SENSOR TECHNOLOGY LTD

SBT/SIT/ORT/RWT/SGR Series
Transducer Communication Protocol

Revision 9 – November 2023

Table of Contents

Contact Details

Sensor Technology Ltd,
Apollo Park,
Ironstone Lane,
Wroxton,
BANBURY,
OX15 6AY,
United Kingdom.

Sales
Email: stlsales@sensors.co.uk
Tel: +44 (0)1869 238400

Technical Support
Email: software@sensors.co.uk
Tel: +44 (0)1869 238400

Introduction

The RS232 and USB interfaces on our Torque Transducers provide a method of extracting digital operational data from the transducer. Functions for controlling aspects of the transducer's operation are also present.

The protocol used is the same for both RS232 and USB, but due to USB's more complex nature, it is recommended that our Windows or Linux library be used.

If development is being done on the Windows platform, you may want to consider using the STCOMM DLL. The DLL simplifies the use of the USB and RS232 interfaces, by providing a unified interface to access transducers connected via either method; it takes care of the low-level driver access, protocol negotiation and data manipulation. A similar, but simpler library is available for Linux.

Compatible Models
The protocol described in this manual is compatible with transducers from the advanced SBT, SIT, ORT, RWT and SGR family of products. Transducers must be running firmware version 3 or higher and have digital communications enabled.

The table below lists the models that are compatible:

Transducer Family	Model Range	Models
Strain Gauge Reaction	SBT140	SBT140
	SIT145	SIT145
Optical (ORT)	ORT240	ORT240/ORT241
Rayleigh Wave (RWT)	RWT320	RWT320/RWT321/RWT322
	RWT340	RWT340/RWT341/RWT342
	RWT420	RWT420/RWT421/RWT422
	RWT440	RWT440/RWT441/RWT442
Strain Gauge Rotary (SGR)	SGR520	SGR520/SGR521/SGR522
	SGR540	SGR540/SGR541/SGR542

Compatible transducers can be identified by the presence of a status LED and serial number greater than 12200.

Hardware Generations
There have been 3 hardware generations of our modern torque platform, designated MK2, MK3 and MK4. Each generation brings a new electronic design, utilising a different processor and firmware. Accompanying each step are additional features and enhancements.

The same platform is shared between our different torque product families, providing a common interface. There will be some features that are family or generation specific.

Each hardware generation can be identified by the firmware version, which is marked on the transducer label.

Hardware Generation	Firmware Release	Firmware Version	Release Window
MK2	R3	3.0 > 3.9	January 2008 > February 2012
MK3	R4/R5	4.0 > 5.9	February 2012 > July 2022
MK4	R6	6.0 > 6.9	July 2022 Onwards

Transducers using older hardware generations maybe upgraded during repair or calibration.

RS232

The RS232 interface provides a full-duplex communication channel; each byte of data is transmitted in a packet of 10bits. The data packet consists of one start bit, 8 data bits, no parity and one stop bit.

Data Packet Format (D0 – Least Significant Bit)

Start Bit	D0	D1	D2	D3	D4	D5	D6	D7	Stop Bit

The RS232 interface can operate at 3 different baud rates, 9600, 38400 and 115200bps (default). The baud rate can be changed by using our “Transducer Control” program which accompanies our advanced Transducers**.**

The digital connector (12 pin Lumberg - Male) on the transducer is wired as a DTE (Data Terminal Equipment) device. The TX pin is an output, and RX pin is an input. When connecting to another DTE device using your own cabling, TX and RX will need to be crossed.

USB

The USB interface on the transducer is a USB 2.0 Full-Speed device running at 12Mbps. Communication is conducted via bulk transfers using two endpoints. A third endpoint is used for high-speed mode, but its use requires optimised code and is not discussed in this manual.

On Windows, the STCOMM DLL should be used to connect to USB and RS232 transducers, as it greatly simplifies access. On Linux a simpler library is available.

It is beyond the scope of this manual to discuss the intricacies of the USB interface, the information provided below is for users with knowledge of the interface, and for those wishing to use an unsupported platform.

The protocol is exactly the same for both RS232 and USB, ignoring the underlying USB transactions, which should be controlled by the operating system, driver and interface library.

If developing your own software, without our libraires, we suggest using the libusb library, or one derived from it.

Vendor ID (VID)

Sensor Technology – 0x26B4

Early versions of our transducers used Vendor ID’s and Product ID’s provided by various silicon manufacturers (Silicon Laboratories – 0x10C4). These transducers used proprietary static libraries compiled into the transducer firmware. These static libraries were accompanied by a driver and DLL library.

We have since transitioned to a unified generic USB driver on Windows (libusb-win32), unfortunately, the transducers which use the proprietary libraries need some coercion to work correctly. A setup routine written for libusb-win32 can be provided if required.

Product ID (PID)

The table below lists the different Product ID's and associated Vendor ID's for our transducers.

A particular model series may have multiple ID's, often to differentiate between different hardware generations.

A particular hardware generation can be identified by the firmware version, which is marked on the transducer label.

The R values denote the major firmware release to which the ID applies (see the Hardware Generation section).

