

**MEGSV.DLL Programmierhandbuch**  
**zum**  
**Gleichspannungsmeßverstärker**  
**GSV-2**





# Inhaltsverzeichnis

<b>1</b>	<b>Beschreibung .....</b>	<b>9</b>
1.1	Installation .....	9
1.2	Anbindung an Programmiersprachen.....	9
1.2.1	C und C++.....	9
1.2.2	Delphi™ .....	10
1.2.3	Basic .....	10
1.3	Zeitliches Verhalten der Befehlsübermittlung.....	10
<b>2</b>	<b>Kategorischer Index.....</b>	<b>11</b>
2.1	Schnittstellenzugriff und Information .....	11
2.2	Datenfluß .....	12
2.3	Grundeinstellungen .....	12
2.4	Zustandsverwaltung .....	13
2.5	Anzeige .....	13
2.6	Weiterführende Einstellungen .....	14
2.7	Dehnungsindikator-Einstellungen .....	15
2.8	GSV-3 Einstellungen .....	15
2.9	GSV-2.1 Einstellungen .....	16
<b>3</b>	<b>Beschreibung der Funktionen.....</b>	<b>18</b>
3.1	GSVversion ( <i>Index 1</i> ) .....	22
3.2	GSVmodel ( <i>Index 2</i> ) .....	24
3.3	GSVrevision ( <i>Index 5</i> ) .....	26
3.4	GSVabortActivate ( <i>Index 7</i> ) .....	28
3.5	GSVgetLocalBaudRate ( <i>Index 8</i> ).....	30
3.6	GSVactivateExtended ( <i>Index 9</i> ).....	32
3.7	GSVactivate ( <i>Index 10</i> ).....	34
3.8	GSVrelease ( <i>Index 11</i> ).....	36
3.9	GSVflushBuffer ( <i>Index 13</i> ).....	38
3.10	GSVgetValues ( <i>Index 17</i> ) .....	40
3.11	GSVreceived ( <i>Index 20</i> ).....	42
3.12	GSVread ( <i>Index 21</i> ) .....	44
3.13	GSVreadMultiple ( <i>Index 22</i> ).....	46
3.14	GSVreadStatus ( <i>Index 25</i> ).....	48

3.15	<b>GSVreadStatusMultiple</b> ( <i>Index 26</i> ) .....	50
3.16	<b>GSVgetOptionsCode</b> ( <i>Index 30</i> ) .....	52
3.17	<b>GSVgetOptionsExtension3</b> ( <i>Index 31</i> ) .....	54
3.18	<b>GSVgetOptionsLinear</b> ( <i>Index 32</i> ).....	56
3.19	<b>GSVgetOptionsExtension21</b> ( <i>Index 33</i> ) .....	58
3.20	<b>GSVgetOptionsSleepMode</b> ( <i>Index 34</i> ) .....	60
3.21	<b>GSVgetOptionsCommandTest</b> ( <i>Index 35</i> ).....	62
3.22	<b>GSVisBipol</b> ( <i>Index 50</i> ) .....	64
3.23	<b>GSVgetFreq</b> ( <i>Index 51</i> ) .....	66
3.24	<b>GSVgetGain</b> ( <i>Index 52</i> ).....	68
3.25	<b>GSVgetChannel</b> ( <i>Index 53</i> ).....	70
3.26	<b>GSVgetModeLock</b> ( <i>Index 55</i> ) .....	72
3.27	<b>GSVsetModeLinear</b> ( <i>Index 56</i> ).....	74
3.28	<b>GSVgetModeLinear</b> ( <i>Index 57</i> ).....	76
3.29	<b>GSVsetModeAverage</b> ( <i>Index 58</i> ).....	78
3.30	<b>GSVgetModeAverage</b> ( <i>Index 59</i> ).....	80
3.31	<b>GSVsetModeText</b> ( <i>Index 62</i> ) .....	82
3.32	<b>GSVgetModeText</b> ( <i>Index 63</i> ) .....	84
3.33	<b>GSVsetModeMax</b> ( <i>Index 64</i> ).....	86
3.34	<b>GSVgetModeMax</b> ( <i>Index 65</i> ).....	88
3.35	<b>GSVsetModeLog</b> ( <i>Index 66</i> ) .....	90
3.36	<b>GSVgetModeLog</b> ( <i>Index 67</i> ).....	92
3.37	<b>GSVsetModeWindow</b> ( <i>Index 68</i> ).....	94
3.38	<b>GSVgetModeWindow</b> ( <i>Index 69</i> ).....	96
3.39	<b>GSVhasLCD</b> ( <i>Index 70</i> ).....	98
3.40	<b>GSVhasADC</b> ( <i>Index 71</i> ) .....	100
3.41	<b>GSVhasUII</b> ( <i>Index 72</i> ) .....	102
3.42	<b>GSVisSI</b> ( <i>Index 73</i> ).....	104
3.43	<b>GSVisWL</b> ( <i>Index 74</i> ).....	106
3.44	<b>GSVhasAF</b> ( <i>Index 75</i> ) .....	108
3.45	<b>GSVsetBridgeType</b> ( <i>Index 80</i> ) .....	110
3.46	<b>GSVgetBridgeType</b> ( <i>Index 81</i> ) .....	112
3.47	<b>GSVresetStatus</b> ( <i>Index 100</i> ).....	114

3.48	<b>GSVgetScale</b> ( <i>Index 101</i> ) .....	116
3.49	<b>GSVgetZero</b> ( <i>Index 102</i> ).....	118
3.50	<b>GSVgetControl</b> ( <i>Index 103</i> ).....	120
3.51	<b>GSVgetOffset</b> ( <i>Index 104</i> ) .....	122
3.52	<b>GSVwriteScale</b> ( <i>Index 105</i> ).....	124
3.53	<b>GSVwriteZero</b> ( <i>Index 106</i> ).....	126
3.54	<b>GSVwriteControl</b> ( <i>Index 107</i> ).....	128
3.55	<b>GSVwriteOffset</b> ( <i>Index 108</i> ) .....	130
3.56	<b>GSVgetAll</b> ( <i>Index 109</i> ) .....	132
3.57	<b>GSVsaveAll</b> ( <i>Index 110</i> ).....	134
3.58	<b>GSVsetCal</b> ( <i>Index 111</i> ) .....	136
3.59	<b>GSVsetZero</b> ( <i>Index 112</i> ).....	138
3.60	<b>GSVsetScale</b> ( <i>Index 113</i> ) .....	140
3.61	<b>GSVsetOffset</b> ( <i>Index 114</i> ).....	142
3.62	<b>GSVDispSetUnit</b> ( <i>Index 115</i> ).....	144
3.63	<b>GSVDispSetNorm</b> ( <i>Index 116</i> ) .....	146
3.64	<b>GSVDispSetDPoint</b> ( <i>Index 117</i> ) .....	148
3.65	<b>GSVsetFreq</b> ( <i>Index 118</i> ).....	150
3.66	<b>GSVsetGain</b> ( <i>Index 119</i> ).....	152
3.67	<b>GSVsetBipolar</b> ( <i>Index 120</i> ).....	154
3.68	<b>GSVsetUnipolar</b> ( <i>Index 121</i> ) .....	156
3.69	<b>GSVDispGetNorm</b> ( <i>Index 126</i> ).....	158
3.70	<b>GSVDispGetUnit</b> ( <i>Index 127</i> ) .....	160
3.71	<b>GSVDispGetDPoint</b> ( <i>Index 128</i> ).....	162
3.72	<b>GSVswitch</b> ( <i>Index 129</i> ).....	164
3.73	<b>GSVgetSerialNo</b> ( <i>Index 131</i> ).....	166
3.74	<b>GSVsetThreshold</b> ( <i>Index 132</i> ).....	168
3.75	<b>GSVgetThreshold</b> ( <i>Index 133</i> ) .....	170
3.76	<b>GSVsetChannel</b> ( <i>Index 134</i> ).....	172
3.77	<b>GSVstopTransmit</b> ( <i>Index 135</i> ) .....	174
3.78	<b>GSVstartTransmit</b> ( <i>Index 136</i> ).....	176
3.79	<b>GSVclearBuffer</b> ( <i>Index 137</i> ).....	178
3.80	<b>GSVsetMode</b> ( <i>Index 138</i> ).....	180

3.81	<b>GSVgetMode</b> ( <i>Index 139</i> ).....	182
3.82	<b>GSVgetEquipment</b> ( <i>Index 141</i> ) .....	184
3.83	<b>GSVfirmwareVersion</b> ( <i>Index 143</i> ).....	186
3.84	<b>GSVsetGageFactor</b> ( <i>Index 144</i> ).....	188
3.85	<b>GSVgetGageFactor</b> ( <i>Index 145</i> ).....	190
3.86	<b>GSVsetPoisson</b> ( <i>Index 146</i> ).....	192
3.87	<b>GSVgetPoisson</b> ( <i>Index 147</i> ).....	194
3.88	<b>GSVsetBridge</b> ( <i>Index 148</i> ).....	196
3.89	<b>GSVgetBridge</b> ( <i>Index 149</i> ).....	198
3.90	<b>GSVgetRange</b> ( <i>Index 151</i> ).....	200
3.91	<b>MEgetOffsetWait</b> ( <i>Index 153</i> ) .....	202
3.92	<b>GSVgetOptions</b> ( <i>Index 154</i> ) .....	204
3.93	<b>GSVgetValue</b> ( <i>Index 159</i> ) .....	206
3.94	<b>GSVclearMaxValue</b> ( <i>Index 160</i> ) .....	208
3.95	<b>GSVDispSetDigits</b> ( <i>Index 161</i> ).....	210
3.96	<b>GSVDispGetDigits</b> ( <i>Index 162</i> ).....	212
3.97	<b>GSVunlockUII</b> ( <i>Index 163</i> ).....	214
3.98	<b>GSVlockUII</b> ( <i>Index 164</i> ) .....	216
3.99	<b>GSVgetLastError</b> ( <i>Index 166</i> ).....	218
3.100	<b>GSVsetSecondThreshold</b> ( <i>Index 167</i> ).....	220
3.101	<b>GSVgetSecondThreshold</b> ( <i>Index 168</i> ).....	222
3.102	<b>GSVgetDeviceType</b> ( <i>Index 169</i> ) .....	224
3.103	<b>GSVDispCalcNorm</b> ( <i>Index 170</i> ).....	226
3.104	<b>GSVsetTxMode</b> ( <i>Index 228</i> ).....	228
3.105	<b>GSVgetTxMode</b> ( <i>Index 229</i> ) .....	230
3.106	<b>GSVsetBaud</b> ( <i>Index 230</i> ) .....	232
3.107	<b>GSVgetBaud</b> ( <i>Index 231</i> ).....	234
3.108	<b>GSVsetSlowRate</b> ( <i>Index 234</i> ).....	236
3.109	<b>GSVgetSlowRate</b> ( <i>Index 235</i> ).....	238
3.110	<b>GSVsetSpecialMode</b> ( <i>Index 236</i> ) .....	240
3.111	<b>GSVgetSpecialMode</b> ( <i>Index 237</i> ).....	242
3.112	<b>GSVwriteSamplingRate</b> ( <i>Index 238</i> ).....	244
3.113	<b>GSVreadSamplingRate</b> ( <i>Index 239</i> ).....	246

<b>3.114</b>	<b>GSVsetCanSetting</b> ( <i>Index 240</i> ).....	<b>248</b>
<b>3.115</b>	<b>GSVgetCanSetting</b> ( <i>Index 241</i> ).....	<b>250</b>
<b>3.116</b>	<b>GSVsetAnalogFilter</b> ( <i>Index 244</i> ).....	<b>252</b>
<b>3.117</b>	<b>GSVgetAnalogFilter</b> ( <i>Index 245</i> ).....	<b>254</b>
<b>3.118</b>	<b>GSVisCommandAvailable</b> ( <i>Index 247</i> ).....	<b>256</b>
<b>3.119</b>	<b>GSVsetNoiseCutThreshold</b> ( <i>Index 248</i> ).....	<b>258</b>
<b>3.120</b>	<b>GSVgetNoiseCutThreshold</b> ( <i>Index 249</i> ).....	<b>260</b>
<b>3.121</b>	<b>GSVsetAutoZeroCounter</b> ( <i>Index 250</i> ) .....	<b>262</b>
<b>3.122</b>	<b>GSVgetAutoZeroCounter</b> ( <i>Index 251</i> ) .....	<b>264</b>
<b>3.123</b>	<b>GSVsetUserTextChar</b> ( <i>Index 252</i> ) .....	<b>266</b>
<b>3.124</b>	<b>GSVgetRanges</b> ( <i>Index 262</i> ).....	<b>268</b>
<b>3.125</b>	<b>GSVsetRanges</b> ( <i>Index 263</i> ).....	<b>270</b>
<b>3.126</b>	<b>GSVgetTxModeConfig</b> ( <i>Index 357</i> ).....	<b>272</b>
<b>3.127</b>	<b>GSVsetTxModeTransmit4</b> ( <i>Index 358</i> ).....	<b>274</b>
<b>3.128</b>	<b>GSVgetTxModeTransmit4</b> ( <i>Index 359</i> ).....	<b>276</b>
<b>3.129</b>	<b>GSVsetTxModeRepeat3</b> ( <i>Index 360</i> ) .....	<b>278</b>
<b>3.130</b>	<b>GSVgetTxModeRepeat3</b> ( <i>Index 361</i> ) .....	<b>280</b>
<b>3.131</b>	<b>GSVsetTxModeTransmit5</b> ( <i>Index 362</i> ).....	<b>282</b>
<b>3.132</b>	<b>GSVgetTxModeTransmit5</b> ( <i>Index 363</i> ).....	<b>284</b>
<b>3.133</b>	<b>GSVsetTxModeReadOnly</b> ( <i>Index 364</i> ).....	<b>286</b>
<b>3.134</b>	<b>GSVgetTxModeReadOnly</b> ( <i>Index 365</i> ).....	<b>288</b>
<b>3.135</b>	<b>GSVsetSpecialModeSlow</b> ( <i>Index 372</i> ).....	<b>290</b>
<b>3.136</b>	<b>GSVgetSpecialModeSlow</b> ( <i>Index 373</i> ) .....	<b>292</b>
<b>3.137</b>	<b>GSVsetSpecialModeAverage</b> ( <i>Index 374</i> ).....	<b>294</b>
<b>3.138</b>	<b>GSVgetSpecialModeAverage</b> ( <i>Index 375</i> ).....	<b>296</b>
<b>3.139</b>	<b>GSVsetSpecialModeFilter</b> ( <i>Index 376</i> ).....	<b>298</b>
<b>3.140</b>	<b>GSVgetSpecialModeFilter</b> ( <i>Index 377</i> ).....	<b>300</b>
<b>3.141</b>	<b>GSVsetSpecialModeMax</b> ( <i>Index 378</i> ).....	<b>302</b>
<b>3.142</b>	<b>GSVgetSpecialModeMax</b> ( <i>Index 379</i> ).....	<b>304</b>
<b>3.143</b>	<b>GSVsetSpecialModeFilterAuto</b> ( <i>Index 380</i> ) .....	<b>306</b>
<b>3.144</b>	<b>GSVgetSpecialModeFilterAuto</b> ( <i>Index 381</i> ).....	<b>308</b>
<b>3.145</b>	<b>GSVsetSpecialModeFilterOrder5</b> ( <i>Index 382</i> ) .....	<b>310</b>
<b>3.146</b>	<b>GSVgetSpecialModeFilterOrder5</b> ( <i>Index 383</i> ) .....	<b>312</b>

3.147	<b>GSVsetSpecialModeSleep</b> ( <i>Index 384</i> ).....	314
3.148	<b>GSVgetSpecialModeSleep</b> ( <i>Index 385</i> ) .....	316
3.149	<b>GSVsetSpecialModeAutoZero</b> ( <i>Index 400</i> ) .....	318
3.150	<b>GSVgetSpecialModeAutoZero</b> ( <i>Index 401</i> ) .....	320
3.151	<b>GSVsetSpecialModeNoiseCut</b> ( <i>Index 402</i> ) .....	322
3.152	<b>GSVgetSpecialModeNoiseCut</b> ( <i>Index 403</i> ) .....	324
3.153	<b>GSVreadSamplingFrequency</b> ( <i>Index 404</i> ).....	326
3.154	<b>GSVreadSamplingFactor</b> ( <i>Index 405</i> ).....	328
3.155	<b>GSVsetBaudRate</b> ( <i>Index 406</i> ) .....	330
3.156	<b>GSVgetBaudRate</b> ( <i>Index 407</i> ).....	332
3.157	<b>GSVsetCanBaudRate</b> ( <i>Index 408</i> ).....	334
3.158	<b>GSVgetCanBaudRate</b> ( <i>Index 409</i> ) .....	336
3.159	<b>GSVsetCanID</b> ( <i>Index 412</i> ).....	338
3.160	<b>GSVgetCanID</b> ( <i>Index 413</i> ).....	340
3.161	<b>GSVsetCanBaud</b> ( <i>Index 414</i> ) .....	342
3.162	<b>GSVgetCanBaud</b> ( <i>Index 415</i> ).....	344
3.163	<b>GSVisCanAvailable</b> ( <i>Index 417</i> ).....	346
3.164	<b>GSVsetCanActive</b> ( <i>Index 418</i> ).....	348
3.165	<b>GSVgetCanActive</b> ( <i>Index 419</i> ).....	350
3.166	<b>GSVsetCanMode20B</b> ( <i>Index 420</i> ) .....	352
3.167	<b>GSVgetCanMode20B</b> ( <i>Index 421</i> ) .....	354
3.168	<b>GSVsetUserText</b> ( <i>Index 448</i> ).....	356
4	<b>Demoprogramm</b> .....	358
4.1	<b>Quelltext in C</b> .....	358
4.2	<b>Quelltext in Delphi™</b> .....	362

Stand: 03.09.2009, MEGSV.DLL Version 1.50



# 1 Beschreibung

Zur Programmierung der GSV-Baugruppe unter Windows™ (Warenzeichen der Microsoft Corp.) steht eine 32-bit Dynamic Link Library (DLL) zur Verfügung. Sie enthält Funktionen, die meist weitgehend den vom GSV bekannten Befehlen entsprechen, oft jedoch zusätzliche Funktionalität beinhalten oder mehrere Befehle kombinieren, um die Programmierung zu vereinfachen. Sie enthält auch Funktionen zum Öffnen einer seriellen Schnittstelle mit gleichzeitigem Aufbau der Kommunikation mit dem GSV und zum Schließen einer so geöffneten seriellen Schnittstelle. Die Funktionen der DLL können über ihren Index importiert werden.

Die aktivierten Schnittstellen und GSVs werden für jedes Programm, das die DLL verwendet, separat verwaltet. Die Benutzung einer Schnittstelle (mit GSV), die von einem anderen Prozeß aktiviert wurde, ist nicht möglich. Verschiedene Threads eines Prozesses können ihre aktivierten Schnittstellen gemeinsam parallel bzw. pseudo-parallel verwenden. Die Funktionsfähigkeit auf Multiprozessorsystemen kann zur Zeit nicht bestätigt werden.

## 1.1 Installation

Bei der Installation wird die DLL im System-Verzeichnis von Windows™ abgelegt. Dort steht sie allen Programmen zur Verfügung. Die DLL kann jedoch auch in das Verzeichnis kopiert werden, in dem sich das aufrufende Programm befindet (.EXE Datei), um sicherzustellen, daß das Programm immer genau eine bestimmte Version der DLL verwendet. Dieses Vorgehen kann wesentliche Vorteile und schwere Nachteile zur Folge haben.

Die weiteren installierten Dateien befassen sich mit der Anbindung an verschiedene Programmiersprachen und den Demoprogrammen (siehe auch Kapitel 4).

## 1.2 Anbindung an Programmiersprachen

### 1.2.1 C und C++

Die Anbindung an die Programmiersprachen C und C++ erfolgt mit der Header-Datei MEGSV.H sowie einer Link-Library, die zum Linken als Bibliothek mit angegeben wird und die Verbindung zur DLL herstellt. Link-Libraries für viele Compiler und Entwicklungsumgebungen können mit Hilfe dazugehöriger Hilfsprogramme direkt aus der DLL erstellt werden. Für Microsoft® VisualC++® 6.0 (Eingetragene Warenzeichen der Microsoft Corp.) liegt eine derartige Bibliothek bereits bei (MEGSV.LIB). Die Anwendung kann dem Demoprogramm und der zugehörigen Stapelverarbeitungsdatei DEMOMSCV.BAT entnommen werden.

Das Erzeugen einer Link-Library durch Hilfsprogramme wird am Beispiel von Borland® C++Builder™ - (eingetragenes) Warenzeichen der Borland Software Corp. - gezeigt in der Stapelverarbeitungsdatei DEMOBC.BAT.

### 1.2.2 Delphi™

Die Anbindung an Delphi™ (Warenzeichen der Borland Software Corp.) erfolgt durch eine entsprechende Unit, die im Quellcode vorliegt. Dieser muß kompiliert werden um eine Unit in binärer Form zu erhalten. Dies geschieht automatisch bei der Erstellung des betreffenden Demoprogramms mit Hilfe der Stapelverarbeitungsdatei DEMOPAS.BAT. Die Unit MEGSV wird in einem Programm verwendet durch das Schlüsselwort **uses** (siehe Dokumentation zu Delphi™).

### 1.2.3 Basic

Die Anbindung an Microsoft® VisualBasic® (Eingetragene Warenzeichen der Microsoft Corp.) erfolgt durch das Modul MEGSV.BAS, das die benötigten Deklarationen enthält. Diese Datei wird einem VisualBasic®-Projekt hinzugefügt. Danach stehen die Deklarationen und damit auch die Funktionen global zur Verfügung.

## 1.3 Zeitliches Verhalten der Befehlsübermittlung

Diejenigen Funktionen der Library (DLL), die Befehle über eine serielle Schnittstelle an die GSV-Baugruppe senden, versuchen durch ihre Implementierung zu erreichen, daß die Übermittlung dieser Befehle bis zur Rückkehr vom Funktionsaufruf durch das Betriebssystem soweit wie möglich vorangetrieben wird. Im günstigsten Fall wird daher der betreffende Befehl bei der Rückkehr vom Funktionsaufruf bereits physikalisch über das Verbindungskabel gesendet worden sein. Es muß jedoch, insbesondere bei Funktionen die nur (oder als letzte Aktion) Befehle zur GSV-Baugruppe senden, berücksichtigt werden, daß verschiedene Gründe, u.a. die Vielfalt der Hardware- und Treiber-Implementierungen, es unmöglich machen, sich im allgemeinen auf das zeitliche Verhalten des Sendevorgangs über eine serielle Schnittstelle zu verlassen. Dies ist bei der Programmentwicklung immer zu berücksichtigen, auch bei der Verwendung von Funktionen, die durch ihre Funktionalität, ihre Implementierung und/oder ihren Zweck anderes vermuten lassen könnten (siehe z.B. **GSVgetValues** auf Seite 40).

## 2 Kategorischer Index

Index der Funktionen nach ihrem Anwendungsbereich.

