WinEasyTORK Manual

Version 1.0

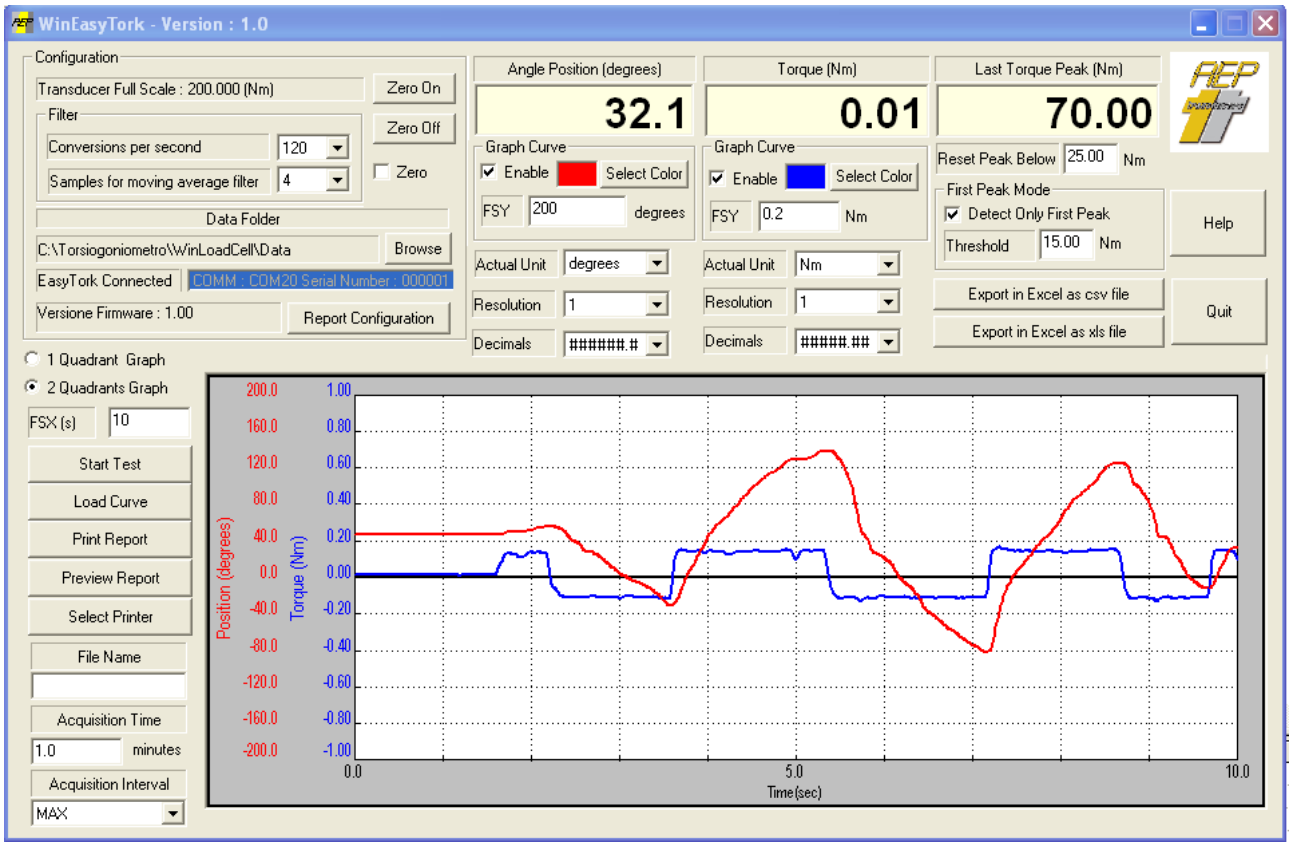
Index

WinEasyTORK is the AEP transducers environment for the real-time monitoring on PC of EasyTORK transmitter on the USB line.

It implements all the communication protocol features, create test charts , print reports, export data in Microsoft Excel .

Data can be acquired in different units at different filters and conversion speed.

The report can be configured with the customer's logo and explanatory notes to describe the test performed.



WinEasyTORK LICENSE AGREEMENT

1.

We grant the customer a non-exclusive and non transferrable right to use the AEP transducers **WinEasyTORK** software on a single computer and at only one place.

