GSV-PROTOKOLLSPEZIFIKATION

GSV-6 / GSV-8

Stand: 18.5.2017

Änderungsnachweis		
Version	**Status**	**Bearbeiter**
00.00.01	initiiert	Dre
00.00.02	erstellt	Sei
00.01.00	überarbeiten	Pasch
00.01.01	überarbeitet	Dre
00.02.00	geprüft	Sei
01.00.00	freigegeben	Pasch
01.00.01	überarbeitet	Kab
01.00.02	Überarbeitet	Kab
01.00.03	überarbeitet	Dre
01.00.04	überarbeitet	Schu
01.00.05	überarbeitet & kommentiert bis Kap. 1.11.5	SW
01.00.06	überarbeitet	Schu
01.00.07	überarbeitet	SW
01.00.08	überarbeitet, korrigiert	SW
01.00.09	überarbeitet, Kap. 1.9 ergänzt	SW
01.00.10	überarbeitet, Kap. 1.9 erweitert	SW
01.00.11	Überarbeitet, Kap. 1.5	Schu
01.00.12	Umformatiert auf ME-Vorlage Ooo	SW
01.00.13	Kleinere Änderungen, Ooo Versionierung eingeführt	Kab/SW
01.00.14	Read / Write Interface Setting ergänzt.	SW
01.00.15	TEDS-Kommandos ergänzt, kl. Korrekturen	SW
01.00.16	Aktualisiert, Anhang gegliedert	SW
01.00.17	Aktualisiert, korrigiert	SW
01.00.18	Aktualisiert	SW
01.00.19	Aktualisiert, korrigiert	SW

Dateiname: PK01079 TB03 01.00.19 GSV-Protokollspez_public.odt
pdf-Datei: ba-gsvcom.pdf

Hinweise zur Verwendung des GSV-6

Hinweise zu Gerätesoftwareversionen bis einschließlich 03 00 05

a) Beim GSV-6 ist die Anzahl der übertragenen Messwerte im Datenframe im Auslieferzustand =6.

b) Durch die Anwendung des Befehls GetAll werden Default Einstellungen geladen, z.B. Messfrequenz 100Hz und andere Parameter; Die Anzahl der übertragenen Kanäle wird durch Anwendung des Kommandos GetAll auf 1 gesetzt.

Inhaltsverzeichnis

Einleitung

Zweck

Dieses Dokument beschreibt die Protokollspezifikation für die Systeme GSV-6 und GSV-8. Es werden der grundlegende Protokollstrukturaufbau und die einzelnen Befehle beschrieben.

Zielgruppe

Dieses Dokument richtet sich an Entwickler.

Konvention

Folgende beschreibenden Elemente werden verwendet:

<num>	Einzelnes Bit mit Position ‚num' gezählt von 0
<high:low>	Bit-String von Position ‚low' bis ‚high' (inclusive)
[low-high]	Wertebereich (Grenzen inklusive)
name[num]	Byte/Zeichen ‚num' aus einem Array ‚name'

GSV-Protokoll

Allgemein

Die Geräte GSV-6 und GSV-8 benutzen für die serielle Kommunikation ein nahezu identisches Protokoll.

Die grundlegende Struktur der Protokollblöcke ist gleich; sie unterscheiden sich bei bestimmten Befehlen in ihren Parametern und dem Befehlsverhalten (systembedingte Abweichungen).

Die Abweichungen sind in den einzelnen Befehlsbeschreibungen markiert.

Numerische Werte werden innerhalb eines Frames im Big-Endian-Format übertragen, d.h. alle Mehrbyte-Datentypen werden vom MSB zum LSB übertragen, bzw. in die Struktur eingefügt. Ebenso werden sie in dieser Reihenfolge in den Befehlsbeschreibungen aufgelistet.

Die Aufstellung der Bits erfolgt in den Struktur-Tabellen immer vom MSBit zum LSBit, bis ein Byte vollständig ist.

Protokoll-Grundstruktur

Die Grundstruktur des Protokolls sieht wie folgt aus:

# Bits	Bezeichnung	Beschreibung
(8)	Präfix	Bei einer nicht Block-Orientierten Datenübertragung (seriell) beginnt ein Paket mit dem Präfix-Byte 0xAA
2	Frame-Type	Beschreibung des übertragenen Frames: 0b00: Messwert-Frame 0b01: Befehl: Antwort 0b10: Befehl: Anfrage 0b11: <Reserviert>
2	Interface	Beschreibt das Interface über den der Frame versendet wird/wurde. 0b00: CAN 0b01: Seriell (RS232, USB ...) 0b10: <Reserviert> 0b11: <Reserviert>
4	Länge	Dieses Feld wird abhängig von den Feldern Frame-Type und Interface unterschiedlich interpretiert. Die entsprechende Beschreibung erfolgt in den folgenden Kapiteln zu den Frame-Typen.
8	Steuerbyte / Statusbyte	Dieses Feld wird abhängig vom Feld Frame-Type unterschiedlich interpretiert. Die entsprechende Beschreibung folgt in den folgenden Kapiteln zu den Frame-Typen.
0-480 bzw. 0-120	Daten	Das Datenfeld besitzt unterschiedliche Längen und wird Frame Type-spezifisch und befehlsspezifisch interpretiert. Bei CAN wird immer ein vollständiges Paket übertragen und die Daten sind auf 48 Bit beschränkt.
(8)	Suffix	Bei einer nicht Block-Orientierten Datenübertragung (seriell) endet ein Paket mit dem Suffix-Byte 0x85

Tabelle: Protokollgrundstruktur

Der Hauptunterschied in der Grundstruktur liegt bei den unterschiedlichen Schnittstellen.

Bei der CAN-Schnittstelle können maximal 8 Byte in einem Paket übertragen werden.

Bei einer seriellen Schnittstelle kann die Anzahl der übertragenen Werte größer gestaltet werden, da es sich um einen Stream handelt. Um hier eindeutige Paket-Grenzen zu erhalten, werden Präfix- und Suffix-Bytes übertragen.

Anfrage-Frame (0b10)

Der Anfrage-Frame wird genutzt, um Befehle (Anfragen) an das Gerät zu übermitteln. Jede Anfrage wird grundsätzlich mit einem Antwort-Frame beantwortet (bis auf eine Ausnahme, nämlich GetValue). Verschiedene Anfragen können u.a. Parameter auslesen („Read...“, „Get...“), schreiben („Write...“, „Set…“) oder Vorgänge auslösen („Set...“). Die Anfragen werden durch die Kommando-Nummer unterschieden; Parameterliste und Rückgabewerte sind diesen Kommandos konstant zugeordnet, d.h. durch diese festgelegt.

Hinweis: Es wird empfohlen, die Antwort auf eine Anfrage erst abzuwarten, bevor eine neue versendet wird.

CAN-Anfrage-Frame

# Bits	Bezeichnung	Beschreibung
2	Frame-Type	0b10: Befehl: Anfrage
2	Interface	0b00: CAN
4	Länge	Anzahl der aktiven Datenbytes [0-6]
8	Steuerbyte	Kommando-Nummer
48	Daten	Kommando-Parameter

Tabelle: CAN-Anfrage Protokoll-Frame

Hinweis: CAN-Anfragen sind immer 64 Bit / 8 Byte groß!

Seriell-Anfrage-Frame

# Bits	Bezeichnung	Beschreibung
8	Präfix	Präfix-Byte 0xAA
2	Frame-Type	0b10: Befehl: Anfrage
2	Interface	0b01: Seriell (RS232, USB ...)
4	Länge	Anzahl der Datenbytes [0-15]
8	Steuerbyte	Kommando-Nummer
0-120	Daten	Kommando-Parameter
8	Suffix	Suffix-Byte 0x85

Tabelle: Seriell-Anfrage Protokoll-Frame

Antwort-Frame (0b01)

Antwort-Frames erhalten ein Statusbyte, in dem der Fehler-Code enthalten ist. Eine fehlerfrei aufgeführter Befehl kann zusätzlich Rückgabewerte übermitteln (meist bei Leseanfragen), d.h. das Statusbyte lautet dann stets ERR_OK =0. Wird hingegen ein Fehler signalisiert (Statusbyte >0), enthält dieser Antwortframe keine Daten.

CAN-Antwort-Frame

# Bits	Bezeichnung	Beschreibung
2	Frame-Type	0b01: Befehlsantwort
2	Interface	0b00: CAN
4	Länge	Anzahl der aktiven Bytes in diesem Rückgabeframe [0-6]
8	Statusbyte	Fehler-Code (siehe Fehlercodes Fehlercodes)
48	Daten	Rückgabewert(e)

Tabelle: CAN-Antwort Protokoll-Frame

Hinweis: CAN-Antworten sind immer 64 Bit = 8 Bytes groß!

Seriell-Antwort-Frame

# Bits	Bezeichnung	Beschreibung
8	Präfix	Präfix-Byte 0xAA
2	Frame-Type	0b01: Befehlsantwort
2	Interface	0b01: Seriell (RS232, USB ...)
4	Länge	Anzahl der Daten-Bytes in diesem Rückgabeframe [0-15]
8	Statusbyte	Fehler-Code (siehe Fehler: Referenz nicht gefunden)
0-120	Daten	Rückgabewert(e)
8	Suffix	Suffix-Byte 0x85

Tabelle: Seriell-Antwort Protokoll-Frame

Messwert-Frame (0b00)

Im Messwert-Frame werden die erfassten Messwerte übermittelt. Messwert-Frames können autonom vom Gerät versendet werden.

CAN-Messwert-Frame

# Bits	Bezeichnung		Beschreibung	
2	Frame-Type		0b00: Messwert-Frame	
2	Interface		0b00: CAN	
4	Kanal		Kanalnummer beginnend mit 0 (GSV-6 [0-5] / GSV-8 [0-7])	
8	Steuerbyte / Statusbyte		<7> Indikator ==0 <6:4> Datentyp siehe Datentyp Datentyp <3:0> Error Bits - <3:2> reserviert • <1> 6-Achsen-Fehler • <0> Übersteuerung des Eingangs	
48	Daten	16 Bit	TimeStamp	Zähler, der bei den zeitgleich erfassten Kanälen identisch ist und mit jedem Zyklus um 1 inkrementiert wird. Es besteht kein Zusammenhang zu einer Zeitbasis.
		32 Bit*	Messwert	Der erfasste Messwert des übertragenen Kanals. * Ist der Datentyp kleiner als 32Bit so werden Null-Bits angehängt.

Tabelle: CAN-Messwert-Frame

Hinweis: CAN-Messwert-Frames sind immer 64 Bit / 8 Byte groß!

Serieller-Messwert-Frame

# Bits	Bezeichnung	Beschreibung
8	Präfix	Präfix-Byte 0xAA
2	Frame-Type	0b00: Messwert-Frame
2	Interface	0b01: Seriell (RS232, ...)
4	Anzahl Kanäle	Anzahl der übertragenen Kanäle (minus 1) GSV-6 [0-5] GSV-8 [0-7]
8	Steuerbyte / Statusbyte	<7> Indikator ==1 <6:4> Datentyp siehe Datentyp Datentyp <3:0> Error Bits - <3:2> reserviert - <1> 6-Achsen-Fehler - <0> Übersteuerung des Eingangs
16-512	Daten	Pro Kanal ein Messwert mit angegebenem Datentyp
8	Suffix	Suffix-Byte 0x85

Tabelle: Serieller-Messwert-Frame

Datentyp

Es sind folgende Datentypen für Messwert-Frames definiert:

Bits<6:4> Statusbyte	Enum Cmd 0x80.1 / 0x81.1	Typenname
0b000	-	reserviert
0b001	1	int16
0b010	2	int24 [nur GSV-8]
0b011	3	float32
0b100	-	reserviert
0b101	-	reserviert
0b110	-	reserviert
0b111	-	reserviert

Tabelle: Messwert-Datentyp Definition

Interpretation der Messdaten

Innerhalb des Messdatenframes werden die Messwerte der Eingangskanäle von niedrigster Kanalnummer zur höchsten übertragen, d.h. der mit niedrigster Eingangskanalnummer kommt zuerst. Die Bytereihenfolge der Messwerte selbst ist Big-endian, d.h. das höchstwertigste Byte kommt zuerst.

Die richtige Interpretation der Messwerte, d.h. die Umrechnung zu physikalischen Werten, hängt vom Messdatentyp und der korrekten Parametrierung anhand des verwendeten Sensors ab. Beim Messdatentyp float32 werden fertig normierte Messwerte übertragen. Sofern der Messverstärker richtig parametriert ist (u.a. richtige Kalibriermatrix beim Sechsachsensensor oder richtige UserScale-Werte), entsprechen diese den physikalischen Messwerten.

Bei den Integer-Datentypen müssen die Werte folgendermaßen umgerechnet werden:

1. **Beim GSV-8** liegen die Rohwerte im sog. Binary Offset-Format vor. Daher muss dieser Offset zunächst subtrahiert werden, um Signed-Int Werte zu erhalten:

int16: ZwischenWert_1 = Rohwert - 0x8000

int24: ZwischenWert_1 = Rohwert - 0x800000

Beim GSV-6 wird dieser Schritt übersprungen: ZwischenWert_1 = Rohwert

2. Dieses Zwischenergebnis wird zur Dezimalzahl (d.h. Fließkommazahl) umgewandelt, dann mit dem Messbereichsoverhead 1,05 multipliziert und durch den binären (unipolaren) Messbereich dividiert:

int16: ZwischenWert_2 = ZwischenWert_1 * 1,05 / 2^15

int24: ZwischenWert_2 = ZwischenWert_1 * 1,05 / 2^23

3. So erhält man Messwerte, die stets auf +-1,0 normiert sind. Der Wert 1,0 entspricht dabei dem Nenn-Eingangsmessbereich. Ist dieser z.B. 2mV/V, so müsste der ZwischenWert_2 mit 2 multipliziert werden, um die Brückenverstimmung direkt in mV/V anzuzeigen. Ist der Eingangsmessbereich z.B. Single-Ended 10V, müsste ZwischenWert_2 mit 10 multipliziert werden, um die Eingangsspannung direkt in Volt anzuzeigen (vorausgesetzt, der Nullpunkt wurde nicht durch SetZero o.a. auf einen Wert ungleich 0 verschoben).

Folgende Tabelle zeigt beispielhaft Rohwerte und ZwischenWert_2 für einen Nenn-Eingangsmessbereich von 2mV/V:

Sensor-auslenkung in mV/V	GSV-8		GSV-6	Lesewert MEGSVxx.dll:: GSVread u.ä. Messwert-Lesefunktionen (ZwischenWert_2)
	Ganzzahliger Messwert, 16-Bit (Uint16) Hex	Ganzzahliger Messwert, 24-Bit (Uint24) Hex	Ganzzahliger Messwert, 16-Bit (Sint16) Hex	
<= -2,1	0x0000	0x000000	0x8000	-1,05
-2,0	0x0618	0x061862	0x8618	-1,0
0	0x8000	0x800000	0x0000	0,0
2,0	0xF9E7	0xF9E79E	0x79E7	1,0
>= 2,1	0xFFFF	0xFFFFFF	0x7FFF	1,05

Fehlercodes

In diesem Kapitel werden alle definierten Fehler-Codes aufgelistet. Im Statusbyte des Antwort-Frame wird einer der Werte, situationsabhängig, hinterlegt.

Einige Fehlercodes sind reserviert für künftige Anwendungen.