Vendor ID	Product ID	Model Range	Hardware Generation	Firmware Release
0x26B4	0x0029	SBT 140	MK4	R6
0x26B4	0x0027	SIT 145	MK4	R6
0x26B4	0x0010	ORT 220	MK3	R5
0x26B4	0x001F	ORT 220	MK4	R6
0x26B4	0x0011	ORT 240	MK3	R5
0x26B4	0x0020	ORT 240	MK4	R6
0x26B4	0x0012	ORT 260	MK3	R5
0x26B4	0x0021	ORT 260	MK4	R6
0x10C4	0x82FB	RWT 320/340/360	MK2	R3
0x26B4	0x0001	RWT 320	MK3	R4
0x26B4	0x0002	RWT 340	MK3	R4
0x26B4	0x0003	RWT 360	MK3	R4
0x26B4	0x0004	RWT 420	MK3	R4/R5
0x26B4	0x0005	RWT 440	MK3	R4/R5
0x26B4	0x0006	RWT 460	MK3	R4/R5
0x26B4	0x0017	SGR 520	MK3	R5
0x26B4	0x0023	SGR 520	MK4	R6
0x26B4	0x0018	SGR 540	MK3	R5
0x26B4	0x0024	SGR 540	MK4	R6
0x26B4	0x0019	SGR 560	MK3	R5
0x26B4	0x0025	SGR 560	MK4	R6

Endpoints
Communication is conducted by IN and OUT bulk endpoints. At a high level, the IN/OUT channels can be thought of in the same terms as the RX and TX lines of the RS232 interface.

Due to different USB stack implementations, employed by different processor ecosystems, there are endpoint differences between firmware versions.

Firmware Version 5 or lower (R3/R4/R5)

Endpoint Type	**Direction**	**ID**
Bulk	IN (Receive)	0x82
Bulk	Out (Transmit)	0x02

Firmware Version 6 or later (R6)

Endpoint Type	**Direction**	**ID**
Bulk	IN (Receive)	0x81
Bulk	Out (Transmit)	0x02

Protocol Description
The transducer uses a simple request and send protocol. Data is transferred using either a binary or ASCII (Firmware v4.2) format. Firmware version 4.2 introduced an ASCII format alongside the original and efficient binary format. The ASCII format was added to provide compatibility with machines that had trouble decoding the binary data.

Binary Format
The binary format is a fast and efficient method of extracting data from the transducer. It has a much lower overhead compared to the ASCII format. Commands are one byte in length, and either request data or switch transducer functions on or off.

To request data, transmit a byte equal to the command number of the function you want, the transducer will then reply with the relevant data, or action your request. Some commands require additional parameters, in these cases, the parameter data should follow the request byte, refer to the command descriptions for more information.

The data returned from request commands will be output in various formats, the format used depends upon the type of data requested. Multi-byte number types are output with the least significant byte (LSB) first, as with Little-Endian systems.

The data types used are C type variables, Int type variables are 2 bytes in size.

The following outlines the variable types used.

Float Data Type (4 bytes): IEEE-754 standard floating-point number format.

Floating-point format:

SEEE EEEE EMMM MMMM MMMM MMMM MMMM MMMM

S – Sign Bit, E – Exponent, M – Mantissa.

LSB Byte 0	Byte 1	Byte 2	**MSB** Byte 3

Unsigned Long Data Type (4 bytes): Long type unsigned integer.

LSB Byte 0	Byte 1	Byte 2	**MSB** Byte 3

Unsigned Int Data Type (2 bytes): Unsigned integer variable.

LSB Byte 0	**MSB** Byte 1

Unsigned Char (1 byte): Single byte unsigned integer variable.

Strings: C has no string variable as such; strings output from the transducer are in the format of an array of Char type (1 byte) integer values, terminated with a NULL character.

PeakMinMax: A structure containing two Float data types (See Float Data Type above).

Torque Peak Max (Float - IEEE-754)			
LSB Byte 0	Byte 1	Byte 2	**MSB** Byte 3
Torque Peak Min (Float - IEEE-754)			
LSB Byte 4	Byte 5	Byte 6	**MSB** Byte 7

Example – Request FastCap Speed:

Request: **0x6F (1 byte)**

Send byte equal to 111.

Transducer Response: **0x000003E8 (4 bytes)**

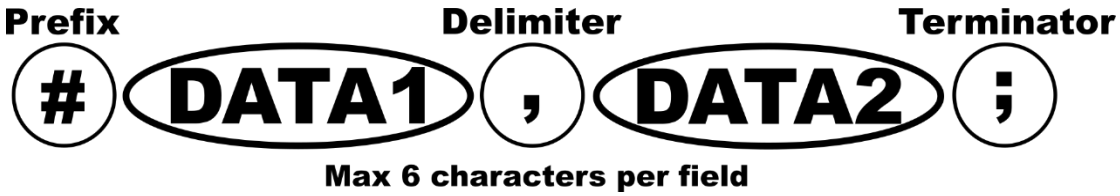
Transducer sends 4 byte unsigned integer, LSB first.