2

The license holder is not permitted:

a) to pass on or otherwise make the software or associated material accessible to third parties without prior written agreement from AEP transducers

b) to modify the software without prior written consent from AEP transducers

c) to create works derived from software or to duplicate the written material

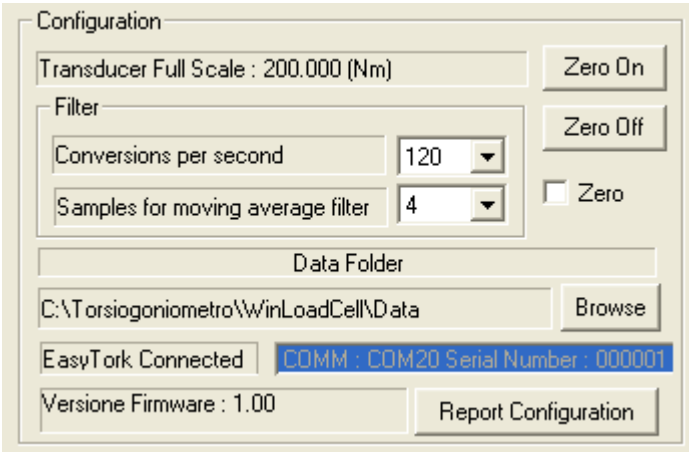
d) to translate or modify it to create works derived from the written material

3

The right of use is dependent on the customer observance of these general conditions of supply

Acceptance of this license agreement is formally required during the installation of the software.

1.0 WinEasyTORK Configuration



The window **Filter** allows you to define the operative mode of the EasyTORK transmitter.

1. **Conversions per second** : 5 – 20 – 120 – 600 – 1200 - 2400 - 4800 Hz

   It defines the conversion speed of the EasyTORK Analog to Digital Converter (ADC)

2. **Samples for moving average filter** : 1- 2- 4 – 8 – 16 -32

   It defines how many samples point will be taken in account in the microprocessor digital filter.

The Zero On – Off set/ reset the transducer torque and angle position zero value

The selection **1 Quadrant Graph** and **2 Quadrant Graph** select the graph type.

Selecting 1 Quadrant the Y axis will be shown for just positive values .

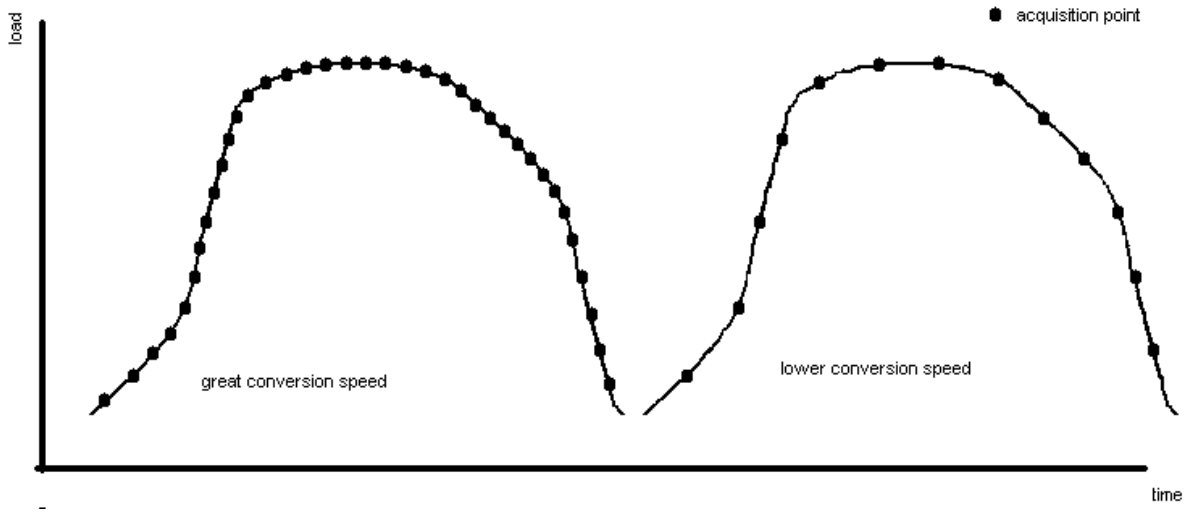
Selecting 2 Quadrant both positive and negative values will be shown .

X axis is always time (is second)

The Data Folder set the directory where curve files will be stored. Use the **Browse** button to change it.

