Force/Torque sensors user manual

This document is a user manual for Bota Systems Force/Torque sensors

Revision1.2

Bota Systems AG – Leonhardstrasse 21, LEE H303, 8092 Zurich, Switzerland – Tel. +41 44 632 78 36, Email: info@botasys.com

1 Table of Contents

1 Table of Contents 2
2 List of Abbreviations and Terms 3
3 Safety 4
3.1 Explanation of notes 4
3.2 General safety guidelines 4
3.3 Safety precautions 5
4 Product overview 6
5 Installation 7
5.1 Adapter plates 7
5.2 Electrical 7
5.2.1 Serial 8
5.2.2 EtherCAT 9
5.3 Mechanical 10
6 Operation 11
6.1 Sensor environment 11
6.2 Sensor operation overview 11
6.3 Sensor configuration 11
6.3.1 Force torque filtering and output rate 11
6.3.2 IMU filtering and output rate 12
6.3.3 Force torque Bias 13
6.4 Serial interface 13
6.4.1 Serial configuration interface 13
6.4.2 Serial data packet 14
6.4.3 Sample code for receiving data (C) 16
6.4.4 Sample code for CRC16 X25 calculation (C) 19
7 EtherCAT interface 20
7.1 PDO interface 20
7.2 SDO interface 20
8 Maintenance 22
8.1 Inspection 22
8.2 Calibrating 22
9 Troubleshooting 23
9.1 Force/Torque reading errors 23
10 Specifications 24
10.1 Serial devices 24
10.2 EtherCAT devices 24
11 Complex loadings 25

2 List of Abbreviations and Terms

Table 1 - list of abbreviations

Term	Definition
ADC	Analog to Digital Converter.
Calibration	The act of measuring the transducer’s raw response to loads and creating data used in converting the response to force and torques.
Calibration Point	Refers to the reference frame at which the sensor is calibrated and the data are valid for. With proper translation this point can be changed if the user desires.
CRC	Cyclic Redundancy Check.
DoF	Degrees of freedom.
IMU	Inertial measurement unit
F/T sensor	Force Torque Sensor
PSU	Power supply unit
POE	Power over Ethernet
SNR	Singal to noise ratio
6σ	6 times the standard deviation
LED	Ligh emmite diodes

3 Safety

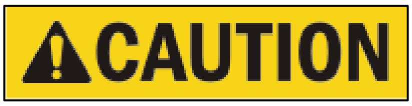
The safety section describes general safety guidelines for the product(s), an explanation of the notifications found in this manual, and the safety precautions applicable to the product(s). More specific notifications are embedded within the sections of the manual where they apply.

3.1 Explanation of notes

The notifications included here are specific to the products covered by this manual. The user should also be aware of the notifications of other components from other manufacturers installed in the system /robot.

DANGER: Indicate[s] a hazardous situation which, if not avoided, will result in death or serious injury. The signal word "DANGER" is to be limited to the most extreme situations. DANGER [signs] should not be used for property damage hazards unless personal injury risk appropriate to these levels is also involved.

WARNING: Indicate[s] a hazardous situation which, if not avoided, could result in death or serious injury. WARNING [signs] should not be used for property damage hazards unless personal injury risk appropriate to this level is also involved.



CAUTION: Indicate[s] a hazardous situation which, if not avoided, could result in minor or moderate injury. CAUTION [signs] without a safety alert symbol may be used to alert against unsafe practices that can result in property damage only.

NOTICE: Notification of specific information or instructions about maintaining, operating, installation, or setup of the product that if not followed could result in damage to equipment. The notification can emphasize but is not limited to specific grease types, good operating practices, or maintenance tips.

3.2 General safety guidelines

The user should verify that the force torque sensor is rated for maximum loads and torques expected from the operation. The user should be aware of the dynamic loads caused by the robot during acceleration or deceleration of the mounted masses.

3.3 Safety precautions

WARNING: Performing maintenance or repair on the sensor, while circuits (e.g. power, water, and air) are energized could result in serious injury. Discharge and verify all circuits are deactivated in accordance with the user's safety practices and policies.

CAUTION: Modifying or disassembly of the sensor could cause damage. Use the robot or adapter mounting bolt pattern and the tool side mounting bolt pattern to mount the sensor to the robot and user tooling to the sensor. Refer to the force/torque sensor model drawings and specifications sheet for more information.

CAUTION: Using fasteners that exceed the user mounting bolt pattern interface depth penetrates the body of the sensor, damages the electronics. Refer to the force/torque sensor model drawings and specifications for more information.