### 2.1 Schnittstellenzugriff und Information

GSVactivate	Seite 34
GSVactivateExtended	Seite 32
GSVrelease	Seite 36
GSVversion	Seite 22
GSVrevision	Seite 26
GSVgetLocalBaudRate	Seite 30
GSVmodel	Seite 24
GSVgetDeviceType	Seite 224
GSVgetEquipment	Seite 184
GSVhasLCD	Seite 98
GSVhasADC	Seite 100
GSVhasUII	Seite 102
GSVisSI	Seite 104
GSVisWL	Seite 106
GSVhasAF	Seite 108
GSVgetSerialNo	Seite 166
GSVfirmwareVersion	Seite 186
GSVgetOptions	Seite 204
GSVgetOptionsCode	Seite 52
GSVgetOptionsLinear	Seite 56
GSVgetOptionsExtension3	Seite 54
GSVgetOptionsExtension21	Seite 58
GSVgetOptionsSleepMode	Seite 60
GSVgetOptionsCommandTest	Seite 62
GSVisCommandAvailable	Seite 256
GSVisCanAvailable	Seite 346
GSVgetRange	Seite 200

GSVgetRanges	Seite 268
MEgetOffsetWait	Seite 202
GSVgetLastError	Seite 218

## 2.2 Datenfluß

GSVreceived	Seite 42
GSVread	Seite 44
GSVreadMultiple	Seite 46
GSVreadStatus	Seite 48
GSVreadStatusMultiple	Seite 50
GSVstopTransmit	Seite 174
GSVstartTransmit	Seite 176
GSVclearBuffer	Seite 178
GSVflushBuffer	Seite 38
GSVresetStatus	Seite 114
GSVclearMaxValue	Seite 208
GSVgetValue	Seite 206
GSVgetValues	Seite 40

## 2.3 Grundeinstellungen

GSVsetFreq	Seite 150
GSVsetCal	Seite 136
GSVsetScale	Seite 140
GSVsetZero	Seite 138
GSVsetOffset	Seite 142
GSVsetGain	Seite 152
GSVsetBipolar	Seite 154
GSVsetUnipolar	Seite 156
GSVsetChannel	Seite 172
GSVsetRanges	Seite 270
GSVsetThreshold	Seite 168

GSVsetModeWindow	Seite 94
GSVgetFreq	Seite 66
GSVgetGain	Seite 68
GSVisBipol	Seite 64
GSVgetChannel	Seite 70
GSVgetThreshold	Seite 170
GSVgetModeWindow	Seite 96

## 2.4 Zustandsverwaltung

GSVgetScale	Seite 116
GSVgetZero	Seite 118
GSVgetOffset	Seite 122
GSVgetControl	Seite 120
GSVwriteScale	Seite 124
GSVwriteZero	Seite 126
GSVwriteOffset	Seite 130
GSVwriteControl	Seite 128
GSVsaveAll	Seite 134
GSVgetAll	Seite 132

## 2.5 Anzeige

GSVDispSetNorm	Seite 146
GSVDispSetDPoint	Seite 148
GSVDispSetUnit	Seite 144
GSVDispGetNorm	Seite 158
GSVDispGetDPoint	Seite 162
GSVDispGetUnit	Seite 160
GSVDispSetDigits	Seite 210
GSVDispGetDigits	Seite 212
GSVDispCalcNorm	Seite 226

## 2.6 Weiterführende Einstellungen

GSVsetMode	Seite 180
GSVsetModeText	Seite 82
GSVsetModeMax	Seite 86
GSVsetModeLog	Seite 90
GSVsetModeAverage	Seite 78
GSVsetModeLinear	Seite 74
GSVgetMode	Seite 182
GSVgetModeText	Seite 84
GSVgetModeMax	Seite 88
GSVgetModeLog	Seite 92
GSVgetModeAverage	Seite 80
GSVgetModeLinear	Seite 76
GSVgetModeLock	Seite 72
GSVswitch	Seite 164
GSVlockUII	Seite 216
GSVunlockUII	Seite 214
GSVsetTxMode	Seite 228
GSVsetTxModeTransmit4	Seite 274
GSVsetTxModeRepeat3	Seite 278
GSVsetTxModeTransmit5	Seite 282
GSVsetTxModeReadOnly	Seite 286
GSVgetTxMode	Seite 230
GSVgetTxModeConfig	Seite 272
GSVgetTxModeTransmit4	Seite 276
GSVgetTxModeRepeat3	Seite 280
GSVgetTxModeTransmit5	Seite 284
GSVgetTxModeReadOnly	Seite 288
GSVsetCanSetting	Seite 248
GSVsetCanActive	Seite 348

GSVsetCanID	Seite 338
GSVsetCanBaud	Seite 342
GSVsetCanBaudRate	Seite 334
GSVsetCanMode20B	Seite 352
GSVgetCanSetting	Seite 250
GSVgetCanActive	Seite 350
GSVgetCanID	Seite 340
GSVgetCanBaud	Seite 344
GSVgetCanBaudRate	Seite 336
GSVgetCanMode20B	Seite 354

## 2.7 Dehnungsindikator-Einstellungen

GSVsetBridge	Seite 196
GSVsetBridgeType	Seite 110
GSVsetGageFactor	Seite 188
GSVsetPoisson	Seite 192
GSVgetBridge	Seite 198
GSVgetBridgeType	Seite 112
GSVgetGageFactor	Seite 190
GSVgetPoisson	Seite 194

## 2.8 GSV-3 Einstellungen

GSVsetBaud	Seite 232
GSVsetSlowRate	Seite 236
GSVsetSpecialMode	Seite 240
GSVsetSpecialModeSlow	Seite 290
GSVsetSpecialModeSleep	Seite 314
GSVsetSpecialModeFilter	Seite 298
GSVsetSpecialModeMax	Seite 302
GSVsetSpecialModeFilterAuto	Seite 306
GSVsetSpecialModeFilterOrder5	Seite 310

GSVwriteSamplingRate	Seite 244
GSVsetBaudRate	Seite 330
GSVgetBaud	Seite 234
GSVgetSlowRate	Seite 238
GSVgetSpecialMode	Seite 242
GSVgetSpecialModeSlow	Seite 292
GSVgetSpecialModeSleep	Seite 316
GSVgetSpecialModeAverage	Seite 296
GSVgetSpecialModeFilter	Seite 300
GSVgetSpecialModeMax	Seite 304
GSVgetSpecialModeFilterAuto	Seite 308
GSVgetSpecialModeFilterOrder5	Seite 312
GSVreadSamplingRate	Seite 246
GSVreadSamplingFrequency	Seite 326
GSVreadSamplingFactor	Seite 328
GSVgetBaudRate	Seite 332

## 2.9 GSV-2.1 Einstellungen

GSVsetSecondThreshold	Seite 220
GSVsetAnalogFilter	Seite 252
GSVsetSpecialModeAutoZero	Seite 318
GSVsetAutoZeroCounter	Seite 262
GSVsetSpecialModeNoiseCut	Seite 322
GSVsetNoiseCutThreshold	Seite 258
GSVsetUserTextChar	Seite 266
GSVsetUserText	Seite 356
GSVgetSecondThreshold	Seite 222
GSVgetAnalogFilter	Seite 254
GSVgetSpecialModeAutoZero	Seite 320
GSVgetAutoZeroCounter	Seite 264
GSVgetSpecialModeNoiseCut	Seite 324
GSVgetNoiseCutThreshold	Seite 260





### 3 Beschreibung der Funktionen

Im folgenden werden die Befehle der Bibliothek MEGSV.DLL beschrieben. Dabei ist jeweils eine verkürzte Form der Funktionsdefinition bzw. -deklaration für die Programmiersprachen C, Delphi™ (Warenzeichen der Borland Software Corp.) und VB angegeben. Dies entspricht dem Inhalt der verfügbaren Dateien MEGSV.H (Header-Datei für C und C++), MEGSV.PAS (Delphi™-Unit) und MEGSV.BAS (Modul für VB) für die Programmierung mit der Bibliothek. Obwohl VisualBasic® (Eingetragenes Warenzeichen der Microsoft Corp.) zeilenorientiert ist, werden in diesem Handbuch die entsprechenden Deklarationen der Einfachheit halber mitunter in freier Form dargestellt!

Als Standardgröße von Datenworten wird die dem Betriebssystem entsprechende Standardgröße angenommen (32 Bit), daher ist bei VB zu beachten, daß alle Werte mit Standardgröße ausdrücklich als 32-bit Werte festgelegt werden müssen, da die Standardgröße von VB 16 Bit beträgt.

Die Nummer der seriellen Schnittstelle, die in den meisten Funktionen als Parameter auftritt, entspricht der Zahl die dem Präfix "COM" folgt, entsprechend der üblichen Namensvergabe für serielle Schnittstellen (z.B. "COM1" für Schnittstelle 1, "COM2" für Schnittstelle 2, usw.).

Bei Funktionen mit Angabe der Anwendbarkeit wird Bezug genommen auf die GSV-Modellnummer sowie, falls erforderlich, auf die Firmware Versionsnummer, ab der die Funktion verwendet werden kann (die Wirkung der Funktion kann jedoch noch anderweitig eingeschränkt sein). Die GSV-Modellnummer kann abgefragt werden mit der Funktion **GSVmodel**. Das Modell 1 des GSV (mit der Firmware Version 3.2-11 - siehe **GSVfirmwareVersion**) wird seit Mitte 2001 ausgeliefert. Das Modell 2 des GSV (mit der Firmware Version 3.3-4) wird seit Mitte 2002 ausgeliefert.

## Konstantendefinitionen für Fehlercodes:

C:	#define	GSV_OK	<i>Ganzzahl-Konstante</i>
	#define	GSV_ERROR	<i>Ganzzahl-Konstante</i>
	#define	GSV_TRUE	<i>Ganzzahl-Konstante</i>
Delphi™:	GSV_OK	=	<i>Ganzzahl-Konstante;</i>
	GSV_ERROR	=	<i>Ganzzahl-Konstante;</i>
	GSV_TRUE	=	<i>Ganzzahl-Konstante;</i>
VB:	GSV_OK	As Long =	<i>Ganzzahl-Konstante</i>
	GSV_ERROR	As Long =	<i>Ganzzahl-Konstante</i>
	GSV_TRUE	As Long =	<i>Ganzzahl-Konstante</i>

## Sprachspezifische Typdefinitionen:

In den Funktionsdefinitionen für Delphi™ werden folgende Typdefinitionen verwendet:

```
PDouble = ^Double;
PInteger = ^Integer;
PByte = ^Byte;
```

## Typdefinition für GSVactivateExtended:

```
C:          typedef struct          GSV_ACTIVATE_EXTENDED_
{
            long          actex_size;
            long          actex_buffersize;
            long          actex_flags;
            long          actex_baudrate;
        }          GSV_ACTIVATE_EXTENDED;
```

```
Delphi™:    GSV_ACTIVATE_EXTENDED = record
            actex_size          : LongInt;
            actex_buffersize   : LongInt;
            actex_flags        : LongInt;
            actex_baudrate     : LongInt;
        end;
```

```
VB:         Type          GSV_ACTIVATE_EXTENDED
            actex_size          As Long
            actex_buffersize   As Long
            actex_flags        As Long
            actex_baudrate     As Long
        End          Type
```

## Bedeutung der Felder der Datenstruktur:

**actex\_size**            Ganze Zahl.  
Größe der Datenstruktur, gesetzt durch die GSV\_ACTEX\_SIZE  
Konstante (bzw. Funktion).

**actex\_buffersize**    Ganze Zahl.  
Anzahl der maximal zwischenzuspeichernden Meßdatenwerte.

**actex\_flags**            Ganze Zahl.  
Kombination von GSV\_ACTEX\_FLAG\_xxx Konstanten, die festlegt,  
welche weiteren Felder verwendet werden.

**actex\_baudrate**        Ganze Zahl (Optional).  
Gewünschte Baudrate für die serielle Schnittstelle (nicht verwendet  
= gleiche Einstellung wie bei GSVactivate).

## Konstantendefinitionen für GSVactivateExtended:

C:           #define GSV\_ACTEX\_FLAG\_BAUDRATE       *Ganzzahl-Konstante*  
              #define GSV\_ACTEX\_FLAG\_WAKEUP       *Ganzzahl-Konstante*  
              #define GSV\_ACTEX\_FLAG\_HANDSHAKE    *Ganzzahl-Konstante*  
  
              #define GSV\_ACTEX\_SIZE    ((long)sizeof(GSV\_ACTIVATE\_EXTENDED))

Delphi™:       GSV\_ACTEX\_FLAG\_BAUDRATE   =       *Ganzzahl-Konstante;*  
                  GSV\_ACTEX\_FLAG\_WAKEUP     =       *Ganzzahl-Konstante;*  
                  GSV\_ACTEX\_FLAG\_HANDSHAKE =       *Ganzzahl-Konstante;*  
  
                  GSV\_ACTEX\_SIZE    =    SizeOf(GSV\_ACTIVATE\_EXTENDED);

VB:           GSV\_ACTEX\_FLAG\_BAUDRATE   As Long =   *Ganzzahl-Konstante*  
                  GSV\_ACTEX\_FLAG\_WAKEUP     As Long =   *Ganzzahl-Konstante*  
                  GSV\_ACTEX\_FLAG\_HANDSHAKE As Long =   *Ganzzahl-Konstante*  
  
                  GSV\_ACTEX\_SIZE( ) As Long

## Bedeutung der Konstanten:

Die Konstanten GSV\_ACTEX\_FLAG\_xxx legen in Gestalt einer Bitmaske fest, welche Felder der GSV\_ACTIVATE\_EXTENDED Datenstruktur **außer actex\_size, actex\_buffersize und actex\_flags** (diese werden immer benötigt) mit gültigen Daten belegt sind. Für nicht belegte Felder werden Defaultwerte verwendet.

Die Konstante GSV\_ACTEX\_FLAG\_WAKEUP steht nicht für ein Feld der GSV\_ACTIVATE\_EXTENDED Datenstruktur, sondern legt fest, daß ein besonderes Verfahren verwendet werden soll zum Beginn einer Kommunikation mit einer Baugruppe, die sich im Energiesparmodus befindet. Dabei wird die Übermittlung des nächsten Datenpaketes abgewartet. Da dies sehr lange dauern kann (im Falle eines Fehlers unendlich lange) kann in diesem Zusammenhang die Funktion **GSVabortActivate** in einem parallelen Thread verwendet werden, um einen Abbruch des Wartens zu erreichen.

Die Konstante GSV\_ACTEX\_FLAG\_HANDSHAKE steht nicht für ein Feld der GSV\_ACTIVATE\_EXTENDED Datenstruktur, sondern legt fest, daß die serielle Kommunikation mit der Baugruppe mit dem CTS/RTS-Handshake erfolgen soll.

Die Konstante GSV\_ACTEX\_SIZE dient zum setzen des Feldes **actex\_size** vor dem Aufruf der Funktion GSVactivateExtended. In **VB** ist dies **keine Konstante** sondern als **Funktion** implementiert, woraus die üblichen Einschränkungen folgen.

### 3.1 GSVversion

(Index 1)

#### **Beschreibung:**

Die Funktion GSVversion ermittelt die Versionsnummer der verwendeten Bibliothek MEGSV.DLL als 32-bit Ganzzahl. Dabei befindet sich die Hauptversionsnummer in den höherwertigen 16 Bit des Ergebnisses, während die Nebenversionsnummer in den niederwertigen 16 Bit liegt. Dadurch können Überprüfungen der Gesamtversionsnummer leicht programmiert werden.

**Verwendete GSV Befehle:** Keine.

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall muß damit gerechnet werden, daß die Bibliotheksdatei MEGSV.DLL beschädigt ist.

**Funktionsdefinition:**

C:                    long GSVversion ( void );

Delphi™:            GSVversion: LongInt;

VB:                    GSVversion ( ) As Long

**Rückgabewert:** Versionsnummer oder Fehlercode.

**Aufrufparameter:** Keine.

## 3.2 GSVmodel

(Index 2)

### **Beschreibung:**

Die Funktion GSVmodel ermittelt die Modellnummer der verwendeten GSV Baugruppe als Ganzzahl. Dadurch kann der Umfang der zur Verfügung stehenden Funktionalität bestimmt werden.

**Verwendete GSV Befehle:** Keine.

### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall liegt wahrscheinlich ein Problem beim Zugriff auf die Baugruppe vor.



## Funktionsdefinition:

C:                               int GSVmodel ( int **no** );

Delphi™:                       GSVmodel ( **no**: Integer ): Integer;

VB:                               GSVmodel ( ByVal **no** As Long ) As Long

**Rückgabewert:** Modellnummer oder Fehlercode.

## Aufrufparameter:

**no**                            Ganze Zahl im Bereich 1..1024.  
                                   Nummer der seriellen Schnittstelle des zu verwendenden GSV, diese muß  
                                   aktiviert sein.

### 3.3 GSVrevision

(Index 5)

#### **Beschreibung:**

Die Funktion GSVrevision ermittelt die Revisionsnummer und die Build-Nummer der verwendeten Bibliothek MEGSV.DLL als 32-bit Ganzzahl. Dabei befindet sich die Revisionsnummer in den höherwertigen 16 Bit des Ergebnisses, während die Build-Nummer in den niederwertigen 16 Bit liegt. Dadurch wird die Genauigkeit der Funktion **GSVversion** erweitert.

**Verwendete GSV Befehle:** Keine.

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall muß damit gerechnet werden, daß die Bibliotheksdatei MEGSV.DLL beschädigt ist.

**Funktionsdefinition:**

C:                    long GSVrevision ( void );

Delphi™:            GSVrevision: LongInt;

VB:                    GSVrevision ( ) As Long

**Rückgabewert:** Revisions- und Build-Nummer oder Fehlercode.

**Aufrufparameter:** Keine.

### 3.4 GSVabortActivate

(Index 7)

#### Beschreibung:

Die Funktion GSVabortActivate bricht einen wartenden Aufruf von **GSVactivateExtended** (mit dem Flag GSV\_ACTEX\_FLAG\_WAKEUP, siehe Anfang von Abschnitt 3) ab. Zu diesem Zweck muß GSVabortActivate in einem parallelen Thread aufgerufen werden.

**Verwendete GSV Befehle:** Keine.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall kann der Aufrufparameter ungültig sein oder es liegt nicht der erwartete Zustand vor.

## Funktionsdefinition:

C:                int GSVabortActivate ( int **no** );

Delphi™:        GSVabortActivate ( **no**: Integer ): Integer;

VB:              GSVabortActivate ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**              Ganze Zahl im Bereich 1..1024.  
                   Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.5 GSVgetLocalBaudRate

(Index 8)

#### **Beschreibung:**

Die Funktion GSVgetLocalBaudRate ermittelt die Geschwindigkeitseinstellung der seriellen Schnittstelle des Rechners (in Baud, d.h. Bit/s). Dadurch können Programme eine eventuelle Begrenzung der Daten ermitteln, die pro Sekunde von der angeschlossenen Baugruppe empfangen werden können.

**Verwendete GSV Befehle:** Keine.

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall liegt wahrscheinlich ein Problem beim Zugriff auf die serielle Schnittstelle vor.

## Funktionsdefinition:

C:     long GSVgetLocalBaudRate ( int **no** );

Delphi™:   GSVgetLocalBaudRate ( **no**: Integer ): LongInt;

VB:           GSVgetLocalBaudRate ( ByVal **no** As Long ) As Long

**Rückgabewert:** Baudrate oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                 Nummer der seriellen Schnittstelle des zu verwendenden GSV, diese muß  
                 aktiviert sein.

## 3.6 GSVactivateExtended

(Index 9)

### Beschreibung:

Die Funktion GSVactivateExtended führt die gleiche Funktion aus wie GSVactivate (siehe 3.7), verfügt jedoch über eine Datenstruktur als Parameter, wodurch eine flexiblere und umfangreichere Übergabe von besonderen Einstellungen möglich ist. Die Datenstruktur und die benötigten Konstanten sind am Anfang des Kapitels bei Konstantendefinitionen und Typdefinitionen beschrieben.

Bei der Aktivierung kann angegeben werden, wieviele Meßwerte des GSV die Bibliothek maximal zwischenspeichern soll, falls das Programm, das die Schnittstelle aktiviert hat, die Meßwerte nicht ausreichend schnell abholt. Ist der Wert kleiner als 1, wird 1 angenommen. Der maximal zulässige Wert hängt vom verfügbaren Speicher ab.

**Verwendete GSV Befehle:** 42, 3, 28, 26, ggf. 54, 64, 136, 137  
(siehe GSV Bedienungsanleitung).

### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört, die Schnittstelle bereits in Benutzung oder das Sperren des UII unmöglich sein.



## Funktionsdefinition:

C:           int GSVactivateExtended ( int **no**, GSV\_ACTIVATE\_EXTENDED \***actex** );

Delphi™:    GSVactivateExtended ( **no**: Integer;  
  var **actex**: GSV\_ACTIVATE\_EXTENDED );  
  Integer;

VB:           GSVactivateExtended ( ByVal **no** As Long,  
  ByRef **actex** As GSV\_ACTIVATE\_EXTENDED )  
  As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                  Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**actex**        Datenstruktur (in C: Zeiger auf Datenstruktur) des Typs  
                  GSV\_ACTIVATE\_EXTENDED.  
                  Enthält Daten für die gewünschten Einstellungen, siehe Typdefinition am  
                  Beginn des Kapitels.

### 3.7 GSVactivate

(Index 10)

#### Beschreibung:

Die Funktion GSVactivate aktiviert eine serielle Schnittstelle mit angeschlossener GSV-Baugruppe, d.h. die Schnittstelle wird geöffnet, konfiguriert und ein Kommunikationsversuch mit dem GSV durchgeführt. Die Schnittstelle bleibt nur geöffnet, wenn die Kommunikation erfolgreich aufgebaut werden kann.

GSVactivate führt bei geeigneten GSV-Baugruppen (ab GSV Modell 2) implizit auch **GSVlockUII** (siehe 3.98) aus. Ist das Sperren nicht möglich, wird die Kommunikation abgebrochen und GSVactivate endet mit dem Rückgabewert GSV\_ERROR.

Bei der Aktivierung kann angegeben werden, wieviele Meßwerte des GSV die Bibliothek maximal zwischenspeichern soll, falls das Programm, das die Schnittstelle aktiviert hat, die Meßwerte nicht ausreichend schnell abholt. Ist der Wert kleiner als 1, wird 1 angenommen. Der maximal zulässige Wert hängt vom verfügbaren Speicher ab.

**Verwendete GSV Befehle:** 42, 3, 28, 26, ggf. 64 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört, die Schnittstelle bereits in Benutzung oder das Sperren des UII unmöglich sein.

## Funktionsdefinition:

C:                           int GSVactivate ( int **no**, long **buffersize** );

Delphi™:                   GSVactivate ( **no**: Integer; **buffersize**: LongInt ): Integer;

VB:                           GSVactivate ( ByVal **no** As Long, ByVal **buffersize** As Long )  
                                  As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                    Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**buffersize**        Ganze Zahl.  
                          Anzahl der maximal zwischenzuspeichernden Meßdatenwerte.

### 3.8 GSVrelease

(Index 11)

#### **Beschreibung:**

Die Funktion GSVrelease deaktiviert eine serielle Schnittstelle mit angeschlossener GSV-Baugruppe, die vorher durch GSVactivate oder GSVactivateExtended aktiviert wurde. Danach können mit der betreffenden Schnittstellennummer keine Operationen mehr ausgeführt werden (außer GSVactivate und GSVactivateExtended). Diese Funktion wird automatisch für alle von einem Programm aktivierten Schnittstellen ausgeführt, wenn das Programm endet.

GSVrelease führt bei geeigneten GSV-Baugruppen (ab GSV Modell 2) implizit auch **GSVunlockUII** (siehe 3.97) aus.

**Verwendete GSV Befehle:** ggf. 63 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Es steht kein Ergebnis zur Verfügung.

**Funktionsdefinition:**

C:                   void GSVrelease ( int **no** );

Delphi™:            GSVrelease ( **no**: Integer );

VB:                   GSVrelease ( ByVal **no** As Long )

**Rückgabewert:** Keiner.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV, muß aktiviert sein.

### 3.9 GSVflushBuffer

(Index 13)

#### **Beschreibung:**

Die Funktion GSVflushBuffer setzt zum Zeitpunkt ihres Aufrufs eine Marke im Datenstrom und verwirft mindestens alle diejenigen Daten, die vor dem Setzen der Marke von der GSV Baugruppe zum Senden bereitgestellt worden sind. Bei korrekter Ausführung liefert die Funktion GSVflushBuffer den Rückgabewert GSV\_TRUE oder GSV\_OK; dabei zeigt GSV\_TRUE an, daß tatsächlich Daten gefunden wurden, die verworfen wurden.

**Verwendete GSV Befehle:** 4 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVflushBuffer ( int **no** );

Delphi™:           GSVflushBuffer ( **no**: Integer ): Integer;

VB:                   GSVflushBuffer ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.10 GSVgetValues

(Index 17)

#### **Beschreibung:**

Die Funktion GSVgetValues löst genauso wie die Funktion **GSVgetValue** die Übertragung (je) eines Meßwertes aus. Die Funktion GSVgetValues löst jedoch Meßwerte bei mehreren GSV-Baugruppen parallel aus. Es ist jedoch nur im günstigsten Fall ein gewisses Maß an Parallelität der Befehlsübertragung erreichbar, die außerdem den bei Nicht-Echtzeit-Systemen üblichen Schwankungen unterliegt (Siehe dazu Abschnitt 1.3).

**Verwendete GSV Befehle:** 59 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1, Firmware Version 3.2-19.



## Funktionsdefinition:

C:                   int GSVgetValues ( int \***no**, int **n** );

Delphi™:           GSVgetValues ( **no**: PInteger; **n**: Integer ): Integer;

VB:                   GSVgetValues ( ByRef **no** As Long, ByVal **n** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**           Adresse eines Vektors von Ganzzahlvariablen. Jede der Variablen enthält einen Wert im Bereich 1..1024.  
(VB: Der Datentyp der Variablen ist Long; es wird das erste Element des Vektors als Referenz übergeben.)  
Die ganzzahligen Werte sind die Nummern der seriellen Schnittstellen der zu verwendenden GSVs.

**n**            Ganze Zahl im Bereich 1..1024.  
Anzahl der gültigen Schnittstellennummern im o.g. Vektor.

### 3.11 GSVreceived

(Index 20)

#### Beschreibung:

Die Funktion GSVreceived liefert GSV\_TRUE, wenn sich Daten im Buffer der Schnittstelle befinden, die mit **GSVread** oder **GSVreadMultiple** abgeholt werden können.

**Verwendete GSV Befehle:** Keine.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                   int GSVreceived ( int **no** );

Delphi™:           GSVreceived ( **no**: Integer ): Integer;

VB:                 GSVreceived ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.12 GSVread

(Index 21)

#### **Beschreibung:**

Die Funktion GSVread gibt GSV\_TRUE zurück, wenn sich Daten im Buffer befinden. Von diesen Daten wird der nächste Meßwert entnommen und unter der Variablen **ad** abgelegt. Im unipolaren Modus werden dabei Werte zwischen 0,0 und +1,05 zurückgeliefert, im bipolaren Modus Werte zwischen -1,05 und +1,05. Der Rückgabewert GSV\_OK bedeutet, daß die Funktion zwar fehlerfrei ausgeführt wurde, aber keine Daten verfügbar sind.

**Verwendete GSV Befehle:** Keine.

#### **Fehler:**

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                               int GSVread ( int **no**, double \***ad** );

Delphi™:                       GSVread ( **no**: Integer; var **ad**: Double ): Integer;

VB:                               GSVread ( ByVal **no** As Long, ByRef **ad** As Double ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                    Ganze Zahl im Bereich 1..1024.  
                           Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**ad**                    Zeiger auf Gleitkommavariablen doppelter Genauigkeit.  
                           Gelesener Meßwert ohne Berücksichtigung von Eingangsempfindlichkeit,  
                           Verstärkung und Anzeigenormierung.  
                           (Delphi™ und VB: Die Gleitkommavariablen werden als Referenz übergeben.)

### 3.13 GSVreadMultiple

(Index 22)

#### Beschreibung:

Die Funktion GSVreadMultiple gibt GSV\_TRUE zurück, wenn sich Daten im Buffer befinden. Von diesen Daten werden die nächsten **count** Meßwerte - oder weniger, wenn nicht ausreichend Daten vorliegen - entnommen und unter der Variablen **ad** (Vektor) abgelegt. Die Zahl der unter **ad** abgelegten Werte wird unter **valsread** gespeichert. Im unipolaren Modus werden dabei Werte zwischen 0,0 und +1,05 zurückgeliefert, im bipolaren Modus Werte zwischen -1,05 und +1,05. Der Rückgabewert GSV\_OK bedeutet, daß die Funktion zwar fehlerfrei ausgeführt wurde, aber keine Daten verfügbar sind.

Durch die Verwendung von GSVreadMultiple an Stelle von **GSVread** kann die Datenabholung effizienter gestaltet werden.

**Verwendete GSV Befehle:** Keine.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



### 3.14 GSVreadStatus

(Index 25)

#### Beschreibung:

Die Funktion GSVreadStatus gibt GSV\_TRUE zurück, wenn sich Daten im Buffer befinden. Von diesen Daten wird der nächste Meßwert entnommen und unter der Variablen **ad** abgelegt. Außerdem wird der zugehörige Status-Wert unter der Variablen **ps** gespeichert. Im unipolaren Modus werden dabei Werte zwischen 0,0 und +1,05 zurückgeliefert, im bipolaren Modus Werte zwischen -1,05 und +1,05. Der Rückgabewert GSV\_OK bedeutet, daß die Funktion zwar fehlerfrei ausgeführt wurde, aber keine Daten verfügbar sind.