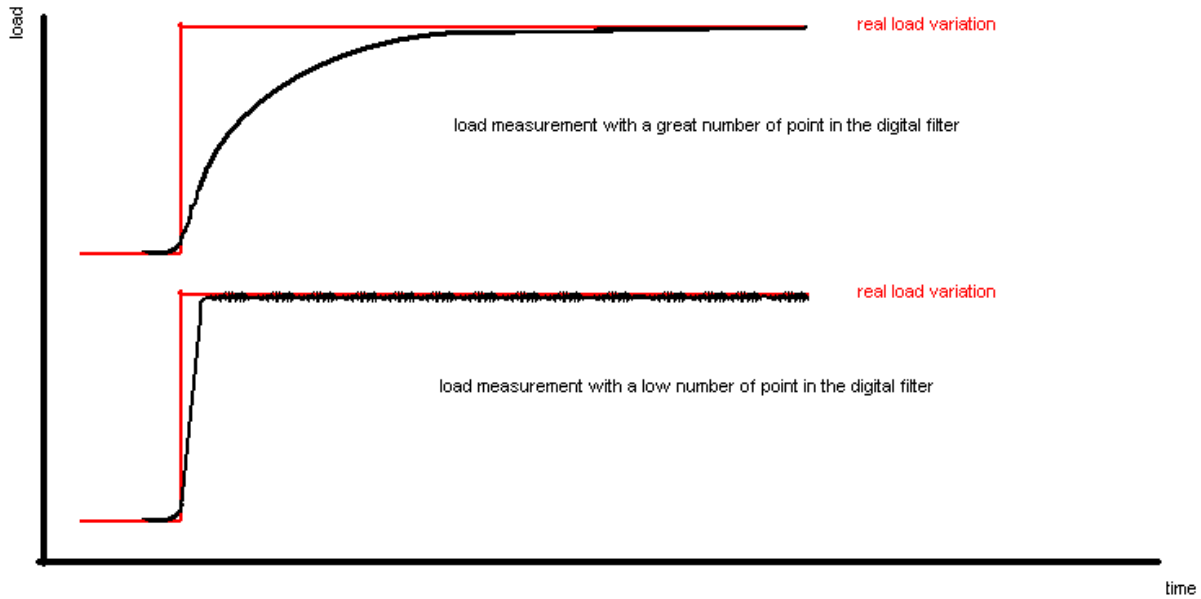
The **Conversion Speed** and the **Samples for moving average filter** parameters must be chosen according to the application.

If you need to detect very fast peaks probably you need to select 2400Hz or 4800Hz conversions per second . If you need to have more accurate measure you are suggested to use 5 or 20 conversions per second. The effect can be described in the picture below. Greater is the conversion speed and you will have more points in the digital curve so you can reconstruct the real curve better. On the other hand greater conversion speed means more residual noise in the measurement so the operator must select the conversion speed keep in mind the right compromise

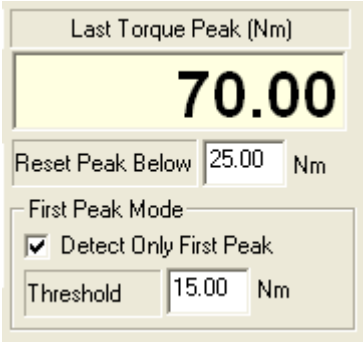


between speed and accuracy.

The effect of changing the number of samples for the moving average filter can be described with the picture below. Great number of point in the digital filter mean a more accurate measure but fast changing in the load are detected with a delay. On the other hand low number of point in the digital filter allow to follow faster any change in the load but it is possible to have a little residual noise in the measure



PEAK MODE



WinEasyTORK always shown Last Peak value detected both for positive and negative values.

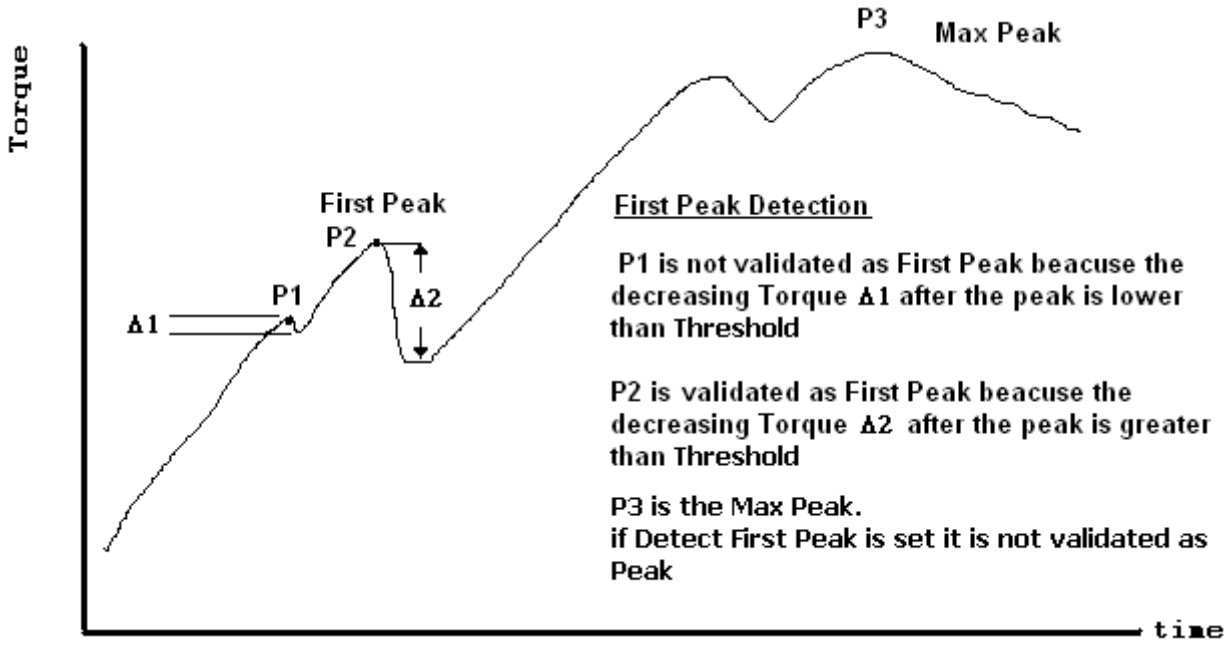
The computed Peak value will be reset for torque values below the Reset Peak Parameter.