Errorbyte im Befehlsanwortframe

Fehlerkürzel	Wert (Hex)	Bedeutung
ERR_OK	0x00	Kommando ohne Fehler ausgeführt.
ERR_OK_CHANGED	0x01	Kommando ohne Fehler ausgeführt, aber zusätzlich wurden weitere Parameter geändert.
ERR_CMD_NOTKNOWN	0x40	Das Kommando ist unbekannt.
ERR_CMD_NOTIMPL	0x41	Das Kommando ist bekannt, wird jedoch nicht unterstützt. Es kann sich hier um Kommandos handeln, die für den jeweiligen Gerätetypen (GSV-6/-8) nicht vorgesehen sind.
ERR_FRAME_ERROR	0x42	Der Frame des Kommandos ist falsch aufgebaut.
ERR_PAR	0x50	Ein falscher Parameter wurde übergeben.
ERR_PAR_ADR	0x51	Der übergebene Index oder die Adresse ist falsch.
ERR_PAR_DAT	0x52	Die Daten eines Datenparameters sind falsch.
ERR_PAR_BITS	0x53	Falsche Bits im Parameter.
ERR_PAR_ABSBIG	0x54	Der Wert eines Parameters ist absolut zu groß.
ERR_PAR_ABSMALL	0x55	Der Wert eines Parameters ist absolut zu klein.
ERR_PAR_COMBI	0x56	Die Parameterkombination ist fehlerhaft.
ERR_PAR_RELBIG	0x57	Ein Parameter ist in Relation zu den anderen zu groß.
ERR_PAR_RELSMALL	0x58	Ein Parameter ist in Relation zu den anderen zu klein.
ERR_PAR_NOTIMPL	0x59	Das Kommando unterstützt den Parameter nicht.
ERR_PAR_TIMEOUT	0x5A	Die Parameter zum Kommando sind nicht innerhalb

Fehlerkürzel	Wert (Hex)	Bedeutung
		von 200 ms eingetroffen.
ERR_WRONG_PAR_NUM	0x5B	Die Anzahl der Parameter passt nicht zum Kommando. Bits 3~0 im ersten Byte des Kommando Blocks zu hoch bzw. niedrig.
ERR_PAR_NOFIT_SETTINGS	0x5C	Parameter entspricht nicht der Geräteeinstellung
ERR_PAR_HW_COLLISION	0x5D	Durch Parameter angeforderte Funktion führt zu einer Hardware-Kollision
ERR_NO_DATA_AVAIL	0x60	Abgewiesen bei Leseanfrage, weil Daten nicht vorhanden
ERR_DATA_INCONSISTENT	0x61	Gespeicherte Daten falsch, bzw. inkonsistent mit Zustandsanforderung
ERR_WRONG_MOD_STATE	0x62	Befehl konnte nicht ausgeführt werden, weil Gerät bzw. Modul im ungeeignetem Zustand
ERR_NOT_SUPPORTED_D	0x63	Abgewiesen, weil angeforderte Funktionalität nicht unterstützt
ERR_FDATA_TOO_HIGH	0x64	Abgewiesen, weil Datenrate zu hoch, bzw. Prozessor wäre zu ausgelastet
ERR_MEMORY_WRONG_COND	0x6E	Abgewiesen bei Memory-Interface-write, weil Bed. nicht erf.
ERR_MEMORY_ACCESS_DENIED	0x6F	Abgewiesen bei Memory-Interface-write
ERR_ACC_DEN	0x70	Kommando wird nicht ausgeführt, weil die Berechtigung fehlt.
ERR_ACC_BLK	0x71	Kommando wurde nicht ausgeführt, da Blocking gesetzt ist.
ERR_ACC_PWD	0x72	Passwort falsch oder nicht gesetzt.
ERR_ACC_MAXWR	0x74	Die maximale Anzahl der Ausführungen ist überschritten.
ERR_ACC_PORT	0x75	Keine Schreibrechte auf dem Port.
ERR_INTERNAL	0x80	Interner Ausnahmefehler.
ERR_ARITH	0x81	Interner arithmetischer Fehler.
ERR_INTER_ADC	0x82	Fehlerhaftes Verhalten des AD-Umsetzers.
ERR_MWERT_ERR	0x83	Zur Befehlsausführung ungeeigneter Messwert
ERR_EEPROM	0x84	Fehlerhaftes Verhalten des EEPROMs
ERR_RET_TXBUF	0x91	Der Sendepuffer ist voll.
ERR_RET_BUSY	0x92	Die CPU ist zu ausgelastet, um einen Befehl auszuführen.
ERR_RET_RXBUF	0x99	Der Empfangspuffer ist voll.
GETTEDS_ERR_NOTEDSEE	0xB1	Kein TEDS-fähiger Speicherbaustein angeschlossen
GETTEDS_ERR_BASICONLY	0xB2	Der TEDS-Speicher enthält nur Basic-Template Daten
GETTEDS_ERR_NOTEDSDAT	0xB3	Daten im TEDS-Speicher entsprechen nicht der Norm
GETTEDS_ERR_ENTRY_INVALID	0xB4	Eintrag im TEDS-Speicher nicht (richtig) gesetzt
GETTEDS_ERR_TOUT	0xB5	Hardware-TEDS-Gerätetreiber timeout
GETTEDS_ERR_CHKSUM	0xB6	Prüfsummenfehler der TEDS Daten
GETTEDS_ERR_UNKNOWN_TEMPL	0xB7	(Noch) nicht unterstütztes TEDS-template

Fehlerkürzel	**Wert** (Hex)	**Bedeutung**
GETTEDS_ERR_VERIFY_FAIL	0xB8	Leseverifizierung nach Schreiben fehlgeschlagen

Tabelle: Fehlercodes des Errorbytes im Befehlsanwortframe

Fehlercodes für „GetLastValueError" (0x43) beim GSV-8

Die 48 Bits (6 Bytes) der Error-Struktur im Einzelnen:

Bits<47:45>

Error-Type: Art des Fehlers (Kategorie)		
Wert	Name	Bedeutung
1	VALERR_TYPE_SATURATED	Analogwert am Sensoreingang gesättigt, d.h. Eingangsmessbereich überschritten
2	VALERR_TYPE_MAX_EXCEED	Maximal erlaubter physikalischer Wert überschritten (insbes. bei 6-Achsensensor)
3	VALERR_TYPE_SENSOR_BROKEN	Brückensensor oder dessen Anschlussleitung defekt
4	HWERR_TYPE_ANA_OUT	Als Stromausgang konfigurierter Analogausgang offen oder Analogausgangstreiber überhitzt
5	HWERR_TYPE_DIO	Als Ausgang konfigurierter GPIO (Digitalanschluss) kurzgeschlossen

Bits<44:16>

Error-Time: Zeit des Fehlers

Gespeichert wird die Fehlerzeit in Minuten des Gerätebetriebs, d.h. dieser Wert/60 sind die (absoluten) Betriebsstunden, zu der der Fehler auftrat.

Bits<15:0> Error-Flags Typabhängige Fehlerkodierung

Beschreibung der Error-Flags im Einzelnen:

1. ErrType = VALERR_TYPE_SATURATED:

Bits<15:8>: Wenn Bit=1: Negative Sättigung ist in einem oder mehreren Eingangskanälen aufgetreten, wobei Bit 8 Kanal 1 entspricht, Bit 9 Kanal 2, usw, bis Bit 15: Kanal 8.

Bits<7:0>: Wenn Bit=1: Positive Sättigung ist in einem oder mehreren Eingangskanälen aufgetreten, wobei Bit 0 Kanal 1 entspricht, Bit 1 Kanal 2, usw, bis Bit 7: Kanal 8.

Bemerkungen:

1. Liegt dieser Fehler aktuell vor (Index 1, s.o.), so leuchtet die "FUNCTION"-LED des Gerätes dauernd rot.

2. Ferner ist dann das Bit 0 des Statusbytes im Messdatenframe gesetzt.

2. ErrType = VALERR_TYPE_MAX_EXCEED:

2.1. Bei Sechsachsensensor

Bit0: Wenn Bit=1: In Fx-Richtung ist ein (positives oder negatives) Überschreiten des in den Sechachsenkalibrierdaten definierten Maximalwertes aufgetreten.

Bit1: Wenn Bit=1: In Fy-Richtung ist ein (positives oder negatives) Überschreiten des in den Sechachsenkalibrierdaten definierten Maximalwertes aufgetreten.

Bit2: Wenn Bit=1: In Fz-Richtung ist ein (positives oder negatives) Überschreiten des in den Sechachsenkalibrierdaten definierten Maximalwertes aufgetreten.

Bit3: Wenn Bit=1: In Mx-Richtung ist ein (positives oder negatives) Überschreiten des in den Sechachsenkalibrierdaten definierten Maximalwertes aufgetreten.

Bit4: Wenn Bit=1: In My-Richtung ist ein (positives oder negatives) Überschreiten des in den Sechachsenkalibrierdaten definierten Maximalwertes aufgetreten.

Bit5: Wenn Bit=1: In Mz-Richtung ist ein (positives oder negatives) Überschreiten des in den Sechachsenkalibrierdaten definierten Maximalwertes aufgetreten.

2.2. Bei PT1000 Temperatursensor:

Bits<7:0>: An Kanal (BitNo+1) ist eine Maximalwertüber- oder Unterschreitung aufgetreten. Der Messwert wird bei Maximalwertüberschreitungf auf 9999 gesetzt.

Bits<15:8>: An Kanal (BitNo-7) ist eine Maximalwertunterschreitung aufgetreten. Der Messwert wird bei Maximalwertüberschreitungf auf -9999 gesetzt.

Bemerkungen:

1. Liegt dieser Fehler aktuell vor (Index 1, s.o.), so leuchtet die "FUNCTION"-LED des Gerätes dauernd rot.

2. Ferner ist dann das Bit 1 des Statusbytes im Messdatenframe gesetzt.

3. ErrType = VALERR_TYPE_SENSOR_BROKEN:

Bit0: Wenn Bit=1: An der Ud+ Leitung des Kanals 1 lag ein Fehler vor.

Bit1: Wenn Bit=1: An der Ud- Leitung des Kanals 1 lag ein Fehler vor.

Bit2: Wenn Bit=1: An der Ud+ Leitung des Kanals 2 lag ein Fehler vor.

Bit3: Wenn Bit=1: An der Ud- Leitung des Kanals 2 lag ein Fehler vor.

Bit4: Wenn Bit=1: An der Ud+ Leitung des Kanals 3 lag ein Fehler vor.

Bit5: Wenn Bit=1: An der Ud- Leitung des Kanals 3 lag ein Fehler vor.

Bit6: Wenn Bit=1: An der Ud+ Leitung des Kanals 4 lag ein Fehler vor.

Bit7: Wenn Bit=1: An der Ud- Leitung des Kanals 4 lag ein Fehler vor.

Bit8: Wenn Bit=1: An der Ud+ Leitung des Kanals 5 lag ein Fehler vor.

Bit9: Wenn Bit=1: An der Ud- Leitung des Kanals 5 lag ein Fehler vor.

Bit10: Wenn Bit=1: An der Ud+ Leitung des Kanals 6 lag ein Fehler vor.

Bit11: Wenn Bit=1: An der Ud- Leitung des Kanals 6 lag ein Fehler vor.

Bit12: Wenn Bit=1: An der Ud+ Leitung des Kanals 7 lag ein Fehler vor.

Bit12: Wenn Bit=1: An der Ud- Leitung des Kanals 7 lag ein Fehler vor.

Bit14: Wenn Bit=1: An der Ud+ Leitung des Kanals 8 lag ein Fehler vor.

Bit15: Wenn Bit=1: An der Ud- Leitung des Kanals 8 lag ein Fehler vor.

Bemerkung: Liegt dieser Fehler aktuell vor (Index 1, s.o.), so leuchtet die "FUNCTION"-LED des Gerätes dauernd rot.

4. ErrType = HWERR_TYPE_ANA_OUT:

Konstanter Wert 0xFFFF: An irgendeinem der Analogausgangskanäle lag ein Fehler vor. Sonst:

Bit0: Wenn Bit=1: An Ausgangskanal 1 liegt ein offener Stromausgang vor.

Bit1: Wenn Bit=1: An Ausgangskanal 2 liegt ein offener Stromausgang vor.

Bit2: Wenn Bit=1: An Ausgangskanal 3 liegt ein offener Stromausgang vor.

Bit3: Wenn Bit=1: An Ausgangskanal 4 liegt ein offener Stromausgang vor.

Bit4: Wenn Bit=1: An Ausgangskanal 5 liegt ein offener Stromausgang vor.

Bit5: Wenn Bit=1: An Ausgangskanal 6 liegt ein offener Stromausgang vor.

Bit6: Wenn Bit=1: An Ausgangskanal 7 liegt ein offener Stromausgang vor.

Bit7: Wenn Bit=1: An Ausgangskanal 8 liegt ein offener Stromausgang vor.

Bit8: Wenn Bit=1: An Ausgangskanal 1 ist der Ausgangstreiber überhitzt

Bit9: Wenn Bit=1: An Ausgangskanal 2 ist der Ausgangstreiber überhitzt

Bit10: Wenn Bit=1: An Ausgangskanal 3 ist der Ausgangstreiber überhitzt

Bit11: Wenn Bit=1: An Ausgangskanal 4 ist der Ausgangstreiber überhitzt

Bit12: Wenn Bit=1: An Ausgangskanal 5 ist der Ausgangstreiber überhitzt

Bit13: Wenn Bit=1: An Ausgangskanal 6 ist der Ausgangstreiber überhitzt

Bit14: Wenn Bit=1: An Ausgangskanal 7 ist der Ausgangstreiber überhitzt

Bit15: Wenn Bit=1: An Ausgangskanal 8 ist der Ausgangstreiber überhitzt

Bemerkungen:

1. Der Überhitzung des Ausgangstreibers liegt möglicherweise ein Kurzschluss des dementsprechenden Spannungsausgangs zugrunde.

2. Liegt dieser Fehler aktuell vor (Index 1, s.o.), so blinkt die "FUNCTION"-LED des Gerätes langsam (ca 1x/Sek) rot.

5. ErrType = HWERR_TYPE_DIO:

Bits<15:0> Wenn Bit=1: An der entsprechneden DIO-No. ist ein Kurzschluss aufgeteten, d.h. wenn dieser als Ausgang und auf High geschaltet ist, ist er mit GNDD kurzgeschlossen, oder wenn er auf Low geschaltet ist, ist eine Spannung >=3V angeschlossen. Bit 0 entspricht dabei DIOno 1 (Group1: 1.1.), Bit 1 DIOno 2 (Group1: 1.2.), usw bis Bit 15: DIOno 16 (Group4: 4.4.)

Bemerkung:

Liegt dieser Fehler aktuell vor (Index 1, s.o.), so blinkt die "FUNCTION"-LED des Gerätes schnell (ca 3x/Sek) rot.

Befehlsbeschreibung

Im folgenden Unterkapitel wird die Struktur der Befehls-Dokumentation erläutert, danach folgt das Unterkapitel mit der Beschreibung jedes Befehls.

Allgemeine Struktur

Bei der Beschreibung der einzelnen Befehle wird folgende Tabellenstruktur genutzt:

Nr.	Richtung (R / W)	Name	Gerätemodel
Beschreibung			
Nr. Byte	Nr. Parameter		
Offset P1	Parameter1 Typ	Paramter1 Name	Parmeter1 Beschreibung
Offset P2	Parameter2 Typ	Paramter2 Name	Paramter2 Beschreibung
Nr. Byte	Nr. Rückgaben		
Offset R1	Rückgabe 1 Typ	Rückgabe1 Name	Rückgabe1 Beschreibung

Tabelle: Allgemeine Befehlsbeschreibung

Zu beachten ist, dass die Anzahl der Bytes und der Offset sich auf die Daten-Bytes im Anfrage-/Antwort-Frame beziehen.

Sollten sich die Befehle von GSV-6 und GSV-8 grundlegend unterscheiden (z.B. in Datentypen), werden separate Beschreibungen aufgeführt. Ebenso bei größeren Kommunikationsweg-Unterschieden (CAN, Seriell).

Als Gerätemodell können folgende Einträge (auch in Kombination) auftreten:

Gerätemodel	Beschreibung
GSV-6	GSV-6 allgemein
GSV-6 (Reset)	GSV-6 Befehl benötigt zur Aktivierung der Änderung einen Reset/Powercycle.
GSV-8	GSV-8 allgemein

Tabelle: Gerätemodelle

Die auftretenden Datentypen in Parameter- und Rückgabetyp sind:

Bezeichner	Bytes	Beschreibung
uint8_t	1	Vorzeichenloser 8-Bit (Byte) Integer
uint16_t	2	Vorzeichenloser 16-Bit Integer
U24	3	Vorzeichenloser 24-Bit Integer
S24	3	Vorzeichenbehafteter 24-Bit Integer
uint32_t	4	Vorzeichenloser 32-Bit Integer
int32_t	4	Vorzeichenbehafteter 32-Bit Integer
S7.24	4	Vorzeichenbehaftete Fixpunkt-Zahl (Vorzeichen-Bit, 7 Vor-, 24 Nachkommabits)
float	4	32-Bit Fließkommazahl
char[x]	x	Zeichenkette

Tabelle: Parameter und Rückgabe Datentypen

Bei den Befehlen treten diverse Elemente als Parameter bzw. Rückgabetyp gehäuft auf, diese logischen Typen sind in der folgenden Tabelle kurz beschrieben:

Bezeichner	Beschreibung
Kanal	Kanal-Nummer werden i.a. mit 1 beginnend gezählt, bis auf die dokumentierten Ausnahmen. Der GSV-6 hat 6 Kanäle mit Kanal-Nr. 1-6. Der GSV-8 hat 8 Kanäle 1-8.

Bezeichner	Beschreibung
	Als Sonderwert kann bei den Schreibbefehlen (Set*, Write*) der Wert 0 übergeben werden, so dass der entsprechende Wert für alle Kanäle gesetzt wird.
Index, Sub Index	Meist ein Wert mit dem auf ein Array zugegriffen, bzw. ein durch Kanal/Adresse spezifiziertes Element ausgewählt wird.
Flags	Bit-Felder, deren Bit-Position einer Konfiguration entsprechen.

Tabelle: Häufige Parameter- bzw. Rückgabewert-Bezeichner

Befehlstypen

Datenfluss konfigurieren

Die autonome permanente Datenübertragung kann mit *StartTransmission* und *StopTransmission* gestartet und gestoppt werden. Die Datenübertragung eines einzelnen Messwertes wird mit *GetValue* ausgelöst.

Die Anzahl der Messwerte, die pro Sekunde übertragen oder intern erzeugt werden (bei mit *StopTransmission* gestoppter Datenübertragung) wird mit *WriteDataRate* eingestellt.

Der mit *StartTransmission* und *StopTransmission* eingestellte Zustand ist flüchtig und wird beim Einschalten zurückgesetzt.

Das Verhalten des Messverstärkers bezüglich der Datenübertragung nach dem Einschalten kann mit SetTXmode gesetzt werden (Messverstärker sendet permanent Daten oder sendet keine Daten).

Nullpunkt verschieben

Die aktuelle Messwertanzeige lässt sich durch Anwendung des Kommandos *SetZero* auf 0 verschieben. Diese Funktion wird auch durch den Steuereingang „Tara“ ausgelöst.

Beim GSV-6 kann über den Parameter des Kommandos *WriteZero* festgelegt werden, auf welchen Wert die Messwertanzeige bei Anwendung des Kommandos *SetZero* verschoben werden soll. Mit *ReadZero* lässt sich dieser Parameter abfragen.