**Verwendete GSV Befehle:** Keine.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



## Funktionsdefinition:

C:                   int GSVreadStatus ( int **no**, double \***ad**, unsigned char \***ps** );

Delphi™:           GSVreadStatus ( **no**: Integer; var **ad**: Double;  
  var **ps**: Byte ): Integer;

VB:                   GSVreadStatus ( ByVal **no** As Long, ByRef **ad** As Double,  
  ByRef **ps** As Byte ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>ad</b>	Zeiger auf Gleitkommavariablen doppelter Genauigkeit. Gelesener Meßwert ohne Berücksichtigung von Eingangsempfindlichkeit, Verstärkung und Anzeigenormierung. (Delphi™ und VB: Die Gleitkommavariablen werden als Referenz übergeben.)
<b>ps</b>	Zeiger auf Byte-Variablen. Kopie des übermittelten Status-Wertes, abhängig vom verwendeten Gerät. (Delphi™ und VB: Die Byte-Variablen werden als Referenz übergeben.)

### 3.15 GSVreadStatusMultiple

(Index 26)

#### Beschreibung:

Die Funktion GSVreadStatusMultiple gibt GSV\_TRUE zurück, wenn sich Daten im Buffer befinden. Von diesen Daten werden die nächsten **count** Meßwerte - oder weniger, wenn nicht ausreichend Daten vorliegen - entnommen und unter der Variablen **ad** (Vektor) abgelegt. Außerdem werden (in der gleichen Reihenfolge) die zugehörigen Statuswerte unter der Variablen **ps** (Vektor) gespeichert. Die Zahl der unter **ad** abgelegten Werte wird unter **valsread** gespeichert. Im unipolaren Modus werden dabei Werte zwischen 0,0 und +1,05 zurückgeliefert, im bipolaren Modus Werte zwischen -1,05 und +1,05. Der Rückgabewert GSV\_OK bedeutet, daß die Funktion zwar fehlerfrei ausgeführt wurde, aber keine Daten verfügbar sind.

Durch die Verwendung von GSVreadStatusMultiple an Stelle von **GSVreadStatus** kann die Datenabholung effizienter gestaltet werden.

**Verwendete GSV Befehle:** Keine.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



### 3.16 GSVgetOptionsCode

(Index 30)

#### Beschreibung:

Die Funktion GSVgetOptionsCode liest die Identifikation einer etwaigen Sonderanwendung. **Ist diese Identifikation verschieden von Null, muß mit Einschränkungen oder Abweichungen der Funktion der Firmware gerechnet werden.**

**Verwendete GSV Befehle:** 54 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:           int GSVgetOptionsCode ( int **no** );

Delphi™:     GSVgetOptionsCode ( **no**: Integer ): Integer;

VB:           GSVgetOptionsCode ( ByVal **no** As Long ) As Long

**Rückgabewert:** Identifikationscode der Sonderanwendung oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.17 GSVgetOptionsExtension3

(Index 31)

#### **Beschreibung:**

Die Funktion GSVgetOptionsExtension3 bestimmt, ob in der Firmware der betreffenden Baugruppe die Erweiterungen implementiert sind, die mit dem GSV-3 eingeführt wurden.

**Verwendete GSV Befehle:** 54 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:    int GSVgetOptionsExtension3 ( int **no** );

Delphi™:

        GSVgetOptionsExtension3 ( **no**: Integer ): Integer;

VB:     GSVgetOptionsExtension3 ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.18 GSVgetOptionsLinear

(Index 32)

#### **Beschreibung:**

Die Funktion GSVgetOptionsLinear bestimmt, ob in der Firmware der betreffenden Baugruppe die Linearisierungsberechnung für einen bestimmten Sensor implementiert ist. Siehe auch **GSVsetModeLinear** und **GSVgetModeLinear**.

**Verwendete GSV Befehle:** 54 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.



## Funktionsdefinition:

C:           int GSVgetOptionsLinear ( int **no** );

Delphi™:    GSVgetOptionsLinear ( **no**: Integer ): Integer;

VB:           GSVgetOptionsLinear ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                   Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.19 GSVgetOptionsExtension21

(Index 33)

#### **Beschreibung:**

Die Funktion GSVgetOptionsExtension21 bestimmt, ob in der Firmware der betreffenden Baugruppe die Erweiterungen implementiert sind, die mit dem GSV-2.1 eingeführt wurden.

**Verwendete GSV Befehle:** 54 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C: int GSVgetOptionsExtension21 ( int **no** );

Delphi™:

GSVgetOptionsExtension21 ( **no**: Integer ): Integer;

VB: GSVgetOptionsExtension21 ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                  Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.20 GSVgetOptionsSleepMode

(Index 34)

#### Beschreibung:

Die Funktion GSVgetOptionsSleepMode bestimmt, ob in der Firmware der betreffenden Baugruppe die Unterstützung des Energiesparmodus (Sleep-Mode) implementiert ist. Dieser Modus setzt den eingeschalteten Slow-Mode voraus (siehe **GSVsetSpecialModeSlow**) und reduziert den Energieverbrauch der GSV Baugruppe während der Wartezeit zwischen den Messungen.

**Verwendete GSV Befehle:** 54 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C: int GSVgetOptionsSleepMode ( int **no** );

Delphi™:

GSVgetOptionsSleepMode ( **no**: Integer ): Integer;

VB: GSVgetOptionsSleepMode ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                  Nummer der seriellen Schnittstelle des zu verwendenden GSV.

## 3.21 GSVgetOptionsCommandTest

(Index 35)

### Beschreibung:

Die Funktion GSVgetOptionsCommandTest bestimmt, ob in der Firmware der betreffenden Baugruppe die Unterstützung des Befehls für die Funktion **GSVisCommandAvailable** implementiert ist.

**Verwendete GSV Befehle:** 54 (siehe GSV Bedienungsanleitung).

### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:

`int GSVgetOptionsCommandTest ( int no );`

Delphi™:

`GSVgetOptionsCommandTest ( no: Integer ): Integer;`VB: `GSVgetOptionsCommandTest ( ByVal no As Long ) As Long`**Rückgabewert:** 0, 1 oder Fehlercode.**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
-----------	---

## 3.22 GSVisBipol

(Index 50)

### **Beschreibung:**

Die Funktion GSVisBipol ermittelt, ob sich der GSV im Bipolar-Modus befindet (Resultat = 1), oder im Unipolar-Modus (Resultat = 0).

**Verwendete GSV Befehle:** 3 (siehe GSV Bedienungsanleitung).

### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



## Funktionsdefinition:

C:                               int GSVisBipol ( int **no** );

Delphi™:                       GSVisBipol ( **no**: Integer ): Integer;

VB:                               GSVisBipol ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**                               Ganze Zahl im Bereich 1..1024.  
                                       Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.23 GSVgetFreq

(Index 51)

#### **Beschreibung:**

Die Funktion GSVgetFreq liest die aktuell eingestellte Frequenz (Erläuterung siehe **GSVsetFreq**) vom GSV. Ergebnis in Hertz.

Siehe auch **GSVreadSamplingRate**.

**Verwendete GSV Befehle:** 3 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                   double GSVgetFreq ( int **no** );

Delphi™:               GSVgetFreq ( **no**: Integer ): Double;

VB:                    GSVgetFreq ( ByVal **no** As Long ) As Double

**Rückgabewert:** Frequenz des GSV oder Fehlercode.

## Aufrufparameter:

**no**                    Ganze Zahl im Bereich 1..1024.  
                           Nummer der seriellen Schnittstelle des zu verwendenden GSV.

## 3.24 GSVgetGain

(Index 52)

### Beschreibung:

Die Funktion GSVgetGain liest die aktuell eingestellte Verstärkung vom GSV. Das Ergebnis ist ein codierter Wert, der bei **GSVsetGain** erläutert ist.

**Verwendete GSV Befehle:** 3 (siehe GSV Bedienungsanleitung).

### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVgetGain ( int **no** );

Delphi™:           GSVgetGain ( **no**: Integer ): Integer;

VB:                 GSVgetGain ( ByVal **no** As Long ) As Long

**Rückgabewert:** Code für die Verstärkung oder Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
                    Nummer der seriellen Schnittstelle des zu verwendenden GSV.

## 3.25 GSVgetChannel

(Index 53)

### Beschreibung:

Die Funktion GSVgetChannel liest den aktuell eingestellten Eingangskanal vom GSV. Das Ergebnis ist die Kanalnummer, die bei **GSVsetChannel** erläutert ist.

**Verwendete GSV Befehle:** 3 (siehe GSV Bedienungsanleitung).

### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                   int GSVgetChannel ( int **no** );

Delphi™:           GSVgetChannel ( **no**: Integer ): Integer;

VB:                   GSVgetChannel ( ByVal **no** As Long ) As Long

**Rückgabewert:** Kanalnummer oder Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.26 GSVgetModeLock

(Index 55)

#### **Beschreibung:**

Die Funktion GSVgetModeLock liest die aktuelle Einstellung der Sperre vom GSV. Bei einem Ergebnis von 1 ist die Sperre eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Die Sperre betrifft die Ausführung von Funktionen, die den Zustand der GSV Baugruppe verändern. Diese Befehle werden dann zwar angenommen, aber nicht mehr ausgeführt. Dies kann als Schutz gegen irrtümliche Veränderung der Einstellungen der GSV Baugruppe verwendet werden. Von der Sperre ausgenommen sind die Funktionen GSVsetZero, GSVsetOffset, GSVstartTransmit und GSVclearBuffer (siehe auch **GSVsetMode**).

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.



## Funktionsdefinition:

C:               int GSVgetModeLock ( int **no** );

Delphi™:       GSVgetModeLock ( **no**: Integer ): Integer;

VB:             GSVgetModeLock ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**             Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.27 GSVsetModeLinear

(Index 56)

#### **Beschreibung:**

Die Funktion GSVsetModeLinear schaltet die Linearisierung für einen bestimmten Sensor ein (Parameter = 1) oder aus (Parameter = 0). Diese Berechnung wird innerhalb der GSV Baugruppe ausgeführt und ist nur für einen bestimmten Sensor gültig.

Falls die GSV Baugruppe die Linearisierung trotz entsprechenden Befehls nicht einschaltet, kann dies folgende Gründe haben (je nach Bauart können manche Gründe grundsätzlich ausgeschlossen werden):

- Der Linearisierungsnullpunkt der Baugruppe wurde noch nie gesetzt.
- Die Linearisierungsdaten enthalten zu wenige Stützstellen.
- Die Eingangsempfindlichkeit und/oder die Verstärkung für den Betrieb der Linearisierung stimmen nicht überein mit der Eingangsempfindlichkeit bzw. der Verstärkung für die die Linearisierungsdaten berechnet wurden.
- Die Eingangsempfindlichkeit für den Betrieb der Linearisierung erfordert eine Änderung der Steckbrückenkonfiguration der Baugruppe.

**Verwendete GSV Befehle:** 39, 38 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Nur wenn das entsprechende Firmware-Optionsflag gesetzt ist.

## Funktionsdefinition:

C:           int GSVsetModeLinear ( int **no**, int **lin** );

Delphi™:       GSVsetModeLinear ( **no**, **lin**: Integer ): Integer;

VB:            GSVsetModeLinear ( ByVal **no** As Long, ByVal **lin** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                 Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**lin**           Ganze Zahl, 0 oder 1.  
                 Logischer Wert, der den Linearisierungsmodus aus- oder einschaltet.

### 3.28 GSVgetModeLinear

(Index 57)

#### **Beschreibung:**

Die Funktion GSVgetModeLinear liest die aktuelle Einstellung des Linearisierungsmodus vom GSV. Bei einem Ergebnis von 1 ist der Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Siehe auch GSVsetModeLinear.

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C:           int GSVgetModeLinear ( int **no** );

Delphi™:       GSVgetModeLinear ( **no**: Integer ): Integer;

VB:            GSVgetModeLinear ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                 Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.29 GSVsetModeAverage

(Index 58)

#### **Beschreibung:**

Die Funktion GSVsetModeAverage schaltet die Mittelung über jeweils 16 Messwerte ein (Parameter = 1) oder aus (Parameter = 0). Diese Berechnung wird innerhalb der GSV Baugruppe ausgeführt.

**Verwendete GSV Befehle:** 39, 38 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2, Firmware Version 3.3-8.

## Funktionsdefinition:

C:           int GSVsetModeAverage ( int **no**, int **avg** );

Delphi™:    GSVsetModeAverage ( **no**, **avg**: Integer ): Integer;

VB:           GSVsetModeAverage ( ByVal **no** As Long, ByVal **avg** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                   Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**avg**           Ganze Zahl, 0 oder 1.  
                   Logischer Wert, der den Mittelwert-Modus aus- oder einschaltet.

### 3.30 GSVgetModeAverage

(Index 59)

#### **Beschreibung:**

Die Funktion GSVgetModeAverage liest die aktuelle Einstellung des Mittelwert-Modus vom GSV. Bei einem Ergebnis von 1 ist der Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Siehe auch GSVsetModeAverage.

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2, Firmware Version 3.3-8.



## Funktionsdefinition:

C:           int GSVgetModeAverage ( int **no** );

Delphi™:     GSVgetModeAverage ( **no**: Integer ): Integer;

VB:           GSVgetModeAverage ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                   Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.31 GSVsetModeText

(Index 62)

#### **Beschreibung:**

Die Funktion GSVsetModeText schaltet den Text-Modus ein (Parameter = 1) oder aus (Parameter = 0). Im Text-Modus werden die Meßdaten in lesbarer Form vom GSV übertragen, im Gegensatz zum binären Übertragungsformat. Die Einstellung des Text-Modus wird zur Zeit noch bei der Herstellung vorgenommen und ist nicht umschaltbar.

**Verwendete GSV Befehle:** 39, 38 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C:                   int GSVsetModeText ( int **no**, int **mt** );

Delphi™:           GSVsetModeText ( **no**, **mt**: Integer ): Integer;

VB:                   GSVsetModeText ( ByVal **no** As Long, ByVal **mt** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
                           Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**mt**                   Ganze Zahl, 0 oder 1.  
                           Logischer Wert, der den Textmodus aus- oder einschaltet.

### 3.32 GSVgetModeText

(Index 63)

#### **Beschreibung:**

Die Funktion GSVgetModeText liest die aktuelle Einstellung des Text-Modus vom GSV. Bei einem Ergebnis von 1 ist der Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Im Text-Modus werden die Meßdaten in lesbarer Form vom GSV übertragen, im Gegensatz zum binären Übertragungsformat.

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:               int GSVgetModeText ( int **no** );

Delphi™:        GSVgetModeText ( **no**: Integer ): Integer;

VB:             GSVgetModeText ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**             Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.33 GSVsetModeMax

(Index 64)

#### **Beschreibung:**

Die Funktion GSVsetModeMax schaltet den Maximum-Modus ein (Parameter = 1) oder aus (Parameter = 0). Im Maximum-Modus enthalten die Meßdaten stets nur den maximal gemessenen Wert.

**Verwendete GSV Befehle:** 39, 38 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C:                   int GSVsetModeMax ( int **no**, int **mx** );

Delphi™:           GSVsetModeMax ( **no**, **mx**: Integer ): Integer;

VB:                   GSVsetModeMax ( ByVal **no** As Long, ByVal **mx** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**mx**                   Ganze Zahl, 0 oder 1.  
Logischer Wert, der den Maximum-Modus aus- oder einschaltet.

### 3.34 GSVgetModeMax

(Index 65)

#### **Beschreibung:**

Die Funktion GSVgetModeMax liest die aktuelle Einstellung des Maximum-Modus vom GSV. Bei einem Ergebnis von 1 ist der Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Im Maximum-Modus enthalten die Meßdaten stets nur den maximal gemessenen Wert.

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.



**Funktionsdefinition:**

C:               int GSVgetModeMax ( int **no** );

Delphi™:       GSVgetModeMax ( **no**: Integer ): Integer;

VB:             GSVgetModeMax ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**             Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.35 GSVsetModeLog

(Index 66)

#### **Beschreibung:**

Die Funktion GSVsetModeLog schaltet den Logger-Modus ein (Parameter = 1) oder aus (Parameter = 0). Im Logger-Modus werden die Meßdaten nicht dauernd gesendet, sondern nur auf besondere Anforderung, z.B. durch ein Schaltsignal am GSV.

**Verwendete GSV Befehle:** 39, 38 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C:                   int GSVsetModeLog ( int **no**, int **lg** );

Delphi™:           GSVsetModeLog ( **no**, **lg**: Integer;): Integer;

VB:                   GSVsetModeLog ( ByVal **no** As Long, ByVal **lg** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**lg**                   Ganze Zahl, 0 oder 1.  
Logischer Wert, der den Logger-Modus aus- oder einschaltet.

### 3.36 GSVgetModeLog

(Index 67)

#### **Beschreibung:**

Die Funktion GSVgetModeLog liest die aktuelle Einstellung des Logger-Modus vom GSV. Bei einem Ergebnis von 1 ist der Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Im Logger-Modus werden die Meßdaten nicht dauernd gesendet, sondern nur auf besondere Anforderung, z.B. durch ein Schaltsignal am GSV.

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C:                   int GSVgetModeLog ( int **no** );

Delphi™:           GSVgetModeLog ( **no**: Integer ): Integer;

VB:                   GSVgetModeLog ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.37 GSVsetModeWindow

(Index 68)

#### **Beschreibung:**

Die Funktion GSVsetModeWindow schaltet den Fenster-Modus ein (Parameter = 1) oder aus (Parameter = 0). Im Fenster-Modus wirkt der Schwellwertschalter als Fensterdiskriminator.

**Verwendete GSV Befehle:** 39, 38 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C:           int GSVsetModeWindow ( int **no**, int **win** );

Delphi™:    GSVsetModeWindow ( **no**, **win**: Integer ): Integer;

VB:           GSVsetModeWindow ( ByVal **no** As Long, ByVal **win** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>win</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Fensterdiskriminator-Modus aus- oder einschaltet.

### 3.38 GSVgetModeWindow

(Index 69)

#### **Beschreibung:**

Die Funktion GSVgetModeWindow liest die aktuelle Einstellung des Fenster-Modus vom GSV. Bei einem Ergebnis von 1 ist der Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Im Fenster-Modus wirkt der Schwellwertschalter als Fensterdiskriminator.

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.



## Funktionsdefinition:

C:           int GSVgetModeWindow ( int **no** );

Delphi™:    GSVgetModeWindow ( **no**: Integer ): Integer;

VB:           GSVgetModeWindow ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**           Ganze Zahl im Bereich 1..1024.  
               Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.39 GSVhasLCD

(Index 70)

#### **Beschreibung:**

Die Funktion GSVhasLCD liest vom GSV, ob eine Flüssigkristallanzeige zur Ausstattung des Geräts gehört (Ergebnis = 1, falls ja).

**Verwendete GSV Befehle:** 41 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:                   int GSVhasLCD ( int **no** );

Delphi™:           GSVhasLCD ( **no**: Integer ): Integer;

VB:                GSVhasLCD ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.40 GSVhasADC

(Index 71)

#### **Beschreibung:**

Die Funktion GSVhasADC liest vom GSV, ob ein Analog/Digital-Wandler zur Ausstattung des Geräts gehört (Ergebnis = 1, falls ja).

**Verwendete GSV Befehle:** 41 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:                   int GSVhasADC ( int **no** );

Delphi™:           GSVhasADC ( **no**: Integer ): Integer;

VB:                 GSVhasADC ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.41 GSVhasUII

(Index 72)

#### **Beschreibung:**

Die Funktion GSVhasUII liest vom GSV, ob ein UII (User Input Interface, d.h. Tastatur o.ä.) zur Ausstattung des Geräts gehört (Ergebnis = 1, falls ja).

**Verwendete GSV Befehle:** 41 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:                           int GSVhasUII ( int **no** );

Delphi™:                   GSVhasUII ( **no**: Integer ): Integer;

VB:                           GSVhasUII ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                        Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.42 GSVisSI

(Index 73)

#### **Beschreibung:**

Die Funktion GSVisSI liest vom GSV, ob das Gerät einer besonderen, reservierten Kategorie von Geräten zugehört, **deren Einstellungen auf keinen Fall umprogrammiert werden dürfen** (Ergebnis = 1, falls ja). Diese Geräte werden nur durch Spezialprogramme konfiguriert und programmiert.

**Verwendete GSV Befehle:** 41 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.



**Funktionsdefinition:**

C:                               int GSVisSI ( int **no** );

Delphi™:                       GSVisSI ( **no**: Integer ): Integer;

VB:                             GSVisSI ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                      Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.43 GSVisWL

(Index 74)

#### **Beschreibung:**

Die Funktion GSVisWL liest vom GSV, ob das Gerät eine Funkübertragung der Datenkommunikation verwendet (Ergebnis = 1, falls ja).

**Verwendete GSV Befehle:** 41 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:                           int GSVisWL ( int **no** );

Delphi™:                   GSVisWL ( **no**: Integer ): Integer;

VB:                           GSVisWL ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                        Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.44 GSVhasAF

(Index 75)

#### **Beschreibung:**

Die Funktion GSVhasAF liest vom GSV, ob das Gerät über ein analoges Filter verfügt (Ergebnis = 1, falls ja).

**Verwendete GSV Befehle:** 41 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:                           int GSVhasAF ( int **no** );

Delphi™:                   GSVhasAF ( **no**: Integer ): Integer;

VB:                           GSVhasAF ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                        Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.45 GSVsetBridgeType

(Index 80)

#### Beschreibung:

Die Funktion GSVsetBridgeType legt die Grundschialtung der verwendeten Widerstandsbrücke gemäß der folgenden Tabelle fest:

Typ	Schaltung
0	Vollbrücke, 4 aktive DMS
1	Halbbrücke, 2 aktive DMS
2	Viertelbrücke, 1 aktiver DMS
3	Halbbrücke, 2 aktive DMS, 1 × längs, 1 × quer
4	Vollbrücke, 4 aktive DMS, 2 × längs, 2 × quer

**Verwendete GSV Befehle:** 49, 48 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:           int GSVsetBridgeType ( int **no**, int **bt** );

Delphi™:       GSVsetBridgeType ( **no**, **bt**: Integer ): Integer;

VB:            GSVsetBridgeType ( ByVal **no** As Long, ByVal **bt** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                   Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**bt**            Ganze Zahl im Bereich 0..4.  
                   Code für die gewünschte Grundschtung der Widerstandsbrücke.

### 3.46 GSVgetBridgeType

(Index 81)

#### **Beschreibung:**

Die Funktion GSVgetBridgeType liest die aktuell eingestellte Grundschtung der Widerstandsbrücke aus dem GSV. Das Ergebnis ist ein codierter Wert, der bei **GSVsetBridgeType** erläutert ist.

**Verwendete GSV Befehle:** 49 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



## Funktionsdefinition:

C:           int GSVgetBridgeType ( int **no** );

Delphi™:       GSVgetBridgeType ( **no**: Integer ): Integer;

VB:            GSVgetBridgeType ( ByVal **no** As Long ) As Long

**Rückgabewert:** Code der Grundschtung der Widerstandsbrücke oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                 Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.47 GSVresetStatus

(Index 100)

#### **Beschreibung:**

Die Funktion GSVresetStatus setzt den Verstärkerstatus zurück (status = 0).

**Verwendete GSV Befehle:** 0 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVresetStatus ( int **no** );

Delphi™:           GSVresetStatus ( **no**: Integer ): Integer;

VB:                   GSVresetStatus ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.48 GSVgetScale

(Index 101)

#### **Beschreibung:**

Die Funktion GSVgetScale ermittelt den Inhalt des Scale-Registers. Der gelesene Wert kann im Rechner gespeichert werden und dann mit Hilfe von **GSVwriteScale** zu einem späteren Zeitpunkt wieder hergestellt werden.

**Verwendete GSV Befehle:** 1 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                    long GSVgetScale ( int **no** );

Delphi™:            GSVgetScale ( **no**: Integer ): LongInt;

VB:                    GSVgetScale ( ByVal **no** As Long ) As Long

**Rückgabewert:** Skalierungswert oder Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.49 GSVgetZero

(Index 102)

#### **Beschreibung:**

Die Funktion GSVgetZero ermittelt den Inhalt des Zero-Registers. Der gelesene Wert kann im Rechner gespeichert werden und dann mit Hilfe von **GSVwriteZero** zu einem späteren Zeitpunkt wieder hergestellt werden.

**Verwendete GSV Befehle:** 2 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                    long GSVgetZero ( int **no** );

Delphi™:            GSVgetZero ( **no**: Integer ): LongInt;

VB:                    GSVgetZero ( ByVal **no** As Long ) As Long

**Rückgabewert:** Null-Einstellungswert oder Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                        Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.50 GSVgetControl

(Index 103)

#### Beschreibung:

Die Funktion GSVgetControl ermittelt die momentane Konfiguration des GSVs. Der zurückgelieferte Wert enthält codiert die Notch-Frequenz, Betriebsart, Polarität sowie Verstärkung des Umsetzers. Dieser Wert kann mit **GSVwriteControl** an den Umsetzer zurückgeschrieben werden.

**Verwendete GSV Befehle:** 3 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



## Funktionsdefinition:

C:                    long GSVgetControl ( int **no** );

Delphi™:            GSVgetControl ( **no**: Integer ): LongInt;

VB:                    GSVgetControl ( ByVal **no** As Long ) As Long

**Rückgabewert:** Kontrollregisterwert oder Fehlercode.

## Aufrufparameter:

**no**                    Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.51 GSVgetOffset

(Index 104)

#### **Beschreibung:**

Die Funktion GSVgetOffset ermittelt die Offseteinstellung des Vorverstärkers. Der erhaltene Wert kann mit **GSVwriteOffset** an den GSV zurückgeschrieben werden.

**Verwendete GSV Befehle:** 4 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                    long GSVgetOffset ( int **no**, );

Delphi™:            GSVgetOffset ( **no**: Integer ): LongInt;

VB:                    GSVgetOffset ( ByVal **no** As Long ) As Long

**Rückgabewert:** Offsetwert oder Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                        Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.52 GSVwriteScale

(Index 105)

#### Beschreibung:

Die Funktion GSVwriteScale setzt die Empfindlichkeits-Kalibrierung des AD-Umsetzers. Der Parameter **scalev** muß dabei einen Wert enthalten, der mit **GSVgetScale** zu einem früheren Zeitpunkt ermittelt wurde.

**Verwendete GSV Befehle:** 5 (siehe GSV Bedienungsanleitung).

**Beeinflußte Register:** Scale.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                   int GSVwriteScale ( int **no**, long **scalev** );

Delphi™:           GSVwriteScale ( **no**: Integer; **scalev**: LongInt ): Integer;

VB:                   GSVwriteScale ( ByVal **no** As Long, ByVal **scalev** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**scalev**             Ganze Zahl.  
                          Skalierungswert, wie durch GSVgetScale erhalten.

### 3.53 GSVwriteZero

(Index 106)

#### Beschreibung:

Die Funktion GSVwriteZero setzt die Null-Kalibrierung des AD-Umsetzers. Der Parameter **zerov** muß dabei einen Wert enthalten, der mit **GSVgetZero** zu einem früheren Zeitpunkt ermittelt wurde.