If the Detect Only First Peak check box is set only the first Peak detected is validated.

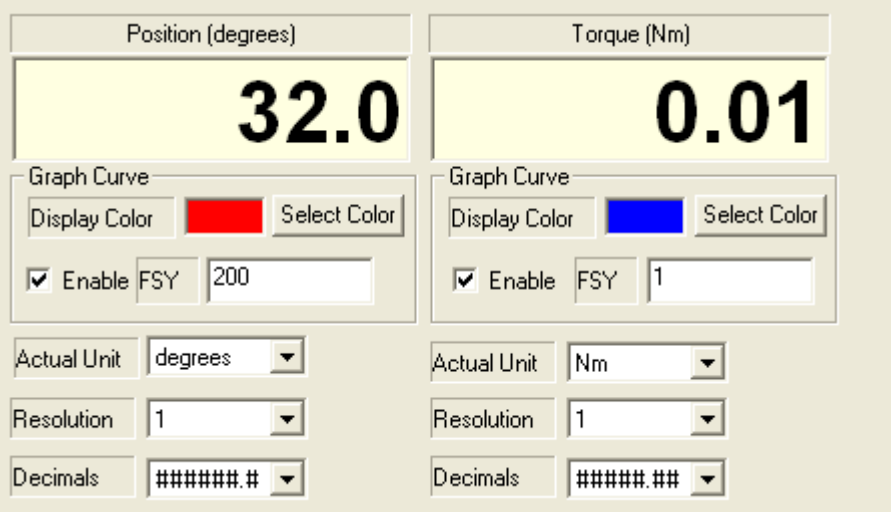
To avoid that not valid first peaks will be computed it is necessary to set a Threshold parameter that establish a decreasing torque value after the peak to validate the first peak.

The figure below better describe the way how the peak is managed

If Detect Only First Peak is not checked the point P3 is always computed as Peak



2.0 Real Time Channels Setup



The EasyTORK transmits in real time 2 type of informations.

1. **Position or Speed**
2. **Torque**

The transmission frequency is determined by the conversion speed parameter. So if the conversion speed is for example 120 the transmission data rate will be 120.

In this window you can select how these informations will be managed by the software.

You can select the graph color, the Y axis full scale and the enabled state.

- **Actual Unit :** selects the Unit for the relevant parameter. Selecting degrees EasyTORK positions will be displayed. Selecting rpm or Hz the EasyTORK speed will be displayed.
- **Resolution :** selects the parameter resolution among 1-2-5-10-20-50-100
- **Decimals :** with this selection you can set the dot position for the relevant parameter

3.0 Make a Test

Pressing the "**Start Test**" button. **WinEasyTORK** begins to store data and to show the relevant graph.

You can define the **Acquisition Time** (always in minutes) that defines the interval time to acquire a point and the **Test Time** that defines how long the test should last.

All curves are shown in the measurement unit defined in the Actual Unit window. Only the enabled curves will be shown.

The Time Axis is always in seconds.

