinguish if a CustomSerNo is set. In general, the GetSerNo (0x1F) command is sufficient to read the product serial number.			
0 Byte	0 Parameter		
4 Bytes	1 return value		
0	uint32_t (4)	Serial number	[1-99999999]. If no serial no is set yet, the response is 0xFFFFFFFF or 0.

GetRawValue (0x3A)

0x3A	R	GetRawValue	GSV-6 / GSV-8
Read raw measuring value of an input			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	[1-7] / [1-10]
4 Bytes	1 return values		
0	int32_t (4)	Raw value	Raw value

GetValue (0x3B)

0x3B	R	GetValue	GSV-6 / GSV-8
Trigger the transmission of a measuring value frame. ***Caution***: This command exceptionally gives **no** response frame, therefore, no error code. Instead, a measured value frame is transmitted, if the permanent value transmission is disabled.			
0 Byte	0 Parameter		
0 Byte	0 return values		

ClearMaxValue (0x3C)

0x3C	A	ClearMaxValue	GSV-6 / GSV-8
Clear peak values memory, i.e. maximum- and minimum value(s)			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel-No: GSV-6: [0-6] / [0-7] GSV-8: [0-8] / [0-10] Channel =0: Clear all max.-and min- values
0 Byte	0 return values		

GetLastProtokollError (0x42)

0x42	R	GetLastError	GSV-6 / GSV-8
GSV-6: Read last protocol error. The (synchronous) protocol error is the error code sent in the response frame. An asynchronous protocol error is an error code, that occurred while sending measuring value frames autonomously. GSV-8: An asynchronous error may be an error, that occurred at the reception of a request frame, when fundamental protocol requirements were not met, so that the request was ignored. A request frame with such severe faults isn't responded exceptionally. The error code is ERR_FRAME_ERROR in that case. Or, it may be a loss of measuring value frames in the USB transmit queue, so that older, not yet transmitted frames were overwritten by newer. This may happen, if the USB communication is fully loaded at high data frequencies (>= 16000/s), e.g. because of many command requests; or when using several USB devices at the same USB hub, where the GSV-8 is connected to. In both cases, the error code is ERR_RET_TXBUF.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: (synchronous) protocol error 1: A synchronous protocol error All other values reserved.
4 Byte	1 return values		
0	uint32_t	Error-Code	see table p.17

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

GetLastValueError (0x43)

0x43	R	GetLastValueError	GSV-6
Read last fault related to measuring value(s)			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: Fault bits Other values without function
6 Bytes	1 return values		
0	uint8_t[6]	Fault	Index 0: - Fault [0] Fault bits (maximum value exceedance with multi-axis sensor) - Fault [1] Fault bits (Input value saturation) - Fault [2..5] =0, see p. 18

0x43	R	GetLastValueError	GSV-8
Read last fault related to measuring value(s), see Table above			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: Faults counter 1-40: Fault structure
6 Bytes	1 return values		

0x43	R	GetLastValueError	GSV-8
0	uint8_t [6] ErrorStruct	Error	**Index 0:** Read the number of faults that occurred since the last power-on session and the number of faults stored in the non-volatile fault memory. Response paramter: Only the 2 "highest" bytes of the 6 returned bytes are meaningful, i.e. the MSbyte and the one that follows it (if see as pseudo-numerical interpretation), the lower 4 bytes are =0. The first byte ("MSbyte") is the number of faults since the actual power-on session. The following byte is the number of faults stored in memory. **Index 1:** ErrorStruct since Power On: Fault that recently occurred and that is not yet stored to non-volatile memory. **Indices 2 to 83:** ErrorStruct of faults stored in non-volatile memory. Higher indices correspond to newer faults, lower to older ones. Definition of the ErrorStruct in C-Notation: { unsigned short ErrFlags:16; unsigned long ErrTime:29; unsigned ErrType:3; } ErrorStruct; (see see Table above for meanings).

EraseErrorMemory (0x44)

0x44	W / A	EraseErrorMemory	GSV-8
Erase the non-volatile fault memory. The USER_ID_LEVEL must be set for successful execution.			
0 Byte	0 Parameter		
0 Byte	0 return values		

GetSensorPlugged (0x45)

0x45	R	GetSensorPlugged	GSV-8
Determine, if a sensor is connected to an analogue input (not usable with Single-ended Input).			
0 Byte	0 Parameter		
4 Bytes	1 return values		
0	uint32_t (4)	Flag value	Bit 0: Channel 1 … Bit 7: Channel 8 Bit=1: Bridge sensor connected, Bit=0: No bridge sensor connected Bits<15:8>: TEDS device connected[5] Channel 8:1. Bits<23:16>: TEDS writeable Channel 8:1. Bits<31:24>: Reserved for coming sensor properties

5 Independently of whether the TEDS device (1wire EEPROM) contains meaningful data. The latter can be determined with the command GetTEDSactive (if the corresponding bit is set in the Mode-Variable<15:8>)

ReadFTSensorCal (0x47)

0x47	R	ReadFTSensorCal	GSV-6 / GSV-8
Read a calibration value of the **F**orce/**T**orque-multi-axis sensor, e.g. a calibration matrix element, serial number, scaling factor of the calibration matrix, sensor maximum value or input sensitivity. **Advice**: 1. Several calibration structures for multi-axis sensors are supported (with GSV-8 up to 5). for reading values from a particular structure, that one should be selected by PrepReadFTsensor before. A 2-dimensional calibration matrix is put row-wise in the array (a11, a12, a13, … a21, …), so that index={0..5} addresses the row vector a11..a15, index={6..11} the row vector a21..a25 and so on.			
2 Bytes	2 Parameters		
0	uint8_t (1)	Type	0: Serial number (index must be =0) uint32 1: Matrix scaling factor (index must be =0) 2: Matrix A values (index: [0-35]), 1st order 3: Geometrical offsets x,y,z (index: [0-2]) 4: Maximum values (index: [0-5]) 5: Input sensitivity (index must be =0) 6: Basic bridge deviation at zero load (GSV-8 only) 7: Sensor type (GSV-8, FWver >= 1.39) Type=0: Standard F/T 6-axis sensor, only matrix A used Type=1: F/T 6-axis sensor with matrix B, 1st and 2nd order matrix used Type=2: 3/4 axis sensor (reserved) 8: Matrix B values (index: [0-35]), 2nd order (GSV-8, FWver >= 1.39) 9: Indices of the 1st input factors (GSV-8, FWver >= 1.39) 10: Indices of the 2nd input factors (GSV-8, FWver >= 1.39) 11: Enum of the unit, see Cmd. 0x0F p.84 (GSV-8, FWver >= 1.54), Index [0-5] 12: Calibration date: index 0..2: Year, Month, Day (GSV-8, FWver >= 1.54)
1	uint8_t (1)	Index	Index of the array addressed by *type*
4 Bytes	1 return values		
0	uint32_t (4) / float (4)	Serial number / Values	With exception of the serial no (uint32_t), all entries are of data type float

WriteFTSensorCal (0x48)

0x48	W (RAM)	WriteFTSensorCal	GSV-6 / GSV-8
Write a calibration value of the **F**orce/**T**orque-multi-axis sensor, e.g. a calibration matrix element, serial number, scaling factor of the calibration matrix, sensor maximum value or input sensitivity. **Advice**: 1. A 2-dimensional calibration matrix is put row-wise in the array (a11, a12, a13, … a21, …), so that index={0..5} addresses the row vector a11..a15, index={6..11} the row vector a21..a25 and so on. 2. GSV-8: Precondition is USER_ID_LEVEL set 3. For storing the data persistently, the command *StoreSensorCal* (0x7F) must be issued, after all values of a calibration struct have been set. The matrix B values are relevant for Sensor type =1 only.			
6 Bytes	3 Parameters		

0x48	W (RAM)	WriteFTSensorCal	GSV-6 / GSV-8
0	uint8_t (1)	Type	0: Serial number (index must be =0) uint32 1: Matrix scaling factor (index must be =0) 2: Matrix A values (index: [0-35]), 1st order 3: Geometrical offsets x,y,z (index: [0-2]) 4: Maximum values (index: [0-5]) 5: Input sensitivity (index must be =0) 6: Basic bridge deviation at zero load (GSV-8 only) 7: Sensor type (GSV-8, FWver >= 1.39) Type=0: Standard F/T 6-axis sensor, only matrix A used Type=1: F/T 6-axis sensor with matrix B, 1st and 2nd order matrix used Type=2: 3/4 axis sensor 8: Matrix B values (index: [0-35]), 2nd order (GSV-8, FWver >= 1.39) 9: Indices of the 1st input factors (GSV-8, FWver >= 1.39) 10: Indices of the 2nd input factors (GSV-8, FWver >= 1.39) 11: Enum of the unit, see Cmd. 0x0F p.84 (GSV-8, FWver >= 1.54), Index [0-5] 12: Calibration date: index 0..2: Year, Month, Day (GSV-8, FWver >= 1.54)
1	uint8_t (1)	Index	Index of the array addressed by *type*
2	uint32_t (4) / float (4)	Serial number / Values	With exception of the serial no (uint32_t), all entries are of data type float
0 Byte	0 return values		

GetTXmapping (0x49)

0x49	R	GetTXMapping	GSV-8 / GSV-6
Read assignment and physical meaning of the values in the measuring data frame. Remark: GSV-8: The indices 1..[no. of mapped objects] are not yet implemented. By now (FWver <=1.38), at index 0 a constant 0 is returned. GSV-6: Implemented as described from FW-Ver. 3.11			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: Determine number of input channels in measuring data frame [1..<No. of mapped objects>] Number of the mapping entry, i.e. position of the object in the measuring value frame
2 Bytes	1 return value		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x49	R	GetTXMapping	GSV-8 / GSV-6
0	uint16_t (2)	Number / Mapping entry	If =0: No mapping defined. >0: If Index=0: Number of mapped objects. Else: Mapping entry as follows: Bits<15:8>, i.e. 1st data byte of the response Physical type. Existing definitions: 0x00: No physical type defined 0x01: Force in X direction (mainly with Six-axis sensors) 0x02: Force in Y direction (mainly with Six-axis sensors) 0x03: Force in Z direction (mainly with Six-axis sensors) 0x04: Torque in X direction (mainly with Six-axis sensors) 0x05: Torque in Y direction (mainly with Six-axis sensors) 0x06: Torque in Z direction (mainly with Six-axis sensors) 0x10: Raw value of six-axis sensor (before calculation) 0x20: Temperature 0x26: Digital inputs, to be interpreted as integer number (bits) 0x28: Quadrature-Encoder-Pulses 0x38: Quadrature-Encoder-Speed Bits<7:4>: Actual value / maximum- / minimum value: NORMAL=0x00: The value-object is an actual value MAX_VAL=0x01: The value-object is a maximum value MIN_VAL=0x02: The value-object is a minimum value Bits<3:0>: Input channel number: 1..8: Analog sensor input (GSV-8) 1..6: Analog sensor input (GSV-6) 7: Quadrature encoder pulse input (GSV-6 BT) 9: Quadrature encoder pulse input No. 1 (GSV-8) 10: Quadrature encoder pulse input No. 2 (GSV-8) 11: Digital input (GSV-8), future function

SetTXmapping (0x4A)

0x4A	W	SetTXmapping	GSV-8 / GSV-6
Set assignment to measuring value frame GSV-6: Implemented with index=0 from FW-Ver. 3.11 GSV-8: Implemented with index=0 from FW-Ver. 1.45			
3 Bytes	2 Parameter		
0	uint8_t (1)	Index	Index =0: Write the number of mapped input channels Index >0 [1-16]: Write mapping entry (reserved)
1	uint16_t (2)	Number / Mapping entry	Number of mapped input channels in the measuring data frame (Index=0) or mapping entry, as described in GetTXmapping.
0 Byte	0 return values		

Value ranges SetTXmapping at Index 0:

GSV-6: Only values 1,2,3,6 (without Counter). From FW.ver 3.33: 1-6

GSV-6BT: With Counter: Only values 2,3,4,7. From FW.ver 3.33: 2-7. Without Counter: see GSV-6

GSV-8 (from FW.ver 1.45): Without Counter: USB: 2..8, UART: 1..8

GSV-8 (from FW.ver 1.45): With 1 Counter channel: 2..9
GSV-8 (from FW.ver 1.45): With 2 Counter channels: 3..10

GetDigiFiltType (0x4B)

0x4B	R	GetDigiFiltType	GSV-6 / GSV-8
Read type of digital filter			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [1-6] GSV-8: [1-8]
1	uint8_t (1)	Sub Type	GSV-6: Unused, shall be =0 GSV-8: 0x00: IIR-Filter 0x80 FIR-Filter (GSV-8 only)
1 Byte	1 return values		
0	uint8_t (1)	Filter type enum	<7>: FIR-Filter-Indicator: =1 FIR filter, =0 IIR <6:4> Filter type: -0: Low pass -1: High pass -2: Band pass -3: Band stop -4: Notch filter (GSV-8) <3:0> Filter length Constant =4 for IIR and for FIR one of the values: 4,6,8,10,12,14

SetDigiFiltType (0x4C)

0x4C	W (RAM)	SetDigiFiltType	GSV-6 / GSV-8
Write type of digital filter ***Advice***: 1. To store the written filter parameters in non-volatile memory, *StoreDigiFilt* (0x7E) has to be issued. That should be done once, after all parameters (coefficients, type, cut-off frequency) were written for all filters that are desired to be configured. 2. The IIR filter is only usable with GSV-8			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-6] GSV-8: [0-8] Channel=0: Set parameters of all channels to the same value.
1	uint8_t (1)	Filter type enum	<7>: FIR-Filter-Indicator: =1 FIR filter, =0 IIR <6:4> Filter type: -0: Low pass -1: High pass -2: Band pass -3: Band stop -4: Notch filter (GSV-8) <3:0> Filter length Constant =4 for IIR and for FIR one of the values: 4,6,8,10,12,14