ASCII Format

The ASCII format (Firmware version 4.2 or later) is a straight forward way of exchanging data with a transducer. All data exchanges are in a human readable format, which enables the use of a terminal program to interact with a transducer. The drawback of using the ASCII format is the increased overhead of encoding the binary data, as well as the additional command formatting required.

All data exchanges consist of a data request and a response. Each message, either to or from the transducer, should be formatted in the following way (request data fields should not exceed 6 characters; response data fields may be more than 6 characters).

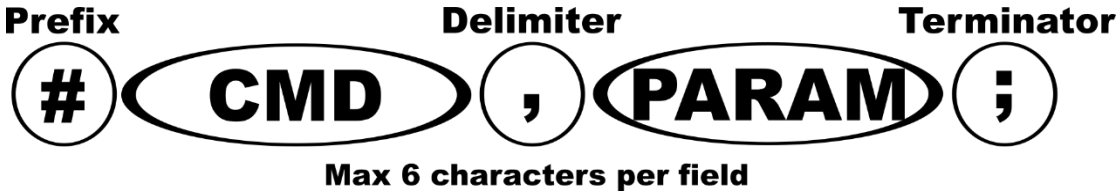
Each message should start with a "**#**" prefix and end with a "**;**" terminator. For readability on a terminal, a carriage return **[CR]** and line feed **[LF]** are appended to all transducer responses. If additional fields are required, each additional field should be separated with "**,**" delimiter.



To request data or an action, transmit a message with the first DATA field set to the command number of the function required, the transsducer will then reply with the relevant data or action your request, some commands may require extra parameters.

Command request message:

Command request message with parameter:



Each request message should encompass a single command, additional fields beyond those required for the requested command, will be ignored. Chaining multiple messages together, e.g. "**#0;#1;**", is permitted, provided the input does exceed the transducers input buffer (256 characters). If multiple requests are chained in this

manner, the transducer will respond to the requests in the order they are received. Each request will have an individual message response.

Each message transaction must be completed within 5 seconds, if it is not, the input buffer will be purged and a "**#NAK;**" will be transmitted. The timer is started as soon as a "**#**" is received.

Requests that do not result in a data response are acknowledged with an "**#ACK;**". Any formatting or validation errors detected in transmitted messages, will receive a "**#NAK;**".

Requests do not need to be any specific length, as long as each field does not exceed 6 characters or contain invalid characters. Response data fields are not limited to 6 characters. Command responses vary in length depending on the command.

Torque, speed, power and temperature requests all output the same number format. The format used is **±0000000.000**, the first character is the sign, while this is irrelevant for speed and power values, it is still output. The sign in torque readings indicates torque direction, negative is counter-clockwise, while positive is clockwise.

To illustrate how to request data from the transducer, two examples are included below:

Request Torque:

Request: **#50;**

Transducer Response: **#+0000000.390;[CR][LF]**

Set Torque Filter to 64 samples:

Request: **#180,64;**

Transducer Response: **#ACK;[CR][LF]**

Torque Modes

The primary purpose of our Torque Transducers is to measure torque. The captured torque value is run through several processes within the firmware, before it is output. These processes include a filter (if enabled), torque scaling, temperature correction and zero offset adjustment. The filter is a running average with a standard deviation cut off to remove spurious readings, the running average enables the sample throughput to be unaffected by filter size.

Once the final torque value is computed, it is run though a peak torque capture algorithm. The peak torque algorithm monitors the incoming data, and compares it against a set of stored values, using various criteria. If the value matches the criteria, that value replaces the stored value. In most cases the criterion is related to whether the captured value is greater than the stored value.

Peak values assume a reset position on start-up, when peak values are reset, they are set to zero, PeakMinMax values are set to the current torque value.

The torque value, unless specified, will always be scaled in the native unit of measurement for the transducer.

The following subsections describe the different types of peak torque.

Peak Torque

The Peak Torque value indicates the highest torque applied to the transducer in either direction. The value is signed to indicate the direction that the torque was applied in.

Peak Torque with Auto Reset

The Peak Torque with Auto Reset value is similar to the Peak Torque feature, it works in the same way, by recording the maximum torque, but automatically resets to zero, when the current torque value drops below a configured percentage of the peak value.

The default auto reset percentage is 80%; the percentage can be configured using our “Transducer Control” program, which accompanies our advanced Torque Transducers. Newer firmware versions offer additional customisations, see the “Transducer Control” manual.

Peak Torque CW

The Peak Torque CW value records the highest torque value measured in the clockwise direction.

Peak Torque CCW

The Peak Torque CCW value records the highest torque value measured in the counter-clockwise direction.

PeakMinMax

The PeakMinMax feature monitors the captured torque values, and records the lowest and highest value from a reference position. This reference is given via a reset command, and assumes zero on power on. An example of the PeakMinMax feature is as follows: if the reference is set to 10, then the torque value goes up by 10 and down by 12, Max would be 20 and Min would be -2.

Speed Modes
Speed is decoded from a square wave signal, produced either from a shaft mounted grating passing through an optical sensor, or from a high-resolution angle encoder. In the case of the angle encoder, the square wave is derived from the XOR of the encoder's quadrature signal.