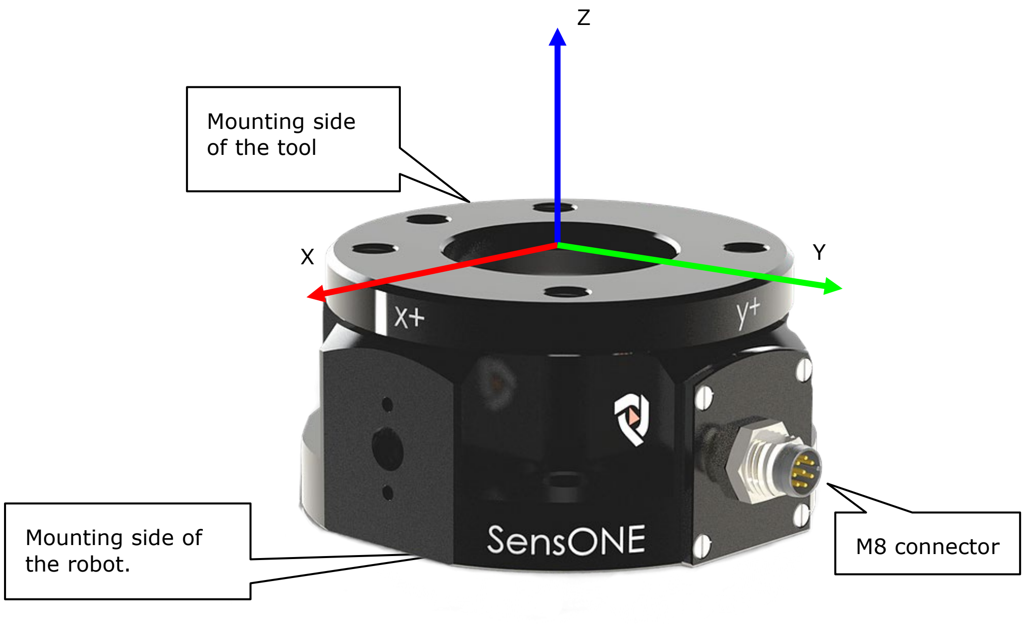
CAUTION: Do not overload the sensor. Exceeding the single-axis overload values of the sensor, causes irreparable damage.

CAUTION: Overload values refer to static overloading the sensor. This shouldn't be confused with the dynamic loading. The sensor should be dynamically loaded at the rated values of force and torque.

4 Product overview

The Bota Systems F/T sensor system measures (6) components of force and torque (Fx, Fy, Fz, Mx, My, Mz) and streams data to user devices that use Serial or EtherCAT communication. The sensor has two mounting sides. These sides are the connecting mechanical interfaces of the sensor. Refer to the force/torque sensor model drawings and specifications for more information. The sensor is IP67 rated. An IP67 rated connector is for the cable assembly provided with the sensor. The sensor is powered through this connector. For the electrical connector pin assignments, refer to 5.2. The Bota Systems sensors provide resolved force and torque data measured in N and Nm accordingly. Each F/T sensor model has its own reference frame which disclosed in its specification sheet.

Figure 1 – Example sensor Illustration. SensONE



NOTICE

NOTICE: The Sensor is powered through the cable connector. The USB version is powered directly from the USB port.

5 Installation

⚠WARNING

WARNING: Performing maintenance or repair on the sensor, while circuits (e.g. power, water, and air) are energized could result in serious injury. Discharge and verify all circuits are deactivated in accordance with the user's safety practices and policies.

⚠CAUTION

CAUTION: Modifying or disassembly of the sensor could cause damage and void the warranty. Use the adapter mounting bolt pattern and the tool side mounting bolt pattern to mount the sensor to the robot and user tooling to the sensor. Refer to the force/torque sensor model drawings and specifications for more information.

⚠CAUTION

CAUTION: Using fasteners that exceed the user mounting bolt pattern interface depth penetrates the body of the sensor, damages the electronics, and voids the warranty. Refer to the force/torque sensor model drawings and specifications for more information.

⚠CAUTION

CAUTION: Thread lockers and fasteners must not be used more than once. Fasteners can become loose and cause damage to the sensor environment or to itself. New fasteners and thread locker should be used.

⚠CAUTION

CAUTION: The cable connector should be mounted gently. No excessive force should be applied to the sensor and the cable connector. Gently align the keyway of the connector and mount the connector to avoid applying excessive force to connector and thus damage the sensor.

5.1 Adapter plates

When designing an adapter plate the following points need to be considered:

- The adapter should include bolt holes for mounting to the sensor, a dowel locating pin and a cylindrical boss for accurate positioning of the adapter to the robot/system.
- The adapter should provide a coaxial alignment of the sensor's z axis and flat and parallel mounting surface for the sensor.
- Sufficient thread length should be provided in order to be properly mounted on the robot/system. The adapter should be designed and calculated to bear the overload force torque of the sensor at least.
- Refer to each sensors specification sheer for correct fasteners types and dimensions.

5.2 Electrical

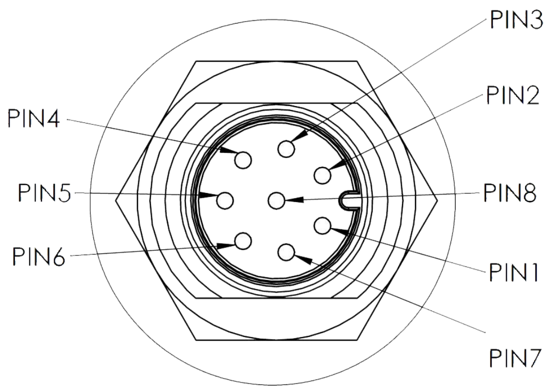
The sensors are provided with 3 meters cable with an RJ45 connector (see Figure 2). For Serial sensors a USB cable is also available under special request, this cable can be customized to customer's length. The static bending radius of the cable should not be less than 30mm, for non-static mounting (for example energy chain) the turning radius should be more than 60mm.
The sensor cable should be routed such that there is no stress applied on it. Stress can be induced from pulling, bending, kinking. The cable jacket material should be protected from sharp edges and hot surfaces.
The cable near the sensor connector should be restricted as stressing the connector mounting can affect the measurements. To connect the M8 end of the cable to the sensor, align the slot with the key of the connector and apply gentle force. Do not rotate the connector while connecting to the sensors as it could bend the pins of the sensor's connector.