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x4C	W (RAM)	SetDigiFiltType	GSV-6 / GSV-8
0 Byte	0 return values		

ReadDigiFiltCutOff (0x4D)

0x4D	R	ReadDigiFiltCutOff	GSV-6 / GSV-8
Read -3dB cut-off frequency of a digital filter This returns the informative values that have been written by WriteDigiFiltCutOff (0x4E)			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [1-6] GSV-8: [1-8]
1	uint8_t (1)	Index	0: Fcl (lower cut-off frequency) 1: Fco (upper cut-off frequency)
4 Bytes	1 return values		
0	float (4)	Cut-off frequency	

WriteDigiFiltCutOff (0x4E)

0x4E	W (RAM)	WriteDigiFiltCutOff	GSV-6 / GSV-8
Write one of the two -3dB cut-off frequency of a digital filter This is an informative value, that has no effect on the filter itself. With low pass and high pass, only Fcl is valid. ***Advice***: To store the written filter parameters in non-volatile memory, *StoreDigiFilt* (0x7E) has to be issued. That should be done once, after all parameters (coefficients, type, cut-off frequency) were written for all filters that are desired to be configured.			
6 Bytes	3 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-6] GSV-8: [0-8] Channel=0: Set parameters of all channels to the same value.
1	uint8_t (1)	Index	0: Fcl (lower cut-off frequency) 1: Fco (upper cut-off frequency)
2	float (4)	Cut-off frequency	
0 Byte	0 return values		

ReadDigiFiltCoeff (0x4F)

0x4F	R	ReadDigiFiltCoeff	GSV-6
Read digital filter coefficients			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-6]
1	uint8_t (1)	Index	Only with IIR: 0x00...0x04: Coefficients A (Input) 0x10...0x13: Coefficients B (Output, feedback)
4 Bytes	1 return values		
0	float (4)	Coefficient	

0x4F	R	ReadDigiFiltCoeff	GSV-8
Read digital filter coefficients			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-8]
1	uint8_t (1)	Index	With IIR: 0x00...0x04: Coefficients A (Input) 0x10...0x13: Coefficients B (Output, feedback) With FIR: -0x80...0x87: Coefficients (up to and including centre of the symmetrical coefficients sequence only)
4 Bytes	1 return values		
0	S7.24 (4)	Coefficient	

WriteDigiFiltCoeff (0x50)

0x50	W (RAM)	WriteDigiFiltCoeff	GSV-6
Write digital filter coefficients[6] ***Advice***: To store the written filter parameters in non-volatile memory, *StoreDigiFilt* (0x7E) has to be issued. That should be done once, after all parameters (coefficients, type, cut-off frequency) were written for all filters that are desired to be configured.			
6 Bytes	3 Parameter		
0	uint8_t (1)	Channel	Channel No. [0-6] Channel =0: Set values for all channels to the same value.
1	uint8_t (1)	Index	Only with IIR: 0x00...0x04: Coefficients A (Input) 0x10...0x13: Coefficients B (Output, feedback)
2	float (4)	Coefficient	
0 Byte	0 return values		

0x50	W	WriteDigiFiltCoeff	GSV-8
Write digital filter coefficients[2] ***Advice***: To store the written filter parameters in non-volatile memory, *StoreDigiFilt* (0x7E) has to be issued. That should be done once, after all parameters (coefficients, type, cut-off frequency) were written for all filters that are desired to be configured.			
6 Bytes	3 Parameter		
0	uint8_t (1)	Channel	Channel No. [0-8] Channel=0: Set parameters of all channels to the same value.
1	uint8_t (1)	Index	With IIR: 0x00...0x04: Coefficients A (Input) 0x10...0x13: Coefficients B (Output, feedback) With FIR: -0x80...0x87: Coefficients (up to and including centre of the symmetrical coefficients sequence only)
2	S7.24 (4)	Coefficient	

6 The determination of these coefficients cannot be done by the device, nor is it subject of this document. However, the Windows DLL MEGSV86xx.dll provides a function to calculate and set the filter coefficients.

0x50	W	WriteDigiFiltCoeff	GSV-8
0 Byte	0 return values		

GetDfiltOnOff (0x51)

0x51	R	GetDfiltOnOff	GSV-6
Determine which digital filters are enabled (Bit mask) Bit =1 means: Filter is enabled			
0 Byte	0 Parameter		
4 Bytes	1 return values		
0	uint32_t (4)	Channel Flags	GSV-6: <5:0> Filter at the corresponding channel (=BitNo+1) enabled <31:6> reserved

0x51	R	GetDfiltOnOff	GSV-8
Determine which digital filters are enabled (Bit mask) Bit =1 means: Filter is enabled			
0 Byte	0 Parameter		
4 Bytes	1 return values		
0	uint32_t (4)	Channel Flags	GSV-8: <7:0> IIR filter at the corresponding channel (=BitNo+1) enabled <15:8> FIR Filter at the corresponding channel (=BitNo-7) enabled <31:16> reserved

SetDfiltOnOff (0x52)

0x52	W	SetDfiltOnOff	GSV-6
Enable / disable digital filter (bit mask) Bit =1 means: Enable filter			
4 Bytes	1 Parameter		
0	uint32_t (4)	Channel Flags	GSV-6: <5:0> Filter at the corresponding channel (=BitNo+1) enabled <31:6> reserved
0 Byte	0 return values		

0x52	W	SetDfiltOnOff	GSV-8
Enable / disable digital filter (bit mask) Bit =1 means: Enable filter			
4 Bytes	1 Parameter		
0	uint32_t (4)	Channel Flags	GSV-8: <7:0> Enable IIR filter at the corresponding channel (=BitNo+1) <15:8> Enable FIR Filter at the corresponding channel (=BitNo-7) <31:16> reserved
0 Byte	0 return values		

ReadMaxMinVal (0x53)

0x53	R (Serial)	ReadMaxMinVal	GSV-6
Read maximum (peak) and minimum measuring value			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-6] / [1-7]
9 Bytes	3 return values		
0	uint8_t (1)	Indicator	=0
1	float (4)	Maximum value	
5	float (4)	Minimum value	

0x53	R (CAN)	ReadMaxMinVal	GSV-6
Read maximum (peak) and minimum measuring value **Advice**: Two response frames are transmitted: First the maximum value, then the minimum value			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-6] / [1-7]
5 Bytes	2 return values		
0	uint8_t (1)	Indicator	=1: Maximum value =2: Minimum value
1	float (4)	Max./min. value	

0x53	R (Serial)	ReadMaxMinVal	GSV-8
Read maximum (peak) and minimum measuring value			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-8] / [1-10]
8 Bytes	2 return values		
0	float (4)	Maximum value	
1	float (4)	Minimum value	

GetFTSensorCalArrNo (0x54)

0x54	R	GetFTSensorCalArrNo	GSV-6 / GSV-8
Request information about multi-axis sensor calibration structures			
0 Byte	0 Parameter		
3 Bytes	3 return values		
0	uint8_t (1)	Maximum	Maximum number of calibration structs available in the device
1	uint8_t (1)	NumActive	Number of structs stored N
2	uint8_t (1)	Struct Index	Index of the active structure [0...N-1]

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

SetFTSensorCalArrNo (0x55)

0x55	W	SetFTSensorCalArrNo	GSV-6 (Reset) / GSV-8
Write the active struct for multi-axis sensor. Calibration data must be present at this index!			
1 Byte	1 Parameter		
0	uint8_t (1)	Struct Index	[0...N-1]
0 Byte	0 return values		

ReadDeviceHours (0x56)

0x56	R	ReadDeviceHours	GSV-8 / GSV-6BT
Read one of the two device hours counter The „absolute“ device hours counter is continuous and cannot be cleared. The „adjustable“ device hours counter is continuous, as long as it isn't set by the command WriteDeviceHours (0x57) to a new value. GSV-6: Implemented from FW-Ver. 3.11, if hardware properly equipped (otherwise: error return)			
1 Byte	1 Parameter		
0	uint8_t (1)	Counter index	0: Absolute 1: Settable Other values reserved
4 Bytes	1 return value		
0	float (4)	Counter	Device hours of device

WriteDeviceHours (0x57)

0x57	W	WriteDeviceHours	GSV-8
Write the adjustable device hours counter. Precondition is, that the USER_ID_LEVEL is set. GSV-6: Implemented from FW-Ver. 3.11, if hardware properly equipped (otherwise: error return)			
1 Byte	1 Parameter		
0	float (4)	Value	Value of the adjustable device hours counter in hours.
0 Byte	0 return values		

SetPassword (0x58)

0x58	W	SetPassword	GSV-8
Change the user password **Advice**: Precondition is, that the USER_ID_LEVEL is set, i.e. give the old password before.			
4 Bytes	1 Parameter		
0	Char[4]	Value	New password. Must be 4 ASCII chars long.
0 Byte	0 return values		

GetDIOdirection (0x59)

0x59	R	GetDIOdirection	GSV-8
Determine the direction (1=Input, 0=Output) of a group of digital I/O lines (see text below)			
1 Byte	1 Parameter		
0	uint8_t (1)	Group-No.	Number of the group [1-4]. =0: Read direction of all 4 groups.
1 Byte	1 return value		=0: Output, =1: Input. If Group-No=0: In Bit 0: Direction of Group-No No 1… Bit 3: Direction of Group-No No 4

SetDIOdirection (0x5A)

0x5A	W	SetDIOdirection	GSV-8
Set the direction (1=Input, 0=Output) of a group of digital I/O lines (see text).			
2 Bytes	2 Parameters		
0	uint8_t (1)	Group-No.	Number of the group [1-4]. =0: Set direction of all 4 groups.
1	uint8_t (1)	Data Direction	=0: Output, =1: Input. If Group-No=0: In Bit 0: Direction of Group-No No 1… Bit 3: Direction of Group-No No 4
0 Byte	0 return values		

The digital I/O lines are organized in groups of 4 lines each. I/O lines 1-4 belong to group 1, 5-8 to group 2, 9-12 to group 3, 13-16 to group 4. Inside a group, all 4 lines have the same direction. This also limits the configuration variety of I/O types.

GetDIOtype (0x5B)

0x5B	R	GetDIOtype	GSV-8 / GSV-6
Determine the type of a digital I/O line. GSV-6: present from FWver 3.27			
1 Byte	1 Parameter		
0	uint8_t (1)	DIO No	Number of the DIO line [1-16] (GSV-8) GSV-6: [1-5].
4 Bytes	2 return values		
0	U24 (3)	DIOtype	Type of the DIO line (see table Annex B, p. 85)
1	uint8_t (1)	Assigned channel	Assigned input channel GSV-8: [1-8] / GSV-6: [1-6] (relevant with threshold switch and single-tare types only)

SetDIOtype (0x5C)

0x5C	W	SetDIOtype	GSV-8
Set the type of a digital I/O line. Advice: Eventually, the data direction must be set first with *SetDIOdirection*. In the table in Annex B, the direction is noted.			
5 Bytes	3 Parameters		
0	uint8_t (1)	DIO No.	Number of the DIO line [1-16]. Value=0 means: Set all lines to the same type.
1	U24 (3)	DIOtype	Type of the DIO line (see table Annex B, p. 85)
2	uint8_t (1)	Assigned channel	Assigned input channel [1-8] (relevant with threshold switch and single-tare types only)
0 Byte	0 return values		

0x5C	W	SetDIOtype	GSV-6
Set the type of a digital I/O line. Present from FWver 3.27 Remark: In- and output finctions are assigned to hardware lines (pins) as follows: DIO-Nr. 1-3: SW1,SW2,SW3[7]: Output functions only DIO-Nr. 4: TRIGGER: Input functions only DIO-Nr. 5: TARA: Input functions only			
5 Bytes	3 Parameters		

7 SW3 isn't present with some models, e.g. GSV-6BT. That can be determined by Cmd. 0x2A.

0x5C	W	SetDIOtype	GSV-6
0	uint8_t	DIO No	Number of DIO line [1-5].
1	Uint_24	DIOtype	Type of the DIO line (see table Annex B, p. 85)
2	uint8_t	Assigned channel	Assigned input channel [1-6] (relevant with threshold switch and single-tare type only)
0 Byte	0 Rückgaben		

GetDIOlevel (0x5D)

0x5D	R	GetDIOlevel	GSV-8 / GSV-6
Read the level of a digital I/O line. GSV-6: present from FWver 3.29			
1 Byte	1 Parameter		
0	uint8_t (1)	DIO No	Number of the DIO line. GSV-8: [1-16] / GSV-6: [1-5]. =0: Get level for all DIO lines
2 Bytes	1 return values		
0	uint16_t (2)	DIOlevel	Level of the DIO line: 0: Low, 1: High. If DIOno=0: Get all 16 lines by flags: Bit 0: Line No. 1 (1.1.) ... Bit15: GPIO No 16 (4.4.) GSV-6: Bit 0: No 1 (SW1)... Bit 3: No 4 (TRIG), Bit 4: DIO No 5 (TARA)

SetDIOlevel (0x5E)

0x5E	W	SetDIOlevel	GSV-8
Set the level of a digital I/O line (if data direction = output and type= GP-out)			
3 Bytes	2 Parameter		
0	uint8_t (1)	DIO No	Number of the DIO line [1-16] =0: Set level for all DIO lines
1	uint16_t (2)	DIOlevel	Level of the DIO line: 0: Low, 1: High. If DIOno=0: Set all lines by flags: Bit 0: Line No. 1 (1.1.) ... Bit15: GPIO No 16 (4.4.)
0 Bytes	0 return values		