Die Funktion *SetZero* wirkt unmittelbar auf die „Rohwerte“ des Analog-Digital-Umsetzers und somit noch vor jeglicher Weiterverarbeitung der Messwerte durch Anwendung von Benutzer-Skalierungen, wie zum Beispiel *SetUserOffset* und *SetUserScale*.

Skalierung einstellen

Mit den Kommandos *WriteUserOffse*t und *WriteUserScale* wird die Skalierung der Messwerte eingestellt.

Die Gesamtskalierung wird außerdem durch die Eingangsbeschaltung des Messverstärkers beeinflusst. Beim GSV-6 kann die Eingangsempfindlichkeit des Messverstärkers über das Kommando *MEwriteInputRange* gesetzt werden, bei GSV-8 wird hierfür *SetInputType* verwendet.

Bei einer Eingangsempfindlichkeit von zum Beispiel 2 mV/V muss der Skalierungsfaktor mit *WriteUserScale* auf 2 eingestellt werden, um eine richtige Anzeige zu erhalten. Die Rohmesswerte werden stets auf einen Bereich von ±1 normiert übertragen. Der Wert 1 entspricht stets 100% der eingestellten Eingangsempfindlichkeit bzw. des eingestellten elektrischen Eingangsbereiches (zum Beispiel +10V oder 2mV/V).

Der Skalierungsfaktor kann mit *WriteUserScale* so eingestellt werden, dass mit dem angeschlossenen Sensor physikalisch richtig skalierte Messwerte ausgegeben werden, da der auf ±1 normierte Rohwert hiermit multipliziert wird:

Physikalischer Messwert mit Datentyp Float = Rohwert * Skalierungsfaktor

Zur Skalierung von Mehrkomponentensensoren (Kraft-/Drehmoment Sensoren, Force-/Torque Sensoren, F/T Sensoren) wird eine Matrizenmultiplikation durchgeführt. Die Matrizenmultiplikation wird aktiviert, indem ein Bit in einem Modus Register mit dem Befehl *SetMode* gesetzt wird. Wenn der Modus für F/T Sensoren aktiviert ist, sind die Skalierungen *WriteUserOffset* (nur GSV-6) und *WriteUserScale* (GSV-6 und -8) für einachsige Sensoren unwirksam. Das Nullsetzen mit *SetZero* ist weiterhin wirksam. Es sollte dann allerdings auf alle Kanäle angewendet werden. Beim GSV-8 wird das Kommando bei Anwendung auf einzelne Kanäle dann abgewiesen.

Das Setzen der Einheit mit *SetUnitNo* hat grundsätzlich keinen Einfluss auf die Skalierung, sondern ist ausschließlich für die Darstellung in der Anzeige von Bedeutung.

Die Skalierung des Analogausgangs erfolgt nach der Skalierung der Messwerte; sie kann mit dem Kommando *WriteAoutScale* gesetzt werden.

Schnittstellen konfigurieren

Der CANbus wird mit dem Kommando *SetCANSetting* konfiguriert.

Sonderfunktionen konfigurieren

Es stehen eine Reihe von Sonderfunktionen zur Verfügung, wie zum Beispiel die Konfiguration eines IIR Filters vierter Ordnung, das Auslesen von Fehlerspeichern, eine Rauschunterdrückung im Bereich des Nullwertes, das Auslesen von Minimal- und Maximalwertspeichern, und vieles mehr.

Zugangsschutz, Schreibschutz

Einige Befehle haben für ihr Funktionieren das Setzen der Benutzer-ID, d.h. des richtigen Passwortes zur Voraussetzung. Dies geschieht beim GSV-8 mit dem Kommando *SetPassword.*

In der Beschreibung ist dies dann vermerkt ("Benutzer-ID erforderlich").

Ferner können beim GSV-8 per Flag im Wert des *SetMode* Kommandos alle Schreibbefehle gesperrt werden. Das Entsperren ist nut mit gesetzter Benutzer-ID möglich. Darüber hinaus kann ein Schreibbefehl abgewiesen werden, weil ein anderes Kommunikationsinterface das Schreibrecht hat. Dies ist z.B. bei Geräten mit Feldbus (EtherCat, CANopen) dann der Fall, wenn der Feldbus aufgrund des „States“ das Schreibrecht besitzt (z.B. ab Pre-Operational State).

Beschreibungen

ResetStatus (0x00)

0x00	**W**	**ResetStatus**	**GSV-6 / GSV-8**
Alle Protokoll-Fehlerzustände löschen. GSV-8: Fehler der Messapplikation (siehe *GetLastValueError)* werden ebenfalls gelöscht, aber nur dann, wenn deren Ursache nicht mehr vorliegt.			
0 Byte	0 Parameter		
0 Byte	0 Rückgaben		

GetInterface (0x01)

0x01	**R**	**GetInterface**	**GSV-6 / GSV-8**
Abfrage der Schnittstellenbeschreibung. Mit diesem Befehl wird nicht nur die aktuelle Schnittstelle u.a. abgefragt, sondern mit einem Parameter auch die gewünschte Messdatenübertragung gesteuert.			
1 Byte	1 Parameter		
0	uint8_t (1)	Request-Flag	<7:2> Reservierte Bits (=0) <1:0> Einzustellende

0x01	R	GetInterface	GSV-6 / GSV-8
			Messdatenübertragung. - 0b00: Keine Änderung • 0b01: Übertragung AUS (OFF) • 0b10: Übertragung AN (ON)
4 Byte	4 Rückgaben		
0	uint8_t (1)	Protokoll und Gerätemodel	<7:6> Protokoll-Typ ==0b01 <5:0> Gerätemodel - 0x00: Unbekannt - 0x06: GSV-6 - 0x08: GSV-8
1	uint8_t (1)	Messwertframe-Info	<7:4> Anzahl der Messwertframe-Objekte minus 1 <3> Aktuelle Einstellung zur Messdatenübertragung (ON/OFF) <2:0>Messwertdatentyp (siehe Datentyp Datentyp) - 0x01: int16 - 0x02: S24 (nur GSV-8) - 0x03: float
2	uint8_t (1)	Schreibschutz	<7> Indikator ob der Schnittstellen-spezifische Schreibschutz aktiv ist. <6> Genereller-Schreibschutz ist aktiv. <5:0> Nummer der Schnittstelle, über die gerade kommuniziert wird (0...N-1)
3	uint8_t (1)	Deskriptor Anzahl	Anzahl der vorhandenen Schnittstellen N. Bestimmt auch den Indexbereich der "Basic settings" für die Kommandos 0x7B/0x7C

ReadZero (0x02)

0x02	R	ReadZero	GSV-6 / GSV-8
Aktuellen Tara-Wert auslesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Abzufragender Kanal GSV-6: [1-6] GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	int32_t (4)	Tara-Wert	Aktueller Tara-Wert Anmerkung: GSV-6 besitzt ‚nur' einen 16-Bit Tara-Wert, der hier auf 32-Bit erweitert zurückgegeben wird, beim GSV-8 wird von 24 auf 32 Bits sign-extended.

WriteZero (0x03)

0x02	W	WriteZero	GSV-6 / GSV-8
Neuen Tara-Wert setzen GSV-8: Voraussetzung ist ein gesetzter USER_ID_LEVEL			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	int32_t (4)	Tara-Wert	Neuen Tara-Wert schreiben. Anmerkung: GSV-6 nutzt nur die niederwertigeren 16Bits. GSV-8: 24 Bits.
0 Byte	0 Rückgaben		

ReadAoutOffset (0x04)

0x04	R	ReadAoutOffset	GSV-6 / GSV-8
Auslesen des Analogausgangsoffset (in Prozent)			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Abzufragender Kanal GSV-6: [1] GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	float (4)	Offset-Wert	Analogausgangsoffset im Bereich: -60.0 bis 60.0

WriteAoutOffset (0x05)

0x04	W	WriteAoutOffset	GSV-6 (Reset) / GSV-8
Setzen eines Analogausgangsoffset (in Prozent)			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-6: [0-1] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	float (4)	Offset-Wert	Analogausgangsoffset im Bereich: -60.0 bis 60.0
0 Byte	0 Rückgaben		

ReadAoutScale (0x06)

0x06	R	ReadAoutScale	GSV-6
Auslesen des User-Scale-Werts			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Abzufragender Kanal GSV-6: [1-6] GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	float (4)	Scale-Wert	Aktueller User-Scale-Wert

0x06	R	ReadAoutScale	GSV-8
Auslesen der Skalierung des analogen Ausgangs			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Abzufragender Kanal GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	float (4)	Scale-Wert	Aktuelle Skalierung des analogen Ausgangs

WriteAoutScale (0x07)

0x07	W	WriteAoutScale	GSV-6
Setzen des User-Scale-Werts			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-6: [0-6] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	float (4)	Scale-Wert	Neuer Wert für die Eingangsskalierung. die die ganze Meßverarbeitungskette betrifft, d.h. auch den analogen Ausgang
0 Byte	0 Rückgaben		

0x07	W	WriteAoutScale	GSV-8
Setzen der Skalierung des analogen Ausgangs			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	float (4)	Scale-Wert	Neue Skalierung des analogen Ausgangs
0 Byte	0 Rückgaben		

WriteAoutDirect (0x08) [GSV-8]

0x08	W	WriteAoutDirect	GSV-8
Ausgang auf einen Wert setzen. Diese Befehlt funktioniert nur, wenn das Gerät mit dem Befehl SetAoutType (0x0E) in den Direkt-Modus versetzt wurde.			
3 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-8: [1-8]
1	uint16_t (2)	DAC-Code	Wert, der direkt in das DAC-Register geschrieben wird.
0 Byte	1 Rückgaben		

LoadConfig (0x09)

0x09	W	GetAll	GSV-6 / GSV-8
Die Betriebsparameter des Geräts aus dem angegebenen Slot lesen und einstellen. ***Achtung***: Beim GSV-6 sind die gelesenen Einstellungen **temporär** und werden **nicht** automatisch als aktuelle Einstellung gespeichert. Zum Sichern der Einstellungen ist beim GSV-6 ein SaveAll mit Datensatz 0 notwendig!			
1 Byte	1 Parameter		
0	uint8_t (1)	Daten Satz-Nr.	Einzustellender Datensatz GSV-6: [0-1] GSV-8: [0-7] [0]: die aktuelle (GSV-8: zuletzt gespeicherte) Einstellung [1]: die Werkseinstellung [2-7]: Benutzer-Speicherplätze.
0 Byte	0 Rückgaben		

StoreConfig (0x0A)

0x0A	W	SaveAll	GSV-6 / GSV-8
Die Betriebsparameter des Geräts in den angegebenen Slot speichern.			
1 Byte	1 Parameter		
0	uint8_t (1)	Daten Satz-Nr.	Einzustellender Datensatz GSV-6: [0-1] GSV-8: [0-7] [0]: die aktuelle Einstellung [1]: die Werkseinstellung [2-7]: Benutzer-Speicherplätze. Die Datensätze sind nicht alle frei für den Benutzer einstellbar. So ist die Werkseinstellung [1] nur mit entsprechender Freigabe beschreibbar (GSV-8).
0 Byte	0 Rückgaben		

SetZero (0x0C)

0x0C	W	SetZero	GSV-6 / GSV-8
Tara auf den aktuellen Eingangswert setzten, so dass der Ausgabewert Null wird.			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle Null gesetzt.
0 Byte	0 Rückgaben		

GetAoutType (0x0D) [GSV-8]

0x0D	R	GetAoutType	GSV-8
Abfrage des Typs eines analogen Ausgangs			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Abzufragender Kanal GSV-8: [1-8]
2 Byte	2 Rückgaben		
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 Andere Werte sind reserviert.
1	uint8_t (1)	Kanal-Modus	<7:2> Reserviert <1> Ausgangskanal inaktiv==1 aktiv ==0 <0> Direktmodus aktiv - ==0: Folgt Messwert - ==1: Direkt Modus (siehe WriteAoutDirect (0x08))

SetAoutType (0x0E) [GSV-8]

0x0E	W	SetAoutType	GSV-8
Setzen des Typs und Modus eines analogen Ausgangs			
3 Byte	3 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem gleichen Werten beschrieben.
1	uint8_t (1)	Type-Enum	siehe GetAoutType (0x0D)
2	uint8_t (1)	Kanal-Modus	siehe GetAoutType (0x0D)
0 Byte	0 Rückgaben		

GetUnitNo (0x0F)

0x0F	R	GetUnitNo	GSV-6 / GSV-8
Abfrage der hinterlegten physikalischen Einheit			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Abzufragenden Kanal GSV-6: [1-6] GSV-8: [1-8]
1 Byte	1 Rückgaben		

0x0F	R	GetUnitNo	GSV-6 / GSV-8
0	uint8_t (1)	PhysUnit-Enum	Enumerator-Wert der physikalischen Einheit (siehe Physikalische Einheiten (Codes) Physikalische Einheiten (Codes))

SetUnitNo (0x10)

0x10	W	SetUnitNo	GSV-6 / GSV-8
Setzen der Kanalspezifischen physikalischen Einheit			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem identischen Wert beschrieben.
0	uint8_t (1)	PhysUnit-Enum	Enumerator der physikalischen Einheit (siehe Physikalische Einheiten (Codes) Physikalische Einheiten (Codes))
0 Byte	0 Rückgaben		

GetUnitText (0x11)

0x11	R (Seriell)	GetUnitNo	GSV-6 / GSV-8
Abfrage der Freitext-Einheit Achtung: Bei diesem Befehl gibt es eine Unterscheidung, die Schnittstellen-Spezifisch ist.			
1 Byte	1 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
9 Byte	2 Rückgaben		
0	uint8_t (1)	Indikator	Bei Seriell immer ==0
1	char[8]	Einheiten-Text	Der hinterlegte Freitext (NULL-terminiert oder genau 8 Zeichen lang)

0x11	R (CAN)	GetUnitNo	GSV-6 / GSV-8
Abfrage der Freitext-Einheit ***Achtung***: Es werden auf eine Anfrage **zwei** Antwort-Paket versendet. Und bei diesem Befehl gibt es eine Unterscheidung, die Schnittstellen-Spezifisch ist.			
1 Byte	1 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
5 Byte	2 Rückgaben		
0	uint8_t (1)	Indikator	==1 ‚Erste Hälfte' ==2 ‚Zweite Hälfte'
1	char[4]	Einheiten-Text	Der hinterlegte Freitext.

SetUnitText (0x12)

0x12	W (Seriell)	SetUnitNo	GSV-6 / GSV-8
Setzen einer Freitext-Einheit (maximal 8 Zeichen) Ähnlich wie bei der Abfrage ist auch beim Setzen ein Unterschied.			
10 Byte	3 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
1	uint8_t (1)	Indikator	Bei Seriell immer ==0
2	char[8]	Einheiten-Text	Der zu setzende Freitext (NULL-terminiert oder genau 8 Zeichen lang).
0 Byte	0 Rückgaben		

0x12	W (CAN)	SetUnitNo	GSV-6 / GSV-8
Setzen einer Freitext-Einheit (maximal 8 Zeichen) Es **muss** immer zuerst ein Befehl mit der Ersten Hälfte und dann mit der zweiten Hälfte versendet werden, damit der Text gespeichert wird! Wird nur die erste Hälfte versendet verweilt diese in einem temporären Speicher. Wird nur die zweite Hälfte versendet gibt es eine Fehlermeldung. Ähnlich wie bei der Abfrage ist auch beim Setzen ein Unterschied.			
6 Byte	3 Parameter		
0	uint8_t (1)	Slot	Text-Slot [0-1]
1	uint8_t (1)	Indikator	==1 ‚Erste Hälfte' ==2 ‚Zweite Hälfte'
2	char[4]	Einheiten-Text	Der zu setzende Freitext.
0 Byte	0 Rückgaben		

ReadUserScale (0x14)

0x14	R	ReadUserScale	GSV-6 / GSV-8
Auslesen des vom Benutzer einstellbaren Skalierungsfaktors			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Abzufragender Kanal GSV-6: [1-6] GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	float (4)	UserScale Faktor	Eingestellter Skalierungsfaktor des Float-Messwertes an der Schnittstelle

WriteUserScale (0x15)

0x14	W	WriteUserScale	GSV-6 / GSV-8
Setzen des vom Benutzer einstellbaren Skalierungsfaktors **Hinweis**: Wenn die 6-Achsen-Berechnung aktiv ist, haben diese Werte keine Auswirkung!			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem identischen Wert beschrieben.