⚠CAUTION

CAUTION: For correct sensor operation the shielding should be connected to GND (Vss-) in the RJ45 end of the cable. The connection can be found in Table 2 and Table 3 for serial and EtherCAT devices respectively.

Figure 2 - RJ45 cable (left), USB cable (right)[1]

Figure 3 - M8 connector pin numbers



5.2.1 Serial

The serial sensors can be provided either with a cable breakout board with wire terminal for wiring the RS-422 signals to native RS-422 devices or with a USB to RS-422 device to directly interface with a PC. The USB to RS-422 device will also power the device with the appropriate supply voltage, it is using an FTDI chip to implement a virtual serial port for maximum compatibility and driver support. In the case of the cable breakout board the user should provide the device with the nominal power supply voltage from a noise free PSU and use proper RS-422 termination. Both options can be found in Figure 4. For easy and stress free installation, the USB to RS-422 is recommended. When the sensor is used with the cable breakout board is recommended to connect the shielding to the Vss- (see Table 2) for achieving the specified signal to noise ratio.

[1] The USB cable is only available for the Serial sensors

Figure 4 - Cable breakout board for RS-422 applications (left), USB to RS422 board (right)

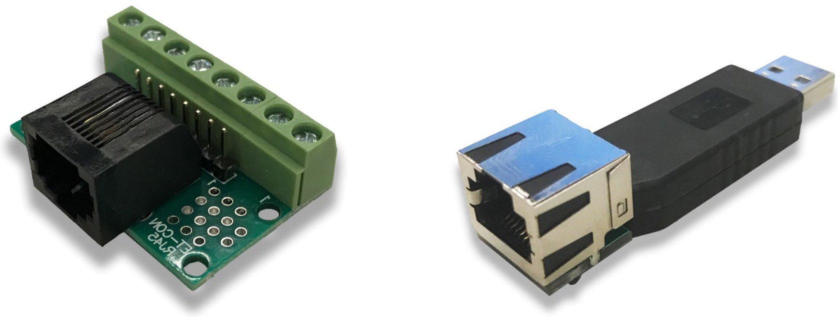
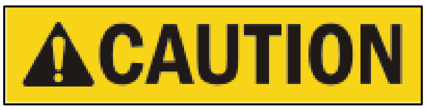


Table 2 - Serial cabling pinout

RJ45 Pins & breakout	Cable wires color	M8 Pins	RS-422 Signals
1	Orange/White	6	Tx+ (sensor output)
2	Orange	4	Tx- (sensor output)
3	Green/White	5	Rx+ (sensor input)
4	Blue	7	Vss+ (5V)
5	Blue/White	1	Vss+ (5V)
6	Green	8	Rx- (sensor input)
7	Brown/White	2	Vss- (0V)
8	Brown	3	Vss- (0V)
Shield	Shield	Shield	Shield

5.2.2 EtherCAT

The EtherCAT device are provided with a POE injector and a 48V AC/DC power supply (see Figure 5). The sensor is complying with **802.3af Mode B** protocol, this means that the power is being delivered by the two unused pair of the Cat6 ethernet cable (see Table 3). The user should connect the sensor cable to the “POE” marked port of the injector and the EtherCAT cable to the “LAN” port. The power supply should be connected to the “DC” port. To interface with the sensor, a computer with an ethernet adapter that supports “raw mode” is required. Other POE Injectors compatibible with the **802.3af Mode B** can be used with either 24V or 48V supply voltage.



CAUTION: The POE injector supplied with the sensor is connecting the shielding of the cable and the sensor body with the Vss-. For this reason an isolated PSU should be considered in the installation.

WARNING: The use of an **802.3af Mode A** POE injector will damage the sensor.

Figure 5 - POE Injector (left), 48V AC/DC PSU (right)

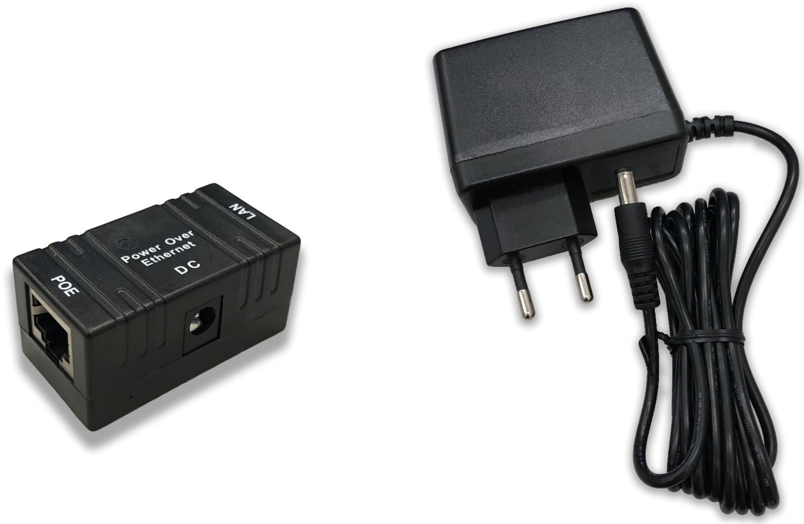


Table 3 - EtherCAT cabling pinout

RJ45 Pins & breakout	Cable wires color	M8 Pins	EtherCAT Signal
1	Orange/White	6	Tx/Rx+
2	Orange	4	Tx/Rx -
3	Green/White	5	Rx/Tx+
4	Blue	7	Vss+ (9-70V)
5	Blue/White	1	Vss+ (9-70V)
6	Green	8	Rx/Tx -
7	Brown/White	2	Vss- (0V)
8	Brown	3	Vss- (0V)
Shield	Shield	Shield	Shield

5.3 Mechanical

Several preparations should be done in order to properly install the sensor to a robot/system.