0x5E	W	SetDIOlevel	GSV-6
Set the level of a digital out line, if type= GP-out. Present from FWver 3.27. The Number of the highest available output-No can be determined with Cmd. 0x2A. With GSV-6BT, it's 2, otherwise, mostly 3.			
3 Bytes	2 Parameter		
0	uint8_t	DIO No	Number of DO line [1-3/2]. =0: Set level of all DO-lines.
1	uint16_t	DIOlevel	Level of DO line: 0: Low, 1: High. If DIO No=0: Bit 0: DO No. 1 (SW1).. Bit2: DO No 3 (SW3)
0 Bytes	0 return values		

ReadDIOthreshold (0x5F)

0x5F	R	ReadDIOthreshold	GSV-8 / GSV-6
Read the threshold value of a digital threshold switch output. GSV-6: present from FWver 3.29			
2 Bytes	2 Parameter		
0	uint8_t (1)	DIO No	Number of the DIO line. GSV-8: [1-16], GSV-6: [1-3]
1	uint8_t (1)	Upper / lower	=0: Lower threshold value, =1: Upper threshold

0x5F	R	ReadDIOthreshold	GSV-8 / GSV-6
4 Bytes	1 return value		
0	Float (4)	DIOthreshold	Threshold value of threshold switch

WriteDIOthreshold (0x60)

0x60	W	WriteDIOthreshold	GSV-8 / GSV-6
Write the threshold value of a digital threshold switch output. GSV-6: present from FWver 3.29			
6 Bytes	3 Parameter		
0	uint8_t (1)	DIO No	Number of the DIO line. GSV-8: [1-16], GSV-6: [1-3] =0: Set all to the same value.
1	uint8_t (1)	Upper / lower	=0: Lower threshold value, =1: Upper threshold
2	Float (4)	DIOthreshold	Threshold value of threshold switch
0 Byte	0 return values		

GetDIOinitialLevel (0x61)

0x61	R	GetDIOinitLevel	GSV-8 / GSV-6
Read the default / initialization level of a digital output. GSV-6: present from FWver 3.29			
1 Byte	1 Parameter		
0	uint8_t (1)	DIO No	Number of the DIO line GSV-8: [1-16], GSV-6: [1-3] =0: Get init. level for all DIO lines
2 Bytes	1 return value		
0	uint16_t (2)	DIO Init-Level	Init level for the DIO line: 0: Low, 1: High. If DIOno=0: GSV-8: Get all 16 init levels by flags: Bit 0: Line No. 1 (1.1.) ... Bit15: GPIO No 16 (4.4.) GSV-6: Bit 0: DIO No. 1 (SW1).. Bit 2: No. 3 (SW3)

SetDIOinitialLevel (0x62)

0x62	W	SetDIOinitLevel	GSV-8 / GSV-6
Write the default / initialization level of a digital output. GSV-6: present from FWver 3.29			
3 Bytes	2 Parameter		
0	uint8_t (1)	DIO No	Number of the DIO line GSV-8: [1-16], GSV-6: [1-3]. =0: Set init. level for all DIO lines
1	uint16_t (2)	DIO Init-Level	Init level for the DIO line: 0: Low, 1: High. If DIOno=0: Set all init levels by flags: GSV-8: Bit 0: Line No. 1 (1.1.) ... Bit15: GPIO No 16 (4.4.) GSV-6: Bit 0: DIO Nr. 1 (SW1).. Bit 2 Nr.3 (SW3)
0 Byte	0 return values		

The DIO default level is set to digital output lines, if none of the defined conditions for setting high or low occurred yet, e.g. with general-purpose outputs after boot-up.

ReadDataRateRange (0x63)

0x63	R	ReadDataRateRange	GSV-8
Determine the maximum or minimum of the adjustable range of the data frequency.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	=0: Determine maximum data frequency =1: Determine minimum data frequency
4 Bytes	1 return value		
0	Float (4)	Max./Min. Drate	Maximum / Minimum data frequency

ReadTEDSdataEntry (0x64)

0x64	R	ReadTEDSdataEntry	GSV-8
Read entry in TEDS data struct			
4 Bytes	4 Parameters		
0	uint8_t (1)	Channel No.	1..8
1	uint8_t (1)	TEDS Template-ID.	0= Basic-TEDS, 33, 35, 25
2	uint8_t (1)	Property-ID	see Annex D, p.91
3	uint8_t (1)	Array-Index	[Reserved, =0]. Planned to use, if a) multiple existence of same PropertyID, or b) multiple existence of same template ID
6 Bytes	3 return values		
0	uint8_t (1)	Next Property-ID	Property-ID of next entry in linked list
1	uint8_t (1)	ValType or Error	Data type of the value (Bytes 2..5) or error codes not returned by Error-Frame.
2	uint32_t (4) or Float (4)	Data	Data. Type=Float, if ValTyp_Err =1. Else: uint32_t

The Property-ID indexes all possible entries, i.e. 1. Properties (see. IEEE1541.4, Annex D), 2. relevant Selectors, 3. Special values; see Annex D, p.91. The order in the linked list corresponds to the order in the TEDS template. If the whole list should be read, best practise is to start with Property-ID=0, then set Property-ID=Next Property-ID for the next request, and so on, until Next Property-ID=0 is returned.

If Next Property-ID =0: Last entry.

ValTyp or Error:

Data type of the value (Bytes 2..5) or error codes not returned by Error-Frame.

Value	Name	Meaning
0	ANSW_IS_UINT	Data is in unsigned U32 format (uint32_t)
1	ANSW_IS_FLT	Data is in unsigned Float format (float32)
2	IS_PACKED_CHR5	Data contains 5 specially packed chars (see IEEE1541.4)
4	IS_DATE_DAYS	Data is an unsigned int value, meaning a date, coded as the number of days since 1st of January in 1998
0x80	ENTRY_HAS_ERROR	Bit7=1: Flag or comparison ValTyp_Err>=0x80: Byte codes an error, don't evaluate data value.
0xFD	ENTRY_INVALID	Entry invalid, e.g. NaN with float value
0xFE	ENTRY_NOT_SET	Entry exists, but it's flagged as "don't care", i.e. all bits =1
0xFF	ENTRY_NOT_EXIST	Entry request doesn't exist in the template. May also be returned at first entry with Property-ID=0.

ReadTEDSrawArray (0x65)

0x65	R	ReadTEDSrawArray	GSV-8 / GSV-6
Read TEDS raw data **Advice**: With GSV-6, this command is present only from FWver 3.10 on.			
3 Bytes	2 Parameter		
0	uint8_t (1)	Channel No.	1..8
1	uint16_t (2)	Byte address	Byte address of the TEDS data in the 1-wire EEPROM without check sum, 4-byte-aligned
4 Bytes	4 return values		

0x65	R	ReadTEDSrawArray	GSV-8 / GSV-6
0	uint8_t (1)	Data @ Address	Data byte 1
1	uint8_t (1)	Data @ Address +1	Data byte 2
2	uint8_t (1)	Data @ Address +2	Data byte 3
3	uint8_t (1)	Data @ Address +3	Data byte 4

With GSV-8, since FW-Version 1.53 and Co-Prozessor Version >=2, the following special functionalities exist, triggered by special adress values.

Adress (Hex)	Function	Applicable 1-wire ICs
8000	Read ROM data array, first 4 Bytes: Data byte 1: Family code, then first 3 bytes of the Serial-No.	all
8004	Read ROM data array, last 4 Bytes: Last 3 bytes of the Serial-No, check sum. Pre-condition: Read at 0x8000 before.	all
FFF0	Read temperature, two bytes. Data byte 1: Lowbyte, 2: High byte. Yields Temperatur in °C x16.	MAX31820, MAX31826
FFFF	Prepare "binary" reading. If this adress is given first (Data then =0), all following read calls are answered with general raw data, so that arbitrary, even non-TEDS-conformant EEPROM data can be read from the 1-wire memory. Otherwise, the check sum defined in the TEDS standard is checked internally and deleted from the reading data (as before with prevoius firmware).	24B33, MAX31826, DS2430

WriteTEDSbytes (0x66)

0x66	W (RAM)	WriteTEDSbytes	GSV-8
Write TEDS raw data to **RAM**			
6 Bytes	6 Parameter		
0	uint8_t (1)	Channel No.	1..8
1	uint8_t (1)	Byte address	Virtual byte address, that starts with 0 at every whole writing process. Must be divisible by 4 and smaller than 64.
2	uint8_t (1)	Data @ Address	Data byte 1
3	uint8_t (1)	Data @ Address +1	Data byte 2
4	uint8_t (1)	Data @ Address +2	Data byte 3
5	uint8_t (1)	Data @ Address +3	Data byte 4
0 Bytes	0 return values		

The data is only copied to the RAM of the device, i.e. not yet stored to the 1-wire EEPROM. Bit fields at Bit 0 at Address 0 are aligned, i.e. most significant bytes/bits may be don't care. Without check sum (will be calculated and set automatically in Cmd 0x67). To copy the data to the 1-wire EEPROM, use command StoreTEDSdata.

StoreTEDSdata (0x67)

0x67	W	StoreTEDSdata	GSV-8
Write TEDS raw data from device RAM to the 1-wire-EEPROM			
5 Bytes	3 Parameter		
0	uint8_t (1)	Channel No.	1..8
1	uint16_t (2)	Bit address	Starting bit address in the 1-wire EEPROM, where that data is written to, without check sum

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x67	W	StoreTEDSdata	GSV-8
2	uint16_t (2)	Bit size	Size of the data field to write, in bits. Bitsize >0: TEDS data, i.e. checksum is calculated and supplemented. Bitsize<0, i.e. size= Bitsize *-1: Non-TEDS raw data: Without checksum "as it is" (from FWver. 1.53)
0 Bytes	0 return values		

Get TEDS active (0x68)

0x68	R	Get TEDS active	GSV-8 / GSV-6
Determine, whether valid data was loaded and used from a sensor with TEDS data **Advice**: With GSV-6, this command is present in firmware-version 3.10 or higher (Bit 0 only).			
0 Bytes	0 Parameter		
4 Bytes	1 return value		
0	uint32_t (4)	TEDS-active Flags	Bits<31:8>: reserved Bits<7:0>: If Bit=1: At the input channel (BitNo+1) a TEDS device with valid and known data is connected to, that was used for calculating the (User-)Scale value (then, the corresponding bit in <15:8> of the mode value is also set).

Read Counter/Freq Mode (0x69)

0x69	R	ReadCountFreqMode	GSV-6BT / GSV-8
Read settings for QEI-encoder / frequency measuring **Advice**: With GSV-8, this command is present in firmware-version 1.45 or higher Bits<7:4> of the Index-paremeter denote the counter number (GSV-8: 0 or 1, GSV-6BT: 0 only)			
1 Byte	1 Parameter		
0	uint8_t (1)	Index<3:0>	0..2
4 Bytes	1 return value		

0x69	R	ReadCountFreqMode	GSV-6BT / GSV-8
0	uint32_t (4)	ModeFlags / GateTime / Value	**Index 0: Mode-Flags:** Bit 0: =1 Module present, =0: Not present. Advice: If not present, the whole value is =0 and the functionality isn't usable Bit 1: =1: Counter measuring enabled Bit 2: =1: Frequency / speed measuring enabled With GSV-6, these two functions can't be used the same time. With GSV-6, they can be used in combination, but for counter No. 0 only. Bits <4:3>: QEI mode: 00 FreeRun, non-QEI mode 01 QEI x1 10 QEI x2 11 QEI x4 Bit 5: =1: Use Home/Index Input Bit 6: =0: Counter value saturation off =1: Counter value saturation on Bit 7: =1: Last counter value stored in non-volatile memory Bit 8: =1: With frequency/speed mode: Period measuring enabled Bit 9: =1: Automatic selection of frequency mode: Period measuring or counter (GSV-8 only) Bit 10: =0: Pullup-Resistors (10kΩ) e**nabled** =1: Pullup- Resistors (10kΩ) **off** (GSV-8 only) Bit 11: =1: Index Input inverted (GSV-8 only) Bit 12: =1: Use input noise filter (GSV-8 only) **Index 1: Gate time** Without period measuring: Number of data periods, until counter value is evaluated for frequency measurement. Period measuring: Number of data periods without change, after output value is set to 0. **Index 2:** Start value for home/index input

Write Counter/Freq Mode (0x6A)

0x6A	W	WrCountFreqMode	GSV-6BT / GSV-8
Read settings for QEI-encoder / frequency measuring **Advice**: With GSV-6, this command requires properly equipped hardware; see ReadCountFreqMode Index 0, bit 0. Bits<7:4> of the Index-paremeter denote the counter number (GSV-8: 0 or 1, GSV-6BT: 0 only)			
5 Bytes	2 Parameter		
0	uint8_t (1)	Index<3:0>	Index 0: Mode-Flags, Index 1: Gate time, Index 2: Home value
1	uint32_t (4)	ModeFlags / GateTime / Value	See ReadCounter/FreqMode, p.63 Advice: After changing general enabled state in Bits<2:1> of Index 0, the number of values in the measuring value frame may change. It can then be determined by GetTXmapping (0x49)
0 Byte	0 return values		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Read Clock Time (0x6B)

0x6B	R (serial)	ReadClockTime	GSV-6BT
Read date and time of the RTC or read interval of the RTC-Alarm. **Advice**: This command requires firmware-version 3.14 or higher and hardware equipped with RTC (otherwise, an error message is returned).			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	[0-3] 0: Actual date and time. 1: Alarm interval 1 (day, hour, minute) 2: Alarm interval 2 (reserved) 3: UTC offset[8]: Entry "Day": Sign: 0=positive, 0xFF= negative. Hour, Minute: Time-zone offset to UTC
7 Bytes	7 return values		
0	uint8_t (1)	Indicator	Always =0
1	uint8_t (1)	Year	Binary value as summand to the year 2000. For example, in 2017 the value is =17. The alarm interval (Index=1) has no year; the value is then =0 and to be ignored
2	uint8_t (1)	Month	Month from 1 (January) to 12. With index =0 only
3	uint8_t (1)	Day	Day of month from 1 to 31 (Index 3: UTC-of. sign)
4	uint8_t (1)	Hour	Hour of day from 0 to 23
5	uint8_t (1)	Minute	Minute from 0 to 59
6	uint8_t (1)	Second	Second from 0 to 59. With index =0 only