0x14	W	WriteUserScale	GSV-6 / GSV-8
1	float (4)	UserScale Faktor	Einzustellender Skalierungsfaktor des Float-Messwertes an der Schnittstelle
0 Byte	0 Rückgaben		

ReadCal (0x17)

0x17	R	MEreadCal	GSV-6
Auslesen des Kalibrierwerts. Der Kalibrierkanal ist die Kanal-Nummer zur „Basis 0“ und ein Offsetwert in die entsprechende Kalibrier-Datenreihe. ***Achtung***: Dieser Befehl erfordert eine MESYS-Freigabe!			
2 Byte	2 Parameter		
0	uint8_t (1)	Kalibrierkanal	0x00-0x05: Eingangsfaktor 0x08-0x0D: Eingangsoffset 0x10: Ausgangsfaktor 0x18: Ausgangsoffset 0x20-0x25: Scale-Werte (Faktor) 0x28-0x2D: Tara-Werte (Offset) Die anderen Werte sind reserviert!
1	uint8_t (1)	SubIndex	Eingangsfaktor/-offset: für Vorverstärkung: - 0x00: Verstärkung 1x - 0x01: Verstärkung 2x - 0x02: Verstärkung 4x - 0x03: Verstärkung 8x Ausgangsfaktor/-offset für Ausgangstyp: - 0x00: 0V – 10V - 0x01: -10V – 10V - 0x02: 0V – 5V - 0x03: -5V – 5V - 0x04: 4mA – 20mA - 0x05: reserviert - 0x06: 0mA – 20 mA - 0x07: reserviert Scale- / Tara-Werte - ungenutzt Die anderen Werte sind reserviert!
4 Byte	1 Rückgaben		
0	float (4) / int32_t (4)	Faktor Offset	

0x17	R	MEreadCal	GSV-8
Auslesen des Hersteller-Kalibrierwerts.			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal	[1-8] Eingangs bzw. Ausgangskanal

0x17	R	MEreadCal	GSV-8
1	uint8_t (1)	SubIndex	- 0x00: Scalecal f. Brückeneingang Us= 8,75V - 0x01: Scalecal f. Brückeneingang Us= 5V - 0x02: Scalecal f. Brückeneingang Us= 2,5V - 0x03: ScaleCal f. SingleEnded Eingang +-10V - 0x04: ScaleCal f. PT1000-Eingang - 0x10: ScaleCal f. Analogausgang 10V - 0x11: OffsetCal f. Analogausgang 10V - 0x12: ScaleCal f. Analogausgang 5V - 0x13: OffsetCal f. Analogausgang 5V - 0x14: ScaleCal f. Analogausgang 4..20mA - 0x15: OffsetCal f. Analogausgang 4..20mA - 0x20: OffsetCal f. Brückeneingang Us= 8,75V - 0x21: OffsetCal f. Brückeneingang Us= 5V - 0x22: OffsetCal f. Brückeneingang Us= 2,5V - 0x23: OffsetCal f. SingleEnded Eingang +-10V - 0x24: OffsetCal f. PT1000-Eingang Andere Werte sind reserviert!
4 Byte	1 Rückgaben		
0	float (4)	Faktor / Offset	

MEwriteCal (0x18)

0x18	W	MEwriteCal	GSV-6 (Reset)
Setzen eines Kalibrierwerts. Der Kalibrierkanal ist die Kanal-Nummer zur „Basis 0“ und ein Offsetwert in die entsprechende Kalibrierdatenreihe. ***Achtung***: Dieser Befehl erfordert eine MESYS-Freigabe!			
6 Byte	3 Parameter		
0	uint8_t (1)	Kalibrierkanal	Siehe ReadCal (0x17)
1	uint8_t (1)	SubIndex	Siehe ReadCal (0x17)
2	float (4) / int32_t (4)	Faktor Offset	
0 Byte	0 Rückgaben		

0x18	W	MEwriteCal	GSV-8
Mit Kanal 0 werden ALLE Kanäle mit dem identischen Wert beschrieben. ***Achtung***: Dieser Befehl erfordert eine MESYS-Freigabe!			
6 Byte	3 Parameter		
0	uint8_t (1)	Kalibrierkanal	Zu setzender Eingangs bzw. Ausgangskanal [0-8]
1	uint8_t (1)	SubIndex	Siehe ReadCal (0x17)
2	float (4)	Faktor / Offset	
0 Byte	0 Rückgaben		

MEsetID (0x19)

0x19	W	MEsetID	GSV-6 / GSV-8
Setzen eines Freischalt-Codes / Setzen des ID-Levels Nur die aufgeführten Codes werden ohne Fehler akzeptiert. Wird ein fehlerhafter Code übergeben wird nach drei Fehlversuchen immer ein Fehler zurückgegeben. Die Freigabe ist bis zu einem Neustart aktiv.			
4 Byte	1 Parameter		
0	uint32_t (4)	Freischalt-Code	ID_EXT_MESYS ID_EXT_USER_CONF ID_EXT_USER_CONF_ALL (reserviert)
0 Byte	0 Rückgaben		

Die Freischaltcodes sind wie folgt definiert:

MEgetIDstate (0x1A)

0x1A	R	MEgetIDstate	GSV-6 / GSV-8
Abfragen des ID-States			
0 Byte	0 Parameter		
1 Byte	1 Rückgaben		
0	uint8_t (1)	Freischalt-Zustand	0x00: Normalbetrieb 0x08: Privilegierte Befehle nutzbar 0x10: ME-Befehle nutzbar

GetSerNo (0x1F)

0x1F	R	GetSerNo	GSV-6 / GSV-8
Abfrage der Seriennummer			
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		
0	uint32_t (4)	Seriennummer	[1-99999999]. GSV-6 liefert 0xFFFFFFFF wenn noch keine Seriennummer gesetzt ist, GSV-8 dann d01234567

MEsetSerNo (0x20)

0x20	W	MEsetSerNo	GSV-6 / GSV-8
Setzen der Seriennummer ***Achtung***: Dieser Befehl erfordert eine MESYS-Freigabe!			
4 Byte	1 Parameter		
0	uint32_t (4)	Seriennummer	[1-99999999]
0 Byte	0 Rückgaben		

StopTransmission (0x23)

0x23	W	StopTrasmission	GSV-6 / GSV-8
Permanente, selbstständige Messdatenübertragung für diese Schnittstelle stoppen. Dieser Zustand ist flüchtig, d.h. er bleibt nach dem Neustart nicht erhalten. Zum dauerhaften Stoppen der permanenten Messdatenübertragung bitte den Befehl SetTXmode verwenden.			

0x23	W	StopTrasmission	GSV-6 / GSV-8
0 Byte	0 Parameter		
0 Byte	0 Rückgaben		

StartTransmission (0x24)

0x24	W	StartTrasmission	GSV-6 / GSV-8
Permanente, selbstständige Messdatenübertragung für diese Schnittstelle starten. Dieser Zustand ist flüchtig, d.h. er bleibt nach dem Neustart nicht erhalten. Zum dauerhaften Stoppen der permanenten Messdatenübertragung bitte den Befehl SetTXmode verwenden.			
0 Byte	0 Parameter		
0 Byte	0 Rückgaben		

ClearBufferAbortTX (0x25) [GSV-8]

0x25	W	ClearBufferAbortTX	GSV-8 (USB)
Übertragungspuffer löschen (USB). Es werden im USB-Sendequeue der Messanwendung befindliche Messdatenframess aus diesem entfernt. Eine "pending transmission" auf USB-Ebene wird jedoch nicht abgebrochen, so dasss Messdatenframes, die zu diesem Zeitpunkt gerade gesendet werden, erhalten bleiben (es entsteht kein "Datenmüll").			
0 Byte	0 Parameter		
0 Byte	0 Rückgaben		

GetMode (0x26)

0x26	R	GetMode	GSV-6
Mode-Flags auslesen. Beim GSV-6 handelt es sich hierbei um die System-Flags.			
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		
0	uint32_t (4)	Mode-Flags	GSV-6: <2:0> Anzahl der aktiven Kanäle. 0b001: 1 Kanal 0b010: 2 Kanäle 0b011: 3 Kanäle 0b110: 6 Kanäle <3> Save-Tara <4> User-Monitor <6> Scale-Negate <7> Spitzenwert-Ausgabe <8> Spitzenwert-Reset und Tara <9> TEDs Auslesen beim Boot <10> 6-Achsen Berechnung Alle anderen Bits sind reserviert.

0x26	R	GetMode	GSV-8
Mode-Flags auslesen. Beim GSV-8 sind es reine Mode-Flags die vom User beschrieben werden können.			

0x26	R	GetMode	GSV-8
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		
0	uint32_t (4)	Mode-Flags	GSV-8: <0> 6-Achsen Berechnung aktiv <1> Analogfilter automatisch setzen <2> Maximalwert berechnen <3> Rauchunterdrückung aktiv <7> Alle Schreibfunktionen sperren <15:8>: Wenn am Eingangskanal (BitNo-7) Sensoren mit TEDS angeschlossen sind, sollen deren Daten zur Verwendung des User-Scale-Wertes verwendet werden (wenn möglich) <16>: Wenn entsprechendes Bit in <15:8> gesetzt, soll auch die Einheit aus den TEDS-Daten gesetzt werden (wenn möglich) <17>: Wenn entsprechendes Bit in <15:8> gesetzt, soll auch die Eingangs-empfindlichkeit anhand der TEDS-Daten gesetzt werden (wenn möglich)

SetMode (0x27)

0x27	W	SetMode	GSV-6 (Reset)
Mode-Flags setzen			
4 Byte	1 Parameter		
0	uint32_t (4)	Mode-Flags	GSV-6: <10> 6-Achsen Berechnung Alle anderen Bits werden ignoriert.
0 Byte	0 Rückgaben		

0x27	W	SetMode	GSV-8
Mode-Flags setzen			
4 Byte	1 Parameter		
0	uint32_t (4)	Mode-Flags	siehe GetMode (0x26) Bit 7 ist nicht veränderbar (read-only). Alle anderen Bits werden ignoriert.
0 Byte	0 Rückgaben		

GetSoftwareConfiguration (0x2A) [GSV-8]

0x2A	R	GetSoftwareConfiguration	GSV-8
Dieser Wert zeigt das Vorhandensein hardwareabhängiger Features an, die mit dementsprechenden Features der Gerätesoftware korrespondieren müssen. Deshalb wird dieser Wert (Hex-kodiert) auch in der Bootloader-Software angezeigt. Weicht er zwischen vorhandener Gerätesoftware und upzudatendem Neuprogramm ab, ist von einem Updatevorgang i.d.R. dringend abzuraten (es sei denn, er erfolgt im Zuge einer Hardwareaufrüstung).			
0 Byte	0 Parameter		
4 Byte	1 Rückgabe		s. Tabelle

Tabelle: Bedeutung BUILD-Val

Bit	Wert	Name	Bedeutung
0	1	HAS_ADC	AD-Umsetzer vorhanden
1	2	HAS_ETHERCAT	Ethercat-Feldbus vorhanden
2	4	HAS_LCD	LC-Display vorhanden
3	8	HAS_TEDS	TEDS-Sensor-Unterstützung
4	16	HAS_DIGI_IN_OUT	Digitale Ein- und Ausgänge vorhanden
5	32		reserviert
6	64	HAS_ANALOG_OUT	Analogausgänge vorhanden
7	128	HAS_SERIAL	Serielles Interface vorhanden (z.B. via Ethernet)
8	256	HAS_FREQ_OUT	Frequenzmodulierter Analogausgang vorhanden
9	512	HAS_AIN_MCU	Sensorüberwachung am Analogeingang
10	1024	HAS_SIXAXIS	Sechsachsensensoren werden unterstützt
11	2048	HAS_CANOPEN	CANopen feldbus vorhanden

FirmwareVersion (0x2B)

0x2B	R	FirmwareVersion	GSV-6 / GSV-8
Firmware Version auslesen			
0 Byte	0 Parameter		
4 Byte	2 Rückgaben		
0	uint16_t (2)	Hauptversion	VER_MAJOR
1	uint16_t (2)	Unterversion	VER_MINOR

MEwriteInputRange (0x34)

0x34	W	MEwriteInputRange	GSV-6 (Reset)
Eingangsempfindlichkeit setzen. ***Achtung***: GSV-6 kann die Kanäle nicht unabhängig voneinander setzen! ***Achtung***: Dieser Befehl erfordert eine MESYS-Freigabe!			
6 Byte	3 Parameter		
0	uint8_t (1)	Kanal	[0] Mit Kanal 0 werden ALLE Kanäle mit dem gleichen Wert beschrieben.
1	uint8_t (1)	SensIndex	ungenutzt (Null übergeben)

0x34	W	MEwriteInputRange	GSV-6 (Reset)
2	uint32_t (4)	Empfindlichkeit	HW-Empfindlichkeit in mV*100 Mögliche Werte sind: 800: 8mV/V 400: 4mV/V 200: 2mV/V 100: 1mV/V
0 Byte	0 Rückgaben		

0x34	W	MEwriteInputRange	GSV-8
Herstellereinstellung: Kommunizierten Wert für die Eingangsempfindlichkeit (=Messbereich) setzen. ***Achtung***: Dieser Befehl erfordert eine MESYS-Freigabe!			
6 Byte	3 Parameter		
0	uint8_t (1)	Kanal	[0-8] Mit Kanal 0 werden ALLE Kanäle mit dem gleichen Wert beschrieben.
1	uint8_t (1)	SensIndex	[1-4] Eingangstyp: - 0x01: Range f. Brückeneingang Us= 8,75V - 0x02: Range f. Brückeneingang Us= 5V - 0x03: Range f. Brückeneingang Us= 2,5V - 0x04: Range f. SingleEnded Eingang +-10V
2	uint32_t	Empfindlichkeit	HW-Empfindlichkeit in mV*100 Standardversion: 2mV/V = 200, 3,5 ->350 7 -> 700. +-10V -> 1000000
0 Byte	0 Rückgaben		

SetInjectValOrOffset (0x35) [GSV-8]

0x35	W	Set Inject Val or Offset	GSV-8
Eingangswerte-Simulation einstellen oder Nullsignaloffsetwerte laden			
1 Byte	1 Parameter		

0x35	W	Set Inject Val or Offset	GSV-8
0	uint8_t (1)	Index	**0:** Zurücksetzen aller Inject Values oder Offsets zum Normalbetrieb **1:** Der halbe Nennaussteuerungswert (Rohwert: 0x003CF3CF) wird bei allen Kanälen als Pseudo-Messwert gesetzt, zugleich wird die Messanwendung vom AD-Umsetzer abgekoppelt. **2:** Die nach dem letzten Nullsetzen ermittelten Offsetwerte werden nicht verwendet; stattdessen werden die für exakt 0mV/V (Brückeneingang) bzw. 0V geltenden Offset-Hersteller-Kalibrierwerte des Messverstärkers geladen. Bei angeschlossenem Sensor unter Nulllast kann so dessen Grundverstimmung abgelesen werden (der AD-Umsetzer ist also weiterhin aktiv). **3:** Ähnlich wie 2., es werden allerdings die Offset-Kalibrierwerte des Sechsachsensensors geladen (sofern Sechsachsenkalibrierdaten vorhanden). Somit kann unter Nulllast des Sechsachsensensors die Abweichung seiner Grundverstimmung von der des Zeitpunktes seiner Kalibrierung - und somit ein Indikator für dessen Abnutzung - abgelesen werden.
0 Byte	0 Rückgaben		

GetHardwareVersion (0x36)

0x36	R	GetHardwareVersion	GSV-6 / GSV-8
Hardware Option auslesen (GSV-6: Liefert hier z.Z. immer 0!)			
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		
0	uint32_t (4)	Optionen (GSV-8)	Bits<31:16>: Hardware-Versionierung des Hauptboards. "Versioniert" werden Hardware-Releases, die einer angepassten, bzw. geänderten Geräte-Software bedürfen. Bits<15:0>: Unteres Byte der Hardwareversion. Vorgesehen f. Geräte-Software relevante Versionierung kommender Tochterboardversionen.

GetRawValue (0x3A)

0x3A	R	GetRawValue	GSV-6
Rohwert eines Eingangs einlesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	[1-6]
4 Byte	1 Rückgaben		
0	int32_t (4)	Rohwert	Rohwert

0x3A	R	GetRawValue	GSV-8
Rohwert eines Eingangs einlesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	[1-8]
3 Byte	1 Rückgaben		
0	int32_t (4)	Rohwert	Rohwert

GetValue (0x3B)

0x3B	R	GetValue	GSV-6 / GSV-8
Absenden eines/der Messwert-Frames anstoßen! ***Achtung***: Dieser Befehl liefert **keinen** Antwort-Frame! Daher auch keinen Fehler-Code!			
0 Byte	0 Parameter		
0 Byte	0 Rückgaben		

ClearMaxValue (0x3C)

0x3C	-	ClearMaxValue	GSV-6 / GSV-8
Extremwertspeicher zurücksetzen, d.h. Maximal- und Minimalwert(e)			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Zu setzender Kanal GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle zurückgesetzt.
0 Byte	0 Rückgaben		

GetLastProtokollError (0x42)

0x42	R	GetLastError	GSV-6
GSV-6: Letzten Protokollfehler auslesen. Bei einem (synchronen) Protokollfehler handelt es sich um den in einem Antwort-Frame versendeten Fehler-Code. Bei einem asynchronen Protokollfehler handelt es sich um den Fehler-Code, der bei der autonomen Messwert-Frame Versendung aufgetreten ist. GSV-8: Bei einem asynchronen Protokollfehler handelt es sich entweder um einen Fehler, der beim Empfang des Anfrageframes aufgetreten ist und der das Ignorieren dieses Anfrageframes zur Folge hatte, da fundamentale Protokollanforderungen verletzt wurden. Ein so stark fehlerhafter Anfrageframe wird also (ausnahmsweise) nicht bestätigt. Als Fehlercode tritt dann ERR_FRAME_ERROR auf. Oder es handelt sich um einen Messwertverlust im Sendepuffer des USB-Sendequeues, wodurch ältere, noch nicht gesendete Messwertframes durch neuere überschrieben wurden. Dies kann auftreten, wenn bei hoher Datenrate (>= 16000/s) die USB-Kommunikation zu stark ausgelastet ist, z.B.: durch häufige Befehlsanfragen; oder auch bei Verwendung mehrerer USB-Geräte an demselben USB-Hub, an dem der GSV-8 angeschlossen ist. In diesem Fall tritt als Fehlercode ERR_RET_TXBUF auf.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: (Synchroner) Protokollfehler 1: Asynchronen Protokollfehler Alle anderen Werte sind reserviert
4 Byte	1 Rückgaben		

0x42	R	GetLastError	GSV-6
0	uint32_t	Fehler-Code	siehe Tabelle <link>

GetLastValueError (0x43)