The frequency of the square wave indicates the rotational speed of the shaft. The transducer uses two methods for the measurement of speed, both methods run simultaneously, offer good accuracy, but differ in measurement time. Speed is always measured in revolutions per minute (RPM).

Slow
The slow method uses a frequency count. Rising edges of the square wave are counted over a period of a second, after each second the count is converted into RPM. As the name suggests, this method is slow, measurements will be produced at a rate of 1 a second. This method is good if you have a fluctuating drive speed and wish to filter the captured speed value.

The speed filter does not apply to the slow method, as it is intrinsically a filtered output.

Fast
The fast method uses a period count. The period count measures the time between rising edges of the square wave, then computes the RPM by turning the time into frequency. The fast methods measurement rate is variable and is directly related to the rotational speed of the transducer. When the rotational speed of the shaft rises above a set threshold, the fast method will increase the number of rising edges over which time is measured, this is done to preserve measurement accuracy.

The fast methods measurement rate can be calculated from the following tables. The measurement rate differs between the different hardware generations, due to differing capabilities.

RWT320/340 (MK2)

Calculations are for a 60-line grating

Rotational Speed (RPM)		**Update Rate (Hz)**
From	**To**	
0		1 Hz
1	2000	RPM / 2
2000	4000	((RPM – 2000) x 0.3227) + 650
4000	8000	((RPM – 4000) x 0.196) + 800
8000	16000	((RPM – 8000) x 0.1117) + 850
16000	32000	((RPM – 16000) x 0.058) + 900

ORT240/RWT420/440/SGR520/540 (MK3)

As standard a transducer is fitted with a 60-line grating.

Rotational Speed (RPM)	**Update Rate**
0	1 Hz
> 0	$SQWaveFreq(Hz) = (\frac{RPM}{60}) x\, GratingSize$ $Update\ Rate(Hz) = \frac{SQWaveFreq}{\lceil \frac{SQWaveFreq}{2000} \rceil}$

ORT240/SGR520/540 (MK4)

As standard a transducer is fitted with a 60-line grating. For high-resolution angle encoders with a quadrature output, double the grating size.

Rotational Speed (RPM)	**Update Rate**
0	1 Hz
> 0	$SQWaveFreq(Hz) = (\frac{RPM}{60}) x\, GratingSize$ $Update\ Rate(Hz) = \frac{SQWaveFreq}{\lceil \frac{SQWaveFreq}{1000} \rceil}$

Temperature Sensors
The transducer monitors temperature from different sensors, these are defined as ambient, shaft and internal. The shaft temperature is the only one which is used for compensation. The transducer measures temperature in degrees Celsius.

On some models, not all sensors may be present; when a sensor is absent, the value returned will be the shaft temperature.

Ambient
The ambient sensor is mounted in free air, stood off from the PCB it is mounted to.

Shaft
The shaft sensor is an infra-red device which is pointed directly at the centre of the shaft. In SGR type transducers, the sensor is on the shaft.

Internal
The internal sensor is part of the communications processor on the main processing board.

Unit Key
Some of the commands use a number to represent transducer units, or to indicate which units to convert to. The table below shows which number represents each unit, e.g. 7 = N.m.

Key Value	Unit
0	ozf.in
1	lbf.in
2	lbf.ft
3	gf.cm
4	Kgf.cm
5	Kgf.m
6	mN.m
7	N.m
8	N.cm*

* N.cm only available with firmware >= 6.

Technology Family Key

The protocol discussed in this manual covers multiple product families of different technologies. It is not necessary to know the actual technology, as it does not affect the commands.

For information purposes, the user may wish to decode the technology used, the table below shows which key values represent each technology.

Key Value	Family Designation	Technology Type
0x001 (1)	RWT	SAW based transducer
0x002 (2)	ORT	Optical based transducer
0x004 (4)	Strain Gauge	Strain Gauge based transducer
0x008 (8)	RWT External	SAW based transducer with external electronics
0x010 (16)	ORT External	Optical based transducer with external electronics
0x020 (32)	SGR	Strain Gauge based transducer
0x040 (64)	SGR External	Strain Gauge based transducer with external electronics.
0x080 (128)	SIT External	
0x100 (256)	SBT External	

It is beyond the scope of this manual to discuss the different technologies and their merits, please refer to the individual product manuals for a technical description.