- The mounting surfaces should be cleaned and free of dust and other debris.
- Always use the correct amount of thread locking adhesive.
- The tightening torque of the M6 bolts should be at least 2.0 Nm. For the M3 bolts the tightening torque should be 1.2Nm. Tighten the bolts in star sequence and with the same torque.
- Always be careful when mounting the cable. The pins are easy to be damaged without proper key alignment.
- The tool mounted should always touch only the mounting surface/side of the tool. Take care that no other side or part is being in contact with the tool.

NOTICE

NOTICE: The sensor will not provide correct data if the tool is not properly mounted on the tool side as indicated in the sensor specifications sheet.

6 Operation

6.1 Sensor environment

To ensure proper operation, the user should take care that the environmental condition should match or be exceeded by the IP67 rating.

CAUTION: Calibration matrix is unique for each sensor.

6.2 Sensor operation overview

Both Serial and EtherCAT Sensor models have two different states. The **CONFIG** state where the user can change the parameters of the sensor according to paragraph. The **RUN** state where the sensor is providing the force torque measurements and the IMU measurements (IMU only in EtherCAT). In the **CONFIG** mode the user can adjust the operational parameters and save them in the internal memory optionally. All the parameters will become active only after a transition from **CONFIG** to **RUN** mode.
All serial devices are featuring LEDs that they indicate the sensors operational state and status. The green LED reflects the operational state and the red LED reflects the status in **RUN** mode. The status flags can be found in Table 13.

Table 4 - Serial sensor LEDs

	States	
LEDs	**CONFIG**	**RUN**
Red	Blinking 1s	Blinking in the publishing frequency of the sensor
Green	off	On (at least one status flag is set)

6.3 Sensor configuration

The Sensor is offering a range of adjustability to ensure perfect fit to a variety of applications. The user can change the output rate and the filtering of the data to match the response frequency required for the application, this way the sensor is providing data with the minimum noise possible for the given required output rate. Paragraph Force torque filtering and output rate below is explaining in detail the different adjustability options for the Force Torque data. The EtherCAT version of Bota Systems sensors also offers IMU measurements, specifically linear accelerations and angular velocities. For these IMU measurements the user can also select from a range of cut-off frequencies versus noise level, more information can be found in the chapter IMU filtering and output rate below.

6.3.1 Force torque filtering and output rate

The Bota Systems force/torque sensors are featuring an advance hardware level filtering combination. This includes a SINC filter, a chop filter and FIR filter for extra smoothing when high sampling frequency is not necessary. The user has to keep in mind that the filter parameters define the amount of noise and the output rate in an inverse fashion. Finally the sensor is offering a special functionality when the FIR filter is enabled, called FAST. FAST is automatic switching of the FIR filter that will allow you to have noise level equivalent to having the FIR enabled and response time equivalent to having the FIR switched off. If FIR filter is disabled FAST status is ignored.

NOTICE: The Chop filter is recommended to be always deactivated.

NOTICE: The FAST filter is not recommended when the sensor is used in real-time control or dynamic measurements.

In Table 5 the user can find some typical cases of filter parameters with the resulting Noise, Output rate, Cut-off frequency. In Table 6 the user can find a more complete selection of parameters. Use this table to decide the filtering parameters given the required response frequency and noise level. If noise level is not absolutely critical it is recommended to disable the FAST filter. Also the Chop filter is recommended to be always deactivated for real time applications.

Table 5 - Typical filter parameters and their results

Sinc length (dec)	**CHOP**	**FAST**	**FIR Disabled**	**Output rate**	**Cut-off frequency**	**Noise 6σ (Fxy, Fz, Txy, Tz)** [2]
64	0	1/0	1	800 Hz	1255 Hz	0.6N, 0.5N, 0.01Nm , 0.005Nm
64	0	0	0	800 Hz	189 Hz	0.3N, 0.25N, 0.005Nm , 0.002Nm
64	0	1	0	800 Hz	1255 Hz	0.5N, 0.4N, 0.008Nm , 0.004Nm
512	0	1/0	1	100 Hz	157 Hz	0.08N, 0.07N, 0.001Nm, 0.0003Nm
512	0	0	0	100 Hz	314 Hz	0.04N, 0.03N, 0.0005Nm, 0.00015Nm
512	0	1	0	100 Hz	157 Hz	0.06N, 0.05N, 0.0008Nm, 0.0002Nm

Table 6 - list of output rates

Sinc length (dec)	**Output Rate**	**Cut-off frequency with FIR disabled, or FIR enable with FAST**	**Cut-off frequency with FIR enabled**
51	1000 Hz	1674 Hz	252 Hz
64	1000 Hz	1255 Hz	189 Hz
128	1000 Hz	628 Hz	94.5Hz
205	250 Hz	393 Hz	59.5 Hz
256	200 Hz	314 Hz	47.5 Hz
512	100 Hz	157 Hz	23.5 Hz

Example

Sinc Length: 64
CHOP: disabled
FAST: disabled
FIR: disabled

Will result in an Output rate of 800 Hz with cut-off frequency of 1255 Hz.