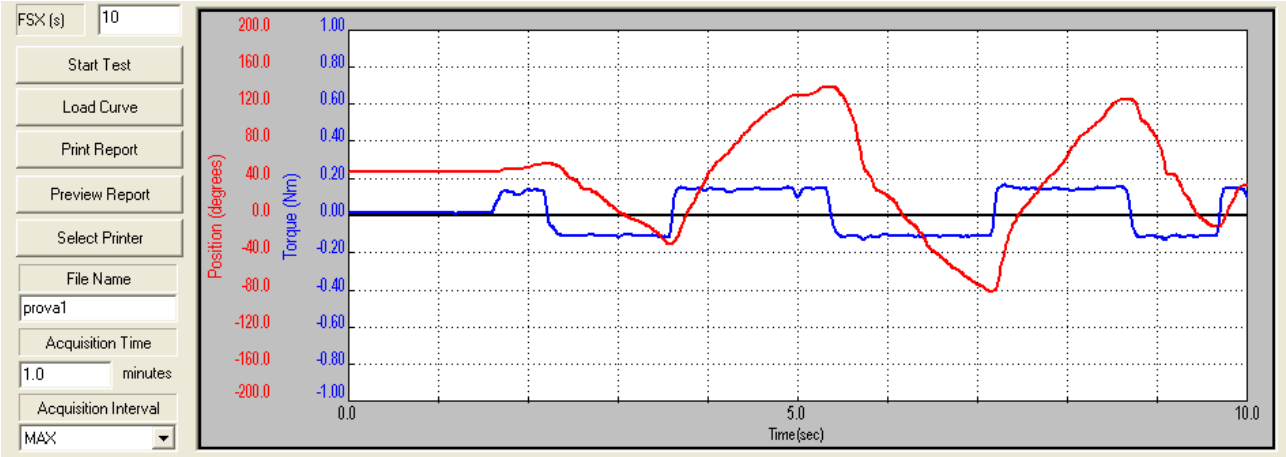
To stop the test you can also press the Button "**STOP Test**" in any moment.

If the field **File Name** is filled with a valid filename a new test file is automatically saved in the folder selected in the Data Folder Window.

At the end of the test all the acquired data can be saved in an **Microsoft Excel** file by pressing "**Export in Excel as xls file**" . This function requires that Microsoft Excel must be installed on your computer. It is possible on the other hand to **Export data in Excel using .csv file**. This mode does not require Microsoft Excel installed on your computer

Warning

For acquisition speed equal or above 1200, due to the great quantity of data the software must manage, not all acquired points are displayed on the graph in real time during a test. Any way the complete acquisition points will redrawn at the end of the test.



You can **Print** a report or just have a preview by pressing the relevant buttons. The test image printed is always the one shown on the screen.

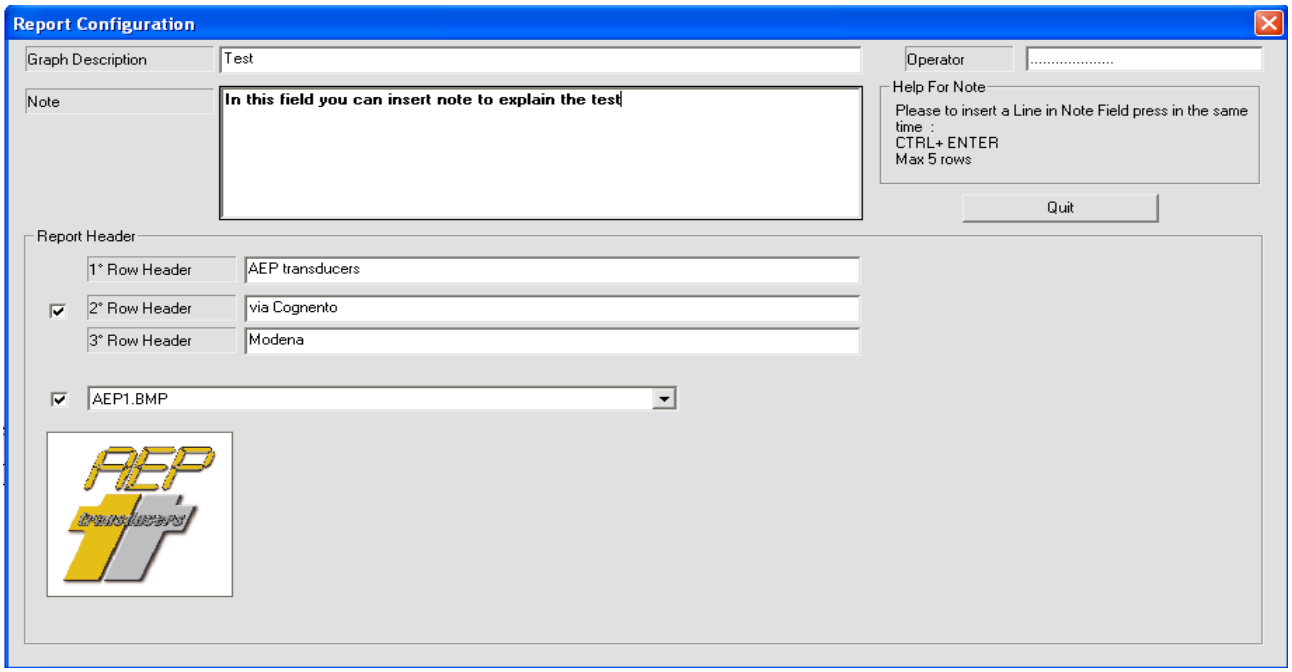
By pressing the **Load Curve** button it is possible to review tests in the archive.