0x43	R	GetLastValueError	GSV-6
Letzten Messwertfehler auslesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: Fehlerbits Andere Werte ohne Funktion
6 Byte	1 Rückgaben		
0	uint8_t[6]	Fehler	Index 0: - Fehler [0] Fehlerbits (Maximalwertüberschreitung 6-Achsen) - Fehler [1] Fehlerbits (Eingangswertsättigung) - Fehler [2..5] =0

0x43	R	GetLastValueError	GSV-8
Letzte(n) Messwertfehler auslesen. Siehe Tabelle oben			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: Fehlerzähler 1-40: Fehler-Struktur
6 Byte	1 Rückgaben		

0x43	R	GetLastValueError	GSV-8
0	uint8_t [6] ErrorStruct	Fehler	Index 0: Anzahl der seit dem letzten Einschalten aufgetretenen Fehler und Anzahl der nichtflüchtig gespeicherten Fehler auslesen. Antwortparameter: Von den 6 gesendeten Bytes sind nur die beiden zuerst gesendeten relevant, d.h. die beiden "höherwertigen" bei pseudo-numerischer Interpretation; die letzten bzw. unteren 4 Bytes sind konstant 0. Das zuallererst gesendete ist die Anzahl der seit dem letzten Einschalten aufgetretenen Fehler. Das darauffolgende ist die Anzahl der nichtflüchtig im EEPROM gespeicherten Fehler. Index 1: ErrorStruct seit Power On: Fehler, der zuletzt seit dem letzten Einschalten aufgetreten ist und noch nicht nichtflüchtig gespeichert wurde. Indizes 2 bis 83: ErrorStruct des nichtflüchtig im EEPROM gespeicherten Fehlers. Höhere Indizes gehören zu neueren Fehlern, niedrigere zu älteren. Definition des ErrorStruct in C-Notation: { unsigned short ErrFlags:16; unsigned long ErrTime:29; unsigned ErrType:3; } ErrorStruct;

EraseErrorMemory (0x44)

0x44	W	EraseErrorMemory	GSV-8
Löschen des nichtflüchtigen Fehlerspeichers. Zur erfolgreichen Ausführung muss der USER_ID_LEVEL gesetzt sein.			
0 Byte	0 Parameter		
0 Byte	0 Rückgaben		

GetSensorPlugged (0x45)

0x45	R	GetSensorPlugged	GSV-8
Ermitteln, ob am Analogeingang ein Sensor angeschlossen ist (nicht nutzbar bei SingleEnded Input).			
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		

0x45	R	GetSensorPlugged	GSV-8
0	uint_32	Flagwert	Bit 0: Kanal 1 … Bit 7: Kanal 8 Bit=1: Brückensensor angeschlossen, Bit=0: Kein Brückensensor angeschlossen Bits<15:8>: TEDS-Baustein angeschlossen[1] Kanal 8:1. Bits<23:16>: TEDS schreibbar Kanal 8:1. Bits<31:24>: Reserviert f. weitere Sensoreigenschaften

ReadFTSensorCal (0x47)

0x47	R	ReadFTSensorCal	GSV-6 / GSV-8
Ein Kalibrierwert des **F**orce/**T**orque-Sechsachsensensors wie z.B. Kalibriermatrixelement, Seriennummer, Normierungsfaktor der Kalibriermatrix, Sensormaximalwert und Eingangsempfindlichkeit wird zurückgegeben. **Hinweise**: 1. Es werden mehrere Sechsachsenkalibrierarrays unterstützt (beim GSV-8 bis zu 5). Um Werte(e) aus einem bestimmten Array auszulesen, sollte dieses vorher mit PrepReadFTsensor ausgewählt worden sein. Die 2-dimensionale Kalibriermatrix wird zeilenweise im Array hinterlegt (a11, a12, a13, … a21, …), so dass index={0..5} den Zeilenvektor a11..a15 adressiert, index={6..11} den Zeilenvektor a21..a25 usw.			
2 Byte	2 Parameter		
0	uint8_t (1)	type	0: Seriennummer (index muss =0 sein) 1: Normierungsfaktor (index muss =0 sein) 2: Matrixwerte (index: [0-35]) 3: Geom. Offsets x,y,z (index: [0-2]) 4: Maximalwerte (index: [0-5]) 5: Eingangsempfindlichkeit (index muss =0 sein) 6: Grundverstimmungswerte bei Nulllast (GSV-8)
1	uint8_t (1)	index	Index des durch *type* adressierten Arrays
4 Byte	1 Rückgaben		
0	uint32_t (4) / float (4)	Seriennummer / Einträge	Bis auf die Seriennummer sind alle Einträge float-Werte.

WriteFTSensorCal (0x48)

0x48	W	WriteFTSensorCal	GSV-6 / GSV-8
Ein Kalibrierwert des **F**orce/**T**orque-Sechsachsensensors wie z.B. Kalibriermatrixelement, Seriennummer, Normierungsfaktor der Kalibriermatrix, Sensormaximalwert und Eingangsempfindlichkeit wird geschrieben. **Hinweis**: Die 2-dimensionale Kalibriermatrix wird zeilenweise im Array hinterlegt (a11, a12, a13, … a21, …), so dass index={0..5} den Zeilenvektor a11..a15 adressiert, index={6..11} den Zeilenvektor a21..a25 usw. ***Achtung***: Zum persistenten Speichern der übertragenen Werte muss die Funktion StoreSensorCal (0x7F) aufgerufen werden, nachdem alle in sich konsistenten Werte geschrieben wurden. GSV-8: Voraussetzung ist ein gesetzter USER_ID_LEVEL			

1 Unabhängig davon, ob dieser gültige Daten enthält. Letzteres kann mit Befehl Get TEDS active ermittelt werden (sofern in der Mode-variablen<15:8> das entsprechnede Bit gesetzt ist.

0x48	W	WriteFTSensorCal	GSV-6 / GSV-8
6 Byte	3 Parameter		
0	uint8_t (1)	type	0: Seriennummer (index muss =0 sein) 1: Normierungsfaktor (index muss =0 sein) 2: Matrixwerte (index: [0-35]) 3: Geom. Offsets x,y,z (index: [0-2]) 4: Maximalwerte (index: [0-5]) 5: Eingangsempfindlichkeit (index muss =0 sein) 6: Grundverstimmungswerte bei Nulllast (index: [0-5], nur GSV-8)
1	uint8_t (1)	index	Index des durch *type* adressierten Arrays
2	uint32_t (4) / float (4)	Seriennummer / Einträge	Bis auf die Seriennummer sind alle Einträge float-Werte.
0 Byte	0 Rückgaben		

GetTXmapping (0x49)

0x49	R	GetTXMapping	GSV-8
Zuordnung und physische Bedeutung der Werte im Messwertframe auslesen. Bemerkung: Die Indizes 1..[Anzahl gemappter Objekte] sind noch nicht implementiert. An Index 0 wird zur Zeit (FW Ver. 1.10) konstant 0 zurückgegeben.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	Nummer des Mappingeintrags [1..<Anzahl gemappter Objekte>] oder =0: Anzahl der gemappten Objekte bestimmen.
2 Byte	1 Rückgaben		

0x49	R	GetTXMapping	GSV-8
0	uint16_t (2)	Anzahl / Mappingeintrag	Wenn =0: Kein Mapping definiert. >0: Wenn Index=0: Anzahl gemappter Objekte. Sonst: Mappingeintrag wie folgt: Bits<15:8>, d.h. erstes Datenbyte der Antwort: Physikalischer Typ. z.Zt. vordefiniert (reserviert): 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: Temparature 0x28: Quadrature-Encoder-Pulses 0x38: Quadrature-Encoder-Speed Bits<7:4>: Aktueller Wert / Maximal- / Minimalwert: 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>: Kanalnummer des Eingangs: 1..8: Analoger Sensoreingang 9: Quadraturenkoder-Pulseingang Nr. 1 10: Quadraturenkoder-Pulseingang Nr. 2

SetTXmapping (0x4A)

0x4A	W	SetTXMapping	GSV-8 (reserviert)
Messwertframe-Zuordnung setzen			
3 Byte	2 Parameter		
0	uint8_t (1)	Index	Index =0: Mit dem Wert in *Anzahl / Mappingeintrag* wird die Anzahl gemappter Objekte im Messdatenframe festgelegt. Index >0 [1-16]: Der Mappingeintrag wird gesetzt.
1	uint16_t (2)	Anzahl / Mappingeintrag	Anzahl gemappter Objekte im Messdatenframe bzw. Mappingeintrag, wie in GetTXmapping definiert.
0 Byte	0 Rückgaben		

GetDigiFiltType (0x4B)

0x4B	R	GetDigiFiltType	GSV-6 / GSV-8
Typ des Digitalfilters ermitteln			
2 Byte	2 Parameter		

0x4B	R	GetDigiFiltType	GSV-6 / GSV-8
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6] GSV-8: [1-8]
1	uint8_t (1)	subTyp	GSV-6: Ungenutzt GSV-8: 0x00: IIR-Filter 0x80 FIR-Filter (nur GSV-8)
1 Byte	1 Rückgaben		
0	uint8_t (1)	Filtertyp-Enum	<7>: FIR-Filter-Indikator: =1 FIR Filter, =0 IIR <6:4> Filtertyp: -0: Tiefpass -1: Hochpass -2: Bandpass -3: Bandstop -4: Kammfilter (GSV-8) <3:0> Filterlänge i.a. für IIR konstant =4 und für FIR einer der Werte: 4,6,8,10,12,14

SetDigiFiltType (0x4C)

0x4C	W	SetDigiFiltType	GSV-6 / GSV-8
Typ des Digitalfilters setzen ***Achtung***: Zum persistenten Speichern der übertragenen Filter-Werte muss die Funktion StoreDigiFilt (0x7E) aufgerufen werden! Dies sollte nach der Übertragung aller Filterwerte (Koeffizienten, Typ, Grenzen) für alle zu setzenden Filter einmalig erfolgen.			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	uint8_t (1)	Filtertyp-Enum	<7>: FIR-Filter-Indikator: =1 FIR Filter, =0 IIR <6:4> Filtertyp: -0: Tiefpass -1: Hochpass -2: Bandpass -3: Bandstop -4: Kammfilter (GSV-8) <3:0> Filterlänge i.a. für IIR konstant =4 und für FIR einer der Werte: 4,6,8,10,12,14
0 Byte	0 Rückgaben		

ReadDigiFiltCutOff (0x4D)

0x4D	R	ReadDigiFiltCutOff	GSV-6 / GSV-8
Digitalfilter-Grenzfrequenz auslesen. Hiermit werden die informativen Werte, die mittels WriteDigiFiltCutOff (0x4E) gesetzt wurden, zurückgegeben.			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6] GSV-8: [1-8]
1	uint8_t (1)	Index	0: Fgu (untere Grenzfrequenz) 1: Fgo (obere Grenzfrequenz)
4 Byte	1 Rückgaben		
0	float (4)	Grenzfrequenz	

WriteDigiFiltCutOff (0x4E)

0x4E	W	WriteDigiFiltCutOff	GSV-6 / GSV-8
Einen der beiden Digitalfilter-Grenzfrequenzen (-3dB) setzen. Hierbei handelt es sich um einen rein informativen Wert, der keine Auswirkung auf das Digitalfilter besitzt. Bei Tief- und Hochpass gilt nur Fgu. ***Achtung***: Zum persistenten Speichern der übertragenen Filter-Werte die Funktion StoreDigiFilt (0x7E) aufgerufen werden! Dies sollte nach der Übertragung aller Filterwerte (Koeffizienten, Type, Grenzen) für alle zu setzenden Digitalfilter einmalig erfolgen.			
6 Byte	3 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem identischen Wert beschrieben.
1	uint8_t (1)	Index	0: Fgu (untere Grenzfrequenz) 1: Fgo (obere Grenzfrequenz)
2	float (4)	Grenzfrequenz	
0 Byte	0 Rückgaben		

ReadDigiFiltCoeff (0x4F)

0x4F	R	ReadDigiFiltCoeff	GSV-6
Digitalfilter-Koeffizienten auslesen			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer [1-6]
1	uint8_t (1)	Index	Bei IIR: 0x00...0x04: Koeffizienten A (Eingang) 0x10...0x13: Koeffizienten B (Ausgang, Rückkopplung)
4 Byte	1 Rückgaben		
0	float (4)	Koeffizient	

0x4F	R	ReadDigiFiltCoeff	GSV-8
Filter-Koeffizienten auslesen			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer [1-8]
1	uint8_t (1)	Index	Bei IIR: -0x00...0x04: Koeffizienten A (Eingang) -0x10...0x13: Koeffizienten B (Ausgang, Rückkopplung) Bei FIR: -0x80...0x87: Koeffizienten (nur bis einschließlich Mitte der symmetrischen Koeffizientenfolge)
4 Byte	1 Rückgaben		
0	S7.24 (4)	Koeffizient	

WriteDigiFiltCoeff (0x50)

0x50	W	WriteDigiFiltCoeff	GSV-6
Filter- Koeffizienten setzen ***Achtung***: Beim GSV-6 muss zum persistenten Speichern der übertragenen Filter-Werte die Funktion StoreDigiFilt (0x7E) aufgerufen werden! Dies sollte nach der Übertragung aller Filterwerte (Koeffizienten, Type, Grenzen) für alle zu setzenden Filter einmalig erfolgen.			
6 Byte	3 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer [0-6] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	uint8_t (1)	Index	Bei IIR: - 0x00...0x04: Koeffizienten A (Eingang) - 0x10...0x13: Koeffizienten B (Ausgang, Rückkopplung)
2	float (4)	Koeffizient	
0 Byte	0 Rückgaben		

0x50	W	WriteDigiFiltCutOff	GSV-8
Filter- Koeffizienten setzen ***Achtung***: Zum persistenten Speichern der übertragenen Filter-Werte die Funktion StoreDigiFilt (0x7E) [GSV-6] aufgerufen werden! Dies sollte nach der Übertragung aller Filterwerte (Koeffizienten, Type, Grenzen) für alle zu setzenden Filter einmalig erfolgen.			
6 Byte	3 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.

0x50	W	WriteDigiFiltCutOff	GSV-8
1	uint8_t (1)	Index	Bei IIR: -0x00...0x04: Koeffizienten A (Eingang) -0x10...0x13: Koeffizienten B (Ausgang, Rückkopplung) Bei FIR: -0x80...0x87: Koeffizienten (nur bis einschließlich Mitte der symmetrischen Koeffizientenfolge)
2	S7.24 (4)	Koeffizient	
0 Byte	0 Rückgaben		

GetDfiltOnOff (0x51)

0x51	R	GetDfiltOnOff	GSV-6
Abfrage der aktiven Filter (Bit Maske) Gesetzte Bits entsprechen einem aktiven Filter.			
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		
0	uint32_t (4)	Kanal Flags	GSV-6: <5:0> Filter auf dem entsprechenden Kanal aktiv <31:6> reserviert

0x51	R	GetDfiltOnOff	GSV-8
Abfrage der aktiven Filter (Bit Maske) Gesetzte Bits entsprechen einem aktiven Filter.			
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		
0	uint32_t (4)	Kanal Flags	GSV-8: <7:0> IIR-Filter auf dem entsprechenden Kanal (Bit0: K1)aktiv <15:8> FIR-Filter auf dem entsprechenden Kanal (Bit8: K1)aktiv <31:16> reserviert

SetDfiltOnOff (0x52)

0x52	W	SetDfiltOnOff	GSV-6
Setzen der aktiven Filter (Bit Maske) Gesetzte Bits entsprechen einem aktiven Filter.			
4 Byte	1 Parameter		
0	uint32_t (4)	Kanal Flags	GSV-6: - <5:0> Filter auf dem entsprechenden Kanal aktiv - <31:6> reserviert
0 Byte	0 Rückgaben		

0x52	W	SetDfiltOnOff	GSV-8
Setzen der aktiven Filter (Bit Maske) Gesetzte Bits entsprechen einem aktiven Filter.			
4 Byte	1 Parameter		
0	uint32_t (4)	Kanal Flags	GSV-8: - <7:0> IIR-Filter auf dem entsprechenden Kanal (Bit0: K1) aktivieren - <15:8> FIR-Filter auf dem entsprechenden Kanal (Bit8: K1) aktivieren - <31:16> reserviert
0 Byte	0 Rückgaben		

ReadMaxMinVal (0x53)

0x53	R (Seriell)	ReadMaxMinVal	GSV-6
Maximal- und Minimalwert eines Kanals auslesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6]
9 Byte	3 Rückgaben		
0	uint8_t (1)	Indikator	==0
1	float (4)	Maximalwert	
5	float (4)	Minimalwert	

ReadMaxMinVal (0x53)

0x53	R (Seriell)	ReadMaxMinVal	GSV-8
Maximal- und Minimalwert eines Kanals auslesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-8: [1-8]
8 Byte	2 Rückgaben		
0	float (4)	Maximalwert	
1	float (4)	Minimalwert	

0x53	R (CAN)	ReadMaxMinVal	GSV-6
Maximal- und Minimalwert eines Kanals auslesen **Achtung**: Es werden zwei Antworten versendet! Erst der Maximalwert, dann der Minimalwert.			
1 Byte	1 Parameter		

0x53	R (CAN)	ReadMaxMinVal	GSV-6
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6] GSV-8: [1-8]
5 Byte	2 Rückgaben		
0	uint8_t (1)	Indikator	==1: Maximalwert ==2: Minimalwert
1	float (4)	Extremwert	

GetFTSensorCalArrNo (0x54)

0x54	R	GetFTSensorCalArrNo	GSV-6 / GSV-8
Abfrage der aktiven 6-Achsen-Matrix			
0 Byte	0 Parameter		
3 Byte	3 Rückgaben		
0	uint8_t (1)	Maximum	Maximale Anzahl der Matrizen, die im Gerät zur Verfügung stehen.
1	uint8_t (1)	NumAktiv	Anzahl der gespeicherten Matrizen
2	uint8_t (1)	Matrix	Nummer der aktiven Matrix

SetFTSensorCalArrNo (0x55)

0x55	W	SetFTSensorCalArrNo	GSV-6 (Reset) / GSV-8
Setzen der aktiven 6-Achsen-Matrix			
1 Byte	1 Parameter		
0	uint8_t (1)	Matrix-Nummer	
0 Byte	0 Rückgaben		

ReadDeviceHours (0x56)

0x56	R	ReadDeviceHours	GSV-8
Einen der beiden Betriebsstundenzähler kann hiermit ausgelesen werden. Der „absolute“ Betriebsstundenzähler ist monoton und kann nicht zurückgesetzt werden. Der „einstellbare“ Betriebsstundenzähler ist monoton, solange er nicht über das Kommando WriteDeviceHours (0x57) auf einen neuen Wert gesetzt wird.			
1 Byte	1 Parameter		
0	uint8_t (1)	Zähler-Index	0: absolut 1: setzbar Andere Werte sind reserviert.
4 Byte	1 Rückgaben		
0	float (4)	Zähler	Betriebsstunden des Geräts.