0x6B	R (CAN)	ReadClockTime	GSV-6
Read date and time of the RTC or read interval of the RTC-Alarm. **Advice**: 1. This command requires firmware-version 3.14 or higher and hardware equipped with RTC (otherwise, an error message is returned). 2. On request, 2 response frames are transmitted			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0..1 0: Actual date and time. 1: Alarm interval (day, hour, minute)
4 Bytes	4 return values		
0	uint8_t (1)	Indicator	=1 ‚First half': Year/Month/Day =2 ‚2nd half': Hour/Min/Sec
1	uint8_t (1)	Year/Hour	see serial
2	uint8_t (1)	Month/Minute	see serial
3	uint8_t (1)	Day/Second	see serial

Write Clock Time (0x6C)

0x6C	W (serial)	WriteClockTime	GSV-6BT
Write date and time of the RTC or interval of the RTC-Alarm. **Advice**: This command requires firmware-version 3.14 or higher and hardware equipped with RTC (otherwise, an error message is returned).			
8 Bytes	8 Parameter		

8 Present from Firmware Version 3.27

0x6C	W (serial)	WriteClockTime	GSV-6BT
0	uint8_t (1)	Index	[0-3] 0: Actual date and time. 1: Alarm interval 1 (day, hour, minute) 2: Alarm interval 2 (reserved) 3: UTC offset[9]: Entry "Day": Sign: 0=positive, 0xFF= negative. Hour, Minute: Time-zone offset to UTC
1	uint8_t (1)	Indicator	Must be =0
2	uint8_t (1)	Year	Binary value as summand to the year 2000. For example, in 2017 the value is =17. The alarm interval (Index=1) has no year; the value is then =0.
3	uint8_t (1)	Month	Month from 1 (January) to 12. With index =0 only
4	uint8_t (1)	Day	Day of month from 1 to 31 (Index 3: UTC-of. sign)
5	uint8_t (1)	Hour	Hour of day from 0 to 23
6	uint8_t (1)	Minute	Minute from 0 to 59
7	uint8_t (1)	Second	Second from 0 to 59. With index =0 only
0 Byte	0 return values		

0x6C	W (CAN)	WriteClockTime	GSV-6
Write date and time of the RTC or interval of the RTC-Alarm. **Advice**: 1. This command requires firmware-version 3.14 or higher and hardware is equipped with RTC (otherwise, an error message is returned). 2. For the value to get stored, the first half of the data has to be written, then the 2nd half.			
5 Bytes	5 Parameter		
0	uint8_t (1)	Index	0..1 0: Actual date and time. 1: Alarm interval (day, hour, minute)
1	uint8_t (1)	Indicator	=1 ‚First half': Year/Month/Day =2 ‚Second half': Hour/Min/Sec
2	uint8_t (1)	Year/Hour	see serial
3	uint8_t (1)	Month/Minute	see serial
4	uint8_t (1)	Day/Second	see serial

Read Logger Settings (0x6D)

0x6D	R	ReadLoggerSettings	GSV-6BT
Read configuration of the measuring data recording to SD card. **Advice**: This command requires firmware-version 3.14 or higher, hardware equipped with RTC and SD-card present			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0..4 0: Basic Flags, some Read-only 1: Alarm Mode 2: B<3:0>: DirectoryCreationMode B<7:4>: File finalization mode B<12:8>: Content formatting flags 3: File length (Number of rows in file) 4: Decimation factor for data frequency to logging rate 5: Multiplication factor for data frequency to logging rate[10]

9 Present from Firmware Version 3.27

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

4 Bytes	1 return value		
0	uint32_t (4)	Value	**Index 0:** Basic Flags: Bit<0>: =1: Suitable SD card inserted Bit<1>: =1: File actually open for recording (read-only) Bit<2>: =1: Recording-enabled-state (persistently stored) Bit<4>: =1: Permanent data recording =0: Single value logging Bit<5>: =1: Recoding by trigger condition (by now, digital input only) **Index 1:** Bit<0>: =1: Alarm Mode: RTC activates alarm-line as configured by alarm interval **Index 2:** Bit<0>: Directory creation mode: Determines, at which time a new directory will be created, when the recording is started: =0: New directory every day =1: New directory every month (by now: **Read only, always=1**) Bit <1>: =1: With single logging, every day a new file will be created. =0: Data will be appended to existing file Bit<2>: With recording by trigger condition: =1: Append to existing file =0: A new file is created, when trigger condition is met Bit<8>: =1: Print date to each line Bit<9>: =1: Print time to each line (**Read only, always=1**) Bit<11>: =1: Print Header. Default=1. Not recommended without header (flag=0), because information will be omitted that may be necessary for later conversion of the file to other formats. **Index 3:** File length lines: Maximum number of rows per file **Index 4:** Decimation factor N: With permanent data recording, if this value is >1, only every Nth measuring value will be logged (value range: 1..65535) **Index 5:** Multiplication factor N: If this is >1, the logging rate is N times higher than the communication data rate with permanent data recording [10]

10 Present from FW-ver 3.35

Write Logger Settings (0x6E)

0x6E	W	WriteLoggerSettings	GSV-6BT
Configure measuring data recording to SD card **Advice**: This command requires firmware-version 3.14 or higher, hardware equipped with RTC and SD-card present			
5 Bytes	2 Parameter		
0	uint8_t (1)	Index	0..4 0: Basic Flags, some Read-only 1: Alarm Mode 2: B<3:0>: DirectoryCreationMode B<7:4>: File finalization mode B<12:8>: Content formatting flags 3: File length (Number of rows in file) 4: Decimation factor for data frequency to logging rate 5: Multiplication factor for data frequency to logging rate[10]
1	uint32_t (4)	Value	See ReadLoggerSettings p.66
0 Bytes	0 return values		

Control Logger (0x6F)

0x6F	A	ControlLogger	GSV-6BT
Trigger immediate action on measuring data recording **Advice**: This command requires firmware-version 3.14 or higher and hardware equipped with RTC and SD-card present			
1 Byte	1 Parameter		
0	uint8_t (1)	Action Enum	=**0**: Finish recording =**1**: Log a single row of measuring values of all configures input channels =**2**: Start permanent data recording at configured data frequency (/decimation factor) =**3**: Open a new file for recording. Then, a coming trigger for single value log can be executed faster. If a file was open in append mode, this creates and opens a new file. =**4**: Initialize and mount file system (if uninitialized or ejected before)[11] =**5**: Uninitialize file system and free SD-card hardware for usage by other SPI-Master 9) =**0x40** Reset RTC-Alarm only =**0x41** Reset RTC-Alarm and restart alarm interval
0 Byte	0 return values		

QueryFileSystem (0x70)

0x70	R	QueryFileSys	GSV-6BT
Determine file and directory names on the SD card **Advice**: This command requires firmware-version 3.14 or higher and hardware equipped with RTC and SD-card present			
2 Byte	2 Parameters		
0	uint8_t (1)	Control value	=0: Start; =1: Next entry =0xFE: Read last directory name of latest logged file =0xFF: Read name of latest logged file
1	uint8_t (1)	Directory level	=0: Root directory. Start here; after opening a directory inside: =1, and so forth. Advice: The length of the internal path string must not exceed 256 chars!
14 Bytes	2 return values		
0	uint8_t (1)	Flags	Bit 0: =1: IsDirectory =0: IsFile Bit 1: =1: Has Known Name Pattern Bit 2: =1: Has LFN (long file name) (reserved) Bit 7: =1: No Entry exists anymore in current directory
13	String	Name	Name in the actual directory: File name 8.3, Null-terminated or directory name 8 chars, Null-terminated

11 Present from firmware version 3.27

OpenFileDir (0x71)

0x71	A	OpenFileDir	GSV-6BT
Open file or directory on SD card for reading **Advice**: This command requires firmware-version 3.14 or higher and hardware equipped with RTC and SD-card present			
14 Bytes	2 Parameters		
0	uint8_t (1)	Control value	=0: Open file, =1: Open directory =2: Open recently logged file (if present). Name ignored
13	String	Name	File name 8.3, Null-terminated Directory name 8 Chars, Null-terminated
0 Bytes	0 return values		

GetFileInfo (0x72)

0x72	R	GetFileInfo	GSV-6BT
Read information about last opened file or directory or file/directory accessed by *QueryFileSystem,* i.e. flags, size and last-changed time stamp. **Advice**: This command requires firmware-version 3.14 or higher, hardware equipped with RTC and SD-card present			
0 Bytes	0 Parameter		
12 Bytes	9 return values		
0	uint8_t (1)	Level	Directory level, e.g. =0: inside Root-Dir
1	uint8_t (1)	Flags	Bit0: =1: isDirectory, =0: isFile Bit1: =1: Read only Bit2: =1: Hidden Bit3: =1: System Bit6: =1: Archive
2	Uint32 (4)	Size	File size in Bytes. With directory: =0
3	uint8_t (1)	Year last changed	Encoded as a binary value that represents a summand to the year 2000. For instance, in 2017 the value is 17. Note: The range of years from 1980 to 1999 will be returned as 0.
4	uint8_t (1)	Month last changed	Month from 1 (January) to 12
5	uint8_t (1)	Day last changed	Day of month from 1 to 31
6	uint8_t (1)	Hour last changed	Hour of day from 0 to 23
7	uint8_t (1)	Minute last changed	Minute from 0 to 59
8	uint8_t (1)	Second last changed	Second from 0 to 58

Read File (0x73)

0x73	R	ReadFile	GSV-6BT
Read file data from SD card. Precondition: File opened for reading **Advice**: 1. This command requires firmware-version 3.14 or higher and hardware equipped with RTC and SD-card present 2. After reading a file, it must be closed by ControlLogger with parameter=0!			
0 Bytes	0 Parameter		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x73	R	ReadFile	GSV-6BT
0..15 Bytes	0 or 1 return value(s)		
0..15	uint8_t (1) [0..15]	Data read	Up to 15 bytes of data is read by consecutive calls to the command. Normally, a file is greater than 15 bytes, so every call to the command yields the next 15 bytes and the last call less than that, i.e. 0 to 14 bytes.

Read File Extended (0x74)

0x74	R	ReadFileExtended	GSV-6BT
Read file data from SD card. Precondition: File opened for reading, Measuring data transmission stopped. File download using this command can be about 6 times faster than with Read File (0x73). **Advice**: 1. This command requires firmware-version 3.20 or higher and hardware equipped with RTC and SD-card present 2. After reading a file, it must be closed by ControlLogger with parameter=0! 3. If the permanent value transmission is not stopped, or if other commands are issued during reading a file, only up to 15 bytes are returned.			
2 Bytes	1 Parameter		
0	uint16_t	Buffer size	Maximum number of bytes to read. Value range: 2..270.
0..15 Bytes	0 or 1 return value(s)		
0..270	uint8_t (1) [0..269]	Data read	Up to <BufferSize> or 270 bytes of data is read by consecutive calls to the command (whatever is smaller). If a file is greater than that number, every call to the command yields the next <BufferSize> or 270 bytes and the last call less than that, e.g. 0 to 269 bytes. See Serial Response Frame for header description of large answer.

Read Value String (0x75)

0x75	R	ReadValueString	GSV-6
Read measuring value as text string **Advice**: 1. This command requires firmware-version 3.13 or higher 2. If an unconfigured input channel is requested, an empty string is returned 3. If Channel=0, the permanent value transmission must be disabled			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel-No. 0..8. 7: Counter/Speed 8: Temperature 9: Supply Voltage GSV-6CPU (FW-ver >= 3.20) 0: Read all configures measuring channels, i.e. up to 7 (FW-ver >= 3.20)
1	uint8_t (1)	Formatting Flags	Bit 0: =1: Include unit to string Bit 1: =1: Fixed total string length= 15 bytes. =0: String length variable from 1 to 15 / 127 bytes
1..15 Bytes 1..127 Bytes	1 return value	Value String	Formatted value string

0x75	R	ReadValueString	GSV-6
1..15 / 1..127	uint8_t[1..15] / uint8_t[1..127]	Value String	If the requested input channel is not configured (i.e. value not present), an empty string is returned, i.e. 1st char =0x00. Otherwise, the value string always starts with the sign '+' or '-', then the pre-decimal places follow (at least one), then the decimal place '.', then the decimal places. The string is printed with 5 digits altogether, if the valid maximum value is <= 99999, otherwise the necessary pre-decimal digits without decimal places. The valid maximum value is the user-scale value or (with multi-axis measuring) the corresponding maximum value of the multi-axis calibration struct. If a unit is to be printed to the string, it's separated by a white-space ' ' from the measuring value. If it's too long, it will be shortened. The string is 0-terminated or exactly 15 chars long (can happen with long unit) and never contains carriage return/linefeed characters. If the string contains several measuring values (Channel=0), they're separated by tabulator char. If Channel=0 or if a user-defined unit is included, a long response frame is transmitted, if possible (see Serial Response Frame). That one can be up to 127 bytes long.