4.0 Report Configuration

This page allows to configure your report.

The Report Header can be filled with your logo or with up to 3 rows of text or both.

Make your choose by using the selection shown.



You can have more than a logo image. You can select the desired one selecting it from the list.

To add a logo image it is very simple.

1. Create a logo image as a bitmap file
2. Copy the file in the folder “Loghi” you can find in the WinEasyTORK installation folder

To better describe your test in the report you can add :

- Notes : Free text (Up to 5 row of rows). Be carefully :To create a new line in this windows press CTRL+ENTER
- A Test Description
- The Operator Name

5.0 **Communication Protocol**

EasyTORK is a USB device. Anyway to simplify the communication, from the PC point of view, it is simulated a virtual serial port.

In this way a programmer that wants to create its own communication program can use the usual function for a serial port.

The only thing it is necessary to know is the COM port Windows has assigned to the EasyTORK device. This can be done in the Windows Control Panel.

Because EasyTORK is not a real COM port, the user program can open the COM port with any serial parameter. The only important parameter is to set 8 Bits for the Data Bits parameter

For the Baud Rate, Parity, Stop Bit it is possible to set any values.

EasyTORK continuously transmits the actual values data in binary form. The transmission frequency is determined by the “Conversion Speed” parameter.

Beside the continuously transmitted actual value, it is possible to require other values such as

- Status information
- The transducers capacity
- The firmware version
- EasyTORK serial number

To require these further data it is necessary to send a command

Other commands allow to set the Filter, the conversion speed, the Zero transducer /position ON/Off , and the peak mode .

For the description of available commands see Chapter 5.2

5.1 RECEIVED DATA PACKET

Each received data packet is composed by 12 byte in binary form.

The first byte of the data packet is recognized by the SYNC bit.

The SYNC bit is the BIT B7 of the first byte of each data packet

BYTE N.	B7	B6	B5	B4	B3	B2	B1	B0	Comment
1	1	Byte – SYNC BIT + OP_CODE							BIT B7 bit is the SYNC for the incoming data packet. The other 7 bits are OP_CODE that describe the meaning of following bytes See Table 1
2	0	Byte 1							OP_CODE : '0' – '2' – '3' – '4' contain the value selected by the OP_CODE in floating point according to IEEE754. OP_CODE : '1' Contains status informations OP_CODE : '7' Contains the first 5 characters of the transducers S/N in ASCII
3	0	Byte 2							
4	0	Byte 3							
5	0	Byte 4							
6	0	Byte 5							
7	0	Byte							OP_CODE '0' : Measurement unit for torque and position/speed OP_CODE '7' : contains th sixth ASCII characters of the transducers S/N
8	0	Byte 1							OP_CODE : '0' Position/speed value in step (long format: 32 bit integer) All others OP_CODE : no meaning
9	0	Byte 2							
10	0	Byte 3							
11	0	Byte 4							
12	0	Byte 5							

5.1.1 SYNC BIT AND OP_CODE

EasyTORK continuously transmits the actual value. The transmission frequency is determined by the “Conversion Speed” parameter.

For example : if you use a conversion speed of 100 conversions per second a new data value will be transmitted every 10ms (1000/100).

Any way it is possible to ask to EasyTORK different data (i.e. the firmware version, the status and so on). In this case, among the actual value data packets, is transmitted the required data packet.

To recognize from the incoming data the relevant packet it is necessary to decode the first byte of the data packet. This byte contains the OP_CODE.

To recognize the OP_CODE byte it is necessary to check Bit B7

B7 bit of the OP_CODE byte is always set to 1.

B7 bit of other byte are always set to 0

Valid OP_CODE are

OP_CODE	Data Packet
‘0’+SYNC 0xB0=0x30+0x80	Actual Values. Continuosly transmitted data packet
‘1’+SYNC 0xB1=0x31+0x80	Read Status
‘2’+SYNC 0xB2=0x32+0x80	Read Transducer Capacity
‘4’+SYNC 0xB4=0x34+0x80	Read Firmware version
‘7’+SYNC 0xB7=0x37+0x80	Read Transducer Serial Number

Table 1

5.1.2 DECODING DATA VALUE

The second step is to decode the data inside the data packet. According to the OP_CODE it is necessary to use different decoding mode.

DECODING OP_CODE : '0' – '2' – '3' – '4'

Byte 1 to Byte 5 and Byte 7 to Byte 11 contain the data values selected by the OP_CODE. The first value is in floating point according to IEEE754. The second byte is in long (32bit integer)

Because each byte is composed by only 7 bits (bit B7 is always set to 0) are necessary 5 bytes to transport the value information.