Command Set
The table below outlines the commands available:

Command	Function	Parameters	Return Value
Transducer Identification			
0	Get Transducer ID	None	Transducer ID String
1	Get Transducer Information	None	Transducer Information
2	Get Firmware Version	None	Firmware Version
10	Get Firmware Version (Legacy)	None	Firmware Version
Transducer Data			
50	Get Torque	None	Torque
51	Get Peak Torque	None	Peak Torque
52	Get Peak Torque Auto Reset	None	Peak Torque Auto Reset
53	Get Peak Torque CW	None	Peak Torque CW
54	Get Peak Torque CCW	None	Peak Torque CCW
55	Get PeakMinMax Max	None	PeakMinMax Max
56	Get PeakMinMax Min	None	PeakMinMax Min
57	Get PeakMinMax	None	PeakMinMax
60	Get Torque Convert Unit To	Units	Torque
61	Get Peak Torque Convert Units To	Units	Peak Torque
62	Get Peak Torque Auto Reset Convert Units To	Units	Peak Torque Auto Reset
63	Get Peak Torque CW Convert Units To	Units	Peak Torque CW
64	Get Peak Torque CCW Convert Units To	Units	Peak Torque CCW
65	Get PeakMinMax Max Convert Units To	Units	PeakMinMax Max
66	Get PeakMinMax Min Convert Units To	Units	PeakMinMax Min
67	Get PeakMinMax Convert Units To	Units	PeakMinMax
100	Get Speed	None	Speed
101	Get Power	None	Power in Watts
102	Get Temperature Ambient	None	Temperature Ambient
103	Get Temperature Shaft	None	Temperature Shaft
110	Get SlowCap Speed	None	Speed
111	Get FastCap Speed	None	Speed
112	Get SlowCap Power in Watts	None	Power in Watts
113	Get FastCap Power in Watts	None	Power in Watts
114	Get SlowCap Power in HP	None	Power in HP
115	Get FastCap Power in HP	None	Power in HP
Transducer Control			
146	Reset specified Peaks	See Description.	
147	Reset All Peak Torque values	None	None
148	Reset All Peaks	None	None
149	Reset System Values	None	None

150	Peak Torque Reset	None	None
152	Peak Torque Auto Reset - Reset	None	None
155	Zero Transducer with Average	None	None
156	Zero Transducer	None	None
173	PeakMinMax Retrieve & Reset	None	PeakMinMax
180	Set Torque filter	Filter Setting	None
181	Get Torque filter	None	Filter Setting
182	Set Speed filter	Filter Setting	None
183	Get Speed filter	None	Filter Setting

Transducer Identification

Get Transducer ID

	Binary	**ASCII**
Command	0	#0;
Description	Requests an ID string from the transducer. The ID string contains the Transducer Model Name, Firmware Revision and Serial Number. The ID string has the following format: RWT321-DA - Firmware Revision: 2.1 Serial Number: 12345678	
Parameters	None	
Return Value	Transducer ID String	
	Char [58]	#ID STRING;

Get Transducer Information

<table>
<tr><td></td><td>Binary</td><td>ASCII</td></tr>
<tr><td>Command</td><td>1</td><td>#1;</td></tr>
<tr><td>Description</td><td colspan="2">Requests information on the transducer’s configuration. The information details the transducers setup.</td></tr>
<tr><td>Parameters</td><td colspan="2">None</td></tr>
<tr><td rowspan="2">Return Value</td><td colspan="2">The transducer returns a structure of values, either as a packed C structure for binary or a delimited list in ASCII. The returned data contains the following fields, dates have the following format, DD/MM/YYYY.

Model Name - Transducer Model Name.
Type - Transducer Family, (In binary mode see Family Key).
FSD - Transducer Full Scale.
Units - Transducer Torque Unit, (In binary mode see Unit Key).
Maximum Speed - Maximum tested/rated speed.
Serial Number - Transducer Serial Number.
Manufacture Date - Transducer Manufacture Date.
Calibration Date - Transducer Calibration Date.
Options - Enabled features.

The options value is a combination of binary flags, the table below shows its makeup. Bit=1 Enabled, Bit=0 Disabled

<table>
<tr><th>Bit0 (L)</th><th>Bit1</th><th>Bit2</th><th>Bit3</th></tr>
<tr><td>USB</td><td>RS232</td><td>Advanced User Control</td><td>Current Output</td></tr>
<tr><th>Bit4</th><th>Bit5</th><th>Bit6</th><th>Bit7 (M)</th></tr>
<tr><td>None</td><td>Speed Encoder</td><td>Angle Encoder</td><td>IP65</td></tr>
</table>
</td></tr>
<tr><td>Struct
{
char Model_Name[10];
unsigned char Type;
unsigned int FSD;
unsigned char Units;
unsigned long Max_Speed;
char Serial_Number[9];
char Manufacture_Date[11];
char Calibration_Date[11];
unsigned char Options;
}</td><td>#Model Name,
Type,
FSD,
Units,
Maximum Speed,
Serial Number,
Manufacture Date,
Calibration Date,
Options;

[CR][LF] added for readability.</td></tr>
</table>

Get Firmware Version

	Binary	**ASCII**
Command	2	Not Available
Description	Requests the transducer firmware information. Available in Firmware Version >= 5.1 only.	
Parameters	None	
Return Value	The transducer returns firmware version information, as a packed C structure. The returned data contains the following fields Firmware Type - Internal ID number for the firmware. Firm Rev - Firmware revision in Binary-Coded Decimal. Firm Build - Firmware build number. The firmware revision has the following format: Format (Hex): 0xMMms M = Major m = Minor s = Sub Minor Example: 0x0122 = 1.2.2	

	Struct
	{
	unsigned long firm_type;
	unsigned int firm_rev;
	unsigned int firm_build;
	}
		Not Available