6.3.2 IMU filtering and output rate

The same way as with Force Torque filtering, sensors equipped with IMU offer similar adjustability for the acceleration and angular velocity data. For this data the sensors provide only predefined filtering profiles for faster and easier setup. The profile numbers correspond to number that the user should set to the sensor using the Interface.

Table 7 - Filter options for angular velocities

Angular velocities			
Filter Profile	Bandwidth(hz)	Delay(ms)	Fs (kHz)
3	184	2.9	1
4	92	3.9	1
5	41	5.9	1
6	20	9.9	1
7	10	17.85	1
8	5	33.48	1
9	3600	0.17	8

[2] σ stands for standard deviation. It is calculated by measuring 10 sec of measurements when the sensor is not loaded. 6σ means that the noise levels will be exceed nominal peak to peak values 0.27% times.

Table 8 - Filter options for accelerations

Accelerations				
Filter Profile	Bandwidth(hz)	Delay(ms)	Fs (kHz)	Noise Density (ug/rtHz)
1	460	1.94	1	250
2	184	5.80	1	250
3	92	7.80	1	250
4	41	11.80	1	250
5	20	19.80	1	250
6	10	35.70	1	250
7	5	66.96	1	250
8	460	1.94	1	250

6.3.3 Force torque bias

Force torque sensors have biases or offsets. They change when the sensor is mounted on the installation. This means that the force torque measurements will have non zero values despite the fact that the sensor is not externally loaded. Pre-tensioning of screws, gravitational and inertial forces are the usual cause. The last appear during dynamic movements of a robot and can be estimated if the accelerations, angular rotations and the matrix of inertia of the object mounted on the force torque sensor are known. The offsets due to gravitational forces can be estimated if the orientation of the sensor is known. Pre-tensioning from masses when the robot or the sensor is not moving can be compensated by just subtracting the actual sensor values. In this phase the user should calculate a correction factor for the forces and torques using the following formula where F is the force vector.

$$F_{expected} = F_{measured} - F_{bias}$$

After calculating all three force and torque biases the user can send them to sensor and the sensor will automatically subtract them from the measurements[3]. You can find in the following paragraphs how to achieve this.

6.4 Serial interface

NOTICE: IGNORE the following section, if you are provided with a ready to use software. The software provided usually, takes care of the data parsing and publishes force torque data in N and Nm.

This chapter is describing the interface with the serial devices. The user can find here the hardware specifications for the serial communication, the serial port interface and parameters and a complete list of all the commands available for the device

Table 9 - Serial communication parameters

Baud rate (selectable)	**460800bps (default)**
Data bits	8
Stop bits	1
Parity	none
Flow control	none

6.4.1 Serial configuration interface

As shown in Table 10 the user can send the "C" character through the serial port to request the sensor to switch to CONFIG mode. After the sensor confirms config mode, then the user can send the required parameters using the syntax from Table 10. All the commands are ASCII encoded, to allow the user to operate the sensor by using a

[3] The automatic substraction applies for the pre-tension forces

terminal software like “Realterm” for Windows or “Cutecom” for Linux. User has to pay attention that all the commands are case sensitive so cannot be used interchangeably with the other case.

Table 10 - List of serial commands

Cmd (ascii)	Cmd hex	State permission	Syntax in ascii (no new line required)	Description
#	0x23	All states	#	This will trigger a hardware reset of the sensor
I	0x49	All states	I	This will trigger a software reset of the sensor bringing it to a known state
C	0x43	All states	C	Will change the sensor to Config mode
R	0x52	All states	R	Will change the sensor to RUN mode
c		Config	c,<temp_comp>,<calib>,<format>,<baud>	Communication setup, baudrate see , data format, see Table 11
f		Config	f,<sinc_len>,<chop>,<fast>,<fir_disable>	Setup filters according to Table 5 & Table 6
b		Config	b, <Fx>,<Fy>,<Fz>,<Tx>,<Ty>,<Tz>	Setup bias according to Force Torque Bias
s		Config	s	save
l		Config	l	load
w		Config	w	Prints all the user configurable parameters

Table 11 –Communication setup command

Temp comp		**Calibration**		**Data format**		**baudrate**	
ascii cmd	description	ascii cmd	description	ascii cmd	description	ascii cmd	description
0	disabled	0	Disabled	0	binary	0	9600
1	enabled	1	enabled	1	CSV	1	57600
						2	115200
						3	230400
						4	460800

6.4.2 Serial data packet

The Sensor will transmit the data frame described in Table 12, when it is in **RUN** mode and configured with binary data format (see Table 11), the data will be transmitted in the output rate frequency with the selected baudrate. The sensor supports also a CSV ascii output for user readable outputs, the user should keep in mind that the CSV output is only suitable for low output rates, due to bandwidth limitations. If the user tries to use the CSV mode in high output rate the sensor will report with the “Insufficient communication bandwidth” status flag (see Table 13).