ME Prepare Special Mode (0x77)

0x77	A	Reset Device	GSV-6
Prepare special action or state., e.g. Bootloader The command is available from Firmware version 3.27. Pre-condition: Manufachturer ID set			
4 Bytes	1 Parameter	Signature	Fixed special values, that prepare or trigger special functions
0 Bytes	0 Rückgabe		

Reset Device (0x78)

0x78	A	Reset Device	GSV-6
Restart the GSV-6 **Advice**: The GSV-6 exceptionally does **not send a response frame** to this request! This command requires firmware-version 3.10 or higher			
0 Bytes	0 Parameter		
0 Bytes	0 return values		

Release Interface (0x7A)

0x7A	A	ReleaseInterface	GSV-8
Release this communication interface. Should be called at the end of a communication session, to grant write access to other interfaces.			
0 Byte	0 Parameter		
0 Byte	0 return values		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

ReadInterfaceSetting (0x7B)

0x7B	R	ReadInterfaceSetting	GSV-6 / GSV-8
Read settings of the communication interface			
1 Byte	1 Parameter		
0	uint8_t (1)	Index / Interface-No.	0.. Number of existing interfaces-1: Basic settings. Above: Extended settings
6 Bytes	3 return values		
0	uint8_t (1)	next index	Next index in the linked list. =0: Last entry.
1	uint8_t (1)	Phys.Type / DataType	Basic settings: Physical interface type (Enum) Extended settings: Bit 7: =1: Entry writeable. =0: Not writeable. Bits<6:0>: Type of data entry (Enum)
2	uint32_t	Data	Basic: Bits<31:24>: Application interface type (Enum) Bits<23:0>: Flags Extended: Bits<31:0>: Data entry

The indices are divided into two sections. The index section 1 ranges from 0 to (Number of existing interfaces -1). Basic settings can be determined here for every existing interface. (see Annex C, p.89). The number of existing interfaces can be determined with the command *GetInterface* (0x01).

The return value "Next index" points to the beginning of the index section of the "Extended settings" for this interface. Within these, the return value 1 "Phys.Type / DataType" is an enum that describes, what the meaning of the return value 2 "Data" is (see Annex C, p.89).

ReadInterfaceSetting and WriteInterfaceSetting are available with GSV-6 from FWver 3.33.

WriteInterfaceSetting (0x7C)

0x7C	W	WriteInterfaceSetting	GSV-6 / GSV-8
Write settings of the communication interface			
5 Bytes	2 Parameter		
0	uint8_t (1)	Index / Interface-No.	0.. Number of existing interfaces-1: Basic settings. Above: Extended settings
1	uint32_t	Data	Basic: Bits<31:24>: Application interface type (Enum) Bits<23:0>: Flags Extended: Bits<31:0>: Data entry
0 Byte	0 return values		

PrepReadFTsensor (0x7D)

0x7D	A	ReadDataRateRange	GSV-6 / GSV-8
Select multi-axis / FT calibration struct for reading. Subsequent calls to ReadFTsensorCal will read data from this struct. The execution doesn't have any influence on the multi-axis sensor calculation.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	Array index GSV-8: FWver >= 1.39: 0-10. FWver<1.39: 0-4 GSV-6: FWver >= 3.11: 0-7
0 Byte	0 return values		

StoreDigiFilt (0x7E)

0x7E	W	StoreDigiFilt	GSV-8
Store configuration for digital filters of all channels. With GSV-8, both parameters are ignored.			

0x7E	W	StoreDigiFilt	GSV-8
2 Bytes	2 Parameter		
0	uint8_t (1)		
1	uint8_t (1)		
0 Byte	0 return values		

0x7E	W	StoreDigiFilt	GSV-6
Store configuration for digital filters for one (Channel Start==Channel End) or several channels (Channel Start < Channel End).			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel Start	Start index (0-5)
1	uint8_t (1)	Channel End	End index (0-5)
0 Byte	0 return values		

StoreFTSensorCal (0x7F)

0x7F	W	StoreSensorCal	GSV-6 / GSV-8
Store a struct of multi-axis /FT sensor calibration data, written to RAM, to the non-volatile memory, at index specified. Precondition: USER_ID_LEVEL set.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	GSV-6: Array-Index [0-7] GSV-8: Array-Index [0-4 / 0-10] or special value 0xFF. Existing structs can be overwritten or new ones can be appended. Memory leaks are not allowed. With the special value 0xFF, the last struct can be erased.
0 Byte	0 return values		

GetTXMode (0x80)

0x80	R	GetTXMode	GSV-6
Read measuring value frame settings			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: TX-Mode Flags 1: Measuring value data type 2: Maximum number of channels All other values reserved
2 Bytes	1 return value		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x80	R	GetTXMode	GSV-6
0	uint16_t (2)	Value	**Index 0**: TX-Mode <0> Transmission temporarily off <1> Transmission always off (stored to non-volatile memory) <2> Transmit maximum values in measured value frame[12] <4:3> reserved <5>: Measured value frame has CRC-16 checksum[13] <6> =1: Writing via UART blocked (read only) 10 <7> reserved <8> =1: TX-Synchonization: Slave (read only) 10 <9> =1: TX-Synchonization: Master (read only) 10 <15:10> reserved **Index 1**: Measuring value data type =1: int16 =3: float All other values reserved **Index 2**: (read-only) Maximum number of channels

0x80	R	GetTXMode	GSV-8
Read measuring value frame settings			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: TX-Mode Flags 1: Measuring value data type 2: Range of channel No. All other values reserved
2 Bytes	1 return value		

12 From firmware version 3.30
13 From firmware version 3.35

0x80	R	GetTXMode	GSV-8
0	uint16_t (2)	Value	**Index 0**: TX-Mode <0> =1: Transmission temporarily off (read only) <1> =1: Transmission always off (stored to non-volatile memory) <2> =1: Maximum values in measuring value frame <3> =1: Minimum values in measuring value frame <4> reserved, internally used <5>: Measured value frame has CRC-16 checksum[14] <6> =1: Write commands from UART blocked (read only) <7> =1: Write commands from USB blocked (read only) <8> =1: TX-Synchronization: Slave (read only) <9> =1: TX-Synchronization: Master (read only) <15:10> reserved **Index 1**: Measuring value data type 1: int16 2: S24 3: float All other values reserved **Index 2:** Existing input channels (Read-only) Bits<7:0>: Smallest channel No. (=2) Bits<15:8>: Highest channel No. (=8 / =10)

SetTXMode (0x81)

0x81	W	SetTXMode	GSV-6 / GSV-8
Set measuring value frame settings			
3 Bytes	2 Parameters		
0	uint8_t (1)	Index	0: TX mode 1: Measuring value data type All other values reserved
1	uint16_t (2)	Value	see GetTXMode (0x80)
0 Byte	0 return values		

Note: Changes to the CRC16 flag (bit 5) only take effect after a restart.

Read Debounce Time (0x86)

0x86	R	ReadDebounceTime	GSV-6
Read time in ms, for that a digital input or a key must be active until the function is executed. Not valid in the menu and with fast trigger inputs, e.g. Sync-Slave.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	[1-3]: 1: Debounce time TARA input 2: Debounce time SCALE key 3: Debounce time TRIGGER input
2 Bytes	1 Rückgaben		
0	uint16_t (2)	Debounce time	Debounce time in ms

14 From Firmware-Version 1.56

Write Debounce Time (0x87)

0x87	W	WriteDebounceTime	GSV-6 (Reset)
Write the time in ms, for that a digital input or a key must be active until the function is executed. Not valid in the menu and with fast trigger inputs, e.g. Sync-Slave. Precondition: User-ID set			
3 Bytes	2 Parameter		
0	uint8_t (1)	Index	[1-3]: 1: Debounce time TARA input 2: Debounce time SCALE key 3: Debounce time TRIGGER input
1	uint16_t (2)	Entprellzeit	Debounce time in ms [1-65535]
0 Byte	0 Rückgaben		

ReadDataRate (0x8A)

0x8A	R	ReadDataRate	GSV-6 / GSV-8
Read data frequency / Number of measuring value frames acquired/transmitted per second			
0 Byte	0 Parameter		
4 Bytes	1 return values		
0	float (4)	Data frequency	Number of value frames in Hz

WriteDataRate (0x8B)

0x8B	W	WriteDataRate	GSV-6 (Reset) / GSV-8
Write data frequency / Number of measuring value frames acquired/transmitted per second			
4 Bytes	1 Parameter		
0	float (4)	Data frequency	Data frequency in Hz
0 Byte	0 return values		

GetCANSetting (0x8C)

0x8C	R	GetCANSetting	GSV-6 / GSV-8
Read CAN settings Advice for GSV-8: If the command returns ERR_CMD_NOTKNOWN, CAN/CANopen is nor supported.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: CAN_IN Command Can-ID (GSV-6 only) 1: CAN_OUT Response Can-ID (GSV-6 only) 2: CAN_CV Measuring value Can-ID (GSV-6 only) 3: CAN_CAST Multicast Can-ID (GSV-6 only) 4: CAN_BAUD Baud rate (in Bits/s) 5: CAN_FLAGS 6: CANopen Node-ID (GSV-8 only) All other values reserved
4 Bytes	1 return values		
0	uint32_t (4)	Can-ID Baud rate Flags	Can-ID: 0x00000000-0x000007FF: Std-ID 0x80000000-0x9FFFFFFF: Ext-ID (GSV-6 only) Baud rates (GSV-6): 1000000, 500000, 250000, 125000, 100000, 50000, 25000, 12500 Baud rate (GSV-8):

0x8C	R	GetCANSetting	GSV-6 / GSV-8
			1000000, 500000, 250000, 125000, 50000 Flags: GSV-6: <31:0> Reserved (=0) GSV-8: Bit 0: CAN on / off:[15] =**1**: CAN switched **off** =0: CAN switched on Bit 1: CAN Application protocol =1: CANopen enabled =0: Proprietary CAN (as described above)

SetCANSetting (0x8D)

0x8D	W	SetCANSetting	GSV-6 (Reset) / GSV-8
Write CAN settings Advice: With GSV-8, baud rate and NodeID can only be changed, if CAN-interface is disabled			
5 Bytes	2 Parameters		
0	uint8_t (1)	Index	see GetCANSetting (0x8C)
1	uint32_t (4)	Can-ID, Baud rate, Flags	see GetCANSetting (0x8C)
0 Byte	0 return values		

ReadAnalogueFilter (0x90)

0x90	R	ReadAnalogueFilter	GSV-8
Read analogue input filter cut-off frequency			
0 Byte	0 Parameter		
2 Bytes	1 return value		
0	uint16_t (2)	Filter frequency	Cut-off frequency in Hz

WriteAnalogueFilter (0x91)

0x91	W	WriteAnalogueFilter	GSV-8
Write analogue input filter cut-off frequency			
4 Bytes	1 Parameter		
0	float (4)	Cut-off frequency in Hz	Three different frequency values can be set: - 28 Hz - 885 Hz - 11,4 kHz If other values are passed, the closest to one of these will be set.
0 Byte	0 return values		

SwitchBlocking (0x92)

0x92	W	SwitchBlocking	GSV-6 / GSV-8
Set write protection To unlock, the USER_ID_LEVEL must be set, i.e. the user password must have been given			
4 Bytes	1 Parameter		

15 With GSV-6: from FW-ver 3.25

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x92	W	SwitchBlocking	GSV-6 / GSV-8
0	uint32_t (4)	Code	BLOCKING_UNLOCK_THIS 0x554c4b74 = "ULKt" (reserved) BLOCKING_UNLOCK_ALL 0x554c4b61 = "ULKa" BLOCKING_LOCK_THIS 0x426c6b54 = "BlkT" (reserved) BLOCKING_LOCK_ALL 0x426c6b41 = "BlkA"
0 Byte	0 return values		

GetCommandAvailable (0x93)

0x93	R	GetCommandAvailable	GSV-6 / GSV-8
Determines, if all commands from the parameter "HighCmd" to (including) "LowCmd" are implemented, and returns the corresponding error code. ***Advice***: LowCmd must be smaller or equal to HighCmd!			
2 Bytes	2 Parameter		
0	uint8_t (1)	HighCmd	Upper command number (inclusive)
1	uint8_t (1)	LowCmd	Lower command number (inclusive)
0 Byte	0 return values (Information in ErrorByte of response frame)		Information in Error-Byte: ERR_OK: All commands are present. ERR_CMD_NOTIMPL: One or more commands are not present

ReadNoiseCutThreshold (0x94)

0x94	R	ReadNoiseCutThreshold	GSV-8
Read the threshold for the noise suppression. Below this threshold and above its negation (i.e. around zero) measuring value will be assigned to 0.			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-8]
4 Bytes	1 return values		
0	float (4)	Threshold	Threshold value

WriteNoiseCutThreshold (0x95)

0x95	W	WriteNoiseCutThreshold	GSV-8
Write the threshold for the noise suppression. Below this threshold and above its negation (i.e. around zero) measuring value will be assigned to 0.			
5 Bytes	2 Parameters		
0	uint8_t (1)	Channel	Channel No. [1-8] =0: Set parameters of all channels to the same value.
1	float (4)	Threshold	Threshold value. Must be >0.
0 Byte	0 return values		

Read AutoZero Setting (0x96)

0x96	R	ReadAutoZeroSetting	GSV-8
Read the counting period or the threshold for the automatic SetZero function. Present with Firmware-version >= 1.39			
2 Bytes	2 Parameters		
0	uint8_t (1)	Type	=0: Auto-Zero count period in seconds (Channel ignored) =1: Auto-Zero Threshold
1	uint8_t (1)	Channel	Channel No. [1-8] (if Type=1)
4 Bytes	1 return value		
0	float (4)	Period / Threshold	

Write AutoZero Setting (0x97)

0x97	W	WriteAutoZeroSetting	GSV-8
Write the counting period or the threshold for the automatic SetZero function. If the measuring value is for a time of <Period> seconds below its correspondent threshold (i.e. around 0 between threshold and -threhold), for certain channels (configurable with SetAutoZeroProperty) a SetZero routine will be performed, if Bit 22 of the Mode-Value is set, see GetMode (0x26) Present with Firmware-version >=1.39			
6 Bytes	3 Parameters		
0	uint8_t (1)	Type	=0: Auto-Zero count period in seconds (Channel ignored) =1: Auto-Zero Threshold
1	uint8_t (1)	Channel	Channel No. [1-8] (if Type=1)
2	float (4)	Wert	Periode / Schwelle
0 Byte	0 return values		

Get AutoZero Property (0x98)

0x98	R	GetAutoZeroProperty	GSV-8
Request flags that specify the behaviour of the Auto-Zero function. Present with Firmware-version >=1.39			
2 Bytes	2 Parameters		

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0x98	R	GetAutoZeroProperty	GSV-8
0	uint8_t (1)	Type	**=0**: **ApplyAZ**: Channel-flags that specify which channels shall be set to zero, or (with Mode=1) for which channels the AutoZero function is enabled. **=1**: **UseThreshold**: With Mode=0: Flags, that specify, which channels shall be compared with their correspondend threshold. The boolan results are then ANDed and if all conditions meet, the SetZero routine will be performed. **=2**: **Mode**: Bit 0: =0: Channels are grouped, i.e. if all thershold and time conditions meet (see UseThreshold), the channels configured with ApplyAZ will be set to Zero. Bit 0 =1: Auto-Zero will be applied on all channels configured with ApplyAZ, independently of each other (UseThreshold ignoriered).
1	uint8_t (1)	Channel	Reserved, (ignored)
2 Bytes	1 return value		
0	uint16_t (2)	Flags	Type =0 und =1: Channel flags: Bit 0: Condition/Action applied to channel 1, and so forth until Bit 7: applied to channel 8.