WriteDeviceHours (0x57)

0x57	W	WriteDeviceHours	GSV-8
Den veränderlichen Betriebsstundenzähler setzen Voraussetzung ist ein gesetzter USER_ID_LEVEL			
1 Byte	1 Parameter		
0	float (4)	Wert	Wert für den veränderlichen Betriebsstundenzähler in Stunden.

0x57	W	WriteDeviceHours	GSV-8
0 Byte	0 Rückgaben		

SetPassword (0x58)

0x58	W	SetPassword	GSV-8
Das Benutzer-Passwort verändern Hinweis: Voraussetzung ist der aktivierte USER_ID_LEVEL, d.h. ein vorher korrekt gegebenes altes Passwort.			
4 Byte	1 Parameter		
0	Char[4]	Wert	Wert des neuen Passwortes. Muss 4 ASCII-Zeichen lang sein.
0 Byte	0 Rückgaben		

GetDIOdirection (0x59)

0x59	R	GetDIOdirection	GSV-8
Die Datenrichtung (1=Input, 0=Output) der Gruppe der digitalen I/O Anschlüsse ermitteln			
1 Byte	1 Parameter		
0	Uint_8	Gruppen-No	Nummer der Gruppe [0-4]. =0: Richtungen aller 4 Gruppen bestimmen
1 Byte	1 Rückgaben		=0: Ausgang, =1: Eingang. Bei Gruppen-No=0: In Bit 0: Gruppenrichtung No 1… Bit 3: Richtung Gruppe 4

SetDIOdirection (0x5A)

0x5A	W	SetDIOdirection	GSV-8
Die Datenrichtung (1=Input, 0=Output) der Gruppe der digitalen I/O Anschlüsse setzen			
2 Bytes	2 Parameter		
0	Uint_8	Gruppen-No	Nummer der Gruppe [0-4]. =0: Richtungen aller 4 Gruppen setzen
1	Uint_8	Datenrichtung	=0: Ausgang, =1: Eingang. Bei Gruppen-No=0: In Bit 0: Gruppenrichtung No 1… Bit 3: Richtung Gruppe 4
0 Byte	0 Rückgaben		

GetDIOtype (0x5B)

0x5B	R	GetDIOtype	GSV-8
Die Funktion des digitalen I/O Anschlusses ermitteln			
1 Byte	1 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses [1-16].
4 Bytes	2 Rückgaben		
0	Uint_24	DIOtype	Typ des DIO Anschlusses (s. Tabelle im Anhang B)
1	Uint_8	Assigned channel	Zugeordneter Eingangskanal [1-8] (nur relevant bei Schwellwertschalter und Einzel-Tara)

SetDIOtype (0x5C)

0x5C	W	SetDIOtype	GSV-8
Die Funktion des digitalen I/O Anschlusses setzen Bemerkung: Eventuell muss erst die Datenrichtung mit *SetDIOdirection* gesetzt werden. Bei den im Anhang B tabellierten Typen ist die Richtung vermerkt.			
5 Bytes	3 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses [1-16]. Wert=0 bedeutet: Alle auf denselben Typ setzen
1	Uint_24	DIOtype	Typ des DIO Anschlusses (s. Tabelle im Anhang B)
2	Uint_8	Assigned channel	Zugeordneter Eingangskanal [1-8] (nur relevant bei Schwellwertschalter und Einzel-Tara)
0 Byte	0 Rückgaben		

GetDIOlevel (0x5D)

0x5D	R	GetDIOlevel	GSV-8
Den Pegel des digitalen I/O Anschlusses ermitteln			
1 Byte	1 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses 0..16. =0: Pegel aller DIO Anschlüsse bestimmen
2 Bytes	1 Rückgaben		
0	Uint_16	DIOlevel	Pegel des DIO Anschlusses: 0: Low, 1: High. Wenn DIOno=0: Bit 0: Anschluss No. 1 bzw. 1.1. ... Bit15: GPIO No 16 bzw. 4.4.

SetDIOlevel (0x5E)

0x5E	W	SetDIOlevel	GSV-8
Den Pegel des digitalen GPIO Anschlusses setzen (wenn Datenrichtung = Output und Typ= GP-out)			
3 Bytes	2 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses 0..16. =0: Pegel aller DIO Anschlüsse setzen
1	Uint_16	DIOlevel	Pegel des DIO Anschlusses: 0: Low, 1: High. Wenn DIOno=0: Bit 0: DIO No. 1.. Bit15: DIO No 16
0 Bytes	0 Rückgaben		

ReadDIOthreshold (0x5F)

0x5F	R	ReadDIOthreshold	GSV-8
Den Schwellwert des digitalen Schwellwertausgangs ermitteln			
2 Bytes	2 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses 1..16.
1	Uint_8	Upper / lower	=0: Unterer Schwellwert, =1: Oberer Schwellwert
4 Bytes	1 Rückgabe		
0	Float	DIOthreshold	Schwellwert des digitalen Schwellwertausgangs

WriteDIOthreshold (0x60)

0x60	W	WriteDIOthreshold	GSV-8
Den Schwellwert des digitalen Schwellwertausgangs schreiben			
6 Bytes	3 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses 1..16. =0: Alle auf denselben Wert setzen.
1	Uint_8	Upper / lower	=0: Unterer Schwellwert, =1: Oberer Schwellwert
2	Float	DIOthreshold	Schwellwert des digitalen Schwellwertausgangs
0 Byte	0 Rückgaben		

GetDIOinitialLevel (0x61)

0x61	R	GetDIOinitLevel	GSV-8
Den Default- bzw Initialisierungspegel des Digitalausgangs bestimmen			
1 Byte	1 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses 0..16. =0: Initpegel aller DIO Anschlüsse bestimmen
2 Bytes	1 Rückgabe		
0	Uint_16	DIO Init-Level	Initpegel des DIO Anschlusses: 0: Low, 1: High. Wenn DIOno=0: Bit 0: Anschluss No. 1 bzw. 1.1. ... Bit15: GPIO No 16 bzw. 4.4.

SetDIOinitialLevel (0x62)

0x62	W	SetDIOinitLevel	GSV-8
Den Default- bzw Initialisierungspegel des Digitalausgangs schreiben			
3 Bytes	2 Parameter		
0	Uint_8	DIO No	Nummer des DIO Anschlusses 0..16. =0: Initpegel aller DIO Anschlüsse setzen
1	Uint_16	DIO Init-Level	Initpegel des DIO Anschlusses: 0: Low, 1: High. Wenn DIOno=0: Bit 0: Anschluss No. 1 bzw. 1.1. ... Bit15: GPIO No 16 bzw. 4.4.
0 Byte	0 Rückgaben		

Der DIO-Default-Pegel wird bei digitalen Ausgängen dann gesetzt, wenn (noch) keine der definierten Bedingungen für High- oder Low-Zustand vorliegen; zB bei General-Perpose-Output nach dem Einschalten.

ReadDataRateRange (0x63)

0x63	R	ReadDataRateRange	GSV-8
Die maximal oder minimal einstellbare Datenrate bestimmen.			
1 Byte	1 Parameter		
0	Uint_8	Index	=0: Maximale Datenrate bestimmen =1: Minimale Datenrate bestimmen
4 Bytes	1 Rückgabe		
0	Float	Max/Min Drate	Maximale / Minimale Datenrate

ReadTEDSdataEntry (0x64)

0x64	R	ReadTEDSdataEntry	GSV-8
TEDS-Einträge lesen			
4 Bytes	4 Parameter		
0	Uint_8	Kanalnummer	1..8

0x64	R	ReadTEDSdataEntry	GSV-8
1	Uint_8	TEDS Template-ID.	0= Basic-TEDS, 33, 35
2	Uint_8	Property-ID	s. Anhang D
3	Uint_8	Array-Index	[Reserviert, z.Zt. =0]. Verwendet, wenn a) gleiche Prop-ID mehrfach vorh., b) wenn gleiches Template mehrfach vorh.
6 Bytes	3 Rückgaben		
0	Uint_8	Next Index	Nächster Index der Liste
1	Uint_8	ValTyp_Err	Datentyp des Wertes (Bytes 2..5) u. Errors, die nicht per Error-Frame zurückgegeben werden
2	Uint_32 oder Float	Data	Daten. Typ=Float, wenn ValTyp_Err =1. Sonst Uint32

Die Property-ID indiziert alle möglichen Einträge, dh 1. Properties (s. IEEE1541.4 Annex B.1), 2. relevante Selectors, 3. Sonderwerte; s. Anhang D. Die Reihenfolge der verketteten Liste entspricht (nach ID 0) der im Template.

Wenn Next ID =0: Letzter Eintrag.

ValTyp_Err:
Datentyp des Wertes (Bytes 2..5) u. Errors, die nicht per Error-Frame zurückgegeben werden:
#define ANSW_IS_UINT 0 /* Ganzzahliger unsigned-Wert */
#define ANSW_IS_FLT 1 /* Wert im Float-Format, bzw darin umgerechnet */
#define IS_PACKED_CHR5 2 /* CHR5-Typ, gepackt */
#define IS_DATE_DAYS 4 /* Datum in Tagen seit 1.1.1998 */
#define ENTRY_HAS_ERROR 0x80 /* Vergleichswert f. Error-Erkennung: Flags>=0x80: Datenwert nicht auswerten */
#define ENTRY_NOT_EXIST 0xFF /* Entry-Anforderung bzw Property-ID existiert nicht im Template. Auch bei Index 0; dann nicht als error anzusehen */
#define ENTRY_NOT_SET 0xFE /* Eintrag existiert, ist aber als "don't care" geflaggt, dh alle Bits=1 */
#define ENTRY_INVALID 0xFD /* Eintrag ungueltig, zB NaN bei Float */

ReadTEDSrawArray (0x65)

0x65	R	ReadTEDSrawArray	GSV-8 / GSV-6
TEDS-Rohdaten lesen **Hinweis**: Der Befehl ist mit dem GSV-6 erst ab Firmware-Version 3.10 verfügbar			
3 Bytes	2 Parameter		
0	Uint_8	Kanalnummer	1..8
1	Uint_16	Byteadresse	Byteadresse des TEDS-Bereiches im 1-wire EEPROM ohne Checksumme, 4-Byte aligned
4 Bytes	4 Rückgaben		
0	Uint_8	Data @ Address	Databyte 1
1	Uint_8	Data @ Address +1	Databyte 2
2	Uint_8	Data @ Address +2	Databyte 3
3	Uint_8	Data @ Address +3	Databyte 4

WriteTEDSbytes (0x66)

0x66	W	WriteTEDSbytes	GSV-8
TEDS-Rohdaten schreiben			
6 Bytes	6 Parameter		
0	Uint_8	Kanalnummer	1..8
1	Uint_8	Byteadresse	Virtuelle Byteadresse, die bei jedem Schreibvorgang mit 0 beginnt. Muss /4 teilbar sein.

0x66	W	WriteTEDSbytes	GSV-8
2	Uint_8	Data @ Address	Databyte 1
3	Uint_8	Data @ Address +1	Databyte 2
4	Uint_8	Data @ Address +2	Databyte 3
5	Uint_8	Data @ Address +3	Databyte 4
0 Bytes	0 Rückgaben		

Die Daten werden nur in das RAM des Gerätes kopiert, d.h. noch nicht in das 1-wire EEPROM geschrieben. Bitfelder auf Bit 0 an Adresse 0 aligned, d.h. höchstwertigste Bytes/Bits ggf. don't care. Ohne Checksumme (wird in Cmd 0x67 berechnet und gesetzt).

StoreTEDSdata (0x67)

0x67	-	StoreTEDSdata	GSV-8
TEDS-Rohdaten aus Geräte-RAM in das 1-wire-EEPROM übertragen			
5 Bytes	3 Parameter		
0	Uint_8	Kanalnummer	1..8
1	Uint_16	Bitadresse	Bitadresse im 1-wire EEPROM, ab dem die Daten geschrieben werden, ohne Checksumme
2	Uint_16	Bitsize	Größe des zu schreibenden Datenfeldes in Bits
0 Bytes	0 Rückgaben		

Get TEDS active (0x68)

0x68	R	Get TEDS active	GSV-8 / GSV-6
Ermitteln, ob ein gültige Daten aus einem Sensor mit TEDS geladen und verwendet werden **Hinweis**: Der Befehl ist mit dem GSV-6 erst ab Firmware-Version 3.10 verfügbar			
0 Bytes	0 Parameter		
4 Bytes	1 Rückgabe		Bits<31:8>: reserviert Bits<7:0>: Wenn Bit=1: Am Eingangskanal (BitNo+1) ist ein TEDS-Baustein mit gültigen und bekannten Daten angeschlossen, die zur Berechnung des UserScale-Wertes verwendet wurden (d.h. dann ist auch entsprechendes Bit in <15:8> des Mode-Wertes gesetzt).

Reset Device (0x78)

0x78	-	Reset Device	GSV-6
Den GSV-6 Neustarten. **Hinweis**: Der GSV-6 sendet auf diesen Befehl ausnahmsweise **keinen Antwortframe!** Der Befehl ist erst ab Firmware-Version 3.10 verfügbar			
0 Bytes	0 Parameter		
0 Bytes	0 Rückgabe		

Release Interface (0x7A)

0x7A	-	ReleaseInterface	GSV-8
Dieses verwendete Kommunikationsinterface freigeben. Sollte am Ende einer Kommunikationssession aufgerufen werden, um anderen Interfaces ggf. das Schreibrecht zu geben.			
0 Byte	0 Parameter		
0 Byte	0 Rückgaben		

ReadInterfaceSetting (0x7B)

0x7B	R	ReadInterfaceSetting	GSV-8
Einstellungen der Kommunikationsinterfaces bestimmen			
1 Byte	1 Parameter		
0	Uint_8	Index / Interface-Nr.	0.. Anzahl vohandener Interfaces-1: Basic settings. Darüber: Extended settings
6 Bytes	3 Rückgaben		
0	Uint_8	next index	Nächster Index in der verketteten Liste. =0: Letzter Eintrag.
1	Uint_8	Phys.Typ / Dat.typ	Basic settings: Physikalischer Schnittstellentyp (Enum) Extended settings: Bit 7: =1: Eintrag schreibbar. =0: Nicht schreibbar. Bits<6:0>: Typ des Dateneintrags (Enum)
2	Uint_32	Data	Basic: Bits<31:24>: Interface-Applikationstyp (Enum) Bits<32:0>: Flags Extended: Bits<31:0>: Dateneintrag

Die Indizes sind in 2 Bereiche unterteilt. Der Indexbereich 1 reicht von 0 bis (Anzahl vorhandener Schnittstellen -1). Hier können "Basic settings" zu jeder der vorhandenen Schnittstellen ermittelt werden (s. Anhang C). Die Anzahl vorhandener Schnittstellen kann mit dem Befehl GetInterface (0x01) ermittelt werden.

Die Rückgabe "Next index" zeigt auf den Beginn des Indexbereiches der "Extended settings". Bei diesen wird im Rückgabeparameter 1 ein Enum zurückgegeben, der bestimmt, um was für Daten es sich handelt (s. Anhang C).