Get Firmware Version (Legacy)

	Binary	**ASCII**
Command	10	Not Available
Description	Requests the transducer firmware version. This command is valid for all firmware versions. For newer firmware versions based on an BCD representation, the output is an abbreviation.	
Parameters	None	
Return Value	Returns the transducer firmware version, in the format Major.Minor (rounding to 2 significant digits maybe required for correct formatting).	
	Float	Not Available

Transducer Data

Get Torque

	Binary	**ASCII**
Command	50	#50;
Description	Requests the current torque value, if averaging has been enabled, then this value is averaged.	
Parameters	None	
Return Value	Torque in the transducer's native units.	
	Float	#±0000000.000;

Get Peak Torque

	Binary	**ASCII**
Command	51	#51;
Description	Requests the current peak torque value.	
Parameters	None	
Return Value	Peak Torque in the transducer's native units.	
	Float	#±0000000.000;

Get Peak Torque Auto Reset

	Binary	**ASCII**
Command	52	#52;
Description	Requests the current Peak Torque Auto Reset value.	
Parameters	None	
Return Value	Peak Torque Auto Reset in the transducer's native units.	
	Float	#±0000000.000;

Get Peak Torque CW

	Binary	**ASCII**
Command	53	#53;
Description	Requests the current clockwise peak torque value.	
Parameters	None	
Return Value	Peak Torque value in the transducer's native units.	
	Float	#±0000000.000;

Get Peak Torque CCW

	Binary	**ASCII**
Command	54	#54;
Description	Requests the current counter-clockwise peak torque value.	
Parameters	None	
Return Value	Peak Torque value in the transducer's native units.	
	Float	#±0000000.000;

Get PeakMinMax Max

	Binary	**ASCII**
Command	55	#55;
Description	Requests the current max value from the PeakMinMax data.	
Parameters	None	
Return Value	PeakMinMax Max Torque value in the transducer's native units.	
	Float	#±0000000.000;

Get PeakMinMax Min

	Binary	**ASCII**
Command	56	#56;
Description	Requests the current min value from the PeakMinMax data.	
Parameters	None	
Return Value	PeakMinMax Min Torque value in the transducer's native units.	
	Float	#±0000000.000;

Get PeakMinMax

	Binary	**ASCII**
Command	57	#57;
Description	Requests the PeakMinMax data.	
Parameters	None	
Return Value	PeakMinMax data consisting of two torque values in the transducer's native units.	
	PeakMinMax structure	#Max,Min; #±0000000.000,±0000000.000;

Get Torque - Convert Units To

	Binary	**ASCII**
Command	60	#60,***UNITS***;
Description	Requests the current torque value and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	Torque in the selected unit. Parameter is acknowledged in ASCII mode.	
	Float	#ACK,±0000000.000;

Get Peak Torque - Convert Units To

	Binary	**ASCII**
Command	61	#61,***UNITS***;
Description	Requests the current peak torque value and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	Peak Torque value in the selected unit. Parameter is acknowledged in ASCII mode.	
	Float	#ACK,±0000000.000;

Get Peak Torque Auto Reset - Convert Units To

	Binary	**ASCII**
Command	62	#62,***UNITS***;
Description	Requests the current Peak Torque Auto Reset value and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	Peak Torque Auto Reset value in the selected unit. Parameter is acknowledged in ASCII mode.	
	Float	#ACK,±0000000.000;

Get Peak Torque CW - Convert Units To

	Binary	**ASCII**
Command	63	#63,***UNITS***;
Description	Requests the current clockwise peak torque value and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	Peak Torque value in the selected unit. Parameter is acknowledged in ASCII mode.	
	Float	#ACK,±0000000.000;

Get Peak Torque CCW - Convert Units To

	Binary	**ASCII**
Command	64	#64,***UNITS***;
Description	Requests the current counter-clockwise peak torque value and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	Peak Torque value in the selected unit. Parameter is acknowledged in ASCII mode.	
	Float	#ACK,±0000000.000;

Get PeakMinMax Max - Convert Units To

	Binary	**ASCII**
Command	65	#65,***UNITS***;
Description	Requests the current max value from the PeakMinMax data and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	PeakMinMax Max Torque value in the selected unit. Parameter is acknowledged in ASCII mode.	
	Float	#ACK,±0000000.000;

Get PeakMinMax Min - Convert Units To

	Binary	**ASCII**
Command	66	#66,***UNITS***;
Description	Requests the current min value from the PeakMinMax data and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	PeakMinMax Min Torque value in the selected unit. Parameter is acknowledged in ASCII mode.	
	Float	#ACK,±0000000.000;

Get PeakMinMax - Convert Units To

	Binary	**ASCII**
Command	67	#67,***UNITS***;
Description	Requests the PeakMinMax data and converts the native units to the selected units.	
Parameters	***UNITS*** - The ***UNITS*** parameter specifies the torque unit to convert the torque value to. Use the Unit key to find the corresponding value for the unit required.	
	Unsigned Char	***UNITS*** value as an additional parameter.
Return Value	PeakMinMax data consisting of two torque values in the selected unit. Parameter is acknowledged in ASCII mode.	
	PeakMinMax structure	#ACK,Max,Min; #ACK, ±0000000.000, ±0000000.000; [CR][LF] added for readability.