**Verwendete GSV Befehle:** 6 (siehe GSV Bedienungsanleitung).

**Beeinflußte Register:** Zero.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                   int GSVwriteZero ( int **no**, long **zerov** );

Delphi™:           GSVwriteZero ( **no**: Integer; **zerov**: LongInt ): Integer;

VB:                   GSVwriteZero ( ByVal **no** As Long, ByVal **zerov** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                Ganze Zahl im Bereich 1..1024.  
                    Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**zerov**            Ganze Zahl.  
                    Null-Einstellungswert, wie durch GSVgetZero erhalten.

### 3.54 GSVwriteControl

(Index 107)

#### **Beschreibung:**

Die Funktion GSVwriteControl stellt eine Konfiguration wieder her, die vorher durch **GSVgetControl** ermittelt wurde. Es werden Kanal, Betriebsart, Polarität, Verstärkung und Notch-Frequenz gesetzt.

**Verwendete GSV Befehle:** 7 (siehe GSV Bedienungsanleitung).

**Beeinflusste Register:** Channel, Frequenz, Gain, Bipolar/Unipolar.

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.





### 3.55 GSVwriteOffset

(Index 108)

#### Beschreibung:

Die Funktion GSVwriteOffset setzt die Offseiteinstellung des Vorverstärkers. Der Parameter **offsetv** muß dabei einen Wert enthalten, der mit **GSVgetOffset** zu einem früheren Zeitpunkt ermittelt wurde.

**Verwendete GSV Befehle:** 8 (siehe GSV Bedienungsanleitung).

#### Bemerkung:

Nur die Prozeduren **GSVSetOffset** und **GSVwriteOffset** haben Einfluß auf den Analogausgang.

**Beeinflußte Register:** Offset.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                   int GSVwriteOffset ( int **no**, long **offsetv** );

Delphi™:           GSVwriteOffset ( **no**: Integer; **offsetv**: LongInt ): Integer;

VB:                   GSVwriteOffset ( ByVal **no** As Long, ByVal **offsetv** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                    Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**offsetv**            Ganze Zahl.  
                          Offsetwert, wie durch GSVgetOffset erhalten.

### 3.56 GSVgetAll

(Index 109)

#### **Beschreibung:**

Die Funktion GSVgetAll liest die Konfiguration, die vorher im Speicher an der Stelle **pos** abgelegt worden ist.

pos = 0: Einstellungen vor dem letzten Abschalten  
pos = 1: Voreinstellung des Herstellers  
pos = 2: Vom Benutzer festgelegte Konfiguration 1  
pos = 3: Vom Benutzer festgelegte Konfiguration 2  
pos = 4: Vom Benutzer festgelegte Konfiguration 3  
pos = 5: Vom Benutzer festgelegte Konfiguration 4  
pos = 6: Vom Benutzer festgelegte Konfiguration 5  
pos = 7: Vom Benutzer festgelegte Konfiguration 6

Nicht alle GSV Baugruppen unterstützen alle Konfigurationsspeicher-Positionen.

**Verwendete GSV Befehle:** 9 (siehe GSV Bedienungsanleitung).

**Beeinflusste Register:** Gain, Frequenz, Offset, Zero, Scale, Schwellwert, Channel.

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                               int GSVgetAll ( int **no**, int **pos** );

Delphi™:                       GSVgetAll ( **no**, **pos**: Integer ): Integer;

VB:                               GSVgetAll ( ByVal **no** As Long, ByVal **pos** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>pos</b>	Ganze Zahl im Bereich 0..7. Code für den Datensatz zu ladender Einstellungen

### 3.57 GSVsaveAll

(Index 110)

#### Beschreibung:

Die Funktion GSVsaveAll sichert alle relevanten Register des GSVs in einem internen Speicher. Diese Daten bleiben auch nach dem Abschalten des Gerätes erhalten. Es können dabei unterschiedliche Konfigurationen gespeichert werden. Nach Einschalten des Verstärkers sind bis zu 64 Speichervorgänge möglich. Der Parameter **pos** gibt dabei das Ziel des Speichervorganges an.

pos = 2: Vom Benutzer festgelegte Konfiguration 1  
pos = 3: Vom Benutzer festgelegte Konfiguration 2  
pos = 4: Vom Benutzer festgelegte Konfiguration 3  
pos = 5: Vom Benutzer festgelegte Konfiguration 4  
pos = 6: Vom Benutzer festgelegte Konfiguration 5  
pos = 7: Vom Benutzer festgelegte Konfiguration 6

Nicht alle GSV Baugruppen unterstützen alle Konfigurationsspeicher-Positionen.

Die Positionen 0 und 1 werden vom Anwender nicht programmiert. Auf der Position 0 (siehe **GSVgetAll**) wird beim Abschalten die Konfiguration des GSVs automatisch gesichert.

Das Laden der gesicherten Daten erfolgt mit der Funktion **GSVgetAll**.

**Verwendete GSV Befehle:** 10 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                           int GSVsaveAll ( int **no**, int **pos** );

Delphi™:                   GSVsaveAll ( **no**, **pos**: Integer ): Integer;

VB:                           GSVsaveAll ( ByVal **no** As Long, ByVal **pos** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>pos</b>	Ganze Zahl im Bereich 2..7. Code für den Datensatz, unter dem die aktuellen Einstellungen gesichert werden sollen.

### 3.58 GSVsetCal

(Index 111)

#### Beschreibung:

Die Funktion GSVsetCal führt eine interne Empfindlichkeitskalibrierung durch. Es gelten nach dieser Kalibrierung die mit **GSVsetGain** gewählten Verstärkungen.

**Verwendete GSV Befehle:** 11, 59 (siehe GSV Bedienungsanleitung).

#### Achtung:

Mit **GSVsetScale** durchgeführte Kalibrierungen gehen verloren. Der Analogausgang wird nicht beeinflusst.

#### Bemerkung:

Nach dem Aufruf dieser Funktion ist die Dauer von ca. 1,0 s abzuwarten (als sicherer, ungefährender Wert). Danach sollte **GSVflushBuffer** aufgerufen werden, um veraltete und verfälschte Messwerte zu verwerfen.

Bei korrekter Ausführung liefert die Funktion GSVsetCal den Rückgabewert GSV\_TRUE; ist der Rückgabewert GSV\_OK, zeigt dies eine Zeitüberschreitung an, die bei vereinzeltem Auftreten durch Wiederholung des Aufrufs behandelt werden kann.

**Beeinflusste Register:** Scale.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



**Funktionsdefinition:**

C:                           int GSVsetCal ( int **no** );

Delphi™:                   GSVsetCal ( **no**: Integer ): Integer;

VB:                           GSVsetCal ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                        Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.59 GSVsetZero

(Index 112)

#### Beschreibung:

Die Funktion GSVsetZero führt bei angeschlossenem Sensor einen System-Nullpunktabgleich durch. Die Empfindlichkeitskalibrierung kann getrennt mit **GSVsetCal** durchgeführt werden. **GSVsetZero** hat keinen Einfluß auf den Analogausgang.

**Verwendete GSV Befehle:** 12, 59 (siehe GSV Bedienungsanleitung).

#### Bemerkung:

Nach dem Aufruf dieser Funktion ist die Dauer von ca. 1,0 s abzuwarten (als sicherer, ungefährender Wert). Danach sollte **GSVflushBuffer** aufgerufen werden, um veraltete und verfälschte Messwerte zu verwerfen.

Bei korrekter Ausführung liefert die Funktion GSVsetZero den Rückgabewert GSV\_TRUE; ist der Rückgabewert GSV\_OK, zeigt dies eine Zeitüberschreitung an, die bei vereinzeltem Auftreten durch Wiederholung des Aufrufs behandelt werden kann.

Falls Log- und Maximalwertmodus (siehe **GSVsetMode**) eingestellt sind, wird vor dem Nullabgleich der letzte Maximalwert einmalig übertragen.

**Beeinflusste Register:** Zero.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVsetZero ( int **no** );

Delphi™:           GSVsetZero ( **no**: Integer ): Integer;

VB:                 GSVsetZero ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.60 GSVsetScale

(Index 113)

#### Beschreibung:

Die Funktion GSVsetScale führt bei angeschlossenem Sensor eine System-Empfindlichkeitsskalierung durch. Im Gegensatz zu **GSVsetCal** wird hier die Empfindlichkeit des gesamten Systems einschließlich des angeschlossenen Aufnehmers eingestellt. Nach einem Nullpunktabgleich mit **GSVsetZero** kann diese Funktion den Aufnehmer bei aufgebrachter Nennlast auf den Endwert (Vollausschlag des AD-Umsetzers) skalieren. Diese Prozedur hat keinen Einfluß auf den Analogausgang.

**Verwendete GSV Befehle:** 13, 59 (siehe GSV Bedienungsanleitung).

#### Achtung:

**GSVsetScale** nicht ohne vorherigen Nullpunktabgleich mit **GSVsetZero** durchführen.

#### Bemerkung:

Nach dem Aufruf dieser Funktion ist die Dauer von ca. 1,0 s abzuwarten (als sicherer, ungefährender Wert). Danach sollte **GSVflushBuffer** aufgerufen werden, um veraltete und verfälschte Messwerte zu verwerfen.

Bei korrekter Ausführung liefert die Funktion GSVsetScale den Rückgabewert GSV\_TRUE; ist der Rückgabewert GSV\_OK, zeigt dies eine Zeitüberschreitung an, die bei vereinzelter Auftreten durch Wiederholung des Aufrufs behandelt werden kann.

**Beeinflusste Register:** Scale.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVsetScale ( int **no** );

Delphi™:           GSVsetScale ( **no**: Integer ): Integer;

VB:                 GSVsetScale ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.61 GSVsetOffset

(Index 114)

#### Beschreibung:

Die Funktion GSVsetOffset führt einen Offsetabgleich der GSV-Eingangsstufe durch. Dieser Abgleich betrifft im Gegensatz zu **GSVsetZero** auch den Analogausgang des GSVs. Bei größeren Verstimmungen der Brücke ist dieser Abgleich für die Funktion des GSVs unerlässlich.

**Verwendete GSV Befehle:** 14, 59 (siehe GSV Bedienungsanleitung).

#### Bemerkung:

Nach dem Aufruf dieser Funktion ist die Dauer von ca. 1,5 s abzuwarten, oder die von durch **MEgetOffsetWait** erhaltene Zeitdauer. Danach sollte **GSVflushBuffer** aufgerufen werden, um veraltete und verfälschte Messwerte zu verwerfen. Bei korrekter Ausführung liefert die Funktion GSVsetOffset den Rückgabewert GSV\_TRUE; ist der Rückgabewert GSV\_OK, zeigt dies eine Zeitüberschreitung an, die bei vereinzeltem Auftreten durch Wiederholung des Aufrufs behandelt werden kann. Nur die Prozeduren **GSVSetOffset** und **GSVwriteOffset** haben Einfluß auf den Analogausgang.

**Beeinflusste Register:** Offset.

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVsetOffset ( int **no** );

Delphi™:           GSVsetOffset ( **no**: Integer ): Integer;

VB:                 GSVsetOffset ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                 Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

## 3.62 GSVDISPSetUnit

(Index 115)

### Beschreibung:

Die Funktion GSVDISPSetUnit setzt die auf dem LC-Display angezeigte Einheit entsprechend dem Wert des Parameters **dispunit**.

0:	: mV/V	
1:	: kg	
2:	: g	
3:	: N	
4:	: cN	
5:	: V	
6:	: µm/m	
7:	:	(keine)
8:	: t	
9:	: kN	
10:	: lb	
11:	: oz	
12:	: kp	
13:	: lbf	
14:	: pdl	
15:	: mm	
16:	: m	
17:	: cNm	
18:	: Nm	

**Verwendete GSV Befehle:** 15 (siehe GSV Bedienungsanleitung).

**Beeinflusste Register:** Einheit

### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

### Anwendbar:

Bei allen GSV Modellen, jedoch sind vor Modell 1 nur die Werte 0..7 gültig, wobei dem Wert 4 die Einheit mN zugeordnet ist.



## Funktionsdefinition:

C:                   int GSVDISPSetUnit ( int **no**, int **dispunit** );

Delphi™:           GSVDISPSetUnit ( **no**, **dispunit**: Integer ): Integer;

VB:                   GSVDISPSetUnit ( ByVal **no** As Long, ByVal **dispunit** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                    Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**dispunit**            Ganze Zahl im Bereich 0..18.  
                          Code für die anzuzeigende Einheit

### 3.63 GSVDISPSetNorm

(Index 116)

#### Beschreibung:

Die Funktion GSVDISPSetNorm normiert die auf dem LC-Display gezeigten Meßwerte. Die Prozedur setzt dafür auch den Dezimalpunkt. Wird für **norm** z.B. 1000 gewählt, so ergibt sich im Bipolar-Modus ein Nennanzeigebereich von -1000...+1000.

**Verwendete GSV Befehle:** 16, 26, 17, 28 (siehe GSV Bedienungsanleitung).

#### Bemerkung:

Nach dem Aufruf dieser Funktion sollte **GSVflushBuffer** aufgerufen werden, um veraltete und verfälschte Messwerte zu verwerfen.

**Beeinflusste Register:** Normierung und Dezimalpunkt

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:               int GSVDISPSetNorm ( int **no**, double **norm** );

Delphi™:        GSVDISPSetNorm ( **no**: Integer; **norm**: Double ): Integer;

VB:               GSVDISPSetNorm ( ByVal **no** As Long, ByVal **norm** As Double )  
                    As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
                    Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**norm**            Gleitkommazahl im Bereich ungefähr 0,15..150000.  
                    Wert, auf den die Anzeige normiert werden soll.

### 3.64 GSVDISPSetDPoint

(Index 117)

#### Beschreibung:

Die Funktion GSVDISPSetDPoint versetzt den im LC-Display angezeigten Dezimalpunkt an die angegebene Stelle.

**Verwendete GSV Befehle:** 17, 28 (siehe GSV Bedienungsanleitung).

#### Bemerkung:

Nach dem Aufruf dieser Funktion sollte **GSVflushBuffer** aufgerufen werden, um veraltete und verfälschte Messwerte zu verwerfen.

**Beeinflusste Register:** Dezimalpunkt

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:           int GSVDISPSetDPoint ( int **no**, int **dpoint** );

Delphi™:       GSVDISPSetDPoint ( **no**, **dpoint**: Integer ): Integer;

VB:            GSVDISPSetDPoint ( ByVal **no** As Long, ByVal **dpoint** As Long )  
                  As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                  Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**dpoint**       Ganze Zahl im Bereich 1..6.  
                  Gewünschte Position des Dezimalpunktes.

### 3.65 GSVsetFreq

(Index 118)

#### Beschreibung:

Die Funktion GSVsetFreq legt die erste Notch-Frequenz des GSV fest. Das implementierte digitale Filter entfernt diese Frequenz und deren Vielfache aus dem Meßsignal. Um eine möglichst hohe Unterdrückung der Netzfrequenz zu erhalten, sollte die Notch-Frequenz kleiner oder gleich der Netzfrequenz sein und in einem einfachen Teilverhältnis zu dieser stehen ( $f_{\text{notch}} = 50/n$  mit  $1 \leq n \leq 5$ ). Es können Frequenzen zwischen 9,766 Hz und ca. 390 Hz gewählt werden. Im Logger-Modus kann die Frequenz auch höher liegen (ca. 930 Hz), im Text-Modus darf eine Frequenz von ca. 100 Hz aus Gründen der Datenübertragungsgeschwindigkeit nicht überschritten werden. Siehe dazu **GSVsetModeLog** und **GSVsetModeText**. Die 3 dB Grenzfrequenz des Filters liegt bei  $f_g = 0,262 \times f_{\text{notch}}$ . Die Meßwertübertragungsrate des GSVs entspricht dabei der eingestellten Notch-Frequenz. Bei einer Frequenz  $f_{\text{notch}} = 100$  Hz werden also 100 Messungen pro Sekunde durchgeführt. Diese Prozedur hat keinen Einfluß auf den Analogausgang.

Siehe auch **GSVwriteSamplingRate**.

**Verwendete GSV Befehle:** 18 (siehe GSV Bedienungsanleitung).

#### Achtung:

Nach einer Änderung der Filterfrequenz ist ein Abgleich mit **GSVsetCal** und **GSVsetZero** notwendig.

**Beeinflusste Register:** Frequenz.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                           int GSVsetFreq ( int **no**, double **freq** );

Delphi™:                   GSVsetFreq ( **no**: Integer; **freq**: Double ): Integer;

VB:                           GSVsetFreq ( ByVal **no** As Long, ByVal **freq** As Double )  
  As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>freq</b>	Gleitkommazahl, der Bereich hängt von weiteren Bedingungen ab. Gewünschte Frequenz des GSV.

### 3.66 GSVsetGain

(Index 119)

#### Beschreibung:

Die Funktion GSVsetGain legt die Verstärkung des GSVs relativ zur Eingangsempfindlichkeit (siehe **GSVgetRange**) gemäß der folgenden Tabelle fest:

gain	Verstärkungsfaktor			
	GSV-2	GSV-3	GSV-2.1 Kanal 0	GSV-2.1 Kanal 1
0	1	1	$\frac{1}{4}$	$\frac{1}{2}$
1	2	—	$\frac{1}{2}$	1
2	4	—	1	2
3	8	—	2	4
4	16	—	4	8
5	32	—	8	16
6	64	—	16	32
7	128	—	—	—

**Verwendete GSV Befehle:** 19 (siehe GSV Bedienungsanleitung).

#### Achtung:

Nach einer Änderung der Verstärkung ist ein Abgleich mit **GSVsetCal** und **GSVsetZero** notwendig.

**Beeinflusste Register:** Gain.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



## Funktionsdefinition:

C:                           int GSVsetGain ( int **no**, int **gain** );

Delphi™:                   GSVsetGain ( **no**, **gain**: Integer ): Integer;

VB:                           GSVsetGain ( ByVal **no** As Long, ByVal **gain** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>gain</b>	Ganze Zahl im Bereich 0..7. Code für die gewünschte Verstärkung.

### 3.67 GSVsetBipolar

(Index 120)

#### Beschreibung:

Die Funktion GSVsetBipolar versetzt den GSV in den Bipolarmodus. In dieser Einstellung werden Ausschläge zwischen -1,05 und +1,05 gemessen. Diese Funktion hat keinen Einfluß auf den Analogausgang.

**Verwendete GSV Befehle:** 20 (siehe GSV Bedienungsanleitung).

#### Achtung:

Nach einer Änderung des Unipolar-/Bipolar-Modus ist ein Abgleich mit **GSVsetCal** und **GSVsetZero** notwendig.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVsetBipolar ( int **no** );

Delphi™:           GSVsetBipolar ( **no**: Integer ): Integer;

VB:                 GSVsetBipolar ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.68 GSVsetUnipolar

(Index 121)

#### Beschreibung:

Die Funktion GSVsetUnipolar versetzt den GSV in den Unipolarmodus. In dieser Einstellung werden nur positive Ausschläge von 0,0 bis +1,05 gemessen. Diese Prozedur hat keinen Einfluß auf den Analogausgang.

**Verwendete GSV Befehle:** 21 (siehe GSV Bedienungsanleitung).

#### Achtung:

Nach einer Änderung des Unipolar-/Bipolar-Modus ist ein Abgleich mit **GSVsetCal** und **GSVsetZero** notwendig.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVsetUnipolar ( int **no** );

Delphi™:           GSVsetUnipolar ( **no**: Integer ): Integer;

VB:                   GSVsetUnipolar ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.69 GSVDISPGetNorm

(Index 126)

#### **Beschreibung:**

Die Funktion GSVDISPGetNorm gibt die mit **GSVDISPSetNorm** eingestellte Normierung zurück.

**Verwendete GSV Befehle:** 26, 28 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:        double GSVDISPGetNorm ( int **no** );

Delphi™:        GSVDISPGetNorm ( **no**: Integer ): Double;

VB:        GSVDISPGetNorm ( ByVal **no** As Long ) As Double

**Rückgabewert:** Anzeigenormierung oder Fehlercode.

**Aufrufparameter:**

**no**        Ganze Zahl im Bereich 1..1024.  
            Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.70 GSVDISPGetUnit

(Index 127)

#### Beschreibung:

Die Funktion GSVDISPGetUnit liefert die Nummer der gewählten Einheit gemäß der obigen Tabelle (siehe **GSVDISPSetUnit**) zurück.

**Verwendete GSV Befehle:** 27 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



**Funktionsdefinition:**

C:                   int GSVDISPGetUnit ( int **no** );

Delphi™:           GSVDISPGetUnit ( **no**: Integer ): Integer;

VB:                   GSVDISPGetUnit ( ByVal **no** As Long ) As Long

**Rückgabewert:** Code der angezeigten Einheit oder Fehlercode.

**Aufrufparameter:**

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.71 GSVDISPGetDPoint

(Index 128)

#### **Beschreibung:**

Die Funktion GSVDISPGetDPoint ermittelt den mit **GSVDISPSetDPoint** gesetzten Dezimalpunkt.

**Verwendete GSV Befehle:** 28 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:           int GSVDISPGetDPoint ( int **no** );

Delphi™:     GSVDISPGetDPoint ( **no**: Integer ): Integer;

VB:           GSVDISPGetDPoint ( ByVal **no** As Long ) As Long

**Rückgabewert:** Position des Dezimalpunktes in der Anzeige oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.72 GSVswitch

(Index 129)

#### Beschreibung:

Die Funktion GSVswitch schaltet den Schaltausgang gemäß des Parameters **swon** ein oder aus. Der Zustand des Schaltausganges bleibt nur erhalten, wenn der Meßwert zwischen den Schaltschwellen liegt. Mit **GSVsetThreshold** können die Schwellwerte gesetzt werden. Schwellwerte am Meßbereichsrand deaktivieren den Schwellwertschalter. Dann kann der Schaltausgang mit GSVswitch unabhängig vom Meßwert genutzt werden.

**Verwendete GSV Befehle:** 29 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                               int GSVswitch ( int **no**, int **swon** );

Delphi™:                       GSVswitch ( **no**, **swon**: Integer ): Integer;

VB:                             GSVswitch ( ByVal **no** As Long, ByVal **swon** As Long )  
  As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>swon</b>	Ganze Zahl im Bereich 0..1. Gewünschter Schalterzustand (aus oder ein).

### 3.73 GSVgetSerialNo

(Index 131)

#### **Beschreibung:**

Die Funktion GSVgetSerialNo ermittelt die Seriennummer des angeschlossenen GSV als ASCII-Text.

**Verwendete GSV Befehle:** 31 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

## Funktionsdefinition:

C:                   int GSVgetSerialNo ( int **no**, char \***number** );

Delphi™:           GSVgetSerialNo ( **no**: Integer; **number**: PChar ): Integer;

VB:                   GSVgetSerialNo ( ByVal **no** As Long, ByRef **number** As String )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**number**           Adresse einer Zeichenkettenvariable mit mindestens 9 Elementen.  
                          Rückgabe der Seriennummer als ASCII-Zeichenkette.  
                          (VB: Eine Zeichenkettenvariable veränderlicher Länge wird als Referenz  
                          übergeben.)

### 3.74 GSVsetThreshold

(Index 132)

#### Beschreibung:

Die Funktion GSVsetThreshold setzt den Ein- und den Ausschaltpunkt des Schwellwertschalters bzw. die obere und die untere Schaltschwelle, wenn der Fensterdiskriminator-Modus gewählt ist (siehe **GSVsetModeWindow**). Die Werte dürfen im Bereich von -1,05...+1,05 im Bipolar-Modus und von 0...+1,05 im Unipolar-Modus liegen. Dabei muß **thon** stets größer als **thoff** sein. Wird das Meßsignal größer als der mit **thon** angegebene Wert, so schaltet der Schwellwertausgang des Verstärkers auf GND. Unterschreitet das Meßsignal den mit **thoff** angegebenen Wert, so wird der Schwellwertausgang hochohmig geschaltet. Im Fensterdiskriminator-Modus erfolgt das Einschalten, wenn der Meßwert außerhalb der Schwellwerte liegt. Schwellwerte können nicht unabhängig für jeden Kanal gewählt werden.

**Verwendete GSV Befehle:** 32 (siehe GSV Bedienungsanleitung).

**Beeinflusste Register:** Schwellwert

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



### Funktionsdefinition:

```
C:      int GSVsetThreshold ( int no, double thon, double thoff );
```

Delphi™:        GSVsetThreshold ( **no**: Integer; **thon**, **thoff**: Double ): Integer;

VB: GSVsetThreshold ( ByVal **no** As Long, ByVal **thon** As Double, ByVal **thoff** As Double ) As Long

**Rückgabewert:** Fehlercode.

### Aufrufparameter:

**no** Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**thn** Gleitkommazahl im Bereich -1,05..+1,05.  
Gewünschte Einschaltsschwelle oder obere Schwelle des Fensterdiskriminators.

**thoff** Gleitkommazahl im Bereich -1,05..+1,05.  
Gewünschte Ausschaltschwelle oder untere Schwelle des  
Fensterdiskriminators.

### 3.75 GSVgetThreshold

(Index 133)

#### **Beschreibung:**

Die Funktion GSVgetThreshold liest die zuvor mit **GSVsetThreshold** gespeicherten Schwellwerte aus dem GSV.

**Verwendete GSV Befehle:** 33 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



### 3.76 GSVsetChannel

(Index 134)

#### Beschreibung:

Die Funktion GSVsetChannel legt den Meßkanal fest und stellt die für diesen Kanal vorher eingestellten Betriebsparameter wieder her (Einheit, Normierung und Schwellwert können nicht kanalunabhängig gewählt werden). Mit diesem Befehl kann zwischen dem Analogmeßeingang und dem Brückeneingang umgeschaltet werden. Nach einer Umschaltung auf den Analogmeßeingang gelten alle Befehle für diesen Eingang. Ein Offsetabgleich hat jedoch keinen Einfluß auf diesen Kanal. **GSVsetChannel** hat keine Wirkung auf den Analogausgang.

Channel = 0: Brückeneingang

Channel = 1: Analogmeßeingang

**Verwendete GSV Befehle:** 34 (siehe GSV Bedienungsanleitung).

**Beeinflusste Register:** Channel

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVsetChannel ( int **no**, int **channel** );

Delphi™:           GSVsetChannel ( **no**, **channel**: Integer ): Integer;

VB:                   GSVsetChannel ( ByVal **no** As Long, ByVal **channel** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**channel**            Ganze Zahl im Bereich 0..1.  
                          Nummer des gewünschten Eingangskanals.

### 3.77 GSVstopTransmit

(Index 135)

#### **Beschreibung:**

Die Funktion GSVstopTransmit stoppt die Datenübertragung vom GSV und löscht den Buffer des GSV und der seriellen Schnittstelle. Im Text-Modus ist diese Funktion zur Zeit ohne Wirkung.