Byte 1 to Byte 4 contain the first 7 bits of each byte for the first data value

Byte 5 contains the 8th bit of each byte for the first data value

Byte 6 contains the measurement unit for the first and second value

Byte 7 to Byte 10 contain the first 7 bits of each byte for the second data value

Byte 11 contains the 8th bit of each byte for the second data value

The table below explain how to rebuild the values from the incoming data packet

First actual value (float)

Float	B7	B6	B5	B4	B3	B2	B1	B0
0	Bit 0 –Byte5	First 7 bit of Byte 1						
1	Bit 1 –Byte5	First 7 bit of Byte 2						
2	Bit 2 –Byte5	First 7 bit of Byte 3						
3	Bit 3 –Byte5	First 7 bit of Byte 4						

Second actual value (long) : only opcode '0'

long	B7	B6	B5	B4	B3	B2	B1	B0
0	Bit 0 –Byte11	First 7 bit of Byte 7						
1	Bit 1 –Byte11	First 7 bit of Byte 8						
2	Bit 2 –Byte11	First 7 bit of Byte 9						
3	Bit 3 –Byte11	First 7 bit of Byte 10						

Values unit

byte	B7	B6	B5	B4	B3	B2	B1	B0
	Index of second value unit				Index of first value unit			

DECODING OP_CODE '0'

The Op_code '0' is transmitted automatically with a data rate equal to the conversion speed selected.

It contains the actual data values Torque and Position or Speed.

Torque is the first actual value and it is transmitted in floating point according to IEEE754. Its value is already in engineering unit. Its unit is represented by the index of first value according to the table below

Index of first value	Torque Unit
0	Nm
1	Nmm
2	kgm
3	kNm
4	in.ft
5	ft.lbf
6	gcm
7	kgmm
8	Nm
9	Nm

Position/ Speed is the second value. It is transmitted as long 32 bit integer.

In case of Unit = 0 it contains the steps counter from last zero command

If Unit <>0 it contains the number of steps detected every 100ms

The numbers of step for every complete EasyTORK revolution (360°) depends on the type

5760 steps/revolution for EasyTork

3520 steps / revolution for RT2 type 1

8000 steps / revolution for RT2 type 2

Index of second value	Unit
0	Steps
1	Steps / 100ms
2	Steps / 100ms

The following C code explain how to decode the data packet

Example in C

RxBuffer is the incoming serial buffer. Indice is the index of the desired OP_CODE inside RxBuffer

```
#define IMPULSI_PER_GIRO      -5760.0          //use minus if you want positive value in clockwise direction
#define DEGREE_RESOLUTION   360.0/IMPULSI_PER_GIRO

void GetFloat(int Indice)
{
union
{
unsigned char Byte[4];
long  L;
float F;
} L;

        L.Byte[0]=RxBuffer[Indice+1] & 0x7f;
        L.Byte[1]=RxBuffer[Indice+2] & 0x7f;
        L.Byte[2]=RxBuffer[Indice+3] & 0x7f;
        L.Byte[3]=RxBuffer[Indice+4] & 0x7f;
        if (RxBuffer[Indice+5] & 0x01) L.Byte[0]|=0x80;
        if (RxBuffer[Indice+5] & 0x02) L.Byte[1]|=0x80;
        if (RxBuffer[Indice+5] & 0x04) L.Byte[2]|=0x80;
        if (RxBuffer[Indice+5] & 0x08) L.Byte[3]|=0x80;
        Torque=L.F;

        ActualUnit[0]= RxBuffer[Indice+6]&0xf;                  // index of Torque Unit
        ActualUnit[1]= (RxBuffer[Indice+6]&0x30)>>4;            //index of position/speed unit

        L.Byte[0]=RxBuffer[Indice+7] & 0x7f;
        L.Byte[1]=RxBuffer[Indice+8] & 0x7f;
        L.Byte[2]=RxBuffer[Indice+9] & 0x7f;
        L.Byte[3]=RxBuffer[Indice+10] & 0x7f;
        if (RxBuffer[Indice+11] & 0x01) L.Byte[0]|=0x80;
        if (RxBuffer[Indice+11] & 0x02) L.Byte[1]|=0x80;
        if (RxBuffer[Indice+11] & 0x04) L.Byte[2]|=0x80;
        if (RxBuffer[Indice+11] & 0x08) L.Byte[3]|=0x80;

        switch(ActualUnit[1])                   //convert speed/position from step to engineering unit
        {
        case 0:
                Position=(float)L.L*DEGREE_RESOLUTION;          // degree
                break;
        case 1:
                Speed=(float)L.L*600.0/IMPULSI_PER_GIRO;        // rpm
                break;
        case 2:
                Speed=(float)L.L*10/IMPULSI_PER_GIRO;           // Hz
                break;
        }
}
```