Get Speed

	Binary	**ASCII**
Command	100	#100;
Description	Requests the current speed value. The slow speed capture mode is used.	
Parameters	None	
Return Value	Speed in RPM.	
	Float	#±0000000.000;

Get Power

	Binary	**ASCII**
Command	101	#101;
Description	Requests the current power value in Watts. The slow speed capture mode is used.	
Parameters	None	
Return Value	Power in Watts.	
	Float	#±0000000.000;

Get Temperature Ambient

	Binary	**ASCII**
Command	102	#102;
Description	Requests the transducer's internal ambient temperature.	
Parameters	None	
Return Value	Temperature in degrees C (°C).	
	Float	#±0000000.000;

Get Temperature Shaft

	Binary	**ASCII**
Command	103	#103;
Description	Requests the transducer's shaft temperature.	
Parameters	None	
Return Value	Temperature in degrees C (°C).	
	Float	#±0000000.000;

Get SlowCap Speed

	Binary	**ASCII**
Command	110	#110;
Description	Requests the current speed value from the slow speed capture system.	
Parameters	None	
Return Value	Speed in RPM.	
	Unsigned Long	#±0000000.000;

Get FastCap Speed

	Binary	**ASCII**
Command	111	#111;
Description	Requests the current speed value from the fast speed capture system.	
Parameters	None	
Return Value	Speed in RPM.	
	Unsigned Long	±0000000.000

Get SlowCap Power in Watts

	Binary	**ASCII**
Command	112	#112;
Description	Requests the current power value in Watts, using the current torque and speed values. Speed from the slow capture system is used.	
Parameters	None	
Return Value	Power in Watts.	
	Float	#±0000000.000;

Get FastCap Power in Watts

	Binary	**ASCII**
Command	113	#113;
Description	Requests the current power value in Watts, using the current torque and speed values. Speed from the fast capture system is used.	
Parameters	None	
Return Value	Power in Watts.	
	Float	#±0000000.000;

Get SlowCap Power in HP

	Binary	**ASCII**
Command	114	#114;
Description	Requests the current power value in Horse Power (HP), using the current torque and speed values. Speed from the slow capture system is used.	
Parameters	None	
Return Value	Power in HP.	
	Float	#±0000000.000;

Get FastCap Power in HP

	Binary	**ASCII**
Command	115	#115;
Description	Requests the current power value in Horse Power (HP), using the current torque and speed values. Speed from the fast capture system is used.	
Parameters	None	
Return Value	Power in HP.	
	Float	#±0000000.000;

Transducer Control

Reset Specified Peaks

	Binary	**ASCII**
Command	146	#146,***FLAGS***;
Description	Resets the stored peak values or zero's the transducer, in accordance with the binary flags specified as an additional parameter. The ASCII implementation of this command is a straight forward request with a single parameter. The binary implementation requires some handshaking, which differs from a standard request. The parameter value is an Unsigned Int which is 2 bytes, because of this, some handshaking is required. The procedure for transmitting the value in binary mode, is outlined below: 1. Transmit the command byte (Unsigned Char) with a value of 146. 2. Receive a byte (Unsigned Char); the byte will have a value of 145, the value has no significance. 3. Transmit the 2 byte (Unsigned Int) reset request parameter, LSB first. 4. Receive a byte (Unsigned Char), this second byte acts as a confirmation, its value will be 145, again the value has no significance.	

<table>
<tr><td>Parameters</td><td colspan="2">
FLAGS - The FLAGS parameter specifies which stored values should be reset. The value is made up of binary flags, each flag signifying a value to reset.

The parameter value is calculated by adding together or OR’ing the flag values. The table below shows the reset flags and their respective values (values are hexadecimal).

<table>
<tr><th>Flag Value</th><th>Value to be reset</th><th>Description</th></tr>
<tr><td>0x01</td><td>Torque Zero</td><td>Zero’s the transducer.</td></tr>
<tr><td>0x02</td><td>Torque Zero with Average</td><td>Zero’s the transducer with an average value.</td></tr>
<tr><td>0x04</td><td>Peak Torque</td><td>Resets the Peak Torque to zero.</td></tr>
<tr><td>0x08</td><td>Peak Torque Auto Reset</td><td>Resets the Peak Torque Auto Reset to zero.</td></tr>
<tr><td>0x10</td><td>Peak Torque CW</td><td>Resets the Peak Torque CW to zero.</td></tr>
<tr><td>0x20</td><td>Peak Torque CCW</td><td>Resets the Peak Torque CCW to zero.</td></tr>
<tr><td>0x40</td><td>PeakMinMax</td><td>Resets the Min and Max values to the current torque value.</td></tr>
<tr><td>0x80</td><td>Peak FastCap Speed</td><td>Resets the Peak FastCap Speed value to zero.</td></tr>
<tr><td>0x100</td><td>Peak SlowCap Speed</td><td>Resets the Peak SlowCap Speed value to zero.</td></tr>
<tr><td>0x200</td><td>Peak FastCap Power</td><td>Resets the Peak FastCap Power value to zero.</td></tr>
<tr><td>0x400</td><td>Peak SlowCap Power</td><td>Resets the Peak SlowCap Power value to zero.</td></tr>
<tr><td>0x800</td><td>Angle</td><td>Resets the Angle position.</td></tr>
<tr><td>0x1000</td><td>Limit Signal</td><td>Resets the Limit signal.</td></tr>
</table>

Example:

To reset all the peak torque values (Peak Torque: 0x04, Peak Torque Auto Reset: 0x08, Peak Torque CW: 0x10, Peak Torque CCW 0x20, PeakMinMax: 0x40), the input value would be 0x7C.