**Verwendete GSV Befehle:** 35 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:               int GSVstopTransmit ( int **no** );

Delphi™:       GSVstopTransmit ( **no**: Integer ): Integer;

VB:             GSVstopTransmit ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**             Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.78 GSVstartTransmit

(Index 136)

#### **Beschreibung:**

Die Funktion GSVstartTransmit startet die Datenübertragung vom GSV und löscht den Buffer des GSV und der seriellen Schnittstelle.

**Verwendete GSV Befehle:** 36 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.



**Funktionsdefinition:**

C:               int GSVstartTransmit ( int **no** );

Delphi™:       GSVstartTransmit ( **no**: Integer ): Integer;

VB:             GSVstartTransmit ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**             Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.79 GSVclearBuffer

(Index 137)

#### Beschreibung:

Die Funktion GSVclearBuffer löscht den Inhalt des mit **GSVactivate** oder **GSVactivateExtended** eingerichteten Buffers sowie aller anderen Buffer im Datenpfad.

**Verwendete GSV Befehle:** 37 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Funktionsdefinition:**

C:                   int GSVclearBuffer ( int **no** );

Delphi™:           GSVclearBuffer ( **no**: Integer ): Integer;

VB:                   GSVclearBuffer ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.80 GSVsetMode

(Index 138)

#### Beschreibung:

Die Funktion GSVsetMode konfiguriert den GSV für diverse Betriebsarten. Der eingestellte Modus bleibt auch nach dem Abschalten erhalten. Vor dem Verändern des Mode-Registers ist dieses mit **GSVgetMode** zu lesen. Empfehlenswerter und einfacher ist es jedoch, die spezifischen Funktionen **GSVsetModeText**, **GSVsetModeLog**, etc. zu benutzen. Es dürfen nur Bit 1..6 der Bitmaske verändert werden. Beschreibung der Mode-Variable:

Bit 0	<b>reserviert, nicht verändern!</b>
Bit 1	Text-Modus
Bit 2	Max-Modus
Bit 3	Log-Modus
Bit 4	Fensterdiskriminator-Modus
Bit 5	Mittelwert-Modus
Bit 6	Linearisierungsmodus
Bit 7	Sperre, <b>nicht verändern!</b>

Die Text-Modus Einstellung wird zur Zeit bei der Herstellung des GSV festgelegt und kann nicht umgeschaltet werden.

Text-Modus = 1: Übertragung in Text-Format ist aktiv (nur für Messwerte)

Max-Modus = 1: Maximalwertübertragung ist aktiv

Log-Modus = 1: Übertragung von Messwerten nur auf Anforderung ist aktiv

Fenster-Modus = 1: Schwellwertschalter wirkt als Fensterdiskriminator

Mittelwert-Modus = 1: Mittelung über jeweils 16 Messwerte

Linearisierungsmodus = 1: Linearisierungsberechnung für einen bestimmten Sensor

Sperre = 1: Sperre der Befehle: 5..10, 15..21, 29, 32, 35, 38, 44, 46, 48, 60, 67, 128  
130, 132, 134, 136, 138, 140, 142, 144, ...

**Verwendete GSV Befehle:** 38 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

## Funktionsdefinition:

C:                   int GSVsetMode ( int **no**, int **mode** );

Delphi™:           GSVsetMode ( **no**, **mode**: Integer ): Integer;

VB:                   GSVsetMode ( ByVal **no** As Long, ByVal **mode** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                Ganze Zahl im Bereich 1..1024.  
                    Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**mode**            Ganze Zahl als Bitmaske zu interpretieren, nur entsprechend  
                    dokumentierte Bits dürfen gesetzt werden.  
                    Gewünschte Betriebsart als Bitmaske aus einzelnen Bits für verschiedene  
                    Teil- bzw. Unterbetriebsarten.

### 3.81 GSVgetMode

(Index 139)

#### Beschreibung:

Die Funktion GSVgetMode liest den eingestellten Modus des GSVs, siehe **GSVsetMode**. Es wird jedoch empfohlen statt dieser Funktion die spezifischen Funktionen **GSVgetModeText**, **GSVgetModeLog**, etc. zu verwenden.

**Verwendete GSV Befehle:** 39 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:                   int GSVgetMode ( int **no** );

Delphi™:           GSVgetMode ( **no**: Integer ): Integer;

VB:                 GSVgetMode ( ByVal **no** As Long ) As Long

**Rückgabewert:** Bitmaske der Betriebsart oder Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.82 GSVgetEquipment

(Index 141)

#### Beschreibung:

Die Funktion GSVgetEquipment liest Informationen über die Hardware-Konfiguration des GSV. Es wird jedoch empfohlen, statt dieser Funktion die spezifischen Funktionen **GSVhasADC**, **GSVhasLCD**, etc. zu verwenden.

Beschreibung der Equipment-Variable:

Bit 0	LCD, Flüssigkristallanzeige installiert
Bit 1	ADC, Analog-Digital-Konverter vorhanden
Bit 2	UII, User Input Interface, d.h. Tastatur o.ä.
Bit 3	SI, reserviert, <b>Einstellungen eines derartigen Gerätes nicht verändern !!</b>
Bit 4	WL, Funkübertragung der Datenkommunikation
Bit 5	AF, Analoges Filter vorhanden
Bit 6	reserviert
Bit 7	reserviert

**Verwendete GSV Befehle:** 41 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.



**Funktionsdefinition:**

C:           int GSVgetEquipment ( int **no** );

Delphi™:       GSVgetEquipment ( **no**: Integer ): Integer;

VB:           GSVgetEquipment ( ByVal **no** As Long ) As Long

**Rückgabewert:** Bitmaske der Ausstattung des GSV oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.83 GSVfirmwareVersion

(Index 143)

#### **Beschreibung:**

Die Funktion GSVfirmwareVersion liest die Versionsnummer der Firmware des GSV. Dabei enthält das zweitniederwertige Byte (Bit 8..15) "das Zehnfache" der Versionsnummer. Das niederwertigste Byte (Bit 0..7) enthält die Revisionsnummer. Alle anderen Bytes des Funktionsergebnisses sind 0 (bei fehlerfreier Ausführung).

**Verwendete GSV Befehle:** 43 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:        long GSVfirmwareVersion ( int **no** );

Delphi™:     GSVfirmwareVersion ( **no**: Integer ): LongInt;

VB:         GSVfirmwareVersion ( ByVal **no** As Long ) As Long

**Rückgabewert:** Firmware-Versionsnummer oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.84 GSVsetGageFactor

(Index 144)

#### Beschreibung:

Die Funktion GSVsetGageFactor speichert den Kennwert  $k$  des verwendeten Dehnungsmeßstreifens im GSV. Dies ermöglicht dem GSV (in Verbindung mit anderen Informationen) die erforderliche Anzeigenormierung eigenständig zu berechnen (Siehe auch **GSVsetBridge**, **GSVsetPoisson** und **GSVDispCalcNorm**).

**Verwendete GSV Befehle:** 44 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

## Funktionsdefinition:

C:           int GSVsetGageFactor ( int **no**, double **gf** );

Delphi™:       GSVsetGageFactor ( **no**: Integer; **gf**: Double ): Integer;

VB:            GSVsetGageFactor ( ByVal **no** As Long, ByVal **gf** As Double ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                 Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**gf**            Gleitkommazahl im Bereich 0,01..655,35.  
                 Kennwert *k* des Dehnungsmeßstreifens.

### 3.85 GSVgetGageFactor

(Index 145)

#### Beschreibung:

Die Funktion GSVgetGageFactor liest den Kennwert  $k$  aus dem GSV, der dort vorher durch **GSVsetGageFactor** gespeichert wurde.

**Verwendete GSV Befehle:** 45 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

### Funktionsdefinition:

C:        double GSVgetGageFactor ( int **no** );

Delphi™:        GSVgetGageFactor ( **no**: Integer ): Double;

VB:                GSVgetGageFactor ( ByVal **no** As Long ) As Double

**Rückgabewert:** Kennwert *k* oder Fehlercode.

### Aufrufparameter:

**no**                Ganze Zahl im Bereich 1..1024.  
                      Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.86 GSVsetPoisson

(Index 146)

#### **Beschreibung:**

Die Funktion GSVsetPoisson speichert die Materialkonstante Querkontraktion im GSV. Dies ermöglicht dem GSV (in Verbindung mit anderen Informationen) die erforderliche Anzeigenormierung eigenständig zu berechnen (Siehe auch **GSVsetBridge**, **GSVsetGageFactor** und **GSVDispCalcNorm**).

**Verwendete GSV Befehle:** 46 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.



## Funktionsdefinition:

C:                   int GSVsetPoisson ( int **no**, double **poiss** );

Delphi™:           GSVsetPoisson ( **no**: Integer; **poiss**: Double ): Integer;

VB:                   GSVsetPoisson ( ByVal **no** As Long, ByVal **poiss** As Double )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**poiss**               Gleitkommazahl im Bereich 0,00..0,50.  
                          Querkontraktion.

### 3.87 GSVgetPoisson

(Index 147)

#### **Beschreibung:**

Die Funktion GSVgetPoisson liest die Materialkonstante Querkontraktion aus dem GSV, die dort vorher durch **GSVsetPoisson** gespeichert wurde.

**Verwendete GSV Befehle:** 47 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

**Funktionsdefinition:**

C:           double GSVgetPoisson ( int **no** );

Delphi™:       GSVgetPoisson ( **no**: Integer ): Double;

VB:            GSVgetPoisson ( ByVal **no** As Long ) As Double

**Rückgabewert:** Querkontraktion oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.88 GSVsetBridge

(Index 148)

#### Beschreibung:

Die Funktion GSVsetBridge legt die Art der verwendeten Widerstandsbrücke fest. Dies ermöglicht dem GSV (in Verbindung mit anderen Informationen) die erforderliche Anzeigenormierung eigenständig zu berechnen (Siehe auch **GSVsetGageFactor**, **GSVsetPoisson** und **GSVDispCalcNorm**).

Die Beschreibung der Widerstandsbrücke erfolgt durch folgende Bitfelder:

Bit 0	Brückentyp (codierter Wert)
Bit 1	
Bit 2	
Bit 3	<b>reserviert, nicht ändern</b>
Bit 4	
Bit 5	
Bit 6	
Bit 7	

Vor dem Ändern von Bitfeldern muß die Art der Widerstandsbrücke durch GSVgetBridge gelesen werden. Empfehlenswerter und einfacher ist es jedoch, die spezifische Funktion **GSVsetBridgeType** zu benutzen. Eine Beschreibung des Brückentyps befindet sich bei der Funktion **GSVsetBridgeType**.

**Verwendete GSV Befehle:** 48 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

## Funktionsdefinition:

C:                   int GSVsetBridge ( int **no**, int **br** );

Delphi™:           GSVsetBridge ( **no**, **br**: Integer ): Integer;

VB:                 GSVsetBridge ( ByVal **no** As Long, ByVal **br** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**br**                Ganze Zahl als Bitmaske zu interpretieren, nur entsprechend  
dokumentierte Bits dürfen verändert werden.  
Gewünschte Art der Widerstandsbrücke als Bitmaske aus einzelnen  
Feldern für verschiedene Teilinformationen.

### 3.89 GSVgetBridge

(Index 149)

#### Beschreibung:

Die Funktion GSVgetBridge liest die aktuell eingestellte Art der Widerstandsbrücke vom GSV. Die Beschreibung der Widerstandbrücke erfolgt durch Bitfelder die bei **GSVsetBridge** beschrieben sind.

Empfehlenswerter und einfacher ist es jedoch, die spezifische Funktion **GSVgetBridgeType** zu benutzen.

**Verwendete GSV Befehle:** 49 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

## Funktionsdefinition:

C:                   int GSVgetBridge ( int **no** );

Delphi™:           GSVgetBridge ( **no**: Integer ): Integer;

VB:                 GSVgetBridge ( ByVal **no** As Long ) As Long

**Rückgabewert:** Bitfelder für die Art der Widerstandsbrücke oder Fehlercode.

## Aufrufparameter:

**no**                Ganze Zahl im Bereich 1..1024.  
                     Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.90 GSVgetRange

(Index 151)

#### Beschreibung:

Die Funktion GSVgetRange liest die Eingangsempfindlichkeit des Eingangskanals 0 (Messbereich) des GSV. Ergebnis in mV/V. Ist das Ergebnis gleich Null, steht die Möglichkeit des Lesens der Eingangsempfindlichkeit nicht zur Verfügung.

Ältere Versionen des GSV-2, die diesen Befehl nicht unterstützen, haben – sofern es sich um kein besonderes oder kundenspezifisches Modell handelt – folgende Eingangsempfindlichkeiten:

GSV-2	JP1 offen	JP1 gesetzt	Sondermodell
Eingangsempfindlichkeit	1 mV/V	2 mV/V	individuell

**Verwendete GSV Befehle:** 51 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.



**Funktionsdefinition:**

C:               double GSVgetRange ( int **no** );

Delphi™:               GSVgetRange ( **no**: Integer ): Double;

VB:               GSVgetRange ( ByVal **no** As Long ) As Double

**Rückgabewert:** Eingangsempfindlichkeit des GSV, Null oder Fehlercode.

**Aufrufparameter:**

**no**               Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.91 MEgetOffsetWait

(Index 153)

#### **Beschreibung:**

Die Funktion MEgetOffsetWait liest die Wartezeit, die nach **GSVsetOffset** mindestens eingehalten werden muß. Das Funktionsergebnis ist als Sekunden zu interpretieren.

**Verwendete GSV Befehle:** 53 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:           double MEgetOffsetWait ( int **no** );

Delphi™:       MEgetOffsetWait ( **no**: Integer ): Double;

VB:            MEgetOffsetWait ( ByVal **no** As Long ) As Double

**Rückgabewert:** Wartezeit oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.92 GSVgetOptions

(Index 154)

#### Beschreibung:

Die Funktion GSVgetOptions liest Informationen über besondere Eigenschaften der Firmware. Die niederwertigen 6 Bit (0..5) sind als (ganze) Zahl im Bereich 0..63 zu interpretieren und enthalten die Identifikation einer etwaigen Sonderanwendung. **Ist diese Identifikation verschieden von Null, muß mit Einschränkungen oder Abweichungen der Funktion der Firmware gerechnet werden.** Die darauf folgenden 18 Bit (6..23) sind folgendermaßen definiert bzw. reserviert:

Bit 6	Erweiterungen, die mit dem GSV-3 eingeführt wurden, sind implementiert
Bit 7	Linearisierungsberechnung ist implementiert
Bit 8	Erweiterungen, die mit dem GSV-2.1 eingeführt wurden, sind implementiert
Bit 9	Stromsparmmodus ist implementiert
Bit 10	<b>GSVisCommandAvailable</b> ist implementiert
Bit 11..23	reserviert (=0)

Es wird jedoch empfohlen statt dieser Funktion die spezifische Funktion **GSVgetOptionsCode** zu verwenden, um die Identifikation einer etwaigen Sonderanwendung zu erhalten.

**Verwendete GSV Befehle:** 54 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1.

**Funktionsdefinition:**

C:                long GSVgetOptions ( int **no** );

Delphi™:            GSVgetOptions ( **no**: Integer ): LongInt;

VB:                GSVgetOptions ( ByVal **no** As Long ) As Long

**Rückgabewert:** Bitmaske der Firmware-Optionen oder Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.93 GSVgetValue

(Index 159)

#### Beschreibung:

Die Funktion GSVgetValue löst die Übertragung eines Meßwertes durch die GSV-Baugruppe aus. Da normalerweise laufend Meßwerte von der Baugruppe übertragen werden, ist diese Funktion nur von Bedeutung, wenn die Meßwertübertragung durch **GSVstopTransmit** ausgeschaltet wurde, oder der Logger-Modus aktiv ist. Der Logger-Modus wird durch **GSVsetModeLog** ein- oder ausgeschaltet (siehe auch **GSVgetModeLog**, **GSVgetMode** und **GSVsetMode**). Der ausgelöste Meßwert wird, wie alle übertragenen Meßwerte, mit **GSVread** abgeholt.

**Verwendete GSV Befehle:** 59 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1, Firmware Version 3.2-19.

**Funktionsdefinition:**

C:                   int GSVgetValue ( int **no** );

Delphi™:           GSVgetValue ( **no**: Integer ): Integer;

VB:                 GSVgetValue ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
                    Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.94 GSVclearMaxValue

(Index 160)

#### Beschreibung:

Die Funktion GSVclearMaxValue setzt im Maximum-Modus den bisher gemessenen Maximalwert zurück, so daß ein neuer Maximalwert gebildet werden kann. Der Maximum-Modus wird durch **GSVsetModeMax** ein- oder ausgeschaltet (siehe auch **GSVgetModeMax**, **GSVgetMode** und **GSVsetMode**).

**Verwendete GSV Befehle:** 60 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 1, Firmware Version 3.2-19.



**Funktionsdefinition:**

C:           int GSVclearMaxValue ( int **no** );

Delphi™:       GSVclearMaxValue ( **no**: Integer ): Integer;

VB:            GSVclearMaxValue ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.95 GSVDISPSetDigits

(Index 161)

#### **Beschreibung:**

Die Funktion GSVDISPSetDigits stellt die Anzahl der in der Anzeige des GSV darzustellenden Ziffern ein (siehe auch **GSVDISPGetDigits**). Bei der GSV-Version mit ASCII-Ausgabe wird gleichzeitig auch die Anzahl der übertragenen Ziffern-Bytes gesetzt.

**Verwendete GSV Befehle:** 61 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.



### 3.96 GSVDISPGetDigits

(Index 162)

#### **Beschreibung:**

Die Funktion GSVDISPGetDigits liest die aktuelle Einstellung der Anzahl der in der Anzeige des GSV darzustellenden Ziffern (siehe auch **GSVDISPSetDigits**).

**Verwendete GSV Befehle:** 62 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

**Funktionsdefinition:**

C:               int GSVDISPGetDigits ( int **no** );

Delphi™:       GSVDISPGetDigits ( **no**: Integer ): Integer;

VB:             GSVDISPGetDigits ( ByVal **no** As Long ) As Long

**Rückgabewert:** Anzahl der darzustellenden Ziffern oder Fehlercode.

**Aufrufparameter:**

**no**             Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.97 GSVunlockUII

(Index 163)

#### Beschreibung:

Die Funktion GSVunlockUII macht das vorangegangene Sperren des UII (User Input Interface, d.h. Tastatur o.ä.) des GSV rückgängig (siehe auch **GSVlockUII**, **GSVhasUII**, **GSVactivate** und **GSVactivateExtended** und **GSVrelease**).

**Verwendete GSV Befehle:** 63 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

**Funktionsdefinition:**

C:                   int GSVunlockUII ( int **no** );

Delphi™:           GSVunlockUII ( **no**: Integer ): Integer;

VB:                 GSVunlockUII ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.98 GSVlockUII

(Index 164)

#### Beschreibung:

Die Funktion GSVlockUII sperrt ein eventuell vorhandenes UII (User Input Interface, d.h. Tastatur o.ä.) des GSV gegen unerwünschte Benutzung. Der Rückgabewert GSV\_OK bedeutet, daß die Funktion zwar fehlerfrei ausgeführt wurde, die Sperre jedoch nicht gesetzt werden konnte. Bei erfolgreicher Sperrung ist der Rückgabewert GSV\_TRUE. Diese Sperre wird durch **GSVunlockUII** rückgängig gemacht (oder durch Ausschalten des GSV). Siehe auch **GSVhasUII**, **GSVactivate** und **GSVactivateExtended** und **GSVrelease**.

**Verwendete GSV Befehle:** 64 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.



**Funktionsdefinition:**

C:                           int GSVlockUII ( int **no** );

Delphi™:                   GSVlockUII ( **no**: Integer ): Integer;

VB:                           GSVlockUII ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                    Ganze Zahl im Bereich 1..1024.  
                        Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.99 GSVgetLastError

(Index 166)

#### **Beschreibung:**

Die Funktion GSVgetLastError fragt vom GSV den Fehlercode des zuletzt ausgeführten Befehls ab.

**Verwendete GSV Befehle:** 66 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 3.

**Funktionsdefinition:**

C:                   int GSVgetLastError ( int **no** );

Delphi™:           GSVgetLastError ( **no**: Integer ): Integer;

VB:                   GSVgetLastError ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode des GSV oder Fehlercode.

**Aufrufparameter:**

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.100 GSVsetSecondThreshold

(Index 167)

#### **Beschreibung:**

Die Funktion GSVsetSecondThreshold setzt den Ein- und den Ausschaltpunkt des zweiten Schwellwertschalters. Die Funktion wird verwendet wie **GSVsetThreshold**, (siehe auch dort) wirkt jedoch auf den zweiten Schwellwertschalter, der z.B. im GSV-2.1 enthalten ist.

Siehe auch **GSVgetOptionsExtension21**.

**Verwendete GSV Befehle:** 67 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-2.1.



### 3.101 GSVgetSecondThreshold

(Index 168)

#### **Beschreibung:**

Die Funktion GSVgetSecondThreshold liest die zuvor mit **GSVsetSecondThreshold** gespeicherten Schwellwerte aus dem GSV.

Siehe auch **GSVgetOptionsExtension21**.

**Verwendete GSV Befehle:** 68 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-2.1.



### 3.102 GSVgetDeviceType

(Index 169)

#### Beschreibung:

Die Funktion GSVgetDeviceType fragt vom GSV einen Code ab, der den Gerätetyp identifiziert. Folgende Gerätetypen sind definiert:

Code	Gerät
2	GSV-2
3	GSV-3
21	GSV-2.1

**Verwendete GSV Befehle:** 69 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 3.



**Funktionsdefinition:**

C:           int GSVgetDeviceType ( int **no** );

Delphi™:     GSVgetDeviceType ( **no**: Integer ): Integer;

VB:           GSVgetDeviceType ( ByVal **no** As Long ) As Long

**Rückgabewert:** Code für den Gerätetyp oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.103 GSVDISPCalcNorm

(Index 170)

#### Beschreibung:

Die Funktion GSVDISPCalcNorm errechnet aus vorher gespeicherten Informationen die Anzeigenormierung des GSV (gemäß Dehnungsindikatorfunktionalität) und setzt diese als aktuelle Anzeigenormierung. Berücksichtigt werden dabei der Kennwert *k*, die Brückenart, die Eingangsempfindlichkeit und eventuell – je nach Brückenart – die Querkontraktionszahl (siehe auch **GSVsetBridge**, **GSVsetGageFactor** und **GSVsetPoisson**).

**Verwendete GSV Befehle:** 70, 26, 28 (siehe GSV Bedienungsanleitung).

#### Bemerkung:

Nach dem Aufruf dieser Funktion sollte **GSVflushBuffer** aufgerufen werden, um veraltete und verfälschte Messwerte zu verwerfen.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 2.

**Funktionsdefinition:**

C:           int GSVDISPCalcNorm ( int **no** );

Delphi™:       GSVDISPCalcNorm ( **no**: Integer ): Integer;

VB:            GSVDISPCalcNorm ( ByVal **no** As Long ) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.104 GSVsetTxMode

(Index 228)

#### Beschreibung:

Die Funktion GSVsetTxMode konfiguriert den Datenübertragungsmodus des GSV. Der eingestellte Modus bleibt auch nach dem Abschalten erhalten. Vor dem Verändern des Übertragungsmodus-Registers ist dieses mit **GSVgetTxMode** zu lesen.

Empfehlenswerter und einfacher ist es jedoch, die spezifischen Funktionen

**GSVsetTxModeTransmit4**, **GSVsetTxModeRepeat3**, etc. zu benutzen.

Beschreibung der Übertragungsmodus-Variable:

Bit 0	Konfigurations-Modus; <b>nicht veränderbar!</b>
Bit 1	4-Byte Protokoll
Bit 2	3-faches Senden im Slow-Modus
Bit 3	5-Byte Protokoll (hat Vorrang vor Bit 1)
Bit 4	Schnittstelle hat nur Lesezugriff
Bit 5..7	<b>reserviert, nicht verändern!</b>

Der Konfigurations-Modus wird durch eine Steckbrücke festgelegt und kann nicht durch Software umgeschaltet werden.

Konfigurations-Modus = 1: Änderung von Übertragungsmodus und Baudrate erlaubt

4-Byte Protokoll = 1: 4 Bytes pro Messwert senden (mit Status/Adresse), statt 3 Byte

3-faches Senden = 1: Im Slow-Modus wird jeder Messwert dreifach gesendet

5-Byte Protokoll = 1: 5 Bytes pro Messwert senden (mit Status), statt 3 oder 4 Byte

Nur Lesezugriff = 1: Einstellungen können nur gelesen werden (sonst auch geändert)

Siehe auch **GSVgetOptionsExtension3**.

#### Achtung:

GSVsetTxMode kann nur im Konfigurationsmodus des GSV erfolgreich durchgeführt werden und wird sonst ignoriert (siehe auch **GSVgetTxModeConfig**).

**Verwendete GSV Befehle:** 128 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3, sowie ab GSV Modell 4.

## Funktionsdefinition:

C:                   int GSVsetTxMode ( int **no**, int **txmode** );

Delphi™:           GSVsetTxMode ( **no**, **txmode**: Integer ): Integer;

VB:                   GSVsetTxMode ( ByVal **no** As Long, ByVal **txmode** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                    Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**txmode**            Ganze Zahl als Bitmaske zu interpretieren, nur entsprechend  
                          dokumentierte Bits dürfen gesetzt werden.  
                          Gewünschte Datenübertragungsart als Bitmaske aus einzelnen Bits für  
                          verschiedene Teil- bzw. Unterübertragungsarten.

### 3.105 GSVgetTxMode

(Index 229)

#### Beschreibung:

Die Funktion GSVgetTxMode liest den eingestellten Datenübertragungsmodus des GSVs, siehe **GSVsetTxMode**. Es wird jedoch empfohlen statt dieser Funktion die spezifischen Funktionen **GSVgetTxModeTransmit4**, **GSVgetTxModeRepeat3**, etc. zu verwenden.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 129 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3, sowie ab GSV Modell 4.

**Funktionsdefinition:**

C:                   int GSVgetTxMode ( int **no** );

Delphi™:           GSVgetTxMode ( **no**: Integer ): Integer;

VB:                   GSVgetTxMode ( ByVal **no** As Long ) As Long

**Rückgabewert:** Bitmaske der Datenübertragungsart oder Fehlercode.

**Aufrufparameter:**

**no**                   Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.106 GSVsetBaud

(Index 230)

#### Beschreibung:

Die Funktion GSVsetBaud legt die Baudrate der Schnittstelle des GSVs gemäß der folgenden Tabelle fest:

baud	Baudrate
0	4800
1	9600
2	19200
3	38400
4	57600
5	115200
6	250000
7	625000
8	1250000

Siehe auch **GSVgetOptionsExtension3**.