WriteInterfaceSetting (0x7C)

0x7C	W	WriteInterfaceSetting	GSV-8
Einstellungen der Kommunikationsinterfaces ändern			
5 Bytes	2 Parameter		
0	Uint_8	Index / Interface-Nr.	0.. Anzahl vohandener Interfaces-1: Basic settings. Darüber: Extended settings
1	Uint_32	Data	Basic: Bits<31:24>: Interface-Applikationstyp (Enum) Bits<23:0>: Flags Extended: Bits<31:0>: Dateneintrag
0 Byte	0 Rückgaben		

PrepReadFTsensor (0x7D)

0x7D		ReadDataRateRange	GSV-8
FT-Kalibrierarray zum Lesen auswählen. Nachfolgende Aufrufe von ReadFTsensorCal lesen aus diesem Array. Die Ausführung hat keine Auswirkung auf die aktuelle Sechachsensensorberechnung.			
1 Byte	1 Parameter		
0	Uint_8	Index	Array-Index [0-4]
0 Byte	0 Rückgaben		

StoreDigiFilt (0x7E)

0x7E	W	StoreDigiFilt	GSV-8
Speichern der Digitalfilterparameter aller Kanäle. Die beiden Parameter werden beim GSV-8 ignoriert.			
2 Byte	2 Parameter		
0	uint8_t (1)		
1	uint8_t (1)		
0 Byte	0 Rückgaben		

0x7E	W	StoreDigiFilt	GSV-6
Speichern der Digitalen-Filter im Flash von einem (Kanal-Beginn==Kanal-Ende) oder mehrerer Kanäle (Kanal-Beginn!=Kanal-Ende).			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal-Beginn	Start-Index (0-5)
1	uint8_t (1)	Kanal-Ende	Stop-Index (0-5)
0 Byte	0 Rückgaben		

StoreFTSensorCal (0x7F)

0x7F	W	StoreSensorCal	GSV-6 / GSV-8
Speichern einer 6-Achsen Kalibriermatrix, die im Matrix-Ram liegt, auf Index im Flash. GSV-8: Voraussetzung ist ein gesetzter USER_ID_LEVEL			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	GSV-6: Array-Index [0-7] GSV-8: Array-Index [0-4] oder Sonderwert 0xFF. Es können bestehende Datenarrays überschrieben oder neue hinten angehängt werden, „Speicherlöcher“ sind nicht erlaubt. Mit dem Sonderwert 0xFF kann das letzte Array gelöscht werden.
0 Byte	0 Rückgaben		

GetTXMode (0x80)

0x80	R	GetTXMode	GSV-6
Aktuelle Messwert-Frame Einstellungen abfragen			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: TX-Mode 1: Messwert-Datenformat 2: Maximale Kanalanzahl Alle anderen Werte sind reserviert.
2 Byte	1 Rückgaben		

0x80	R	GetTXMode	GSV-6
0	uint16_t (2)	Rückgabe	Index 0: TX-Mode - <0> TX temporär AUS - <1> TX permanent AUS - <15:2> reserviert Index 1: Messwert-Datenformat - 1: int16 - 3: float - andere Werte reserviert Index 2: - Maximalanzahl der Kanäle ==6

0x80	R	GetTXMode	GSV-8
Aktuelle Messwert-Frame Einstellungen abfragen			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: TX-Mode 1: Messwert-Datenformat 2: Kanalnummern-Bereich Alle anderen Werte sind reserviert.
2 Byte	1 Rückgaben		
0	uint16_t (2)	Rückgabe	Index **0**: TX-Mode <0> =1: Messwert-Übertragung temporär AUS (read only) <1> =1: Messwert-Übertragung permanent AUS <2> =1: Maximalwerte im Messwertframe übertragen <3> =1: Minimalwerte im Messwertframe übertragen <5:4> reserviert <6> =1: Schreiben via UART blockiert (read only) <7> =1: Schreiben via USB blockiert (read only) <8> =1: TX-Synchonisation: Slave (read only) <9> =1: TX-Synchonisation: Master (read only) <15:10> reserviert Index **1**: Messwert-Datenformat 1: int16 2: S24 3: float - andere Werte reserviert Index **2:** Vorhandene Eingangskanäle: Bits<7:0>: Kleinste Kanalnummer (=1) Bits<15:8>: Höchste Kanalnummer (=8)

SetTXMode (0x81)

0x81	W	SetTXMode	GSV-6 / GSV-8
Setzen der Messwert-Frame Einstellungen.			
3 Byte	2 Parameter		

0x81	W	SetTXMode	GSV-6 / GSV-8
0	uint8_t (1)	Index	0: TX_mode 1: Messwert-Datenformat Alle anderen Werte reserviert.
1	uint16_t (2)	Wert	siehe GetTXMode (0x80)
0 Byte	0 Rückgaben		

ReadDataRate (0x8A)

0x8A	R	ReadDataRate	GSV-6 / GSV-8
Datenrate, mit der Messwertframes ausgegeben werden, auslesen.			
0 Byte	0 Parameter		
4 Byte	1 Rückgaben		
0	float (4)	Datenrate	Messwertframes pro Sekunde in Hz

WriteDataRate (0x8B)

0x8B	W	WriteDataRate	GSV-6 (Reset) / GSV-8
Datenrate, mit der Messwertframes ausgegeben werden sollen, setzen			
4 Byte	1 Parameter		
0	float (4)	Datenrate	Datenrate in Hz
0 Byte	0 Rückgaben		

GetCANSetting (0x8C)

0x8C	R	GetCANSetting	GSV-6 / GSV-8
CAN Einstellung abfragen Hinweis für GSV-8: Wenn der Aufruf dieser Funktion ERR_CMD_NOTKNOWN ergibt, wird CAN/CANopen nicht unterstützt.			
1 Byte	1 Parameter		
0	uint8_t (1)	Index	0: CAN_IN Befehl Can-ID 1: CAN_OUT Antwort Can-ID 2: CAN_CV Messdaten-Frame Can-ID 3: CAN_CAST Multicast Can-ID 4: CAN_BAUD Baudrate (in Hz) 5: CAN_FLAGS 6: CANopen Node-ID (nur bei GSV-8) Alle anderen Werte sind reserviert
4 Byte	1 Rückgaben		
0	uint32_t (4)	Can-ID Baudrate Flags	Can-ID: 0x00000000-0x000007FF: Std-ID 0x80000000-0x9FFFFFFF: Ext-ID Baudrate (GSV-6): 1000000, 500000, 250000, 125000, 100000, 50000, 25000, 12500 Baudrate (GSV-8): 1000000, 500000, 250000, 125000, 50000 Flags: GSV-6: <31:0> Reserviert (=0)

0x8C	R	GetCANSetting	GSV-6 / GSV-8
			GSV-8: Bit 0: CAN on / off: =**1**: CAN **aus**geschaltet =0: CAN eingeschaltet Bit 1: CAN-Applikationsprotokoll: =1: CANopen aktiv =0: serieller CAN (wie hier beschrieben) aktiv

SetCANSetting (0x8D)

0x8D	W	SetCANSetting	GSV-6 (Reset) / GSV-8 (ggf. Reset)
CAN Einstellungen ändern. Hinweis: Beim GSV-8 können Baudrate und CAN- bzw. NodeIDs nur bei ausgeschaltetem CAN-Interface eingestellt werden.			
5 Byte	2 Parameter		
0	uint8_t (1)	Index	siehe GetCANSetting (0x8C)
1	uint32_t (4)	Can-ID, Baudrate, Flags	siehe GetCANSetting (0x8C)
0 Byte	0 Rückgaben		

ReadAnalogueFilter (0x90) [GSV-8]

0x90	R	ReadAnalogueFilter	GSV-8
Analog-Filter Frequenz auslesen			
0 Byte	0 Parameter		
2 Byte	1 Rückgaben		
0	uint16_t (2)	Filter-Frequenz	Grenzfrequenz in Hz

WriteAnalogueFilter (0x91) [GSV-8]

0x91	W	WriteAnalogueFilter	GSV-8
Grenzfrequenz des analogen Vorfilters setzen.			
4 Byte	1 Parameter		
0	float (4)	Grenzfrequenz in Hz	Es können 3 verschiedene Frequenzen gesetzt werden: - 28 Hz - 885 Hz – 11,4 kHz Bei Übergabe anderer Werte wird die nächste vorhandene Frequenz gesetzt
0 Byte	0 Rückgaben		

SwitchBlocking (0x92)

0x92	W	SwitchBlocking	GSV-6 / GSV-8
Schreibschutz setzen Zum Entsperren muss der USER_ID_LEVEL aktiviert sein, d.h. das Passwort gegeben worden sein.			
4 Byte	1 Parameter		
0	uint32_t (4)	Schutz-Code	BLOCKING_UNLOCK_THIS 0x554c4b74 = "ULKt" (reserviert) BLOCKING_UNLOCK_ALL 0x554c4b61 = "ULKa" BLOCKING_LOCK_THIS 0x426c6b54 = "BlkT" (reserviert) BLOCKING_LOCK_ALL 0x426c6b41 = "BlkA"
0 Byte	0 Rückgaben		

GetCommandAvailable (0x93)

0x93	W	GetCommandAvailable	GSV-6 / GSV-8
Überprüft, ob alle Kommandos von dem Index HighCmd bis einschließlich dem Index LowCmd implementiert sind, und liefert den entsprechenden Fehler-Code. ***Achtung***: LowCmd muss kleiner oder gleich HighCmd sein!			
2 Byte	2 Parameter		
0	uint8_t (1)	HighCmd	Oberer Kommando-Index (inklusive)
1	uint8_t (1)	LowCmd	Unterer Kommando-Index (inklusive)
0 Byte	0 Rückgaben (Information im ErrorByte des Antwortframes)		Information im Error-Byte: ERR_OK: Alle Befehle vorhanden. ERR_CMD_NOTIMPL: Ein oder mehrere Befehle nicht vorhanden.

ReadNoiseCutThreshold (0x94) [GSV-8]

0x94	R	ReadNoiseCutThreshold	GSV-8
Abfragen der Schwelle für die Rauschunterdrückung. Unterhalb dieser Schwelle und oberhalb der negierten Schwelle (d.h. „um 0 herum“) werden Messwerte =0 gesetzt.			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer [1-8] =0: Alle Kanäle auf denselben Wert setzen
4 Byte	1 Rückgaben		
0	float (4)	Schwelle	

WriteNoiseCutThreshold (0x95) [GSV-8]

0x95	W	WriteNoiseCutThreshold	GSV-8
Setzen der Schwelle für die Rauschunterdrückung Unterhalb dieser Schwelle und oberhalb der negierten Schwelle (d.h. „um 0 herum“) werden Messwerte =0 gesetzt.			

ReadUserOffset (0x9A)

0x9A	R	ReadUserOffset	GSV-6 / GSV-8
Benutzerdefinierten Offset auslesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6] GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	float (4)	Offset	Offset des Kanals

WriteUserOffset (0x9B)

0x9B	W	SetUserOffset	GSV-6 / GSV-8
Benutzerdefinierten Offset setzen **Hinweis für GSV-6:** Wenn die 6-Achsen-Berechnung aktiv ist, haben diese Werte keine Auswirkung!			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	float (4)	Offset	
0 Byte	0 Rückgaben		

GetInputType (0xA2) [GSV-6]

0xA2	R	GetInputType	GSV-6
Aktuellen Eingangstyp abfragen Bei GSV-6 wird hiermit der elektrische Gesamt-Messbereich des Analogeingangs bestimmt.			
2 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6]
1	uint8_t (1)	Res.	=0 (ungenutzt)
4 Byte	1 Rückgabe		
1	Uint32_t (4)	Sens	Eingangsmessbereich in mV*100 (bzw. mV/V*100)

GetInputType (0xA2) [GSV-8]

0xA2	R	GetInputType	GSV-8
Aktuellen Eingangstyp abfragen Bei GSV-8 werden hiermit die möglichen Messbereiche der analogen Eingangsbeschaltung abgefragt, die zu dem angeschlossenen Sensor zusammenpassen sollte. Es können sowohl alle vorhandenen Messbereiche abgefragt werden, als auch der aktuell gesetzte Messbereich und Eingangstyp.			
2 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer [1-8]

0xA2	R	GetInputType	GSV-8
1	uint8_t (1)	SensIndex	[0xFF..0x05] Der Sensindex ist wie folgt definiert: 0xFF Aktuell eingestellten Eingangstyp und dessen Messbereich lesen 0x00 Eingangsmessbereich für Eingangstyp = Brückensensor mit Speisespannung 8,75V lesen 0x01 Eingangsmessbereich für Eingangstyp = Brückensensor mit Speisespannung 5V lesen 0x02 Eingangsmessbereich für Eingangstyp = Brückensensor mit Speisespannung 2,5V lesen 0x03 Eingangsmessbereich für Eingangstyp = Single-ended Input lesen 0x04 Eingangsmessbereich für Eingangstyp = Temperatursensor PT1000 lesen 0x05 Eingangsmessbereich für Eingangstyp = Temperatursensor K-Type lesen (reserviert)
5 Byte	2 Rückgaben		
0	uint8_t (1)	Type	Aktueller Eingangs-Vorbeschaltungstyp Eingangstyp wie oben unter SensIndex definiert (Bereich: 0..5)
1	Uint32_t (4)	Sens	Eingangsmessbereich in mV*100 (bzw. mV/V*100)

SetInputType (0xA3) [GSV-8]

0xA3	W	SetInputType	GSV-8
Ändern des Eingangstyps Hinweis: Beim Ändern des Eingangstyps werden die Kalibrier-Nullpunkte (Hersteller) und die Defaulr-UserScale Werte geladen, die vorherigen Parameter für Nullpunkt und UserScale werden dabei überschrieben.			
2 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem Identischen Wert beschrieben.
1	Uint_8	Eingangstyp	Der Eingangstyp ist wie folgt definiert: 0x00 Eingangstyp = Brückensensor mit Speisespannung 8,75V setzen 0x01 Eingangstyp = Brückensensor mit Speisespannung 5V setzen 0x02 Eingangstyp = Brückensensor mit Speisespannung 2,5V setzen 0x03 Eingangstyp = Single-ended Input setzen 0x04 Eingangstyp = Temperatursensor PT1000 setzen 0x05 Eingangstyp = Temperatursensor K-Type setzen (reserviert)

0xA3	W	SetInputType	GSV-8
0 Byte	0 Rückgaben		

ReadSensorCapacity (0xA4) [reserviert]

0xA4	R	ReadSensorCapacity	GSV-6 / GSV-8
Physikalischen Sensor-Nennwert (Sensor-Messbereich) auslesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6] GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	float (4)	Nennwert	

WriteSensorCapacity (0xA5) [reserviert]

0xA5	W	WriteSensorCapacity	GSV-6 / GSV-8
Physikalischen Sensor-Nennwert (Sensor-Messbereich) setzen			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem identischen Wert beschrieben.
1	float (4)	Nennwert	
0 Byte	0 Rückgaben		

ReadRatedSensorOutput (0xA6) [reserviert]

0xA6	R	ReadRateOutput	GSV-6 / GSV-8
Elektrischen Kennwert [mV/V] des Sensors auslesen			
1 Byte	1 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [1-6] GSV-8: [1-8]
4 Byte	1 Rückgaben		
0	float (4)	Kennwert	

WriteRatedSensorOutput (0xA7) [reserviert]

0xA7	W	WriteRatedOutput	GSV-6 / GSV-8
Elektrischen Kennwert [mV/V] des Sensors setzen			
5 Byte	2 Parameter		
0	uint8_t (1)	Kanal	Kanalnummer GSV-6: [0-6] GSV-8: [0-8] Mit Kanal 0 werden ALLE Kanäle mit dem identischen Wert beschrieben.
1	float (4)	Kennwert	
0 Byte	0 Rückgaben		

Aufbau und Verwendung der CAN-DLL (GSV-6)

Im Rahmen der Hardwareerweiterungen ist der serielle CAN Bus hard- und softwareseitig integriert worden. Mit ihm besteht jetzt die Möglichkeit, die Sensoren in klassische Feldbussysteme zu einzubinden.

Mit der Integration des Bussystems besteht die Erfordernis, existierende oder neu zu erstellende Erfassungssoftware auf diese Kommunikationsvariante anzupassen. Mit dem Vorliegen der DLL besteht die Möglichkeit, auf die neue Hardwareschnittstelle zuzugreifen.

Die CAN-DLL berücksichtigt viele der o.g. Gerätebefehle. Diese sind als exportoierte Funktion implementiert und kann von der Erfassungsapplikation importiert werden. Bespielsweise kann über die DLL die Konfiguration der Kalibriermatrix der Sechsachsensensoren per Konfigurationsdatei erfolgen.

Schichten der CAN-DLL

Die CAN-DLL ist in vier folgende Schichten aufgebaut:

1. GSV-Schicht

Die GSV-Schicht ist die Schnittstelle zur Applikation, die die DLL einbindet. In dieser Schicht sind die Funktionen implementiert, die für die GSV-Protokollbearbeitung wichtig sind. Ferner erfolgt in dieser Schicht die Verarbeitung der Messdaten, sofern dies notwendig ist (Filter) sowie das Einlesen und Anwenden der 6-Achsen Kalibriermatrizen-Dateien durchgeführt.

2. Geräteschicht

Die Geräteschicht übernimmt das Handling der unterschiedlichen GSV-Geräte. Sie verwaltet die Geräteinformationen und die Nachrichtenzuordnung, und erlaubt eine Zuordnung der Geräte wie bei der COM-Schnittstelle.

3. CAN-Schicht

Die CAN-Schicht abstrahiert die herstellerspezifischen CAN-Funktionen und stellt eine Schnittmenge an typischen Funktionen für die Geräteschicht bereit. In dieser Schicht werden die herstellerspezifischen Funktionen der darunterliegenden Schicht aufgerufen.

4. XCAN-Wrapper-Schicht

Die XCAN-Schicht ist stellvertretend die herstellerspezifische Schicht, die die entsprechende DLL des Herstellers einbindet und die CAN-Funktionen für die CAN-Schicht bereitstellt. Diese Schicht kann, mit Anpassung der CAN-Schicht, gegen andere CAN-Hersteller ausgetauscht werden. Aktuell wird die PCAN (Peak CAN) DLL unterstützt.