Set AutoZero Property (0x99)

0x99	W	WriteNoiseCutThreshold	GSV-8
Set flags that specify the behaviour of the Auto-Zero function. Present with Firmware-version >=1.39			
4 Bytes	3 Parameters		
0	uint8_t (1)	Type	see GetAutoZeroProperty
1	uint8_t (1)	Channel	Reserved, (ignored)
2	uint16_t (2)	Flags	Type =0 und =1: Channel flags: Bit 0: Condition/Action applied to channel 1, and so forth until Bit 7: applied to channel 8.
0 Byte	0 return values		

ReadUserOffset (0x9A)

0x9A	R	ReadUserOffset	GSV-6 / GSV-8
Read user-defined offset. This value is added to measuring values of data-type Float, after all scalings have been performed.			
1 Byte	1 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [1-6] / [1-7] (FW-ver >=3.11) GSV-8: [1-10]
4 Bytes	1 return values		
0	float (4)	Offset	User offset value

WriteUserOffset (0x9B)

0x9B	W	SetUserOffset	GSV-6 / GSV-8
Write user-defined offset. This value is added to measuring values of data-type Float, after all scalings have been done. **Advice for GSV-6:** If the FT/multi-axis calculation is enabled, this value is not used!			
5 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [0-6] / [0-7] (FW-ver >=3.11) GSV-8: [0-10] Channel=0: Set parameters of all channels to the same value.
1	float (4)	Offset	User offset value
0 Byte	0 return values		

GetInputType (0xA2)

0xA2	R	GetInputType	GSV-6
Read type of analogue input With GSV-6, this determines the electrical overall input range.			
2 Bytes	1 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-6: [1-6] / [1-7] (FW-ver >=3.11)
1	uint8_t (1)	Reserved	=0 (unused) .
4 Bytes	1 return value		
1	Uint32_t (4)	Sens	Input range in mV/V*100

0xA2	R	GetInputType	GSV-8
Read type of analogue input With GSV-8, this determines the measuring range of the analogue input circuit, that should fit to the sensor connected. All available input ranges can be requested, as well as the configured type/range.			
2 Bytes	1 Parameter		
0	uint8_t (1)	Channel	Channel No. [1-8]
1	uint8_t (1)	SensIndex	[0xFF..0x05] 0xFF: Read input type and its range actually configured <0xFF: As Type enum: 0x00: Read input range for type= bridge sensor with excitation voltage 8.75V 0x01: Read input range for type= bridge sensor with excitation voltage 5V 0x02: Read input range for type= bridge sensor with excitation voltage 2.5V 0x03: Read input range for type= single ended input 0x04: Read input range for type= temperature sensor PT1000 0x05: Input type = Temperature sensor K-Type absolute (requires reference sensor PT1000 connected to channel 8), supported with FWver >= 1.39 0x06: Input type = Temperature sensor K-Type relative (doesn't need reference sensor), supported with FWver >= 1.39 0x07: Counter/Frequency

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

0xA2	R	GetInputType	GSV-8
5 Bytes	2 return values		
0	uint8_t (1)	Type	Actually configured input type enum as listed above (SensIndex=0xFF) Input type as given in SensIndex (SensIndex: 0..7).
1	Uint32_t (4)	Sens	Input range in mV*100 (or mV/V*100, respectively). Counter: Maximum*100, Temperature: °C*100

SetInputType (0xA3)

0xA3	W	SetInputType	GSV-8
Change the input type Advice: When changing the input type, the calibrated offsets (manufacturer default) and the default UserScale values are loaded, the previous parameters for offset and UserScale are overwritten.			
2 Bytes	2 Parameter		
0	uint8_t (1)	Channel	Channel No. GSV-8: [1-8] Channel=0: Set parameters of all channels to the same value.
1	uint8_t (1)	InputType	The InputType is defined as follows: 0x00: Input type = bridge sensor with excitation voltage of 8.75V 0x01: Input type = bridge sensor with excitation voltage of 5V 0x02: Input type = bridge sensor with excitation voltage of 2.5V 0x03: Input type = Single-ended Input 0x04: Input type = Temperature sensor PT1000 0x05: Input type = Temperature sensor K-Type absolute (requires reference sensor PT1000 connected to channel 8), supported with FWver >= 1.39 0x06: Input type = Temperature sensor K-Type relative (doesn't need reference sensor), supported with FWver >= 1.39
0 Byte	0 return values		

Annex

A: Physical Units (Codes)

In the following table the ME-Codes for units are listed.

Unit	Code
mV/V	0
kg	1
g	2
N	3
cN	4
V	5
µm/m	6
(none)	7
t	8
kN	9
lb	10
oz	11
kp	12
lbf	13
pdl	14
mm	15
m	16
cNm	17
Nm	18
°C	19
°F	20
K	21
oztr	22
dwt	23
kNm	24
%	25
‰	26
W	27
kW	28
rpm	29
bar	30
Pa	31
hPa	32
MPa	33
N/mm²	34
°	35
Hz	36
m/s	37
km/h	38
m³/h	39
mA	40
A	41
m/s²	42

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Unit	Code
flbs	43
ftlb	44
J	45
kWh	46
UnitText 2	254 (-2)
UnitText 1	255 (-1)

Table: Units

B: Types of digital In- and Outputs

The following functionalities can be configured:

No.	Function	Data direction	Value Device command Get/SetDIOtype =value of DLL-Function GSV86get/setDIOtype	Short description
0	None	n.a.	0x000000	I/O line not present (only GSV-6). Read-only
1	General-Purpose Input	Input	0x000004	General input. Logic level can be determined by GetDIOlevel / GSV86getDIOlevel. *Default setting.*
2	Set Zero, single channel	Input	0x000010	Active level at input sets measuring value of one analogue input channel to zero
3	Set Zero, all channels	Input	0x000020	Active level at input sets measuring value of all analogue input channels to zero
4	Reset the maximum and minimum value acquisition	Input	0x000040	Active level at input clears all maximum and minimum values
5	Resetting digital outputs to default levels	Input	0x000050	The active input level sets all digital IOs configured as outputs to their configured default levels
6	Trigger Send actual value	Input	0x000080	Trigger the transmission of a frame with actual measuring values via serial/USB-CDC at active-to-inactive edge of the digital input. GSV-8 only.
7	Trigger Send maximum value	Input	0x000100	At inactive-to-active edge of the digital input, the maximum value acquisition

No.	Function	Data direction	Value Device command Get/SetDIOtype =value of DLL-Function GSV86get/setDIOtype	Short description
				is started (all input channels) and at the active-to-inactive edge a frame with these maximum values is sent to the serial/USB-CDC interface. GSV-8 only.
8	Trigger Send minimum value	Input	0x000200	At inactive-to-active edge of the digital input, the minimum value acquisition is started (all input channels) and at the active-to-inactive edge a frame with these minimum values is sent to the serial/USB-CDC interface. GSV-8 only.
9	Trigger Send mean value	Input	0x000400	At inactive-to-active edge of the digital input, a decimating mean value acquisition is started (all input channels) and at the active-to-inactive edge a frame with these mean values is sent to the serial/USB-CDC interface. GSV-8 only.
10	Trigger Send actual value	Input	0x000800	While the level at the digital input is active, frames with actual measuring values are sent to the serial/USB-CDC interface, with data frequency configured. GSV-8 only.
11	Sync Slave	Input	0x000002	At low-to-high edge at digital input, a measuring value frame is transmitted. Only if used with a second GSV-8, configured as Sync Master, and whose sync output (see No.19) is wired to the Sync Input of the Slave.
12	QEI-Encoder	Input	0x000008	Input for quadrature-encoder / counter frequency. Read-Only. To change, the command Write Counter/Freq Mode can be used at index 0. GSV-8 only.
13	File-Logging	Input	0x000001	Trigger-Input for logging measured data to file on SD-card. Only GSV-6 with Logger function.

No.	Function	Data direction	Value Device command Get/SetDIOtype =value of DLL-Function GSV86get/setDIOtype	Short description
14	General-Purpose Output	Output	0x001000	General output. Logic level can be set with SetDIOlevel / GSV86setDIOlevel.
15	Threshold output actual value	Output	0x010000	Threshold switch output: Output is set to active level, if the measuring value of the assigned input channel is greater than the upper threshold, and set to inactive level, if it is smaller than the lower threshold.
16	Threshold output Maximum Value	Output	0x014000	Threshold switch output: Output is set to active level, if the maximum measuring value of the assigned input channel is greater than the upper threshold and set to inactive level, if it is smaller than the lower threshold.
17	Threshold output Minimum Value	Output	0x018000	Threshold switch output: Output is set to active level, if the minimum measuring value of the assigned input channel is smaller than the lower threshold and set to inactive level, if it is greater than the upper threshold.
18	Window comparator output actual measuring value	Output	0x012000	Window comparator switch output: Output is set to active level, if the measuring value of the assigned channel is smaller than the upper threshold and greater than the lower threshold; otherwise (outside window) set to inactive.
19	Window comparator output maximum measuring value	Output	0x016000	Window comparator switch output: Output is set to active level, if the maximum measuring value of the assigned channel is smaller than the upper threshold and greater than the lower threshold; otherwise (outside window) set to inactive.
20	Window comparator output minimum	Output	0x01A000	Window comparator switch output: Output is set to active level, if the

No.	Function	Data direction	Value Device command Get/SetDIOtype =value of DLL-Function GSV86get/setDIO type	Short description
	measuring value			minimum measuring value of the assigned channel is smaller than the upper threshold and greater than the lower threshold; otherwise (outside window) set to inactive.
21	Sync-Master	Output	0x020000	The Sync Master generates a synchronization signal synchronously to its measuring frame transmission. The Slaves are wired to this digital sync output (see No. 11). When transmission is started, the level edge is low-to-high, at the half of the data period, it's high-to-low.

The DIOs are equipped with pull-up resistors that generate a high level, if an input is left open. If a trigger input function shall be operated using a switch, the key/switch should be connected between the DIO line and GNDD. For the function being executed with the switch closed, the line must be inverted by software. To do this, the value mentioned above in the column "Value" must be ORed with 0x800000 when using this device interface or the Windows DLL.

The threshold outputs can be inverted like this, too.

The level states mentioned in the table mean:

Level	Not inverted	Inverted
Active	Logic 1 = High = 5V	Logic 0 = Low = 0V
Inactive	Logic 0 = Low = 0V	Logic 1 = High = 5V

Only with the General-purpose- and the Master-slave-sync functions (No. 1,11, 14 and 21 in the table), inverting has no effect. The commands/DLL-functions Get/SetDIOlevel and GSV86get/setDIOlevel read/write the level always directly, i.e. non-inverted.

For threshold switch and widow comparator thre's an additional possibility to "mirror" the thresholds in the nagative range. The threshold switch is then active, if measured value > upper threshold or measured value < -upper threshold.

To enable this function, the value in the above table must be ORed by 0x400000.[16]

Further notes can be found in the general manual, chapter "Digital I/O Functions".

C: Flags and Enumerations for Read/Write Interface Settings (Cmd 0x7B/0x7C)

Basic Settings: Physical interface type

Enum-No.	Meaning
1	RS232 Interface (asynchronous, 8N1) with V24 Level
2	UART Interface (asynchronous, 8N1) with 3.3V Level (Low=0V, High=3.3V)
3	USB Interface
4	CAN Fieldbus- Interface
5	100BaseTX ("Ethernet")
6	RS422 Interface (5-wire, asynchronous, 8N1) with differential level
7	SPI master Interface, eventually for measured values only

Basic Settings: Type of application layer

Enum-No.	Meaning
1	GSV8/6 protocol, as described in this document
2	CANopen
3	EtherCAT CoE
4	Textual "Monitor" Protocol (GSV-6 only)
5	Scaled SINT16 measuring values (GSV-6 only)

This value is communicated in Bits<31:24> of the data value in the Basic Settings.

Basic Settings: Flags in Flag value

Bit-No.	Meaning
0	=1: Interface is enabled and ready to receive
1	=1: Interface is enabled and ready to send
2	=1: Interface has write access
3	=1: Integer measuring value in Binary offset-Format (only valid with App.Enum =1) =0: Integer measuring value in Signed-Int Format (only valid with App.Enum =1)
4	=1: Permanent measuring value transmission on (volatile state in RAM)

16 Present with GSV-6 from Firmware-Ver. 3.29 and with GSV-8 from 1.54

Bit-No.	Meaning
5	=1: Permanent measuring value transmission on (non-volatile state in EEPROM)
16	=1: Interface switchable on/off
17	=1: Interface uses device or service addresses (field bus)
18	=1: Address(es) is/are changeable

This flag value is communicated in Bits<23:0> of the data value in the Basic Settings.