#### Achtung:

GSVsetBaud kann nur im Konfigurationsmodus des GSV erfolgreich durchgeführt werden und wird sonst ignoriert (siehe auch **GSVgetTxModeConfig**).

**Verwendete GSV Befehle:** 130 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3, sowie ab GSV Modell 4.



### Funktionsdefinition:

```
C:      int GSVsetBaud ( int no, int baud );
```

Delphi™:                    GSVsetBaud ( **no**, **baud**: Integer ): Integer;

VB: GSVsetBaud ( ByVal **no** As Long, ByVal **baud** As Long )  
As Long

**Rückgabewert:** Fehlercode.

### Aufrufparameter:

**no** Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

<b>baud</b>	Ganze Zahl im Bereich 0..8. Code für die gewünschte Baudrate.
-------------	--

### 3.107 GSVgetBaud

(Index 231)

#### **Beschreibung:**

Die Funktion GSVgetBaud liest die aktuell eingestellte Baudrate vom GSV. Das Ergebnis ist ein codierter Wert, der bei **GSVsetBaud** erläutert ist.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 131 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3, sowie ab GSV Modell 4.

**Funktionsdefinition:**

C:                   int GSVgetBaud ( int **no** );

Delphi™:           GSVgetBaud ( **no**: Integer ): Integer;

VB:                GSVgetBaud ( ByVal **no** As Long ) As Long

**Rückgabewert:** Code für die Baudrate oder Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.108 GSVsetSlowRate

(Index 234)

#### Beschreibung:

Die Funktion GSVsetSlowRate legt die Anzahl der Sekunden fest, die im Slow-Mode (siehe auch **GSVsetSpecialModeSlow**) zwischen zwei Messungen (d.h. zwischen zwei Meßdaten-Übertragungen) vergehen. Der Slow-Modus ermöglicht Meßraten < 1 Hz. Die niedrige Transferrate kann den Stromverbrauch senken.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 134 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:                   int GSVsetSlowRate ( int **no**, long **secs** );

Delphi™:           GSVsetSlowRate ( **no**, **secs**: LongInt ): Integer;

VB:                   GSVsetSlowRate ( ByVal **no** As Long, ByVal **secs** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>secs</b>	Ganze Zahl im Bereich 1..65535. Anzahl der Sekunden zwischen zwei Messungen, d.h. der Kehrwert der Meßrate.

### 3.109 GSVgetSlowRate

(Index 235)

#### Beschreibung:

Die Funktion GSVgetSlowRate liest die Anzahl der Sekunden aus dem GSV, die im Slow-Mode (siehe auch **GSVgetSpecialModeSlow**) zwischen zwei Messungen (d.h. zwischen zwei Meßdaten-Übertragungen) vergehen. Der Slow-Modus ermöglicht Meßraten < 1 Hz. Die niedrige Transferrate kann den Stromverbrauch senken.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 135 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:            long GSVgetSlowRate ( int **no** );

Delphi™:        GSVgetSlowRate ( **no**: Integer ): LongInt;

VB:            GSVgetSlowRate ( ByVal **no** As Long ) As Long

**Rückgabewert:** Anzahl der Sekunden zwischen zwei Messungen oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.110 GSVsetSpecialMode

(Index 236)

#### Beschreibung:

Die Funktion GSVsetSpecialMode konfiguriert den besonderen Messmodus des GSV. Der eingestellte Modus bleibt auch nach dem Abschalten erhalten. Vor dem Verändern des Messmodus-Registers ist dieses mit **GSVgetSpecialMode** zu lesen.

Empfehlenswerter und einfacher ist es jedoch, die spezifischen Funktionen **GSVsetSpecialModeSlow**, **GSVsetSpecialModeFilter**, etc. zu benutzen.

Beschreibung der Messmodus-Variable:

Bit 0	Slow-Modus, für Übertragungsrate < 1 Hz, siehe <b>GSVsetSpecialModeSlow</b>
Bit 1	Mittelung innerhalb der Baugruppe, siehe <b>GSVgetSpecialModeAverage</b>
Bit 2	FIR-Tiefpass (2./5. Ordnung) ist aktiv, siehe <b>GSVsetSpecialModeFilter</b>
Bit 3	Maximum-Ereignis-Modus, siehe <b>GSVsetSpecialModeMax</b>
Bit 4	Automatische Einstellung des Analogfilters
Bit 5	FIR-Tiefpass hat 2. Ordnung, statt 5. Ordnung
Bit 6	Sleep-Modus, Energiesparmodus, siehe <b>GSVsetSpecialModeSleep</b>
Bit 7	Unipolar, es werden nur positive Ausschläge gemessen
Bit 8..13	<b>reserviert, nicht verändern!</b>
Bit 14	Nullnachführung (auto zero), siehe <b>GSVsetSpecialModeAutoZero</b>
Bit 15	Rauschunterdrückung (noise cut), siehe <b>GSVsetSpecialModeNoiseCut</b>

Der Mittelwert-Modus wird durch **GSVwriteSamplingRate** und **GSVsetFreq** festgelegt und kann beim GSV-3 in diesem Register nicht umgeschaltet werden.

Der Unipolar-Modus wird durch **GSVsetUnipolar** und **GSVsetBipolar** festgelegt und kann beim GSV-3 in diesem Register nicht umgeschaltet werden.

Maximum-Ereignis-Modus = 1: Nur neue Maximalwerte übertragen (Wirkt nur in Verbindung mit dem Maximum-Modus.)

Nullnachführung = 1: Aktiv, Zeitintervall setzen mit **GSVsetAutoZeroCounter**

Rauschunterdrückung = 1: Aktiv, Schwelle setzen mit **GSVsetNoiseCutThreshold**

Siehe auch **GSVgetOptionsExtension3**, **GSVsetModeMax**, **GSVsetSlowRate**.

**Verwendete GSV Befehle:** 136 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.





### 3.111 GSVgetSpecialMode

(Index 237)

#### Beschreibung:

Die Funktion `GSVgetSpecialMode` liest den eingestellten besonderen Messmodus des GSVs, siehe ***GSVsetSpecialMode***. Es wird jedoch empfohlen statt dieser Funktion die spezifischen Funktionen ***GSVgetSpecialModeSlow***, ***GSVgetSpecialModeAverage***, etc. zu verwenden.

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch `GSV_ERROR` sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:           int GSVgetSpecialMode ( int **no** );

Delphi™:     GSVgetSpecialMode ( **no**: Integer ): Integer;

VB:           GSVgetSpecialMode ( ByVal **no** As Long ) As Long

**Rückgabewert:** Bitmaske der besonderen Messart oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.112 GSVwriteSamplingRate

(Index 238)

#### Beschreibung:

Die Funktion GSVwriteSamplingRate legt gleichzeitig die ADC-Abtastrate, sowie die Meßdaten-Übertragungsrate fest. Angegeben wird dazu die gewünschte ADC-Abtastrate und der Faktor, um den die Meßdaten-Übertragungsrate niedriger als die ADC-Abtastrate sein soll. Die gegebenenfalls höhere Rate der Meßwerte (verglichen mit der Rate der übertragenen Meßdaten) wird zur Mittelung verwendet (siehe auch **GSVgetSpecialModeAverage**).

Der zulässige Bereich für die ADC-Abtastrate ist beim GSV-3 ca. 76,3 .. 10080 Hz. Die zulässigen Faktoren sind beim GSV-3: 1, 2, 4, 8, 16, 32, 64, 128 und 256. Die sich ergebende Datenübertragungsrate muß in jedem Fall kleiner als 1220,7 Hz sein und unterliegt außerdem bei niedrigeren eingestellten Baudraten folgenden Beschränkungen:

Baudrate	Maximale Datenrate [Hz]
4800	158,79
9600	315,02
19200	630,04

Im Logger-Modus (siehe **GSVsetModeLog**) sowie im Maximum-Ereignis-Modus (siehe **GSVsetSpecialModeMax**) darf stets die höchste Datenrate von 1220,7 Hz gesetzt werden.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 138 (siehe GSV Bedienungsanleitung).

#### Achtung:

Nach einer Änderung der Abtastrate ist ein Abgleich mit **GSVsetCal** und **GSVsetZero** notwendig.

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:        int GSVwriteSamplingRate ( int **no**, double **freq**, int **factor** );

Delphi™:    GSVwriteSamplingRate ( **no**: Integer; **freq**: Double; **factor**: Integer ): Integer;

VB:        GSVwriteSamplingRate ( ByVal **no** As Long, ByVal **freq** As Double, ByVal **factor** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>freq</b>	Gleitkommazahl, der Bereich hängt von weiteren Bedingungen ab. Gewünschte ADC-Abtastfrequenz des GSV.
<b>factor</b>	Ganze Zahl im Bereich $2^{0..8}$ . Verhältnis zwischen ADC-Abtastrate und Meßdaten-Übertragungsrate.

### 3.113 GSVreadSamplingRate

(Index 239)

#### **Beschreibung:**

Die Funktion GSVreadSamplingRate liest gleichzeitig die ADC-Abtastrate, sowie das Verhältnis zwischen ADC-Abtastrate und Meßdaten-Übertragungsrate aus dem GSV. Bei einem Verhältnis  $> 1$  wird die höhere Rate der Meßwerte (verglichen mit der Rate der übertragenen Meßdaten) zur Mittelung verwendet (siehe auch **GSVgetSpecialModeAverage**).

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 139 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:        int GSVreadSamplingRate ( int **no**, double **\*freq**, int **\*factor** );

Delphi™:    GSVreadSamplingRate ( **no**: Integer; var **freq**: Double; var **factor**: Integer ): Integer;

VB:        GSVreadSamplingRate ( ByVal **no** As Long, ByRef **freq** As Double, ByRef **factor** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>freq</b>	Zeiger auf Gleitkommavariablen doppelter Genauigkeit. Rückgabe der ADC-Abtastfrequenz des GSV. (Delphi™ und VB: Die Variable wird als Referenz übergeben.)
<b>factor</b>	Zeiger auf ganzzahlige Variable. Rückgabe des Verhältnisses zwischen ADC-Abtastfrequenz und Meßdaten-Übertragungsrate. (Delphi™ und VB: Die Variable wird als Referenz übergeben.)

### 3.114 GSVsetCanSetting

(Index 240)

#### Beschreibung:

Die Funktion GSVsetCanSetting setzt verschiedene Einstellungen der CAN-Bus Schnittstelle des GSV. Der entsprechende Zahlenwert wird als Parameter übergeben. Welche Einstellung gesetzt werden soll, muss ebenfalls als Parameter angegeben werden entsprechend folgender Tabelle:

Zu setzende Einstellung	stype
Befehls-ID	GSV_CANSET_COMMAND_ID
Meldungs-ID	GSV_CANSET_MESSAGE_ID
Baudrate (codierter Wert)	GSV_CANSET_BAUD
Bitmaske diverser Einstellungen	GSV_CANSET_FLAGS

Diese Einstellungen sind allgemeiner Art oder beziehen sich auf den Betrieb entsprechend CAN2.0A. Alternative oder zusätzliche Einstellungen für den Betrieb entsprechend CAN2.0B können durch Verändern des **stype** mit der Konstanten GSV\_CANSET\_20B gelesen werden.

Siehe auch **GSVisCanAvailable**.

**Verwendete GSV Befehle:** 140 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.





### 3.115 GSVgetCanSetting

(Index 241)

#### Beschreibung:

Die Funktion GSVgetCanSetting liest aktuelle Einstellungen der CAN-Bus Schnittstelle vom GSV. Das Ergebnis ist der entsprechende Zahlenwert. Welche Einstellung gelesen werden soll, muss als Parameter angegeben werden entsprechend folgender Tabelle:

Zu lesende Einstellung	stype
Hersteller-ID	GSV_CANSET_MANUFACTURER_ID
Befehls-ID	GSV_CANSET_COMMAND_ID
Meldungs-ID	GSV_CANSET_MESSAGE_ID
Baudrate (codierter Wert)	GSV_CANSET_BAUD
Bitmaske diverser Einstellungen	GSV_CANSET_FLAGS

Diese Einstellungen sind allgemeiner Art oder beziehen sich auf den Betrieb entsprechend CAN2.0A. Alternative oder zusätzliche Einstellungen für den Betrieb entsprechend CAN2.0B können durch Verändern des **stype** mit der Konstanten GSV\_CANSET\_20B gelesen werden.

Siehe auch **GSVisCanAvailable**.

**Verwendete GSV Befehle:** 141 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.

**Funktionsdefinition:**

C:               int GSVgetCanSetting ( int **no**, int **stype** );

Delphi™:       GSVgetCanSetting ( **no**, **stype**: Integer ): Integer;

VB:             GSVgetCanSetting ( ByVal **no** As Long,  
  ByVal **stype** As Long ) As Long

**Rückgabewert:** Zahlenwert der Einstellung oder Fehlercode.

**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>stype</b>	Ganze Zahl im Bereich 0..255. Code der Einstellung, die gelesen werden soll.

### 3.116 GSVsetAnalogFilter

(Index 244)

#### Beschreibung:

Die Funktion GSVsetAnalogFilter setzt die -3dB-Grenzfrequenz des analogen Filters der GSV-Baugruppe. Die Firmware der Baugruppe wählt dabei die nächstliegende vorhandene Filterfrequenz aus. Die eingestellte Filterfrequenz kann durch **GSVgetAnalogFilter** abgefragt werden.

Siehe auch **GSVgetOptionsExtension21**.

**Verwendete GSV Befehle:** 144 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-2.1.

## Funktionsdefinition:

C:           int GSVsetAnalogFilter ( int **no**, double **freq** );

Delphi™:       GSVsetAnalogFilter ( **no**: Integer; **freq**: Double ): Integer;

VB:            GSVsetAnalogFilter ( ByVal **no** As Long,  
  ByVal **freq** As Double ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                  Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**freq**         Gleitkommazahl im Bereich 0,5..32767,5.  
                  Gewünschte Filtergrenzfrequenz in Hertz.

### 3.117 GSVgetAnalogFilter

(Index 245)

#### **Beschreibung:**

Die Funktion GSVgetAnalogFilter liest die zuvor mit **GSVsetAnalogFilter** gespeicherte Grenzfrequenz des analogen Filters aus dem GSV.

Siehe auch **GSVgetOptionsExtension21**.

**Verwendete GSV Befehle:** 145 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-2.1.

**Funktionsdefinition:**

C:       double GSVgetAnalogFilter ( int **no** );

Delphi™:       GSVgetAnalogFilter ( **no**: Integer ): Double;

VB:            GSVgetAnalogFilter ( ByVal **no** As Long ) As Double

**Rückgabewert:** Filtergrenzfrequenz oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.118 GSVisCommandAvailable

(Index 247)

#### **Beschreibung:**

Die Funktion GSVisCommandAvailable ermittelt, ob die Befehle in einem bestimmten Zahlenbereich in der GSV Baugruppe implementiert sind. Bei einem Ergebnis von GSV\_TRUE sind alle Befehle im angegebenen Bereich implementiert, bei einem Ergebnis von GSV\_OK ist mindestens ein Befehl im angegebenen Bereich nicht implementiert.

Dieser Befehl stellt eine eigenständige Erweiterung dar, siehe auch ***GSVgetOptionsCommandTest***.

**Verwendete GSV Befehle:** 147 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_TRUE, GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit dieser Erweiterung.





### 3.119 GSVsetNoiseCutThreshold

(Index 248)

#### Beschreibung:

Die Funktion `GSVsetNoiseCutThreshold` setzt die Schwelle für die Rauschunterdrückung, die durch ***GSVsetSpecialModeNoiseCut*** geschaltet werden kann. Die Rauschunterdrückung setzt alle Messwerte, deren Betrag unterhalb dieser Schwelle liegt, auf Null. Der Schwellwert hat den gleichen Wertebereich wie die Messwerte, die z.B. ***GSVread*** liefert.

Siehe auch ***GSVisCommandAvailable***.

**Verwendete GSV Befehle:** 148 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann `GSV_OK` oder `GSV_ERROR` sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei erfolgreicher Prüfung auf Verfügbarkeit.

## Funktionsdefinition:

C: int GSVsetNoiseCutThreshold ( int **no**, double **thr** );

Delphi™:

GSVsetNoiseCutThreshold ( **no**: Integer; **thr**: Double ): Integer;

VB: GSVsetNoiseCutThreshold ( ByVal **no** As Long,  
ByVal **thr** As Double ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>thr</b>	Gleitkommazahl im Bereich -1,05..+1,05 (bzw. unipolar: 0..1,05). Gewünschte Schwelle.

### 3.120 GSVgetNoiseCutThreshold

(Index 249)

#### Beschreibung:

Die Funktion GSVgetNoiseCutThreshold liest die zuvor mit **GSVsetNoiseCutThreshold** eingestellte Schwelle der Rauschunterdrückung aus dem GSV.

Siehe auch **GSVisCommandAvailable** und **GSVgetSpecialModeNoiseCut**.

**Verwendete GSV Befehle:** 149 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei erfolgreicher Prüfung auf Verfügbarkeit.

**Funktionsdefinition:**

C:

`double GSVgetNoiseCutThreshold ( int no );`

Delphi™:

`GSVgetNoiseCutThreshold ( no: Integer ): Double;`VB: `GSVgetNoiseCutThreshold ( ByVal no As Long ) As Double`**Rückgabewert:** Schwellwert oder Fehlercode.**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.121 GSVsetAutoZeroCounter

(Index 250)

#### Beschreibung:

Die Funktion GSVsetAutoZeroCounter setzt das Zeitintervall für die automatische Nullnachführung, die durch **GSVsetSpecialModeAutoZero** geschaltet werden kann. Die automatische Nullnachführung setzt in regelmäßigen Zeitabständen einen neuen Nullpunkt, sofern der laufende Messwert unterhalb des unteren Schaltpunktes des Schwellwertschalters ist; dazu muss der Fensterdiskriminator-Modus ausgeschaltet sein. Die Dauer des Zeitintervalls wird in Einheiten der Anzahl gesendeter Datenpakete festgelegt.

Siehe auch **GSVisCommandAvailable**, **GSVsetModeWindow** und **GSVsetThreshold**.

**Verwendete GSV Befehle:** 150 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei erfolgreicher Prüfung auf Verfügbarkeit.

## Funktionsdefinition:

C:     int GSVsetAutoZeroCounter ( int **no**, long **ctr** );

Delphi™:  
        GSVsetAutoZeroCounter ( **no**: Integer; **ctr**: LongInt ): Integer;

VB:     GSVsetAutoZeroCounter ( ByVal **no** As Long, ByVal **ctr** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>ctr</b>	Ganze Zahl im Bereich 1..65535. Gewünschte Anzahl gesendeter Datenpakete.

### 3.122 GSVgetAutoZeroCounter

(Index 251)

#### **Beschreibung:**

Die Funktion GSVgetAutoZeroCounter liest die zuvor mit **GSVsetAutoZeroCounter** eingestellte Anzahl der gesendeten Datenpakete, die ein Zeitintervall für die automatische Nullnachführung beschreibt, aus dem GSV.

Siehe auch **GSVisCommandAvailable** und **GSVgetSpecialModeAutoZero**.

**Verwendete GSV Befehle:** 151 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei erfolgreicher Prüfung auf Verfügbarkeit.



**Funktionsdefinition:**

C: long GSVgetAutoZeroCounter ( int **no** );

Delphi™:

GSVgetAutoZeroCounter ( **no**: Integer ): LongInt;

VB: GSVgetAutoZeroCounter ( ByVal **no** As Long ) As Long

**Rückgabewert:** Datenpaketanzahl oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.123 GSVsetUserTextChar

(Index 252)

#### Beschreibung:

Die Funktion GSVsetUserTextChar setzt ein Zeichen des benutzerdefinierten Textes, der in der Anzeige der GSV Baugruppe dargestellt werden kann. Dabei legt der Adressparameter fest, an welcher Stelle das Zeichen stehen soll; die Adressierung beginnt links bei Null. Das Ende des Textes wird durch ein Null-Zeichen bestimmt, das hinter jenem Zeichen folgt, das am weitesten rechts steht. Soll kein Text eingeblendet werden, wird ein Null-Zeichen an Adresse 0 gesetzt; dies entspricht einem Text der Länge Null. Die maximale nutzbare Textlänge beträgt typisch 16 Zeichen, kann aber geräteabhängig sein. Die Übertragung erfolgt durch einen Aufruf pro Zeichen bzw. terminierendes Null-Zeichen und beginnt stets mit der höchsten Adresse, d.h. beim Null-Zeichen, danach folgt das am weitesten rechts stehende Zeichen usw.

Der Zeichensatz kann geräteabhängig sein, jedoch wird in jedem Fall der ASCII Zeichensatz im Bereich von 32 (Leerzeichen) bis 125 ("}") mit Ausnahme von 92 ("\"") dargestellt.

Siehe auch **GSVisCommandAvailable**.

**Verwendete GSV Befehle:** 152 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei erfolgreicher Prüfung auf Verfügbarkeit.



### 3.124 GSVgetRanges

(Index 262)

#### Beschreibung:

Die Funktion GSVgetRanges liest mögliche und aktuelle Eingangsempfindlichkeiten für alle Eingangskanäle vom GSV. Das Ergebnis ist die Nummer der Steckbrückenkonfiguration, die für die Eingangsempfindlichkeit verwendet wird. Die Eingangsempfindlichkeit selbst wird über den Parameter **prange** zurückgegeben.

Die aktuellen Eingangsempfindlichkeiten der Kanäle werden durch einzelne Aufrufe mit dem Parameter **channel** gleich Kanalnummer und dem Parameter **inx** gleich Null gelesen. Ist die Rückgabe > 0, dann handelt es sich um die Nummer der Steckbrücken-Konfiguration für die gelesene Eingangsempfindlichkeit. Ist die Rückgabe gleich GSV\_OK, dann gilt: ist der unter **prange** gelieferte Wert < 0, dann existiert kein solcher Kanal; ist er > 0, dann wird das Lesen der aktuellen Eingangsempfindlichkeit für diesen Kanal nicht unterstützt und die Eingangsempfindlichkeit ist unbekannt.

Die möglichen Eingangsempfindlichkeiten eines Kanals werden durch einzelne Aufrufe mit aufsteigenden Werten für Parameter **inx** (beginnend bei 1) gelesen. Ist die Rückgabe > 0, dann handelt es sich um die Nummer der Steckbrücken-Konfiguration für die gelesene Eingangsempfindlichkeit. Ist die Rückgabe gleich GSV\_OK, dann gilt: ist der unter **prange** gelieferte Wert < 0, dann existiert keine Eingangsempfindlichkeit für diesen oder höhere Indizes (Ende der Liste); ist er > 0, dann existiert keine Eingangsempfindlichkeit für diesen Index, aber möglicherweise für höhere Indizes.

**Verwendete GSV Befehle:** 162 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 5.

## Funktionsdefinition:

C:                   int GSVgetRanges ( int **no**, int **channel**, int **inx**, double \***prange** );

Delphi™:           GSVgetRanges ( **no**, **channel**, **inx**: Integer;  
  var **prange**: Double ): Integer;

VB:                   GSVgetRanges ( ByVal **no** As Long, ByVal **channel** As Long,  
  ByVal **inx** As Long, ByRef **prange** As Double )  
  As Long

**Rückgabewert:** Steckbrückenkonfigurationsnummer oder Fehlercode.

## Aufrufparameter:

**no**                   Ganze Zahl im Bereich 1..1024.  
                          Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**channel**           Ganze Zahl im Bereich 0..15.  
                          Nummer des Eingangskanals, dessen Eingangsempfindlichkeit gelesen werden soll.

**inx**                   Ganze Zahl im Bereich 0..15.  
                          Index der Eingangsempfindlichkeit, die gelesen werden soll (0: aktuelle Einstellung, 1..15: mögliche Einstellungen).

**prange**            Zeiger auf Gleitkommavariablen doppelter Genauigkeit.  
                          Gelesene Eingangsempfindlichkeit (wenn Rückgabewert > 0):  
                          Negative Werte in mV/V (Brückeneingang relativ zur Speisespannung),  
                          Positive Werte in mV (Analogeingang absolut).  
                          Codierter Wert (wenn Rückgabewert = GSV\_OK).  
                          (Delphi™ und VB: Die Gleitkommavariablen werden als Referenz übergeben.)

### 3.125 GSVsetRanges

(Index 263)

#### Beschreibung:

Die Funktion GSVsetRanges stellt die Eingangsempfindlichkeit eines Eingangskanals ein. Dabei gibt der Parameter **channel** den Kanal an, der Parameter **inx** muss Null sein, und der Parameter **range** gibt die gewünschte Eingangsempfindlichkeit an. Ist der Wert für die Eingangsempfindlichkeit negativ, handelt es sich um eine Angabe relativ zur Speisespannung eines Brückeneingangs in mV/V; ist der Wert positiv handelt es sich um eine absolute Angabe für einen Analogeingang in mV.

**Verwendete GSV Befehle:** 163 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 5.



### 3.126 GSVgetTxModeConfig

(Index 357)

#### Beschreibung:

Die Funktion GSVgetTxModeConfig liest die aktuelle Einstellung des Konfigurations-Modus vom GSV. Bei einem Ergebnis von 1 ist der Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Der Konfigurations-Modus wird durch eine Steckbrücke festgelegt und kann nicht durch Software umgeschaltet werden. Dieser Modus erlaubt das Ändern des Datenübertragungsmodus mit **GSVsetTxMode** (und den daraus abgeleiteten Funktionen), sowie das Ändern der Baudrate mit **GSVsetBaud** (und den daraus abgeleiteten Funktionen).

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 129 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:        int GSVgetTxModeConfig ( int **no** );

Delphi™:    GSVgetTxModeConfig ( **no**: Integer ): Integer;

VB:        GSVgetTxModeConfig ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**        Ganze Zahl im Bereich 1..1024.  
            Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.127 GSVsetTxModeTransmit4

(Index 358)

#### Beschreibung:

Die Funktion GSVsetTxModeTransmit4 schaltet das 4-Byte Protokoll zur Meßdatenübertragung ein (Parameter = 1) oder aus (Parameter = 0, 3-Byte Protokoll wird verwendet). Beim 4-Byte Protokoll sind zusätzlich zu den Meßdaten auch Adress- bzw. Status-Informationen enthalten.

Siehe auch **GSVgetOptionsExtension3**.