DECODING OP_CODE 1 :

The OP_CODE 1 is a STATUS Information data packet. It is an answer to the READ STATUS Command

OP_CODE 1 does not contain a float value but each byte has its own meaning. By decoding this data packet it is possible to check the validity of some other commands

Byte 1 : Number of samples in the moving average filter :

value	samples
0	1
1	2
2	4
3	8
4	16
5	32

Byte 2 : Conversion Speed

value	conversions per second
0	5
1	20
2	120
3	600
4	1200
5	2400
6	4800

Byte 3: reserved

Byte 4 : It contains the Zero On/Off Status and the EasyTORK operative mode

B7	B6	B5	B4	B3	BIT 2	BIT 1	BIT 0
0	0	0	0	0	00 : Normal Mode 01 : Peak - On 11 : Peak+ On		0 = Zero off 1 = Zero On

Byte 5 to byte 11 : reserved

DECODING OP_CODE 2 :

The OP_CODE 2 contains the Torque Full Scale value . It is an answer to the READ TORQUE FULL SCALE Command .

The Full Scale Torque is the first actual value and it is transmitted in floating point according to IEEE754. Its unit is always Nm

DECODING OP_CODE 4 :

The OP_CODE 4 contains the firmware version . It is an answer to the READ FIRMWARE VERSION Command

The Firmware Version is the first actual value and it is transmitted in floating point according to IEEE754.

DECODING OP_CODE 7

OP_CODE 7 contains the Transducers Serial Number .

It is a 6 characters ASCII code .

Byte 1 to Byte 6 contains each a character.

Byte 7 contains the code that defines the type of transducer connected:

‘0’ : identifies an EasyTork Transducers

‘1’ : identifies an RT2 transducers.type 1

‘2’ : identifies an RT2 trasducers type 2

It is very important to decode this byte because the number of steps for revolution depends on this value

5760 steps/revolution for EasyTork

3520 steps / revolution for RT2 type 1

8000 steps / revolution for RT2 type 2

All other characters have no meaning

5.2 COMMANDS

Transmitting commands to EasyTORK allows to set operative modes and to read data different from actual values.

Each command is composed by 15 characters starting with the Character '$' and terminating with the character <CR> (0x0d)

5.2.1 Read Commands

Read Status : **$C100000000000<CR>** answer : OP_CODE 1

Read Full Scale Torque : **$C200000000000<CR>** answer : OP_CODE 2

Read Firmware Version : $**C400000000000<CR>** answer : OP_CODE 4

Read Serial Number : **$C700000000000<CR>** answer : OP_CODE 7

The answer to these commands are discussed in the previous paragraph

5.2.2 Operative Commands

Zero ON : **$A100000000000<CR>**

Zero OFF : **$A000000000000<CR>**

SET Normal Operative Mode : **$A200000000000<CR>**

SET PEAK+ Operative Mode : **$A211000000000<CR>**

SET PEAK- Operative Mode : **$A210000000000<CR>**

These commands does not have an answer.

To read the new Status please use the Read Status Command

SET PARAMETER Command : **$L2ABCD0000000<CR>**

This command allows to set some working parameter of EasyTORK

This command does not have an answer.

To read the new Status please use the Read Status Command

A , B, C and D are single characters field that must be coded as shown in the tables below

Field A : Torque Measurement Unit : valid range : '0'..'9'

Measurement Unit	
'0'	Nm
'1'	Nmm
'2'	kgm
'3'	kNm
'4'	in.lbf
'5'	ft.lbf
'6'	gcm
'7'	kgmm
'8'	Nm
'9'	Nm

Field B: Number of samples in the mobile average filter

Value	samples
'0'	1
'1'	2
'2'	4
'3'	8
'4'	16
'5'	32

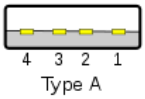
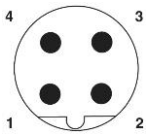
Field C : Conversion Speed

value	conversions per second
‘0’	5
‘1’	20
‘2’	120
‘3’	600
‘4’	1200
‘5’	2400
‘6’	4800

Field D : Position/Speed Selection

value	conversions per second
‘0’	Select Position
‘1’	Select Speed
‘2’	Select Speed

6.0 Electrical connections

USB	Pin		Description	Color cable		Pin	M12
 Type A	1	---	VBUS (5Vdc)	**ROSSO** RED	---	1	
	2	---	D-	**BIANCO** WHITE	---	2	
	3	---	D+	**VERDE** GREEN	---	3	
	4	---	GND	**NERO** BLACK	---	4	