Extended Settings: Type of data content (enum)

Enum-No.	Meaning
1	Mask for settings a BasicSettings flag value: Bit=1: Same BitNo can be set =1
2	Mask for clearing a BasicSettings flag value: Bit=1: Same BitNo can be set =0
3	Number of the interface(s), with which it is present in a mutually-exclusive manner. In every of the 4 bytes an interface number >0 can be present, beginning with the LSbyte. Value =0x00000000 means: With no interface mutually-exclusive.
4	Active baud rate in bps. Value =0 means: No baud rate applicable or not changeable.
5	Existing baud rate in bps
6	Number of active Service-IDs or (device-) addresses
7	CAN-ID of the command request service Host->Device
8	CAN-ID of the command response service Device-> Host
9	CAN-ID of the measuring value frame Device-> Host
10	CAN-ID Multicast Host->Devices
11	CANopen NodeID
16	Device state, especially with field bus; see next table
17	Bits<31:24>: CANopen Transmission-Type. Bits<15:0>: CANopen Event-Timer
18	Bits<31:16>: CANopen Inhibit-Time. Bits<15:0>: CANopen Heartbeat Timer
19	Number of communicated channels. Must be <= Basic No. of Channels setting
20	Scaling factor fpr communicated Integer measured value (Float value -> Int)

This value is communicated in Bits<6:0> of the second return parameter with extended-Settings request of the command ReadInterfaceSetting (0x7B).

Enumeration of the device state

Value	Meaning
0	Interface is switched off

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Value	Meaning
2	Interface on, State = "Init" / "Stopped"
4	Interface on, State = "Pre-Operational"
8	Interface on, State = "Save-Operational"
12	Interface on, State = "Operational"

This value is communicated in the data value with type-enum =16 in the extended Settings.

D: IDs for ReadTEDSdataEntry

ID Property name and Description

0 Placeholder in Dictionary, points to first entry index (Special value: evaluate NextID only)

1 TEMPLATE ID

2 Separator

3 Select Case—Physical Measurand

4 Select Case—Full-Scale Electrical Value Precision

5..9 reserved for coming Select-cases & special IDs

10 Sensitivity and mapping properties General %Sens Sensitivity of transducer

11 %Sens@Ref Sensitivity of transducer at reference conditions

12 %Reffreq Reference frequency (f ref)

13 %RefTemp Reference temperature (T ref)

14 %Sign Phase inversion (0° or 180°)

15 %Direction Direction, or axis, of sensitivity (x, y, or z)

16 %MapMeth Mapping Method of physical to electrical units

17 %MinPhysVal Minimum value of physical measurement/control range

18 %MaxPhysVal Maximum value of physical measurement/control range

19 %MinElecVal Minimum value of electrical signal range

20 %MaxElecVal Maximum value of electrical signal range

21 Strain/Bridge %GageType Topology and rosette orientation of gage

22 %BridgeType Type of bridge (quarter, half, or full)

23 %GageFactor Sensitivity of strain gage

24 %GageTransSens Transverse sensitivity of strain gage

25 %GageOffset Zero offset of gage circuit after installation

26 %PoissonCoef Poisson Coefficient of strain gage

27 %YoungsMod Youngs Modulus of material to which gage is attached

28 %GageArea Area of gage element

29 RTD and thermistor %RTDCoef_R0 RTD or thermistor resistance at 0 °C

30 %RTDCoef_A Coefficient A of Callendar Van-Dusen equation for RTDs

31 %RTDCoef_B Coefficient B of Callendar Van-Dusen equation for RTDs

32 %RTDCoef_C Coefficient C of Callendar Van-Dusen equation for RTDs

33 %SteinhartA Coefficient A of Steinhart-Hart equation for thermistors

34 %SteinhartB Coefficient B of Steinhart-Hart equation for thermistors

35 %SteinhartC Coefficient C of Steinhart-Hart equation for thermistors

36 %SelfHeating Coefficient of self-heating, intended for thermistors

37 TC %TCType Thermocouple calibration type (J, K, T, etc.)

38 %CJSource Cold-junction compensation method

39 Electrical signal properties General %ElecSigType Type of electrical signal (enumerated)

40 %RespTime Response Time

41 %ACDCCoupling Coupling of electrical signal (AC or DC)

42 %SensorImped Electrical impedance of sensor (of each element in case of bridge)

43 %DiscSigType Discrete signal type

44 %DiscSigAmpl Discrete signal voltage amplitude

45 %PulseMeasType Pulse signal measurement type (frequency, period, count, etc.)

46 %Gain Gain of preamplifier

47 %Filter Indicates selectable filter

48 %TempCoef Temperature coefficient

49 Sensitivity and mapping properties Mic/Preamp %Prepolarized Prepolarized (yes or no)

50 %RefPol Polarization voltage

51 %Rin Input resistance of amplifier

52 %Rout Output resistance of amplifier

53 %Cin Input capacitance of amplifier

54 %Cmic Microphone capacitance

55 %Cstray Microphone stray capacitance

56 %Rleakage Microphone leakage resistance

57 %MicType Microphone type

58 %MicSize Microphone size

59 %Resp_Type Frequency response type

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

60 %RefPress Reference pressure

61 %Equi_Vol Equivalent microphone volume

62 %Gate Gate present

63 Excitation and power %ExciteAmplNom Excitation or power-supply level, nominal

64 %ExciteAmplMin Excitation or power-supply level, minimum

65 %ExciteAmplMax Excitation or power-supply level, maximum

66 %ExciteType Type of excitation or power (DC, AC, or bipolar DC)

67 %ExciteCurrentDraw Maximum current required to power/excite transducer

68 %ExciteFreqNom Excitation signal frequency, nominal

69 %ExciteFreqMin Excitation signal frequency, minimum

70 %ExciteFreqMax Excitation signal frequency, maximum

71 %LoopSupplyMin Supply for current loop transducers, minimum

72 %LoopSupplyMax Supply for current loop transducers, maximum

73 Calibration properties Mg. %CalDate The date of the last calibration

74 %CalInitials Calibration initials

75 %CalPeriod Amount of time recommended between calibrations

76 Sensitivity and mapping properties Calibration table and curves %CalTable_Domain Indicates calibration table domain as electrical or physical

77 %CalPoint_DomainValue Domain calibration value

78 %CalPoint_RangeValue Range calibration deviation

79 %CalCurve_Domain Indicates calibration curve domain as electrical or physical

80 %CalCurve_PieceStart Start of calibration curve segment

81 %CalCurve_Power Power of domain value

82 %CalCurve_Coef Coefficient of polynomial

83 Transfer function %TF_SZ Single zero

84 %TF_SP Single pole (low-pass filter of first order)

85 %TF_KZr Complex zero

86 %TF_KZq Quality factor parameter Qz of a complex zero

87 %TF_KPr Complex pole at Fpres (F mounted resonance)

88 %TF_KPq Quality factor Qp for the complex pole (mounted quality factor)

89 %TF_HP_S Single zero at 0 and a single pole (high-pass filter)

90 %TF_SL Constant relative slope

91 %TF_SZm Single zero dependent on previous property

92 %TF_SPm Single pole dependent on previous property

93 %PhaseCorrection Phase correction at the reference condition

94 %TF_Table_Freq Frequency point value for tabular transfer function

95 %TF_Table_Ampl Amplitude point value for tabular transfer function

96 Miscellaneous properties Attached transducer %Attached_MfgrID Manufacturer ID of transducer attached to amplifier

97 %Attached_ModelNum Model number of transducer attached to amplifier

98 %Attached_VersionLetter Version letter of transducer attached to amplifier

99 %Attached_VersionNum Version number of transducer attached to amplifier

100 %Attached_SerialNum Serial number of transducer attached to amplifier

101 %System_MfgrID Manufacturer ID of system

102 %System_ModelNum Model number of system

103 %System_VersionLetter Version letter of system

104 %System_VersionNum Version number of system

105 %System_SerialNum Serial number of system

106 Sensitivity and mapping properties Miscellaneous %Stiffness Stiffness of transducer

107 %Mass_below Mass below gage

108 %Weight Weight of transducer

109 %TestGain Test gain

110 %Passive Indicates support of passive mode

111 %PollFreq The frequency with which the host shall update the FR

112 %MeasID Measurand ID

113 %Ccable Capacitance of cable

114 %CableLen Length of cable

115 %Appended_TEDS Indicates if an Appended TEDS exists

116 %Appended_TEDS_location Indicates location of the Appended TEDS if it exists

117 %EMBTPL Indicates that the next portion of the TEDS is in Embedded Template format

118 %DefaultFR Defines the default setting of the FR. Cannot coexist with subproperty Default.

119 %XML XML format

120 %MDEF Prefix for manufacturer-defined parameters

121 %TDL_CHKSUM User template validation checksum

122 %user Freeform TEDS format

123 Grouping properties %PhysicalParameterType Describes the parameter type

124 %MemberIndex Indicate the order in which XdcrChannels are grouped within a PublicXdcr

E: Initial start-up of the GSV-6 Device (example)

Hardware preconditions for using GSV-6CPU:

a) Supply voltage 3,7V …5.0V at Pin 14 VCC_IN

b) GND 0V at Pin 15;

c) Pin13 „Supply Warning“ connected to Pin14 “VCC_IN”

d) If necessary, connect level shifter to UART2_RX and UART2_TX (3,3V TTL level)

In the following, the communication behaviour after power-up is shown, the commands give are suggested as an example.

The device starts sending measuring values of all 6 channels, if data frame is configured accordingly, at data frequency configured[17]; in this example: 6 channels from GSV-6. The data sent by the device is shown in red, commands sent to the device are printed in blue.

```
AA 15 B0 3A 49 9B 2C BF 86 66 66 BF 5C D4 2D BF 4E E3 26 B9 A8 01 50 BF 86 66 66 85
AA 15 B0 BC 40 27 E6 BF 86 66 66 BE DC 40 93 BE 50 BA A2 BC 8C B4 4C BF 86 66 66 85
AA 15 B0 BC EA 28 3A BF 86 66 66 3E 1A 85 C4 3F 1B 51 A7 BD 23 8A E0 BF 86 66 66 85
AA 15 B0 BD 30 24 93 BF 86 66 66 3F 23 BF 6C 3F 86 66 66 BD 72 4B 7E BF 86 66 66 85
AA 15 B0 BD 58 4E 7D BF 86 66 66 3F 75 9F 22 3F 86 66 66 BD 93 44 59 BF 86 66 66 85
AA 15 B0 BD 6E 5B 76 BF 86 66 66 3F 86 66 66 3F 86 66 66 BD A1 4F A9 BF 86 66 66 85
AA 15 B0 BD 78 11 EF BF 86 66 66 3F 86 66 66 3F 86 66 66 BD A6 F4 81 BF 86 66 66 85
```

Stop Transmission

```
By issuing the command StopTransmission, the data transmission is stopped.
AA 90 23 85
AA 50 00 85
```

GetValue

```
By issuing the command GetValue, the transmission of a measuring value frame is requested.
AA 90 3B 85
AA 15 B0 BD FA 09 F3 BF 86 66 66 3F 86 66 66 3F 86 66 66 BE 1E E2 0A BF 86 66 66 85
```

17 Unless, autonomous data transmission was disabled permanently by SetTXmode

F: Default settings

The following table lists the default settings that apply after the Load Config (0x09) command is executed. These can differ for special versions.

Parameter	Default value GSV-8	Default value GSV-6	Rd,Wr Cmd. (hex)
Input sensitivity	3,5 mV/V	2 mV/V	GSV-8: A2, A3 GSV-6: 6, 7
Measuring range / Typ	Bridge input 3,5 mV/V	Bridge input 4 mV/V	A2, A3
Data frequency	10 Frames/s	10 Frames/s	8A, 8B
Offset (Zero point)	Calibrated value for 0 mV/V	Calibrated value for 0 mV/V	2, 3
User-Scale	3,5	2	14, 15
User-Offset	0	0	9A, 9B
Unit	mV/V (Enum: 0)	mV/V (Enum: 0)	0F, 10
Unit, free text	"" [none=empty string]	"unknown0", "unknown1"	11, 12
Permanent measuring value transmission?	Yes	From v.3.11: Unchanged Factory setting: Yes	80, 81
Full-cal evalue f. SCALE Key Scaling function	---	Unchanged Factory setting: 100%	---
Sync. Master/Slave	No	No	5B, 5C
TX Max / Min?	No, i.e. actual value	No, i.e. actual value	GSV-8: 80, 81 GSV-6: 26, 27
Digital filter	Off	Off	51, 52
6-Axis sensor aktive	No	No	26, 27
6-Axis sensor: Aktivee Data set-No.	0	0	54, 55
Se analog filter according to data freq.	Yes	---	26, 27 (only GSV-8)
Noisecut aktive	No	---	26, 27 (only GSV-8)
Noisecut threshold	5% of. FS, i.e. 0,175 mV/V	---	94, 95 (only GSV-8)
Max. value is absolute value of maximum?	No	---	26, 27 (only GSV-8)
Write protection set?	No	No	92
Use TEDS data	Yes	Yes	26, 27
Set unit from TEDS	Yes	Yes (not configurable)	26, 27 (only GSV-8)
Set InType from TEDS	No	Yes (not configurable)	26, 27 (only GSV-8)
Set AoutScale from TEDS	Yes	Yes (not configurable)	26, 27 (nur GSV-8)
Set offset from TEDS	No	No	26, 27