Input Value: 0x7C = 0x04 + 0x08 + 0x10 + 0x20 + 0x40.

In ASCII mode, the parameter must be a decimal integer, i.e. a base 10 number. A hexadecimal number will cause a NAK.
</td></tr>
<tr><td></td><td>Unsigned Int</td><td>FLAGS value as an additional parameter.</td></tr>
<tr><td>Return Value</td><td>See Description.</td><td>#ACK;</td></tr>
</table>

Reset All Peak Torque Values

	Binary	**ASCII**
Command	147	#147;
Description	Resets the entire memory bank of stored peak values related to torque, values are reset to zero, except for the PeakMinMax values, which are reset to the current torque value.	
Parameters	None	
Return Value	None	#ACK;

Reset All Peak Values

	Binary	**ASCII**
Command	148	#148;
Description	Resets the entire memory bank of stored peak values related to torque, speed and power. Values are reset to zero, except for the PeakMinMax values, which are reset to the current torque value.	
Parameters	None	
Return Value	None	#ACK;

Reset System Values

	Binary	**ASCII**
Command	149	#149;
Description	Resets the entire memory bank of stored peak values related to torque, speed and power, then zero's the transducer using an averaged zero. All subsequent torque values are offset by the zero value. Peak values are reset to zero.	
Parameters	None	
Return Value	None	#ACK;

Peak Torque Reset

	Binary	**ASCII**
Command	150	#150;
Description	Resets the stored Peak Torque value to zero.	
Parameters	None	
Return Value	None	#ACK;

Peak Torque Auto Reset - Reset

	Binary	**ASCII**
Command	152	#152;
Description	Resets the stored Peak Torque Auto Reset value to zero.	
Parameters	None	
Return Value	None	#ACK;

Zero Transducer with Average

	Binary	**ASCII**
Command	155	#155;
Description	Zero’s the transducer torque value; all subsequent torque readings will be offset by an averaged torque value. When the command is sent to the transducer, the firmware will average over the next 32 torque samples, the averaged torque value is then stored as the zero offset.	
Parameters	None	
Return Value	None	#ACK;

Zero Transducer

	Binary	**ASCII**
Command	156	#156;
Description	Zero’s the transducer torque value; all subsequent torque readings will be offset by the torque value present when zeroed.	
Parameters	None	
Return Value	None	#ACK;

Get PeakMinMax & Reset

	Binary	**ASCII**
Command	173	#173;
Description	Requests the PeakMinMax data, then resets the stored values to the current torque reading.	
Parameters	None	
Return Value	PeakMinMax data consisting of two torque values in the transducer's native units. The reset is acknowledged in ASCII mode.	
	PeakMinMax structure	#Max,Min,ACK; #±0000000.000, ±0000000.000, ACK; [CR][LF] added for readability.

Set Torque Filter

	Binary	**ASCII**
Command	180	#180,***FILTER***;
Description	Enables and configures the torque filtering system.	
Parameters	***FILTER*** - Torque filter setting. Disables or sets the filtering sample level. Valid filter settings: 0 = OFF, 2, 4, 8, 16, 32, 64, 128, 256 (In binary format, send 255 for 256).	
	Unsigned Char	***FILTER*** value as an additional parameter.
Return Value	None	#ACK;

Get Torque Filter

	Binary	**ASCII**
Command	181	#181;
Description	Retrieves the current torque filter setting.	
Parameters	None	
Return Value	Torque filter setting. The value returned indicates the filter level. If zero is received, the filter is disabled. In binary format, a filter level of 256 is output as 255.	
	Unsigned Char	#000;

Set Speed Filter

	Binary	**ASCII**
Command	182	#182,***FILTER***;
Description	Enables and configures the speed filtering system.	
Parameters	***FILTER*** - Speed filter setting. Disables or sets the filtering sample level. Valid filter settings: 0 = OFF, 2, 4, 8, 16, 32, 64, 128, 256 (In binary format, send 255 for 256).	
	Unsigned Char	***FILTER*** value as an additional parameter.
Return Value	None	#ACK;

Get Speed Filter

	Binary	**ASCII**
Command	183	#183;
Description	Retrieves the current speed filter setting.	
Parameters	None	
Return Value	Speed filter setting. The value returned indicates the filter level. If zero is received, the filter is disabled. In binary format, a filter level of 256 is output as 255.	
	Unsigned Char	#000;