Table 12 - data frame payload

Section	Variable	Description	Data type	Size in bytes
header	identifier	packet identifier 0xAA (first byte of packet)	Uint8	1
Data	Status	Refer to table below	16bit bitmask	2
	Force X	X component of force in N	Float	4
	Force Y	Y component of force in N	Float	4
	Force Z	Z component of force in N	Float	4
	Torque X	X component of torque in Nm	Float	4
	Torque Y	Y component of torque in Nm	Float	4
	Torque Z	Z component of torque in Nm	Float	4
	Timestamp	In microseconds	Uint32	4
	Temperature	Temperature of sensor in degrees Celsius	Float	4
checksum	CRC16 X25	CRC16 X25 checksum of the above data section	Uint16	2

Table 13 - Status flags

Bit index	Severity	Description	Solution
0	Warning	Insufficient communication bandwidth	Decrease the output rate or Increase Baudrate or switch to Binary data format
1	Warning	Sensor range exceeded	Reduce sensor loading
2	Error	Sensor measurements invalid, Damage risk	Reduce sensor loading immediately if error persist the sensor is damaged
3	Info	Not calibrated data is transmitted	Enable calibration see Table 11

NOTICE

NOTICE: Calibration matrix is unique for each sensor.

NOTICE

NOTICE: If you are provided with a USB version. The FTDI latency timer should be changed from 16ms (Driver Default) to 1ms to work properly.

6.4.3 Sample code for receiving data (C)

```c
// C library headers
#include <stdio.h>
#include <string.h>

// Linux headers
#include <fcntl.h> // Contains file controls like O_RDWR
#include <errno.h> // Error integer and strerror() function
#include <termios.h> // Contains POSIX terminal control definitions
#include <unistd.h> // write(), read(), close()

struct DataStatus  __attribute__((__packed__))
 {
    uint16_t app_took_too_long:1;
    uint16_t overrange:1;
    uint16_t invalid_measurements:1;
    uint16_t raw_measurements:1;
    uint16_t:12; //reserved
 };

struct AppOutput __attribute__((__packed__))
{
    DataStatus status;
    float forces[6];
    uint32_t timestamp;
    float temperature;
};

struct __attribute__((__packed__)) RxFrame
  {
    uint8_t header;
    AppOutput data;
    uint16_t crc16_ccitt;
}frame;

const uint8_t frame_header = 0xAA;

Int main()
{
  /* Open the serial port. Change device path as needed.
   * Currently set to an standard FTDI USB-UART converter
   */
  int serial_port = open("/dev/ttyUSB0", O_RDWR);

  // Create new termios struc, we call it 'tty' for convention
  struct termios tty;
  memset(&tty, 0, sizeof tty);

  // Read in existing settings, and handle any error
  if(tcgetattr(serial_port, &tty) != 0) {
      printf("Error %i from tcgetattr: %s\n", errno, strerror(errno));
  }
```

```
  tty.c_cflag &= ~PARENB; // Disable parity
  tty.c_cflag &= ~CSTOPB; // 1 stop bit
  tty.c_cflag |= CS8; // 8 bits per byte
  tty.c_cflag &= ~CRTSCTS; // Disable RTS/CTS hardware flow control
  tty.c_cflag |= CREAD | CLOCAL; // Turn on READ & ignore ctrl lines (CLOCAL = 1)

  tty.c_lflag &= ~ICANON; // Disable canonical mode
  tty.c_lflag &= ~ECHO; // Disable echo
  tty.c_lflag &= ~ECHOE; // Disable erasure
  tty.c_lflag &= ~ECHONL; // Disable new-line echo
  tty.c_lflag &= ~ISIG; // Disable interpretation of INTR, QUIT and SUSP
  tty.c_iflag &= ~(IXON | IXOFF | IXANY); // Turn off s/w flow ctrl
  tty.c_iflag &= ~(IGNBRK|BRKINT|PARMRK|ISTRIP|INLCR|IGNCR|ICRNL); // Disable any special
handling of received bytes

  tty.c_oflag &= ~OPOST; // Prevent special interpretation of output bytes (e.g. newline c
hars)
  tty.c_oflag &= ~ONLCR; // Prevent conversion of newline to carriage return/line feed

  tty.c_cc[VTIME] = 10;    // Wait for up to 1s (10 deciseconds), returning as soon as any
 data is received.
  tty.c_cc[VMIN] = 0;

  // Set in/out baud rate to be 460800
  cfsetispeed(&tty, B460800);
  cfsetospeed(&tty, B460800);

  // Save tty settings, also checking for error
  if (tcsetattr(serial_port, TCSANOW, &tty) != 0) {
      printf("Error %i from tcsetattr: %s\n", errno, strerror(errno));
  }

  while (1) {
    /* read the next available byte and check if is the header
     * make sure to unget it after. emulating peek. Using static
     * variable for the sync flag will preserve status of sync
     * between calls */
    bool frameSync_ = false;
    while (!frameSync_) {
      uint8_t possible_header;
      /* read bytes 1 by 1 to find the header */
      serial_port.read((char*)&possible_header, sizeof(possible_header));
      if (possible_header == frame_header) {
        /* read the remaining frame to make check also CRC */
        serial_port.read((char*)&frame.bytes[sizeof(frame.header)], sizeof(frame) - sizeof
(frame.header));
        if (frame.crc16_ccitt == calcCrc16X25(frame.data.bytes, sizeof(frame.data))) {
          printf("Frame synced with 0x%X header", frameHeader);
          frameSync_ = true;
        }
        else {
          /* if there is a frame that included the header 0xAA in
         * a fixed position. Could be the above checking mechanism
         * will get stuck because will find the wrong value as header
```

```
       * then will remove from the buffer n bytes where n the size
       * of the frame and then will find again exactly the same
       * situation the wrong header. So we read on extra byte to make
       * sure next time will start from the position that is size of frame
       * plus 1. It works */
          char dummy;
          serial_port.read((char*)&dummy, sizeof(dummy));
          printf("Skipping incomplete frame");
        }
      }
    }
    while(frameSync_) {
      /* Read the sensor measurements frame assuming that is alligned with the RX buffer *
/
      serial_port.read((char*)frame.bytes, sizeof(frame));

      /* Check if the frame is still alligned, otherwise exit */
      if (frame.header != frameHeader) {
        frameSync_ = false;
        break;
      }
      // Read and check CRC 16-bit
      uint16_t crc_received = frame.crc16_ccitt;
      uint16_t crc_computed = calcCrc16X25(frame.data.bytes, sizeof(frame.data));
      if (crc_received != crc_computed) {
        printf("CRC missmatch received: 0x%04x calculated: 0x%04x", crc_received, crc_comp
uted);
        break; //skip this measurements
      }
      // print measurements
      if (frame.data.status.app_took_too_long) {
        printf("Warning force sensor is skipping measurements, Increase sinc length");
      }
      if (frame.data.status.overrange) {
        printf("Warning measuring range exceeded");
      }
      if (frame.data.status.invalid_measurements) {
        printf("Warning measurements are invalid, Permanent damage may occur");
      }
      if (frame.data.status.raw_measurements) {
        printf("Warning raw force torque measurements from gages");
      }
      printf("Fx: %f",frame.data.forces[0]);
      printf("Fy: %f",frame.data.forces[1]);
      printf("Fz: %f",frame.data.forces[2]);
      printf("Tx: %f",frame.data.forces[3]);
      printf("Ty: %f",frame.data.forces[4]);
      printf("Tz: %f",frame.data.forces[5]);
      printf("temperature: %f", frame.data.temperature);
      printf("timestamp: %f", frame.data.timestamp);
    } // while synced
  } // while app run
  close(serial_port);
}
```