#### Achtung:

GSVsetTxModeTransmit4 kann nur im Konfigurationsmodus des GSV erfolgreich durchgeführt werden und wird sonst ignoriert (siehe auch **GSVgetTxModeConfig**). Außerdem muß für die Aktivierung des 4-Byte Protokolls die Baudrate mindestens 57600 Baud betragen, sonst wird der Befehl ebenfalls ignoriert.

**Verwendete GSV Befehle:** 129, 128 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:     int GSVsetTxModeTransmit4 ( int **no**, int **t4** );

Delphi™:

GSVsetTxModeTransmit4 ( **no**, **t4**: Integer ): Integer;

VB:     GSVsetTxModeTransmit4 ( ByVal **no** As Long, ByVal **t4** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>t4</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der das 4-Byte Protokoll aus- oder einschaltet.

### 3.128 GSVgetTxModeTransmit4

(Index 359)

#### **Beschreibung:**

Die Funktion GSVgetTxModeTransmit4 liest die aktuelle Einstellung des 4-Byte Protokolls vom GSV. Bei einem Ergebnis von 1 ist das 4-Byte Protokoll eingeschaltet, bei einem Ergebnis von 0 ist das 3-Byte Protokoll eingeschaltet. Beim 4-Byte Protokoll sind zusätzlich zu den Meßdaten auch Adress- bzw. Status-Informationen enthalten.

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 129 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:     int GSVgetTxModeTransmit4 ( int **no** );

Delphi™:

      GSVgetTxModeTransmit4 ( **no**: Integer ): Integer;

VB:     GSVgetTxModeTransmit4 ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.129 GSVsetTxModeRepeat3

(Index 360)

#### Beschreibung:

Die Funktion GSVsetTxModeRepeat3 schaltet das 3-fache Senden der Meßdaten im Slow-Modus ein (Parameter = 1) oder aus (Parameter = 0, Meßdaten werden nur einmal gesendet). Das 3-fache Senden der Meßdaten im Slow-Modus (siehe **GSVsetSlowRate** und **GSVsetSpecialModeSlow**) kann die Gefahr zufälliger Datenverluste verringern.

Siehe auch **GSVgetOptionsExtension3**.

#### Achtung:

GSVsetTxModeRepeat3 kann nur im Konfigurationsmodus des GSV erfolgreich durchgeführt werden und wird sonst ignoriert (siehe auch **GSVgetTxModeConfig**).

**Verwendete GSV Befehle:** 129, 128 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:      int GSVsetTxModeRepeat3 ( int **no**, int **r3** );

Delphi™: GSVsetTxModeRepeat3 ( **no**, **r3**: Integer ): Integer;

VB:      GSVsetTxModeRepeat3 ( ByVal **no** As Long, ByVal **r3** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>r3</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der das 3-fache Senden der Meßdaten aus- oder einschaltet.

### 3.130 GSVgetTxModeRepeat3

(Index 361)

#### **Beschreibung:**

Die Funktion GSVgetTxModeRepeat3 liest die aktuelle Einstellung des 3-fachen Sendens im Slow-Modus vom GSV. Bei einem Ergebnis von 1 ist das 3-fache Senden eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Das 3-fache Senden der Meßdaten im Slow-Modus (siehe **GSVsetSlowRate** und **GSVsetSpecialModeSlow**) kann die Gefahr zufälliger Datenverluste verringern.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 129 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:     int GSVgetTxModeRepeat3 ( int **no** );

Delphi™: GSVgetTxModeRepeat3 ( **no**: Integer ): Integer;

VB:     GSVgetTxModeRepeat3 ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.131 GSVsetTxModeTransmit5

(Index 362)

#### Beschreibung:

Die Funktion GSVsetTxModeTransmit5 schaltet das 5-Byte Protokoll zur Meßdatenübertragung ein (Parameter = 1) oder aus (Parameter = 0, 3- oder 4-Byte Protokoll entsprechend **GSVsetTxModeTransmit4** wird verwendet). Beim 5-Byte Protokoll sind 24 Bit Messdaten sowie auch Status-Informationen enthalten.

Siehe auch **GSVgetOptionsExtension3**.

#### Achtung:

Die Umschaltung des 5-Byte-Protokolls mittels GSVsetTxModeTransmit5 ist möglicherweise nicht implementiert.

**Verwendete GSV Befehle:** 129, 128 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:     int GSVsetTxModeTransmit5 ( int **no**, int **t5** );

Delphi™:

GSVsetTxModeTransmit5 ( **no**, **t5**: Integer ): Integer;

VB:     GSVsetTxModeTransmit5 ( ByVal **no** As Long, ByVal **t5** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>t5</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der das 5-Byte Protokoll aus- oder einschaltet.

### 3.132 GSVgetTxModeTransmit5

(Index 363)

#### Beschreibung:

Die Funktion GSVgetTxModeTransmit5 liest die aktuelle Einstellung des 5-Byte Protokolls vom GSV. Bei einem Ergebnis von 1 ist das 5-Byte Protokoll eingeschaltet, bei einem Ergebnis von 0 ist entweder das 3-Byte Protokoll oder das 4-Byte-Protokoll eingeschaltet (hängt vom Ergebnis von **GSVgetTxModeTransmit4** ab). Beim 5-Byte Protokoll sind 24 Bit Messdaten sowie auch Status-Informationen enthalten.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 129 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:     int GSVgetTxModeTransmit5 ( int **no** );

Delphi™:

      GSVgetTxModeTransmit5 ( **no**: Integer ): Integer;

VB:     GSVgetTxModeTransmit5 ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.133 GSVsetTxModeReadOnly

(Index 364)

#### **Beschreibung:**

Die Funktion GSVsetTxModeReadOnly schaltet den Schreibzugriff der jeweiligen Schnittstelle der GSV-Baugruppe aus (Parameter = 1, nur Lesezugriff) oder ein (Parameter = 0).

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 129, 128 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 3 oder bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:     int GSVsetTxModeReadOnly ( int **no**, int **ro** );

Delphi™:

GSVsetTxModeReadOnly ( **no**, **ro**: Integer ): Integer;

VB:     GSVsetTxModeReadOnly ( ByVal **no** As Long,  
                                  ByVal **ro** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>ro</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der die Schreibsperre der Schnittstelle aus- oder einschaltet.

### 3.134 GSVgetTxModeReadOnly

(Index 365)

#### **Beschreibung:**

Die Funktion GSVgetTxModeReadOnly liest die aktuelle Einstellung des Schreibzugriffs der jeweiligen Schnittstelle der GSV-Baugruppe vom GSV. Bei einem Ergebnis von 0 ist der Schreibzugriff der Schnittstelle eingeschaltet, bei einem Ergebnis von 1 ausgeschaltet.

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 129 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 3 oder bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:     int GSVgetTxModeReadOnly ( int **no** );

Delphi™:

      GSVgetTxModeReadOnly ( **no**: Integer ): Integer;

VB:     GSVgetTxModeReadOnly ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.135 GSVsetSpecialModeSlow

(Index 372)

#### Beschreibung:

Die Funktion GSVsetSpecialModeSlow schaltet eine Meßdaten-Transferrate  $< 1$  Hz (Slow-Modus) ein (Parameter = 1) oder aus (Parameter = 0). Die niedrige Transferrate kann den Energieverbrauch senken, wenn zusätzlich **GSVsetSpecialModeSleep** verwendet wird und diese Funktion von der GSV Baugruppe unterstützt wird. Die Anzahl der Sekunden zwischen zwei Messungen (d.h. zwischen zwei Meßdaten-Übertragungen) wird durch **GSVsetSlowRate** festgelegt.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:     int GSVsetSpecialModeSlow ( int **no**, int **slow** );

Delphi™:

GSVsetSpecialModeSlow ( **no**, **slow**: Integer ): Integer;

VB:     GSVsetSpecialModeSlow ( ByVal **no** As Long, ByVal **slow** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>slow</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Slow-Modus aus- oder einschaltet.

### 3.136 GSVgetSpecialModeSlow

(Index 373)

#### Beschreibung:

Die Funktion GSVgetSpecialModeSlow liest die aktuelle Einstellung einer Meßdaten-Transferrate < 1 Hz (Slow-Modus) vom GSV. Bei einem Ergebnis von 1 ist der Slow-Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Die niedrige Transferrate kann den Stromverbrauch senken, wenn zusätzlich **GSVsetSpecialModeSleep** verwendet wird und diese Funktion von der GSV Baugruppe unterstützt wird. Die Anzahl der Sekunden zwischen zwei Messungen (d.h. zwischen zwei Meßdaten-Übertragungen) wird durch **GSVsetSlowRate** festgelegt.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:     int GSVgetSpecialModeSlow ( int **no** );

Delphi™:  
        GSVgetSpecialModeSlow ( **no**: Integer ): Integer;

VB:     GSVgetSpecialModeSlow ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                 Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.137 GSVsetSpecialModeAverage

(Index 374)

#### Beschreibung:

Die Funktion GSVsetSpecialModeAverage schaltet eine Meßdaten-Mittelung ein (Parameter = 1) oder aus (Parameter = 0). Diese Mittelung erfolgt durch Verwendung mehrerer Messresultate zur Berechnung eines übertragenen Messwertes, d.h. die Messwert-Übertragungen sind seltener als die Messungen. Die Mess- und Datenübertragungsfrequenzen werden durch **GSVwriteSamplingRate** festgelegt.

Beim GSV-3 hat die Funktion GSVsetSpecialModeAverage keine Wirkung, da die Meßdaten-Mittelung nur implizit durch **GSVwriteSamplingRate** ein- oder ausgeschaltet werden kann, d.h. das betreffende Register-Bit ist nur lesbar.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:

```
int GSVsetSpecialModeAverage ( int no, int avg );
```

Delphi™:

```
GSVsetSpecialModeAverage ( no, avg: Integer ): Integer;
```

```
VB: GSVsetSpecialModeAverage ( ByVal no As Long, ByVal avg As Long ) As Long
```

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>avg</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der die Mittelung aus- oder einschaltet.

### 3.138 GSVgetSpecialModeAverage

(Index 375)

#### Beschreibung:

Die Funktion GSVgetSpecialModeAverage liest die aktuelle Einstellung einer Meßdaten-Mittelung vom GSV. Bei einem Ergebnis von 1 ist die Mittelung eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Diese Mittelung erfolgt durch Verwendung mehrerer Messresultate zur Berechnung eines übertragenen Messwertes, d.h. die Messwert-Übertragungen sind seltener als die Messungen. Die Mess- und Datenübertragungsfrequenzen werden durch **GSVwriteSamplingRate** festgelegt.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:

`int GSVgetSpecialModeAverage ( int no );`

Delphi™:

`GSVgetSpecialModeAverage ( no: Integer ): Integer;`VB: `GSVgetSpecialModeAverage ( ByVal no As Long ) As Long`**Rückgabewert:** 0, 1 oder Fehlercode.**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.139 GSVsetSpecialModeFilter

(Index 376)

#### **Beschreibung:**

Die Funktion GSVsetSpecialModeFilter schaltet einen digitalen Tiefpaß zweiter Ordnung ein (Parameter = 1) oder aus (Parameter = 0). Der Tiefpaß ist als FIR-Filter ausgeführt.

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:     int GSVsetSpecialModeFilter ( int **no**, int **flt** );

Delphi™:

      GSVsetSpecialModeFilter ( **no**, **flt**: Integer ): Integer;

VB:     GSVsetSpecialModeFilter ( ByVal **no** As Long, ByVal **flt** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>flt</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Tiefpaß aus- oder einschaltet.

### 3.140 GSVgetSpecialModeFilter

(Index 377)

#### **Beschreibung:**

Die Funktion GSVgetSpecialModeFilter liest die aktuelle Einstellung des Tiefpasses zweiter Ordnung vom GSV. Bei einem Ergebnis von 1 ist der Tiefpaß eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Der Tiefpaß ist als FIR-Filter ausgeführt.

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:     int GSVgetSpecialModeFilter ( int **no** );

Delphi™:

      GSVgetSpecialModeFilter ( **no**: Integer ): Integer;

VB:     GSVgetSpecialModeFilter ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.141 GSVsetSpecialModeMax

(Index 378)

#### **Beschreibung:**

Die Funktion GSVsetSpecialModeMax schaltet den Maximum-Ereignis-Modus ein (Parameter = 1) oder aus (Parameter = 0). Dieser Modus wird nur wirksam, wenn gleichzeitig der Maximum-Modus (siehe **GSVsetModeMax**) eingeschaltet ist. In diesem Fall werden Meßdaten nur dann übertragen, wenn ein neuer Maximalwert aufgetreten ist, d.h. ein Messwert größer war als der vorher bestehende Maximalwert.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:     int GSVsetSpecialModeMax ( int **no**, int **mx** );

Delphi™: GSVsetSpecialModeMax ( **no**, **mx**: Integer ): Integer;

VB:       GSVsetSpecialModeMax ( ByVal **no** As Long, ByVal **mx** As Long ) As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>flt</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Maximum-Ereignis-Modus aus- oder einschaltet.

### 3.142 GSVgetSpecialModeMax

(Index 379)

#### Beschreibung:

Die Funktion GSVgetSpecialModeMax liest die aktuelle Einstellung des Maximum-Ereignis-Modus vom GSV. Bei einem Ergebnis von 1 ist der Maximum-Ereignis-Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Dieser Modus wird nur wirksam, wenn gleichzeitig der Maximum-Modus (siehe **GSVgetModeMax**) eingeschaltet ist. In diesem Fall werden Meßdaten nur dann übertragen, wenn ein neuer Maximalwert aufgetreten ist, d.h. ein Messwert größer war als der vorher bestehende Maximalwert.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:     int GSVgetSpecialModeMax ( int **no** );

Delphi™: GSVgetSpecialModeMax ( **no**: Integer ): Integer;

VB:         GSVgetSpecialModeMax ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**             Ganze Zahl im Bereich 1..1024.  
                  Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.143 GSVsetSpecialModeFilterAuto

(Index 380)

#### Beschreibung:

Die Funktion GSVsetSpecialModeFilterAuto schaltet die automatische Einstellung des analogen Filters ein (Parameter = 1) oder aus (Parameter = 0). Wenn eingeschaltet, wird das Analogfilter automatisch entsprechend der Datenrate und dem FIR-Filter gesetzt. Sonst wird das Analogfilter durch **GSVsetAnalogFilter** gesetzt.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:

```
int GSVsetSpecialModeFilterAuto ( int no, int fltauto );
```

Delphi™:

```
GSVsetSpecialModeFilterAuto ( no, fltauto: Integer ): Integer;
```

```
VB: GSVsetSpecialModeFilterAuto ( ByVal no As Long,  
                                ByVal fltauto As Long ) As Long
```

**Rückgabewert:** Fehlercode.**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>fltauto</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Tiefpaß aus- oder einschaltet.

### 3.144 GSVgetSpecialModeFilterAuto

(Index 381)

#### **Beschreibung:**

Die Funktion GSVgetSpecialModeFilterAuto liest die aktuelle Einstellung für die automatische Einstellung des analogen Filters vom GSV. Bei einem Ergebnis von 1 wird das Analogfilter automatisch entsprechend der Datenrate und dem FIR-Filter gesetzt. Bei einem Ergebnis von 0 wird das Analogfilter durch **GSVsetAnalogFilter** gesetzt.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:

```
int GSVgetSpecialModeFilterAuto ( int no );
```

Delphi™:

```
GSVgetSpecialModeFilterAuto ( no: Integer ): Integer;
```

```
VB: GSVgetSpecialModeFilterAuto ( ByVal no As Long ) As Long
```

**Rückgabewert:** 0, 1 oder Fehlercode.**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
-----------	---

### 3.145 GSVsetSpecialModeFilterOrder5

(Index 382)

#### **Beschreibung:**

Die Funktion GSVsetSpecialModeFilterOrder5 wählt die Ordnung des digitalen Tiefpasses aus. Der Tiefpass ist 5. Ordnung (Parameter = 1) oder 2. Ordnung (Parameter = 0).

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

## Funktionsdefinition:

C:

```
int GSVsetSpecialModeFilterOrder5 ( int no, int fltord5 );
```

Delphi™:

```
GSVsetSpecialModeFilterOrder5 ( no, fltord5: Integer ): Integer;
```

VB:

```
GSVsetSpecialModeFilterOrder5 ( ByVal no As Long,  
                                ByVal fltord5 As Long ) As Long
```

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>fltord5</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der die Auswahl des Tiefpasses 5. Ordnung aus- oder einschaltet.

### 3.146 GSVgetSpecialModeFilterOrder5

(Index 383)

#### **Beschreibung:**

Die Funktion GSVgetSpecialModeFilterOrder5 liest die aktuelle Einstellung der Ordnung des digitalen Tiefpasses vom GSV. Der Tiefpass ist 5. Ordnung (Ergebnis = 1) oder 2. Ordnung (Ergebnis = 0).

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:

int GSVgetSpecialModeFilterOrder5 ( int **no** );

Delphi™:

GSVgetSpecialModeFilterOrder5 ( **no**: Integer ): Integer;

VB:

GSVgetSpecialModeFilterOrder5 ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.147 GSVsetSpecialModeSleep

(Index 384)

#### Beschreibung:

Die Funktion GSVsetSpecialModeSleep schaltet den Energiesparmodus (Sleep-Mode) ein (Parameter = 1) oder aus (Parameter = 0), sofern dieser von der GSV Baugruppe unterstützt wird. Dieser Modus setzt den eingeschalteten Slow-Mode voraus (siehe **GSVsetSpecialModeSlow**) und reduziert den Energieverbrauch der GSV Baugruppe während der Wartezeit zwischen den Messungen.

Siehe auch **GSVgetOptionsExtension3** und **GSVgetOptionsSleepMode**.

#### Achtung:

Die Einstellung des Energiesparmodus (Ein/Aus), wie sie mit den Funktionen **GSVsetSpecialMode** (Bit 6) und GSVsetSpecialModeSleep vorgenommen, sowie mit den Funktionen **GSVgetSpecialMode** (Bit 6) und **GSVgetSpecialModeSleep** gelesen werden kann, wird — entgegen dem sonstigen Stil der vorliegenden Funktionsbibliothek — virtualisiert. Da ein aktiver Energiesparmodus die Kommunikation mit der Baugruppe weitgehend verhindern würde, wird der Energiesparmodus sofort am Beginn der Kommunikation (siehe auch **GSVactivateExtended**) abgeschaltet, die vorherige Einstellung aber bibliotheksintern gespeichert. Etwaige Veränderungen der Einstellung werden nur am gespeicherten Wert vorgenommen und dieser dann erst am Ende der Kommunikation (siehe **GSVrelease**) zur Baugruppe übermittelt.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3 und des Sleep-Mode.

## Funktionsdefinition:

C:    int GSVsetSpecialModeSleep ( int **no**, int **sleep** );

Delphi™:

      GSVsetSpecialModeSleep ( **no**, **sleep**: Integer ): Integer;

VB:     GSVsetSpecialModeSleep ( ByVal **no** As Long, ByVal **sleep** As Long )  
          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>sleep</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Schlaf-Modus aus- oder einschaltet.

### 3.148 GSVgetSpecialModeSleep

(Index 385)

#### Beschreibung:

Die Funktion GSVgetSpecialModeSleep liest die aktuelle Einstellung des Energiesparmodus (Sleep-Mode) vom GSV. Bei einem Ergebnis von 1 ist der Sleep-Modus eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Dieser Modus setzt den eingeschalteten Slow-Mode voraus (siehe **GSVsetSpecialModeSlow**) und reduziert den Energieverbrauch der GSV Baugruppe während der Wartezeit zwischen den Messungen.

Weitere Erläuterungen siehe **GSVsetSpecialModeSleep**.

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:    int GSVgetSpecialModeSleep ( int **no** );

Delphi™:

      GSVgetSpecialModeSleep ( **no**: Integer ): Integer;

VB:     GSVgetSpecialModeSleep ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.149 GSVsetSpecialModeAutoZero

(Index 400)

#### Beschreibung:

Die Funktion GSVsetSpecialModeAutoZero schaltet die automatische Nullnachführung (Auto-Zero-Modus) ein (Parameter = 1) oder aus (Parameter = 0), sofern dieser von der GSV Baugruppe unterstützt wird. In diesem Modus wird in regelmäßigen Zeitabständen ein neuer Nullpunkt gesetzt, sofern der laufende Messwert unterhalb des Einschaltpunktes des Schwellwertschalters ist; dazu muss der Fensterdiskriminator-Modus ausgeschaltet sein. Die Dauer des Zeitintervalls wird mit der Funktion **GSVsetAutoZeroCounter** festgelegt.

Siehe auch **GSVgetOptionsExtension3**, **GSVsetThreshold**, **GSVsetModeWindow** und **GSVsetAutoZeroCounter**.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:

```
int GSVsetSpecialModeAutoZero ( int no, int autozero );
```

Delphi™:

```
GSVsetSpecialModeAutoZero ( no, autozero: Integer ): Integer;
```

```
VB: GSVsetSpecialModeAutoZero ( ByVal no As Long,  
                                ByVal autozero As Long ) As Long
```

**Rückgabewert:** Fehlercode.**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>autozero</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Auto-Zero-Modus aus- oder einschaltet.

### 3.150 GSVgetSpecialModeAutoZero

(Index 401)

#### Beschreibung:

Die Funktion GSVgetSpecialModeAutoZero liest die aktuelle Einstellung der automatischen Nullnachführung (Auto-Zero-Modus) vom GSV. Bei einem Ergebnis von 1 ist die Nullnachführung eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Dieser Modus setzt gegebenenfalls in regelmäßigen Zeitabständen einen neuen Nullpunkt.

Weitere Erläuterungen siehe **GSVsetSpecialModeAutoZero**.

Siehe auch **GSVgetOptionsExtension3**, **GSVsetSpecialModeAutoZero** und **GSVgetAutoZeroCounter**.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:

`int GSVgetSpecialModeAutoZero ( int no );`

Delphi™:

`GSVgetSpecialModeAutoZero ( no: Integer ): Integer;`VB: `GSVgetSpecialModeAutoZero ( ByVal no As Long ) As Long`**Rückgabewert:** 0, 1 oder Fehlercode.**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.151 GSVsetSpecialModeNoiseCut

(Index 402)

#### Beschreibung:

Die Funktion GSVsetSpecialModeNoiseCut schaltet die Rauschunterdrückung (Noise-Cut-Modus) ein (Parameter = 1) oder aus (Parameter = 0), sofern dieser von der GSV Baugruppe unterstützt wird. In diesem Modus werden alle Messwerte, deren Betrag unter einer einstellbaren Schwelle liegt, auf Null gesetzt. Die Schwelle wird durch **GSVsetNoiseCutThreshold** gesetzt.

Siehe auch **GSVgetOptionsExtension3** und **GSVsetNoiseCutThreshold**.

**Verwendete GSV Befehle:** 137, 136 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:

```
int GSVsetSpecialModeNoiseCut ( int no, int noisecut );
```

Delphi™:

```
GSVsetSpecialModeNoiseCut ( no, noisecut: Integer ): Integer;
```

```
VB: GSVsetSpecialModeNoiseCut ( ByVal no As Long,  
                                ByVal noisecut As Long ) As Long
```

**Rückgabewert:** Fehlercode.**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>noisecut</b>	Ganze Zahl, 0 oder 1. Logischer Wert, der den Noise-Cut-Modus aus- oder einschaltet.

### 3.152 GSVgetSpecialModeNoiseCut

(Index 403)

#### Beschreibung:

Die Funktion GSVgetSpecialModeNoiseCut liest die aktuelle Einstellung der Rauschunterdrückung (Noise-Cut-Modus) vom GSV. Bei einem Ergebnis von 1 ist die Rauschunterdrückung eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet. Dieser Modus setzt alle Messwerte Null, deren Betrag unter einer einstellbaren Schwelle liegt.

Weitere Erläuterungen siehe **GSVsetSpecialModeNoiseCut**.

Siehe auch **GSVgetOptionsExtension3**, **GSVsetSpecialModeNoiseCut** und **GSVgetNoiseCutThreshold**.

**Verwendete GSV Befehle:** 137 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:

`int GSVgetSpecialModeNoiseCut ( int no );`

Delphi™:

`GSVgetSpecialModeNoiseCut ( no: Integer ): Integer;`VB: `GSVgetSpecialModeNoiseCut ( ByVal no As Long ) As Long`**Rückgabewert:** 0, 1 oder Fehlercode.**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
-----------	---

### 3.153 GSVreadSamplingFrequency

(Index 404)

#### **Beschreibung:**

Die Funktion GSVreadSamplingFrequency liest die ADC-Abtastrate aus dem GSV.

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 139 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C: double

GSVreadSamplingFrequency ( int **no** );

Delphi™:

GSVreadSamplingFrequency ( **no**: Integer ): Double;

VB: GSVreadSamplingFrequency ( ByVal **no** As Long ) As Double

**Rückgabewert:** ADC-Abtastrate oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.154 GSVreadSamplingFactor

(Index 405)

#### Beschreibung:

Die Funktion GSVreadSamplingFactor liest das Verhältnis zwischen ADC-Abtastrate und Meßdaten-Übertragungsrate aus dem GSV. Bei einem Verhältnis > 1 wird die höhere Rate der Meßwerte (verglichen mit der Rate der übertragenen Meßdaten) zur Mittelung verwendet (siehe auch **GSVgetSpecialModeAverage**).

Siehe auch **GSVgetOptionsExtension3**.

**Verwendete GSV Befehle:** 139 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



**Funktionsdefinition:**

C:     int GSVreadSamplingFactor ( int **no** );

Delphi™:  
      GSVreadSamplingFactor ( **no**: Integer ): Integer;

VB:     GSVreadSamplingFactor ( ByVal **no** As Long ) As Long

**Rückgabewert:** Abtastrate/Datenrate-Verhältnis oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.155 GSVsetBaudRate

(Index 406)

#### **Beschreibung:**

Die Funktion GSVsetBaudRate legt die Baudrate der Schnittstelle des GSVs in Einheiten von Baud fest.

Siehe auch **GSVgetOptionsExtension3**.

#### **Achtung:**

GSVsetBaudRate kann nur im Konfigurationsmodus des GSV erfolgreich durchgeführt werden und wird sonst ignoriert (siehe auch **GSVgetTxModeConfig**).

**Verwendete GSV Befehle:** 130 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.



### 3.156 GSVgetBaudRate

(Index 407)

#### **Beschreibung:**

Die Funktion GSVgetBaudRate liest die aktuell eingestellte Baudrate vom GSV.

Siehe auch ***GSVgetOptionsExtension3***.