Die nachfolgende Grafik illustriert den Aufbau der DLL:

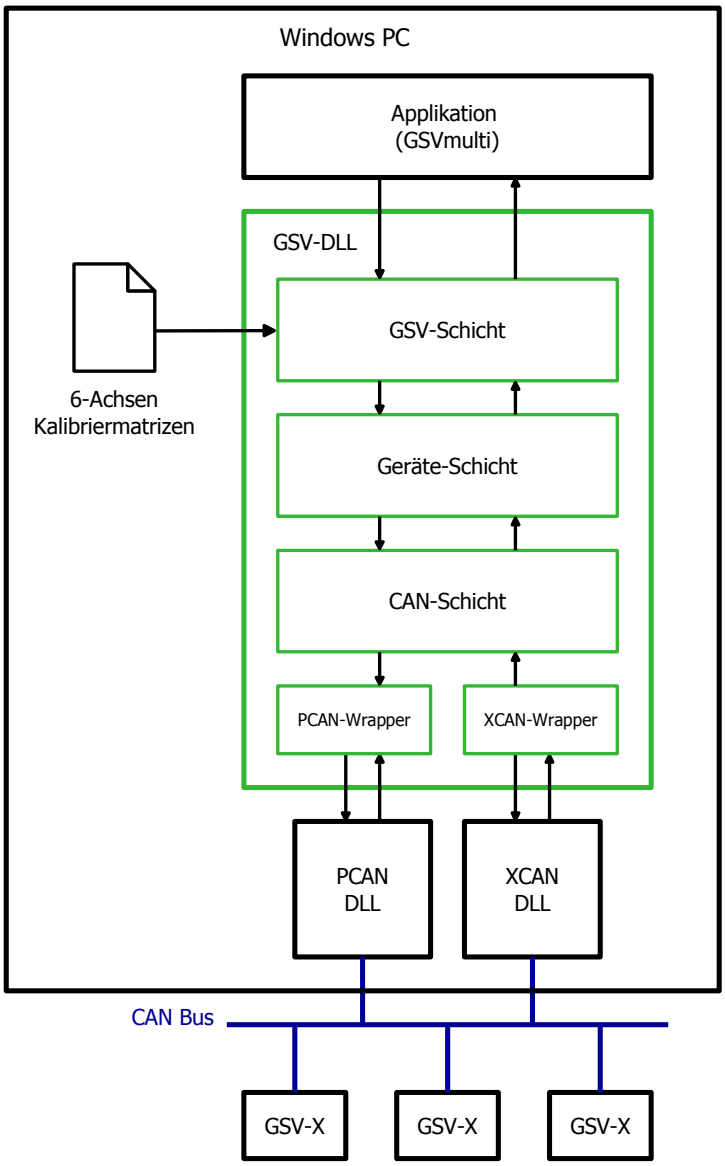

Abbildung Architektur der CAN-DLL

CAN-DLL einbinden

Eine Applikation, die diese DLL verwenden soll, muss folgende Aspekte berücksichtigen:

. die DLL muss eingebunden werden

```
hinstLib = LoadLibrary(TEXT("GSVDLL.dll"));
```

. die Namen der Funktion werden von der DLL exportiert

. die Funktionen müssen einzeln in der Applikation importiert werden

```
typedef int(__stdcall *GSV86CANACTIVATEEXTENDED) (int DevNo, int CanType, unsigned long Bi-
trate, unsigned long BufSize, int *Count, unsigned long CommandIndentifier, unsigned long Respon-
seIndentifier, long Flags);
GSV86CANACTIVATEEXTENDED GSV86CANactivateExtended;
GSV86CANactivateExtended = (GSV86CANACTIVATEEXTENDED)GetProcAddress(hinstLib, "GS-
V86CANxActivateExtended");
```

. anwenden:

```
GSV86CANactivateExtended (
```

```
int DevNo,                          //Deviceenumerator (ungleich 0)
int CanType,                        // Can-Adpter CANADAPTERTYPE_PCAN
unsigned long Bitrate,              //CAN Baudrate in Bit/Sek.
unsigned long BufSize,              //Buffergroesse pro Kanal bis 0xFFFE (Ringbuf-
fer)
int *Count,                         //Anzahl der Messkanaele (1-4, 6)
unsigned long CommandIndentifier,   //CAN-ID fuer Kommandos
unsigned long ResponseIndentifier,  //CAN-ID fuer Antworten des GSV
long Flags)                         //AE_FLAG - wenigstens _TYPEGSV6 oder _TY-
PEGSV8
```

Hinweis: Die GSV-DLL sowie die PCAN-DLL müssen entweder im gleichen Verzeichnis wie die Applikation oder im Windows Systemverzeichnis liegen.
Zum Anzeigen der Daten auf dem CAN-Bus kann das Flag AE_FLAG_SHOWCAN (0x002) aktiviert werden. Das Logging der PCAN-Schnittstelle kann mit AE_FLAG_LOGCAN (0x0004) aktiviert werden. In diesem Falle wird eine "PCANBasic.log" im Verzeichnis der DLL abgelegt.
Die Dokumentation der exportierten Funktionen kann deren Headern entnommen werden .

Anhang

A: Physikalische Einheiten (Codes)

In der folgenden Tabelle sind die Standard ME-Codes für die Einheiten aufgeführt:

Einheit	Code
mV/V	0
kg	1
g	2
N	3
cN	4
V	5
µm/m	6
(keine)	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, Neuendorfstr. 18a, DE-16761 Hennigsdorf
Tel +49 (0)3302 78620 60, Fax +49 (0)3302 78620 69, info@me-systeme.de, www.me-systeme.de

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

Tabelle: Einheiten-Tabelle

B: Typen der digitalen Ein- und Ausgänge des GSV-8

Es können folgende Funktionen konfiguriert werden:

Nr	Funktion	Daten-richtung	**Wert** Gerätebefehl Get/SetDIOtype = Wert DLL-Funktion GSV86get/setDIOtype	Kurzbeschreibung
1	General-Purpose Input	Eingang	0x000004	Allgemeiner Eingang. Logikpegel kann mit GetDIOlevel / GSV86getDIOlevel abgefragt werden.
2	Nullsetzen Einzelkanal	Eingang	0x000010	Aktiver Input-Pegel setzt einen analogen Eingangskanal Null.
3	Nullsetzen alle Kanäle	Eingang	0x000020	Aktiver Input-Pegel setzt alle analogen Eingangskanäle Null.
4	Rücksetzen der Maximal- und Minimalwert-ermittlung	Eingang	0x000040	Aktiver Input-Pegel setzt alle Maximal- und Minimalwerte zurück.
5	Trigger Send actual value	Eingang	0x000080	Auslösen des Sendens eines Messwertframes mit aktuellen Messwerten über die USB-Schnittstelle an inaktiv-zu-aktiv Flanke des digitalen Eingangs.
6	Trigger maximum value	Eingang	0x000100	Bei inaktiv-zu-aktiv Flanke am digitalen Eingang wird die Maximalwertermittlung (alle Eingangskanäle) begonnen und an aktiv-zu-inaktiv Flanke wird ein Frame mit diesen Maximalwerten an die USB-Schnittstelle gesendet.
7	Trigger minimum value	Eingang	0x000200	Bei inaktiv-zu-aktiv Flanke am digitalen Eingang wird die

				Minimalwertermittlung (alle Eingangskanäle) begonnen und an aktiv-zu-inaktiv Flanke wird ein Frame mit diesen Minimalwerten an die USB-Schnittstelle gesendet.
8	Trigger mean value	Eingang	0x000400	Bei inaktiv-zu-aktiv Flanke am digitalen Eingang wird eine dezimierende Mittelwertbildung (alle Eingangskanäle) begonnen und an aktiv-zu-inaktiv Flanke wird ein Frame mit diesen Mittelwerten an die USB-Schnittstelle gesendet.
9	Trigger Send actual value	Eingang	0x000800	Während der Input-Pegel aktiv ist, werden Messwertframes mit aktuellen Messwerten über die USB-Schnittstelle gesendet, mit der eingestellten Datenrate.
10	Sync Slave	Eingang	0x000002	An Low-to-High Flanke wird der zuletzt ermittelte Messwertframe übertragen. Nur zur Verwendung mit einem zweiten GSV-8, der als Sync-Master konfiguriert ist. Dessen Sync-Output (s. Nr. 18) muss mit dem Sync-Eingang des oder der Slave(s) verdrahtet sein.
11	General-Purpose Output	Ausgang	0x001000	Allgemeiner Ausgang. Aktueller Logikpegel kann mit SetDIOlevel / GSV86setDIOlevel festgelegt werden.
12	Threshold output aktueller Messwert	Ausgang	0x010000	Schwellwertausgang: Ausgang wird aktiviert, wenn der zugeordnete Messwert größer als der obere Schwellwert ist und deaktiviert, wenn er kleiner als der untere Schwellwert ist.
13	Threshold output Maximalwert	Ausgang	0x014000	Schwellwertausgang: Ausgang wird aktiviert, wenn der zugeordnete Maximalwert größer als der obere Schwellwert ist und deaktiviert, wenn er kleiner als der untere Schwellwert ist.
14	Threshold output Minimalwert	Ausgang	0x018000	Schwellwertausgang: Ausgang wird aktiviert, wenn der zugeordnete Minimalwert größer als der obere Schwellwert ist und deaktiviert, wenn er kleiner als der untere

				Schwellwert ist.
15	Fensterkomparator -ausgang aktueller Messwert	Ausgang	0x012000	Fensterkomparator: Ausgang wird aktiviert, wenn der zugeordnete Messwert kleiner als der obere Schwellwert und größer als der untere Schwellwert ist; sonst deaktiviert.
16	Fensterkomparator -ausgang Maximalwert	Ausgang	0x016000	Fensterkomparator: Ausgang wird aktiviert, wenn der zugeordnete Maximalwert kleiner als der obere Schwellwert und größer als der untere Schwellwert ist; sonst deaktiviert.
17	Fensterkomparator -ausgang Minimalwert	Ausgang	0x01A000	Fensterkomparator: Ausgang wird aktiviert, wenn der zugeordnete Minimalwert kleiner als der obere Schwellwert und größer als der untere Schwellwert ist; sonst deaktiviert.
18	Sync-Master	Ausgang	0x020000	Der Sync-Master erzeugt sychron zu seiner Messwertframe-übertragung ein Synchronisations-signal, an dem die Slaves angeschlossen sind (s. Nr. 10). Bei Messwertübetragung ist der Pegelwechsel Low-zu-High, nach der halben Datenperiode High-zu-Low.

Die DIOs besitzen Pullup-Widerstände, die bei offenem Eingang High-Pegel erzeugen. Bei Eingangstrigger-Funktionen, die mit einem Ein/Aus-Schalter bedient werden sollen, ist daher der Schalter/Taster zwischen der DIO-Klemme und GNDD anzuschließen. Damit die Funktion bei geschlossenem Schalter ausgeführt wird, muss der Anschluss per Software funktional invertiert werden. Dazu ist bei Verwendung des Geräteinterfaces oder der DLL der in o.g. Spalte "Wert" genannte **Wert mit 0x800000 zu verodern**.

Auch die Schwellwertausgänge können so invertiert werden.

In o.g. Tabelle bedeutet:

Pegel	Nicht-Invertiert	Invertiert
Aktiv	Logisch 1 = High = 5V	Logisch 0 = Low = 0V
Inaktiv	Logisch 0 = Low = 0V	Logisch 1 = High = 5V

Nur bei Verwendung der GeneralPurpose-Funktionen und der Master-slave-sync Funktionen (Nr. 1,10, 11 und 18 in o.g. Tabelle) hat die Invertierung keine Wirkung; die Funktionen GSV86get/setDIOlevel und Get/SetDIOlevel lesen bzw. schreiben den Pegel stets direkt, d.h. nicht-invertiert.

C: Flags und Enumerationen für Read / Write Interface Settings (Cmd 0x7B / 0x7C)

Basic Settings: Physikalischer Schnittstellentyp

Enum-No.	Bedeutung
1	RS232 Schnittstelle (asynchron, 8N1) mit V24 Pegel
2	RS232 Schnittstelle (asynchron, 8N1) mit 3,3V Pegel (Low=0V, High=3,3V)
3	USB Schnittstelle
4	CAN Feldbus-Schnittstelle
5	100BaseTX ("Ethernet")

Basic Settings: Typ des Applikationslayers

Enum-No.	Bedeutung
1	GSV8/6 Protokoll, wie in diesem Dokument beschrieben
2	CANopen
3	EtherCAT CoE
4	Textuelles "Monitor" Protokoll (nur GSV-6)

Dieser Wert wird in Bits<31:24> des Datenwertes in den Basic Settings kommuniziert

Basic Settings: Flags im Flagwert

Bit-Nr.	Bedeutung
0	=1: Inteface ist aktiviert und empfangsbereit
1	=1: Inteface ist aktiviert und sendebereit
2	=1: Interface hat Schreibrecht
3	=1: Integer-Messdatenwert ist im Binary offset-Format (nur bei App.Enum =1 gültig) =0: Integer-Messdatenwert ist im SignedInt Format (nur bei App.Enum =1 gültig)
16	=1: Interface zu- und abschaltbar
17	=1: Interface verwendet Geräte- oder Dienstadressen (Feldbus)
18	=1: Adresse(n) ist/sind änderbar

Dieser Flagwert wird in Bits<23:0> des Datenwertes in den Basic settings kommuniziert

Extended Settings: Typ des Dateninhalts (Enum)

Enum-No.	Bedeutung
1	Maske zum Setzen des BasicSettingsFlagwertes: Bit=1: Gleiche BitNo darf =1 gesetzt werden
2	Maske zum Löschen des BasicSettingsFlagwertes: Bit=1: Gleiche BitNo darf =0 gesetzt werden
3	Nr. der Schnittstelle(n), mit denen dieses wechslseitig auschließend (mutually

Enum-No.	Bedeutung
	exclusive) vorhanden ist. Dabei kann in jedem der 4 Bytes eine Schnittstellennummer >0 stehen, auffüllend beginnend mit dem LSbyte. Wert =0x00000000 bedeutet: Mit keiner Schnitttselle wechslseitig auschließend.
4	Aktive Baudrate in Bits/s. Wert =0 bedeutet: Keine Baudrate anzuwenden bzw nicht veränderbar
5	Vorhandene Baudrate in Bits/s
6	Anzahl der aktiven Dienst-IDs bzw. (Geräte-) Adressen
7	CAN-ID des Kommandodienstes Host->Device
8	CAN-ID des Kommandodienstes Befehlsantworten Device-> Host
9	CAN-ID der Messwertframes Device-> Host
10	CAN-ID Multicast Host->Devices
11	CANopen NodeID
16	Gerätezustand, bes. bei Feldbus. s. nächste Tabelle
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

Dieser Wert wird in Bits<6:0> des zweiten Rückgabeparameters bei Extended-Settings Anfragen des Befehls ReadInterfaceSetting (0x7B) kommuniziert.

Kodierung des Gerätezustands

Wert	Bedeutung
0	Interface ist abgeschaltet.
2	Interface an, Zustand = "Init" / "Stopped"
4	Interface an, Zustand = "Pre-Operational"
8	Interface an, Zustand = "Save-Operational"
12	Interface an, Zustand = "Operational"

Dieser Code wird im Datenwert bei Typ-Enum =16 in den Extended Settings kommuniziert.

D: IDs für ReadTEDSdataEntry

ID Property name and Description

0 Placeholder in Dictionary, points to first entry index (Sonderwert: nur NextID auszuwerten)

1 TEMPLATE ID

2 Separator

3 Select Case—Physical Measurand

4 Select Case—Full-Scale Electrical Value Precision

5..9 reserviert f. weitere Select-cases u. Sonder-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
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: Inbetriebnahme des GSV-6

Voraussetzungen zur Inbetriebnahme sind von GSV-6CPU sind:

a) Versorgungsspannung 3,7V …5.0V an Pin 14VCC_IN

a) GND 0V an Pin 15;

b) Pin13 „Supply Warnung“ mit Pin14 “VCC_IN” verbunden

c) UART2_RX und UART2_TX (3,3V TTL Pegel) ggfs mit Pegelumsetzer beschalten

Die folgende Sequenz protokolliert das Verhalten ab dem Einschalten.

Die GSV6CPU sendet zunächst die Messwerte aller 6 Kanäle mit der konfigurierten Datenfrequenz. Die vom GSV-6 gesendeten Daten sind rot dargestellt. Die Kommandos, die zum GSV-6 gesendet werden, sind blau dargestellt.

```
Hinweis: nach dem Einschalten werden 6 Kanäle übertragen mit 100Hz:
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

```
Durch die Anwendung des Kommandos StopTransmission wird die Datenübertragung angehalten:
AA 90 23 85
AA 50 00 85
```

GetValue

```
Durch die Anwendung des Kommandos GetValue wird ein Datenframe mit 6 Kanälen angefordert
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
```

GetAll 01

```
Durch die Anwendung des Kommandos GetAll werden Defaulteinstellungen geladen. In der aktuellen
Version wird die Anzahl der Kanäle auf 1 zurückgesetzt. Nicht anwenden!
AA 91 09 01 85
AA 50 00 85
AA 10 B0 BE 05 9D D8 85
AA 10 B0 BE 05 7C 3E 85
AA 10 B0 BE 05 9D D8 85
AA 10 B0 BE 05 7C 3E 85
AA 10 B0 BE 05 6B 71 85
AA 10 B0 BE 05 7C 3E 85
AA 10 B0 BE 05 6B 71 85
AA 10 B0 BE 05 8D 0B 85
AA 10 B0 BE 05 7C 3E 85
AA 10 B0 BE 05 7C 3E 85
```

Stop Transmission

```
AA 90 23 85
AA 10 B0 BE 05 D0 3E 85
AA 50 00 85
```

GetValue

```
AA 90 3B 85
AA 10 B0 BE 06 45 D9 85
```