6.4.4 Sample code for CRC16 X25 calculation (C)

```
//CRC16 checksum calculation
#define lo8(x) ((x)&0xFF)
#define hi8(x) (((x)>>8)&0xFF)
inline uint16_t calcCrc16_x25(uint8_t *data, int len)
{
  uint16_t crc = 0xFFFF;
  while(len--)
    crc = crc_ccitt_update(crc, *data++);
  return ~crc;
}

uint16_t crc_ccitt_update (uint16_t crc, uint8_t data)
```

7 EtherCAT interface

The EtherCAT interface is used to give the users the following capabilities:

- Read the Product Number, Serial Number, etc.
- Read Force Torque Data
- Read IMU data
- Enable/Disable filters and calibration matrices.
- Change the sampling rate

7.1 PDO interface

The PDO interface exchanges data in real time with the F/T sensor.

- TxPDO Map / Output Data
  - Object 0x6000: Sensor Data
- RxPDO Map / Input Data
  - Object 0x7000: Digital output

7.2 SDO interface

The SDO data configures the sensor and reads the manufacturing and calibration data. This section lists dictionary objects specific to the EtherCAT F/T sensor application; it does not list objects that are a required part of the EtherCAT standard. These dictionary objects can also be found in the ESI file on the botasys.com

Table 14 - EtherCAT Object dictionary

Object		Name	Type	Description	Default	Options
0x2000	0	Calibration	RO	Used to read the device specific calibration values. Gains, offsets etc.		
0x6000	0	Sensor Data	RO	Used to read the Force Torque, IMU etc. data from the sensor		
0x7000	0	Digital Output	RO			
0x8000	0	Force Torque Offset	RO	Write the Force Torque Offset values		
0x8001	0	Acceleration Offset	RO	Read the Linear Acceleration Offset values		
0x8002	0	Angular Rate Offset	RO	Read the Angular Rate Offset values		
0x8003	0	Force Torque Range	RO	Read the Force Torque Sensor range		
0x8004	0	Acceleration Range	RW	Read and Write the Linear Acceleration range		
0x8005	0	Angular Rate Range	RW	Read and Write the Angular Rate range		
0x8006	0	Force Torque Filter	RO	Read and Write the Force Torque Sensor filtering		
	01	Sinc Length	RW	Sets the Sinc filtering	0x0040 (800sps)	Observe object: 0x8011 after setting this object to get the current sampling See Table 6
	02	FIR disable	RW	Disables FIR filter when bit is HIGH	0x01	
	03	FAST enable	RW	Enables spike detection filter when bit is HIGH	0x00	
	04	CHOP enable	RW	Enable CHOP filtering when bit is HIGH	0x00	
0x8007	0	Acceleration Filter	RW	Read and Write the Linear Acceleration filtering		
0x8008	0	Angular Rate Filter	RW	Read and Write the Angular Rate filtering		
0x8010	0	Device configuration	RO	To activate/deactivate the device features such as calibration matrix, temperature compensation, IMU etc		
	01	Force Torque Cal. Matrix Active	RW	Activate Calibration Matrix on device	0	0 : Raw data 1 : N/Nm data
	02	Temperature Compensation Active	RW	Enable temperature Compensation on device	0	0 : Not activated 1 : Activated
	03	IMU Active	RW	Activate IMU Readings	0	0: Not activated 1: Activated
	04	Coordinate System Cong. Active	RW	reserved	0	
	05	Inertia and Gravity Comp. Active	RW	reserved	0	
	06	Orientation Estimation Active	RW	reserved	0	
0x8011	0	Sampling Rate	RO	Read the Sampling rate of the sensor with the current configuration	0x0320	Observe this object after changing object 0x8006