**Verwendete GSV Befehle:** 131 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit der Erweiterungen des GSV-3.

**Funktionsdefinition:**

C:            long GSVgetBaudRate ( int **no** );

Delphi™:        GSVgetBaudRate ( **no**: Integer ): LongInt;

VB:            GSVgetBaudRate ( ByVal **no** As Long ) As Long

**Rückgabewert:** Baudrate oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.157 GSVsetCanBaudRate

(Index 408)

#### **Beschreibung:**

Die Funktion GSVsetCanBaudRate legt die Baudrate der CAN-Bus Schnittstelle des GSVs in Einheiten von Baud fest.

Siehe auch ***GSVisCanAvailable***.

**Verwendete GSV Befehle:** 140 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.

## Funktionsdefinition:

C:           int GSVsetCanBaudRate ( int **no**, long **baudrate** );

Delphi™:    GSVsetCanBaudRate ( **no**: Integer; **baudrate**: LongInt ): Integer;

VB:                   GSVsetBaudRate ( ByVal **no** As Long, ByVal **baudrate** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**                Ganze Zahl im Bereich 1..1024.  
                    Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**baudrate**        Ganze Zahl > 0.  
                    Die gewünschte Baudrate.

### 3.158 GSVgetCanBaudRate

(Index 409)

#### **Beschreibung:**

Die Funktion GSVgetCanBaudRate liest die aktuell eingestellte Baudrate vom GSV.

Siehe auch **GSVisCanAvailable**.

**Verwendete GSV Befehle:** 141 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.



**Funktionsdefinition:**

C:        long GSVgetCanBaudRate ( int **no** );

Delphi™:    GSVgetCanBaudRate ( **no**: Integer ): LongInt;

VB:        GSVgetCanBaudRate ( ByVal **no** As Long ) As Long

**Rückgabewert:** Baudrate oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
              Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.159 GSVsetCanID

(Index 412)

#### Beschreibung:

Die Funktion GSVsetCanID setzt die gewünschten IDs der CAN-Bus Schnittstelle des GSV. Die ID wird als Zahlenwert übergeben. Welche ID gesetzt werden soll, muss ebenfalls als Parameter angegeben werden entsprechend folgender Tabelle:

Zu lesende ID	idtype
Befehls-ID	GSV_CANSET_COMMAND_ID
Meldungs-ID	GSV_CANSET_MESSAGE_ID

Siehe auch **GSVisCanAvailable**.

**Verwendete GSV Befehle:** 140 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.

### Funktionsdefinition:

```
C:      int GSVsetCanID ( int no, int idtype, long id );
```

Delphi™:      GSVsetCanID ( **no**, **idtype**: Integer; **id**: LongInt ): Integer;

VB: GSVsetCanID ( ByVal **no** As Long, ByVal **idtype** As Long  
ByVal **id** As Long ) As Long

**Rückgabewert:** Fehlercode.

### Aufrufparameter:

**no** Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

<b>idtype</b>	Ganze Zahl im Bereich 1..2. Typ der ID, die gesetzt werden soll.
---------------	---

<b>id</b>	Ganze Zahl. Zahlenwert der ID, die gesetzt werden soll.
-----------	--

### 3.160 GSVgetCanID

(Index 413)

#### Beschreibung:

Die Funktion GSVgetCanID liest die aktuell eingestellten IDs der CAN-Bus Schnittstelle vom GSV. Das Ergebnis ist die ID als Zahlenwert. Welche ID gelesen werden soll, muss als Parameter angegeben werden entsprechend folgender Tabelle:

Zu lesende ID	idtype
Hersteller-ID	GSV_CANSET_MANUFACTURER_ID
Befehls-ID	GSV_CANSET_COMMAND_ID
Meldungs-ID	GSV_CANSET_MESSAGE_ID

Siehe auch **GSVisCanAvailable**.

**Verwendete GSV Befehle:** 141 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.



### 3.161 GSVsetCanBaud

(Index 414)

#### Beschreibung:

Die Funktion GSVsetCanBaud legt die Baudrate der CAN-Bus Schnittstelle des GSVs gemäß der folgenden Tabelle fest:

baud	Baudrate
1	50000
2	62500
3	83333
4	100000
5	125000
6	250000
7	500000
8	1000000

Siehe auch **GSVisCanAvailable**.

**Verwendete GSV Befehle:** 140 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.

**Funktionsdefinition:**

C:                   int GSVsetCanBaud ( int **no**, int **baud** );

Delphi™:           GSVsetCanBaud ( **no**, **baud**: Integer ): Integer;

VB:                   GSVsetCanBaud ( ByVal **no** As Long, ByVal **baud** As Long )  
                          As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

<b>no</b>	Ganze Zahl im Bereich 1..1024. Nummer der seriellen Schnittstelle des zu verwendenden GSV.
<b>baud</b>	Ganze Zahl im Bereich 1..8. Code für die gewünschte Baudrate.

### 3.162 GSVgetCanBaud

(Index 415)

#### **Beschreibung:**

Die Funktion GSVgetCanBaud liest die aktuell eingestellte Baudrate der CAN-Bus Schnittstelle vom GSV. Das Ergebnis ist ein codierter Wert, der bei **GSVsetCanBaud** erläutert ist.

Siehe auch **GSVisCanAvailable**.

**Verwendete GSV Befehle:** 141 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.



**Funktionsdefinition:**

C:                int GSVgetCanBaud ( int **no** );

Delphi™:        GSVgetCanBaud ( **no**: Integer ): Integer;

VB:              GSVgetCanBaud ( ByVal **no** As Long ) As Long

**Rückgabewert:** Code für die Baudrate oder Fehlercode.

**Aufrufparameter:**

**no**              Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.163 GSVisCanAvailable

(Index 417)

#### **Beschreibung:**

Die Funktion GSVisCanAvailable ermittelt, ob die GSV Baugruppe über eine CAN-Bus Schnittstelle verfügt. Bei einem Ergebnis von 1 ist die Schnittstelle verfügbar, bei einem Ergebnis von 0 nicht.

**Verwendete GSV Befehle:** 141 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Ab GSV Modell 3.

**Funktionsdefinition:**

C:           int GSVisCanAvailable ( int **no** );

Delphi™:       GSVisCanAvailable ( **no**: Integer ): Integer;

VB:            GSVisCanAvailable ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.164 GSVsetCanActive

(Index 418)

#### **Beschreibung:**

Die Funktion GSVsetCanActive schaltet die CAN-Bus Schnittstelle des GSV Baugruppe ein (Parameter = 1) oder aus (Parameter = 0).

Siehe auch ***GSVisCanAvailable***.

**Verwendete GSV Befehle:** 140 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.



### 3.165 GSVgetCanActive

(Index 419)

#### **Beschreibung:**

Die Funktion GSVgetCanActive liest den aktuellen Einschaltzustand der CAN-Bus Schnittstelle vom GSV. Bei einem Ergebnis von 1 ist die Schnittstelle eingeschaltet, bei einem Ergebnis von 0 ausgeschaltet.

Siehe auch ***GSVisCanAvailable***.

**Verwendete GSV Befehle:** 141 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.

**Funktionsdefinition:**

C:               int GSVgetCanActive ( int **no** );

Delphi™:       GSVgetCanActive ( **no**: Integer ): Integer;

VB:             GSVgetCanActive ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**             Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.166 GSVsetCanMode20B

(Index 420)

#### **Beschreibung:**

Die Funktion GSVsetCanMode20B schaltet die den Betriebszustand der CAN-Bus Schnittstelle des GSV Baugruppe entsprechend CAN 2.0B ein (Parameter = 1) oder aus (Parameter = 0, d.h. Betrieb entsprechend CAN 2.0A).

Siehe auch ***GSVisCanAvailable***.

**Verwendete GSV Befehle:** 140 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.



## Funktionsdefinition:

C:           int GSVsetCanMode20B ( int **no**, int **mode20b** );

Delphi™:     GSVsetCanMode20B ( **no**, **mode20b**: Integer ): Integer;

VB:           GSVsetCanMode20B ( ByVal **no** As Long, ByVal **mode20b** As Long )  
                  As Long

**Rückgabewert:** Fehlercode.

## Aufrufparameter:

**no**            Ganze Zahl im Bereich 1..1024.  
                Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**mode20b**     Ganze Zahl, 0 oder 1.  
                Logischer Wert, der den Betriebszustand der CAN-Bus Schnittstelle  
                entsprechend CAN 2.0B aus- oder einschaltet.

### 3.167 GSVgetCanMode20B

(Index 421)

#### **Beschreibung:**

Die Funktion GSVgetCanMode20B liest den aktuellen Betriebszustand der CAN-Bus Schnittstelle vom GSV. Bei einem Ergebnis von 1 arbeitet die Schnittstelle entsprechend CAN 2.0B, bei einem Ergebnis von 0 entsprechend CAN 2.0A.

Siehe auch ***GSVisCanAvailable***.

**Verwendete GSV Befehle:** 141 (siehe GSV Bedienungsanleitung).

#### **Fehler:**

Das Ergebnis kann auch GSV\_ERROR sein. In diesem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei Verfügbarkeit einer CAN-Bus Schnittstelle.

**Funktionsdefinition:**

C:           int GSVgetCanMode20B ( int **no** );

Delphi™:     GSVgetCanMode20B ( **no**: Integer ): Integer;

VB:           GSVgetCanMode20B ( ByVal **no** As Long ) As Long

**Rückgabewert:** 0, 1 oder Fehlercode.

**Aufrufparameter:**

**no**           Ganze Zahl im Bereich 1..1024.  
Nummer der seriellen Schnittstelle des zu verwendenden GSV.

### 3.168 GSVsetUserText

(Index 448)

#### Beschreibung:

Die Funktion GSVsetUserText legt den benutzerdefinierten Text fest, der in der Anzeige der GSV Baugruppe dargestellt werden kann. Soll kein Text eingeblendet werden, wird ein Text der Länge Null übergeben. Die maximale nutzbare Textlänge beträgt typisch 16 Zeichen, kann aber geräteabhängig sein.

Der Zeichensatz kann geräteabhängig sein, jedoch wird in jedem Fall der ASCII Zeichensatz im Bereich von 32 (Leerzeichen) bis 125 ("}") mit Ausnahme von 92 ("\") dargestellt.

Siehe auch ***GSVisCommandAvailable***.

**Verwendete GSV Befehle:** 152 (siehe GSV Bedienungsanleitung).

#### Fehler:

Das Ergebnis kann GSV\_OK oder GSV\_ERROR sein. In letzterem Fall können die Aufrufparameter ungültig, die Kommunikation gestört oder die Schnittstelle inaktiv sein.

**Anwendbar:** Bei erfolgreicher Prüfung auf Verfügbarkeit.

**Funktionsdefinition:**

C:                   int GSVsetText ( int **no**, char \***txt** );

Delphi™:           GSVsetText ( **no**: Integer; **txt**: PChar ): Integer;

VB:                   GSVsetText ( ByVal **no** As Long, ByVal **txt** As String) As Long

**Rückgabewert:** Fehlercode.

**Aufrufparameter:**

**no**                Ganze Zahl im Bereich 1..1024.  
                  Nummer der seriellen Schnittstelle des zu verwendenden GSV.

**txt**               Null-terminierte Zeichenkette (ASCII mit o.g. Einschränkungen).  
                  Gewünschter Anzeigetext.  
                  (VB: Die Zeichenkette ist nicht null-terminiert, sondern ein VB-String.)

## 4 Demoprogramm

Das Demoprogramm zeigt die Grundlagen der Programmierung mit der MEGSV.DLL indem es einige Funktionen des GSV auf Tastendruck verfügbar macht, sowie die laufenden Meßdaten anzeigt. Das Demoprogramm liegt in C (GSVDEMO.C in den Unterverzeichnissen VC und BC) und Delphi™ (GSVDEMO.DPR im Unterverzeichnis Delphi™) vor. Zum Kompilieren der Demoprogramme liegen Stapelverarbeitungsdateien (DEMOMSV.C.BAT, DEMOBC.BAT und DEMOPAS.BAT) in den jeweiligen Unterverzeichnissen bereit, die in ihren Unterverzeichnissen ausgeführt werden können. Es ist empfehlenswert, vor Ausführung der Stapelverarbeitungsdateien diese gründlich zu lesen bzw. zu überprüfen.

### 4.1 Quelltext in C

```

/*****
/*
/* Demonstrationsprogramm für die
/* Funktionen in MEGSV.DLL
/*
/* Copyright (C) Dr. Holger Kabelitz 1999-2009
/* Alle Rechte vorbehalten.
/*
/* Dr. Holger Kabelitz
/* D-13507 Berlin
/*
*****/

#include <stdio.h>
#include <conio.h>
#include <windows.h>
#include "megsv.h"

#define ComNr 1          /* Nummer der seriellen Schnittstelle */
#define BuffSize 1000    /* Größe des Puffers, in dem die AD-Werte
                          abgelegt werden, bis sie mit dem Befehl
                          Read gelesen werden */

static char *befehle[] =
{
    "o      : SetOffset(ComNr) ",
    "c      : SetCal(ComNr) ",
    "z      : SetZero(ComNr) ",
    "s      : SetScale(ComNr) ",
    "b      : SetBipolar(ComNr) ",
    "u      : SetUnipolar(ComNr) ",
    "f      : SetFreq(ComNr, Freq) ",
    "g      : SetGain(ComNr, Gain) ",
    (char *)NULL
};

static void Ausgabe(char *str)
{
    char temp[80];

    CharToOem(str, temp);
    cputs(temp);
}

```

```

static void BefehlsUebersicht(int einzel, int taste)
{
    int i;

    if (!einzel)
        Ausgabe("Befehlsauswahl\r\n");

    for (i = 0; befehle[i] != NULL; i++)
        if (!einzel || taste == befehle[i][0])
            cprintf(einzel ? "\r\n\n%s\r\n" : "%s\r\n", befehle[i]);

    if (!einzel)
        Ausgabe("ESC   : Beenden\r\n\n");
}

static int Taste(void)
{
    if (!kbhit())
        return 0;

    return getch();
}

static void LiesZeile(char *z, int l)
{
    int k, i = 0;

    do
    {
        k = getch();

        if (k >= ' ' && k < 127)
        {
            if (i < l - 1)
            {
                z[i++] = (char)k;
                putchar(k);
            }
        }
        else if (k == '\b' || k == 127)
        {
            if (i > 0)
            {
                i--;
                cputs("\b \b");
            }
        }
    } while (k != '\r');

    z[i] = (char)0;
}

static void WerteLesen(void)
{
    double ad;

    if (GSVread(ComNr, &ad) == GSV_TRUE)
        cprintf("  %12.6f\r", ad);
}

static void FehlerMeldung(int code, int ref, int warn)

```

```

{
    if (code == ref)
        Ausgabe("-> Erfolgreich durchgeführt\r\n\n");
    else if (code == GSV_OK)
        Ausgabe("-> Unerwartete Zeitüberschreitung!\r\n\n");
    else
        Ausgabe("-> Fehler! Nicht durchgeführt\r\n\n");

    BefehlsUebersicht(0, 0);

    if (warn)
    {
        Ausgabe("NACH DIESEM BEFEHL MÜSSEN SetCal UND");
        Ausgabe(" SetZero AUSGEFÜHRT WERDEN!!!\r\n\n");
    }
}

static void TastaturAbfragen(int ca)
{
    char zeile[64];
    double freq;
    int gain;

    BefehlsUebersicht(1, ca);

    switch (ca)
    {
        case 'o':
            FehlerMeldung(GSVsetOffset(ComNr), GSV_TRUE, 0);
            break;

        case 'c':
            FehlerMeldung(GSVsetCal(ComNr), GSV_TRUE, 0);
            break;

        case 'z':
            FehlerMeldung(GSVsetZero(ComNr), GSV_TRUE, 0);
            break;

        case 's':
            FehlerMeldung(GSVsetScale(ComNr), GSV_TRUE, 0);
            break;

        case 'b':
            /* Nach GSVsetBipolar müssen GSVsetCal und */
            /* GSVsetZero ausgeführt werden!          */
            FehlerMeldung(GSVsetBipolar(ComNr), GSV_OK, 1);
            break;

        case 'u':
            /* Nach GSVsetUnipolar müssen GSVsetCal und */
            /* GSVsetZero ausgeführt werden!          */
            FehlerMeldung(GSVsetUnipolar(ComNr), GSV_OK, 1);
            break;

        case 'f':
            Ausgabe("Bitte Frequenz eingeben: ");
            LiesZeile(zeile, (int)sizeof(zeile));
            cputs("\r\n");
            if (sscanf(zeile, "%lg", &freq) != 1)
                Ausgabe("Ungültige Eingabe\r\n\n");
            else
            {

```



```

        /* Nach GSVsetFreq müssen GSVsetCal und */
        /* GSVsetZero ausgeführt werden!          */
        FehlerMeldung(GSVsetFreq(ComNr, freq), GSV_OK, 1);
    }
    while (Taste());
    break;

case 'g':
    Ausgabe("Bitte Verstärkung (Gain) eingeben: ");
    LiesZeile(zeile, (int)sizeof(zeile));
    cputs("\r\n");
    if (sscanf(zeile, " %i", &gain) != 1)
        Ausgabe("Ungültige Eingabe\r\n\n");
    else
    {
        /* Nach GSVsetGain müssen GSVsetCal und */
        /* GSVsetZero ausgeführt werden!          */
        FehlerMeldung(GSVsetGain(ComNr, gain), GSV_OK, 1);
    }
    while (Taste());
    break;

default: ;
}
}

int main()                /* Hauptprogramm */
{
    int ch;
    long tmp;

    Ausgabe("\r\nDemonstrationsprogramm für MEGSV.DLL");
    Ausgabe("\r\nVersion der Bibliothek: ");
    tmp = GSVversion();
    cprintf("%ld.", tmp >> 16);
    tmp &= 0xFFFF;
    cprintf("%ld%ld\r\n", tmp / 10, tmp % 10);

    if (GSVactivate(ComNr, BuffSize) != GSV_OK)
    {
        Ausgabe("\r\nInitialisierung der Baugruppe fehlgeschlagen\r\n");
    }
    else
    {
        tmp = GSVmodel(ComNr);
        Ausgabe("Baugruppen-Modell: ");
        cprintf("%ld\r\n\n", tmp);

        BefehlsUebersicht(0, 0);

        do
        {
            ch = Taste();
            WerteLesen();
            TastaturAbfragen(ch);
        } while (ch != 27);

        GSVrelease(ComNr);

        cputs("\r\n");
    }

    return 0;
}

```

```
}
```

## 4.2 Quelltext in Delphi™

```
{ $A+, B-, D-, H-, I+, J-, L-, M-, O-, P-, Q-, R-, T-, U-, V+, W-, X+, Y-, Z4 }
{ $APPTYPE CONSOLE }
{ $DESCRIPTION 'Demonstration program for MEGSV.DLL' }

(*****)
(*                                           *)
(* Demonstrationsprogramm für die          *)
(* GSV-Funktionen in MEGSV.DLL            *)
(*                                           *)
(* Copyright (C) ME-Meßsysteme GmbH 1999-2009 *)
(* Alle Rechte vorbehalten.                *)
(*                                           *)
(* ME-Meßsysteme GmbH                      *)
(* Neuendorfstr. 18a                      *)
(* D-16761 Hennigsdorf                    *)
(*                                           *)
(*****)

program GSVDemo;

uses
  Windows, MEGSV;

const
  ComNr = 1;          (* Nummer der seriellen Schnittstelle *)
  BuffSize = 1000;    (* Größe des Puffers, in dem die AD-Werte
                        abgelegt werden, bis sie mit dem Befehl
                        GSVread gelesen werden *)

const
  befehle : array [0..8] of PChar =
  (
    'o' : GSVsetOffset(ComNr) ',
    'c' : GSVsetCal(ComNr) ',
    'z' : GSVsetZero(ComNr) ',
    's' : GSVsetScale(ComNr) ',
    'b' : GSVsetBipolar(ComNr) ',
    'u' : GSVsetUnipolar(ComNr) ',
    'f' : GSVsetFreq(ComNr, Freq) ',
    'g' : GSVsetGain(ComNr, Gain) ',
    nil
  );

var
  ch : Char;
  tmp : LongInt;
  konsole : THandle;
  modus : DWORD;
  zeichenda : Boolean;
  zeichen : packed record case Boolean of
    false: (IR : TInputRecord);
    true: (EventType : array [0..1] of Word;
           KeyEvent : TKeyEventRecord)
  end;

procedure InitKonsole;
begin
  konsole := GetStdHandle(STD_INPUT_HANDLE);
  GetConsoleMode(konsole, modus);
```

```

        SetConsoleMode(konsole, ENABLE_PROCESSED_INPUT);
        zeichenda := False
    end;

procedure Ausgabe(str: PChar);
var
    temp : array [0..80] of Char;
begin
    CharToOem(str, temp);
    Write(temp)
end;

procedure BefehlsUebersicht(einzel: Boolean; taste: Char);
var
    i : Integer;
begin
    if not einzel then
        begin
            Ausgabe('Befehlsauswahl');
            Writeln
        end;

    i := 0;
    while befehle[i] <> nil do
        begin
            if not einzel or (taste = befehle[i][0]) then
                begin
                    if einzel then
                        begin
                            Writeln;
                            Writeln
                        end;
                    Writeln(befehle[i])
                end;
            i := i + 1
        end;

    if not einzel then
        begin
            Ausgabe('ESC : Beenden');
            Writeln;
            Writeln
        end
    end;
end;

function Taste(warte: Boolean): Char;
var
    n : DWORD;
begin
    if not zeichenda then
        begin
            GetNumberOfConsoleInputEvents(konsole, n);
            while (n > 0) or warte do
                begin
                    ReadConsoleInput(konsole, zeichen.IR, 1, n);
                    if (n = 1) and (zeichen.EventType[0] = KEY_EVENT) and
                        zeichen.KeyEvent.bKeyDown then
                        begin
                            zeichenda := True;
                            break
                        end;
                    GetNumberOfConsoleInputEvents(konsole, n)
                end
            end
        end
    end;
end;

```

```

    end;

with zeichen.KeyEvent do
begin
    if zeichenda and (wRepeatCount > 0) and
        (AsciiChar > Chr(0)) and (AsciiChar <= Chr(127)) then
        begin
            Taste := AsciiChar;
            wRepeatCount := wRepeatCount - 1;
            zeichenda := wRepeatCount > 0
        end
    else
        begin
            Taste := Chr(0);
            zeichenda := False
        end
    end
end;

procedure LiesZeile(var z: array of Char);
var
    i : Integer;
    k : Char;
begin
    i := 0;

    repeat
        k := Taste(True);

        if (k >= ' ') and (k < Chr(127)) then
            begin
                if i < high(z) then
                    begin
                        z[i] := k;
                        i := i + 1;
                        Write(k)
                    end
                end
            else if (k = Chr(8)) or (k = Chr(127)) then
                begin
                    if i > 0 then
                        begin
                            i := i - 1;
                            Write(Chr(8), ' ', Chr(8))
                        end
                    end
                end
            until k = Chr(13);

            z[i] := Chr(0)
        end;

procedure WerteLesen;
var
    ad : Double;
begin
    if GSVread(ComNr, ad) = GSV_TRUE then
        Write(' ', ad:12:6, Chr(13))
    end;

procedure FehlerMeldung(code, ref: Integer; warn: Boolean);
begin
    if code = ref then
        Ausgabe('-> Erfolgreich durchgeführt')

```

```

else if code = GSV_OK then
    Ausgabe('-> Unerwartete Zeitüberschreitung!')
else
    Ausgabe('-> Fehler! Nicht durchgeführt');
Writeln;
Writeln;

BefehlsUebersicht(False, Chr(0));

if warn then
begin
    Ausgabe('NACH DIESEM BEFEHL MÜSSEN SetCal UND');
    Ausgabe(' SetZero AUSGEFÜHRT WERDEN!!!');
    Writeln;
    Writeln
end
end;

procedure TastaturAbfragen(ca: Char);
var
    zeile : array [0..64] of Char;
    freq : Double;
    gain, fehler : Integer;
begin
    BefehlsUebersicht(True, ca);

    case ca of
        'o': FehlerMeldung(GSVsetOffset(ComNr), GSV_TRUE, False);

        'c': FehlerMeldung(GSVsetCal(ComNr), GSV_TRUE, False);

        'z': FehlerMeldung(GSVsetZero(ComNr), GSV_TRUE, False);

        's': FehlerMeldung(GSVsetScale(ComNr), GSV_TRUE, False);

        'b': FehlerMeldung(GSVsetBipolar(ComNr), GSV_OK, False);

        'u': FehlerMeldung(GSVsetUnipolar(ComNr), GSV_OK, False);

        'f': begin
            Ausgabe('Bitte Frequenz eingeben: ');
            LiesZeile(zeile);
            Writeln;
            Val(zeile, freq, fehler);
            if fehler <> 0 then
                begin
                    Ausgabe('Ungültige Eingabe');
                    Writeln;
                    Writeln
                end
            else
                begin
                    (* Nach GSVsetFreq müssen GSVsetCal und *)
                    (* GSVsetZero ausgeführt werden! *)
                    FehlerMeldung(GSVsetFreq(ComNr, freq), GSV_OK, True)
                end;
                while (Taste(False) <> Chr(0)) do
            end;

        'g': begin
            Ausgabe('Bitte Verstärkung (Gain) eingeben: ');
            LiesZeile(zeile);
            Writeln;

```

```

        Val(zeile, gain, fehler);
        if fehler <> 0 then
            begin
                Ausgabe('Ungültige Eingabe');
                Writeln;
                Writeln
            end
        else
            begin
                (* Nach GSVsetGain müssen GSVsetCal und *)
                (* GSVsetZero ausgeführt werden! *)
                FehlerMeldung(GSVsetGain(ComNr, gain), GSV_OK, True)
            end;
            while (Taste(False) <> Chr(0)) do
                end
        else
            end
    end;

begin
    (* Hauptprogramm *)
    InitKonsole;

    Writeln;
    Ausgabe('Demonstrationsprogramm für MEGSV.DLL');
    Writeln;
    Ausgabe('Version der Bibliothek: ');
    tmp := GSVversion;
    Write(tmp shr 16:1, '.');
    tmp := tmp and $FFFF;
    Writeln(tmp div 10:1, tmp mod 10:1);

    if GSVactivate(ComNr, BuffSize) <> GSV_OK then
        begin
            Writeln;
            Ausgabe('Initialisierung des GSV fehlgeschlagen');
            Writeln
        end
    else
        begin
            tmp := GSVmodel(ComNr);
            Ausgabe('Baugruppen-Modell: ');
            Writeln(tmp:1);
            Writeln;

            BefehlsUebersicht(False, Chr(0));

            repeat
                ch := Taste(False);
                WerteLesen();
                TastaturAbfragen(ch)
            until ch = Chr(27);

            GSVrelease(ComNr);

            Writeln
        end;

        SetConsoleMode(konsole, modus)
    end.

```