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Parameter		Default value GSV-8	Default value GSV-6	Rd,Wr Cmd. (hex)
Menu output 0-20mA, if Output-type = Current		---	Unchanged Factory setting: OFF	26, 27
Autozero aktive		No	---	26, 27 (nur GSV-8)
Autozero threshold		5% of. FS, i.e. 0,175 mV/V	---	96, 97 (nur GSV-8)
Autozero period		5 Sec.	---	96, 97 (nur GSV-8)
Analog output: Type		±10 V	±10 V	0D, 0E
Analog output: Offset		0	0	4, 5
Analog output: Scaling		1	2 (also effective for digital meas. value)[18]	6, 7
Digital I/O: Type		GP-Input	No. 1-3: Threshold switch with hysteresis 4: TRIG: SD-File Log. Trigger 5: TARA: TaraAll: Set all chan. zero	5B, 5C
Digital I/O: Flags		Not inverted, unipolar	Not inverted, unipolar	5B, 5C
Digital I/O: Default level		0 (Low)	0 (Low)	61, 62
Threshold, upper		105% of. FS, i.e. 3,675 mV/V, i.e. not aktive	SW1: 90% of. FS, i.e. 1,8 mV/V SW2: 50% of. FS, i.e. 1 mV/V SW3: 75% of. FS, i.e. 1,5 mV/V	5F, 60
Threshold, lower		-105% of. FS, i.e. -3,675 mV/V, i.e. not aktive	SW1: 88% of. FS, i.e. 1,76 mV/V SW2: 48% of. FS, i.e. 0,96 mV/V SW3: 73% of. FS, i.e. 1,46 mV/V	5F, 60
Digital filter used:		No	No	51, 52
Counter/Frequency measuring mode ON/OFF		Unchanged Factory setting: OFF	Unchanged Factory setting: OFF	69, 6A
Counter	Mode flags	0b0000.11001uu1	0b0000.11001uu1 (u= unchanged)	69, 6A (0)
	Gate time	0	0	69, 6A (1)
	Start value	0	0	69, 6A (2)
Keys / Digital inputs debounce time		DIO1-16: 40 ms Sampling period Keys: 120-159 ms (both unchangable)	TARA: 1 Sec. SCALE: 1 Sec. TRIG: 10 ms	86, 87 (only GSV-6)
Logging to SD-Card		---	OFF	6D, 6E (only GSV-6 with SD-Slot)
Logger	Mode flags	---	0b000000u	6D, 6E (0)
	Alarm-Mode	---	0b0	6D, 6E (1)
	Dir./Form.Flags	---	0b1011.00000001	6D, 6E (2)
	Meas. value rows per file	---	30000	6D, 6E (3)
	Decimation	---	1	6D, 6E (4)
	Factor	---	1	6D, 6E (5)

18 With GSV-1L/K: calibrated value, i.e. may differ slightly from 2.

Parameter	Default value GSV-8	Default value GSV-6	Rd,Wr Cmd. (hex)
Prohibit Set Zero	0x0000 (=all allowed)	---	32, 33

The following parameters also remain unchanged upon calling Load Config: Communication parameters: Bit rate, CAN-IDs, Number of channels in measuring value frame[19], Measuring value data type. Also Digital filter coefficients, Six-Axis sensor data, User-Passwort and Flags that indicate the presence of hardware components.

Parameter		Default value GSV-8	Default value GSV-6	Rd,Wr Cmd (hex)
CAN-IDs	Cmd-Request	-	0x100	8C, 8D (0)
	Cmd-Response	-	0x101	8C, 8D (1)
	Measured values	-	0x101	8C, 8D (2)
	CANopen NodeID	0x40 (GSV-8 CANopen)	-	8C, 8D (6)
CAN Bitrate		500 kBits/s (GSV-8 CANopen)	1 MBits/s	8C, 8D (4)
UART-Bitrate		115200	230400	7B, 7C
Number of channels in Frame		8	6[20]	49, 4A (0)
Measured values type		Float	Float	80, 81 (1)
User-Password		"Beln"	"USC1"	58 (only GSV-8, Wr)

19 With GSV-6: from firmware version 3.11. Before, ONE channel with data type= FLOAT was set!

20 With GSV-6: from firmware version 3.11. Before, ONE channel with data type= FLOAT was set. Depending on the model, only the first channel may be calibrated. With analog models: Only channel 1 (channel number=1).

ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

G: Functions in C for checksum calculation

```
const uint16_t u16CrcTable [256] = {
        0x0000, 0xC0C1, 0xC181, 0x0140, 0xC301, 0x03C0, 0x0280, 0xC241,
        0xC601, 0x06C0, 0x0780, 0xC741, 0x0500, 0xC5C1, 0xC481, 0x0440,
        0xCC01, 0x0CC0, 0x0D80, 0xCD41, 0x0F00, 0xCFC1, 0xCE81, 0x0E40,
        0x0A00, 0xCAC1, 0xCB81, 0x0B40, 0xC901, 0x09C0, 0x0880, 0xC841,
        0xD801, 0x18C0, 0x1980, 0xD941, 0x1B00, 0xDBC1, 0xDA81, 0x1A40,
        0x1E00, 0xDEC1, 0xDF81, 0x1F40, 0xDD01, 0x1DC0, 0x1C80, 0xDC41,
        0x1400, 0xD4C1, 0xD581, 0x1540, 0xD701, 0x17C0, 0x1680, 0xD641,
        0xD201, 0x12C0, 0x1380, 0xD341, 0x1100, 0xD1C1, 0xD081, 0x1040,
        0xF001, 0x30C0, 0x3180, 0xF141, 0x3300, 0xF3C1, 0xF281, 0x3240,
        0x3600, 0xF6C1, 0xF781, 0x3740, 0xF501, 0x35C0, 0x3480, 0xF441,
        0x3C00, 0xFCC1, 0xFD81, 0x3D40, 0xFF01, 0x3FC0, 0x3E80, 0xFE41,
        0xFA01, 0x3AC0, 0x3B80, 0xFB41, 0x3900, 0xF9C1, 0xF881, 0x3840,
        0x2800, 0xE8C1, 0xE981, 0x2940, 0xEB01, 0x2BC0, 0x2A80, 0xEA41,
        0xEE01, 0x2EC0, 0x2F80, 0xEF41, 0x2D00, 0xEDC1, 0xEC81, 0x2C40,
        0xE401, 0x24C0, 0x2580, 0xE541, 0x2700, 0xE7C1, 0xE681, 0x2640,
        0x2200, 0xE2C1, 0xE381, 0x2340, 0xE101, 0x21C0, 0x2080, 0xE041,
        0xA001, 0x60C0, 0x6180, 0xA141, 0x6300, 0xA3C1, 0xA281, 0x6240,
        0x6600, 0xA6C1, 0xA781, 0x6740, 0xA501, 0x65C0, 0x6480, 0xA441,
        0x6C00, 0xACC1, 0xAD81, 0x6D40, 0xAF01, 0x6FC0, 0x6E80, 0xAE41,
        0xAA01, 0x6AC0, 0x6B80, 0xAB41, 0x6900, 0xA9C1, 0xA881, 0x6840,
        0x7800, 0xB8C1, 0xB981, 0x7940, 0xBB01, 0x7BC0, 0x7A80, 0xBA41,
        0xBE01, 0x7EC0, 0x7F80, 0xBF41, 0x7D00, 0xBDC1, 0xBC81, 0x7C40,
        0xB401, 0x74C0, 0x7580, 0xB541, 0x7700, 0xB7C1, 0xB681, 0x7640,
        0x7200, 0xB2C1, 0xB381, 0x7340, 0xB101, 0x71C0, 0x7080, 0xB041,
        0x5000, 0x90C1, 0x9181, 0x5140, 0x9301, 0x53C0, 0x5280, 0x9241,
        0x9601, 0x56C0, 0x5780, 0x9741, 0x5500, 0x95C1, 0x9481, 0x5440,
        0x9C01, 0x5CC0, 0x5D80, 0x9D41, 0x5F00, 0x9FC1, 0x9E81, 0x5E40,
        0x5A00, 0x9AC1, 0x9B81, 0x5B40, 0x9901, 0x59C0, 0x5880, 0x9841,
        0x8801, 0x48C0, 0x4980, 0x8941, 0x4B00, 0x8BC1, 0x8A81, 0x4A40,
        0x4E00, 0x8EC1, 0x8F81, 0x4F40, 0x8D01, 0x4DC0, 0x4C80, 0x8C41,
        0x4400, 0x84C1, 0x8581, 0x4540, 0x8701, 0x47C0, 0x4680, 0x8641,
        0x8201, 0x42C0, 0x4380, 0x8341, 0x4100, 0x81C1, 0x8081, 0x4040};

/****************************************************************************
* Function Name  : u16MakeCRC16
* Description    : Calculates CRC16 Checksum for Modbus
* Input          : *pu8data : Pointer to the data array
*                : u16length : Length of data in Bytes
* Output         : None
* Return         : CRC16
****************************************************************************/

uint16_t u16MakeCRC16(const uint8_t *pu8data, uint16_t u16length)
{
        uint16_t count;
        uint8_t u8erg1;
        uint16_t u16erg2;
        union {
           uint16_t u16Buffer;
           uint8_t u8Buffer[2];
        } uCrc;
        uCrc.u16Buffer = 0xFFFF;

        for (count = 0; count < u16length; ++count)
        {
                u8erg1 =        pu8data[count] ^ uCrc.u8Buffer[0] ;
                u16erg2 = u16CrcTable[u8erg1];
                uCrc.u16Buffer = uCrc.u8Buffer[1] ^ u16erg2;
        }
        return uCrc.u16Buffer;
}
```

```
const uint8_t u8CRC8Table[256]= {
    0x00, 0x07, 0x0E, 0x09, 0x1C, 0x1B, 0x12, 0x15, 0x38, 0x3F, 0x36, 0x31, 0x24, 0x23, 0x2A, 0x2D,
    0x70, 0x77, 0x7E, 0x79, 0x6C, 0x6B, 0x62, 0x65, 0x48, 0x4F, 0x46, 0x41, 0x54, 0x53, 0x5A, 0x5D,
    0xE0, 0xE7, 0xEE, 0xE9, 0xFC, 0xFB, 0xF2, 0xF5, 0xD8, 0xDF, 0xD6, 0xD1, 0xC4, 0xC3, 0xCA, 0xCD,
    0x90, 0x97, 0x9E, 0x99, 0x8C, 0x8B, 0x82, 0x85, 0xA8, 0xAF, 0xA6, 0xA1, 0xB4, 0xB3, 0xBA, 0xBD,
    0xC7, 0xC0, 0xC9, 0xCE, 0xDB, 0xDC, 0xD5, 0xD2, 0xFF, 0xF8, 0xF1, 0xF6, 0xE3, 0xE4, 0xED, 0xEA,
    0xB7, 0xB0, 0xB9, 0xBE, 0xAB, 0xAC, 0xA5, 0xA2, 0x8F, 0x88, 0x81, 0x86, 0x93, 0x94, 0x9D, 0x9A,
    0x27, 0x20, 0x29, 0x2E, 0x3B, 0x3C, 0x35, 0x32, 0x1F, 0x18, 0x11, 0x16, 0x03, 0x04, 0x0D, 0x0A,
    0x57, 0x50, 0x59, 0x5E, 0x4B, 0x4C, 0x45, 0x42, 0x6F, 0x68, 0x61, 0x66, 0x73, 0x74, 0x7D, 0x7A,
    0x89, 0x8E, 0x87, 0x80, 0x95, 0x92, 0x9B, 0x9C, 0xB1, 0xB6, 0xBF, 0xB8, 0xAD, 0xAA, 0xA3, 0xA4,
    0xF9, 0xFE, 0xF7, 0xF0, 0xE5, 0xE2, 0xEB, 0xEC, 0xC1, 0xC6, 0xCF, 0xC8, 0xDD, 0xDA, 0xD3, 0xD4,
    0x69, 0x6E, 0x67, 0x60, 0x75, 0x72, 0x7B, 0x7C, 0x51, 0x56, 0x5F, 0x58, 0x4D, 0x4A, 0x43, 0x44,
    0x19, 0x1E, 0x17, 0x10, 0x05, 0x02, 0x0B, 0x0C, 0x21, 0x26, 0x2F, 0x28, 0x3D, 0x3A, 0x33, 0x34,
    0x4E, 0x49, 0x40, 0x47, 0x52, 0x55, 0x5C, 0x5B, 0x76, 0x71, 0x78, 0x7F, 0x6A, 0x6D, 0x64, 0x63,
    0x3E, 0x39, 0x30, 0x37, 0x22, 0x25, 0x2C, 0x2B, 0x06, 0x01, 0x08, 0x0F, 0x1A, 0x1D, 0x14, 0x13,
    0xAE, 0xA9, 0xA0, 0xA7, 0xB2, 0xB5, 0xBC, 0xBB, 0x96, 0x91, 0x98, 0x9F, 0x8A, 0x8D, 0x84, 0x83,
    0xDE, 0xD9, 0xD0, 0xD7, 0xC2, 0xC5, 0xCC, 0xCB, 0xE6, 0xE1, 0xE8, 0xEF, 0xFA, 0xFD, 0xF4, 0xF3 };

/*****************************************************************************
* Function Name  : makeCRC8
* Description    : Calculates CRC8 Checksum
* Input          : * u8Tele : Pointer to the data array
*                : len : Length of data in Bytes
* Output         : None
* Return         : CRC8
*****************************************************************************/

uint8_t makeCRC8(const uint8_t* u8Tele, uint16_t len)
{
        uint8_t u8Result = 0;
        uint16_t i;
        for (i = 0; i<len; i++)
        {
                u8Result = u8CRC8Table[u8Result ^ u8Tele[i]];
        }
        return u8Result;
}
```


ME-Meßsysteme GmbH
Eduard-Maurer Str. 9
16761 Hennigsdorf
Phone: +49 3302 89824 60
Fax: +49 3302 89824 69
Mail: info@me-systeme.de
Web: www.me-systeme.de

Made in Germany

Subject to changes.
All details describe our products in a general form.

They constitute no warranty of characteristics as defined by § 459, Paragraph 2 of the German Civil Code nor grounds for liability.