0x8030	0	Control	RO	Control codes for reading/writing on the device		
	01	Command	RW		0	01: save parameters after power-down 02: reserved 03: reserved 04: reserved
	02	status	RO		0	Error code if command fails

8 Maintenance

8.1 Inspection

With applications that frequently move the system's cabling and dismount the connector, you the cable jacket and connector pins should be checked for signs of wear. The Bota Systems sensors are IP67 rated. Debris and dust should be kept from accumulating on or in the sensor if the Seals that provide dust and water protection are broken.

8.2 Calibrating

Periodic calibration of the sensor and its electronics is required to maintain traceability to international standards.
Applicable ISO-9000-type standards for calibration should be followed. It is recommended annual re-calibrating, especially when the sensor is used in cycle loads applications. The best practice would be to send it back to us for recalibration.

9 Troubleshooting

9.1 Force/Torque reading errors

Inaccurate data from the transducer's strain gages can cause errors in force/torque readings. These errors can result in problems with transducer biasing and accuracy. Listed below there are the basic problems of inaccurate data.

Table 15 - Troubleshooting

Error	Explanation
Noise	Peaks in data readings, with the sensor untouched and at stable environmental conditions, greater than 0.2% of full scale counts is abnormal. Noise can be caused by mechanical vibrations and electrical disturbances and a poor ground is most of the times an issue to robotic systems. Make sure that the DC supply voltage for the sensor is a regulated USB standard DC voltage. A component failure within the system can also be the issue.
Drift	After a load is removed or applied, the raw gage reading does not stabilize but continues to increase or decrease. A shift in the raw gage reading is observed more easily in the resolved data mode using the bias command. Some drift from a change in temperature or mechanical coupling is normal. Mechanical coupling occurs when the sensor's degrees of freedom are restricted. The upper part is in contact with the bottom part. That can be a tape mounted between them, a wrong mounting in the system etc. See Installation section for the last one.
Hysteresis	When the sensor is loaded and then unloaded, gauge readings do not return quickly and completely to their original readings. Hysteresis is caused by mechanical coupling or internal failure.
Sensor not streaming data	The sensor should be correctly installed and powered. Refer to Installation section for more information. The sensor is not in RUN mode (see Sensor Operation Overview)

The sensor has an inherent low temperature drift and a temperature drift from external thermal sources. The two of them are combined and can be seen as an overall temperature drift. The first which is from its inside electronics power consumption can be calibrated if let the sensor work for an amount of time (5minutes is adequate) in stabilized room temperature. The second has to do with external temperature changes. In each case, the sensor is subjected to offset phenomena and need to be calibrated (or else biased) systematically. To do this, the sensor need to be in a reference condition of temperature and force as soon as the user application demands it.

10 Specifications

10.1 Serial devices

The specifications for the Bota Systems USB or RS485/RS422 FT Sensors are covered in the following tables. Individual products have their own specification sheet.

Table 16 - Serial devices specifications

Other characteristics		Unit	Min	Typical	Max	Comments
Power						
	Voltage	V		5		
	Current	mA		150		
	Power consumption	W		0.8		
Bandwidth						
		Hz	50	310	1000	
Dimensions						
	Height	mm		spec sheet		
	Diameter	mm		spec sheet		
Operation at environmental challenging temperatures			-10°C		60°C	

10.2 EtherCAT devices

The specifications for the Bota Systems EtherCAT FT Sensors are covered in the following tables. Individual products have their own specification sheet

Table 17 - EtherCAT devices specifications

IMU				
	Component		Sensing range	Bandwidth 1 to 4 kHz
	Accelerometer		±2g, 4g, 8g, 16g,	
	Gyroscope		±250°/sec, ±500°/sec, ±1000°/sec, ±2000°/sec	
	Magnetometer		3-axis (not used)	

Other characteristics		Unit	Min	Typical	Max	Comments
Power						
	Voltage	V	9		70	
	Current	mA		250		
	Power consumption	W		1.5		
Bandwidth						
		Hz	50	310	1000	
Dimensions						
	Height	mm		spec sheet		
	Diameter	mm		spec sheet		
Operation at environmental challenging temperatures:			0°C		50°C	

11 Complex loadings

The complex loadings should be in consideration for two reasons. One is the overloading and the other is the saturation of the ADCs.

The sensor's overload values above are calculated only for single axis overloading. For complex loading care should be taken in order to avoid overload the sensor. With more torque than specified. Induced torque from translated force to the calibration point from the application should be in consideration. Hereby complex loading graphs for the sensor should be shown.

The ADCs are also saturated when complex loading is applied to the sensor and thus the rated forces cannot be measured. The sensor is saturated for the given load and all forces and torques read by the sensor will be invalid until the load is changed so that the sensor is no longer saturated.