PapillArray Tactile Sensor Communications Protocol (v2.0)

Specification

Document #: PTSCOM_2.0_SPEC_NOV21

November, 2021

Foreword

Information contained in this document is the property of Contactile Pty Ltd. and shall not be reproduced in whole or in part without prior written approval of Contactile Pty Ltd. The information herein is subject to change without notice and should not be construed as a commitment on Contactile Pty Ltd. This manual is periodically revised to reflect and incorporate changes made to the PapillArray Tactile Sensor Development Kit.

Contactile Pty Ltd assumes no responsibility for any errors or omissions in this document. Users' critical evaluation is welcome to assist in the preparation of future documentation.

Conditions of Sale

Contactile's conditions of sale apply to all products sold by Contactile to the Distributor under this Agreement. The conditions of sale that apply are provided on the USB flash drive shipped with the product in the folder 'LEGAL' in the root directory.

End User Licence Agreement

Contactile's end user license agreement applies to all software and algorithms included with the products sold by Contactile. The end user license agreement that applies is provided on the USB flash drive shipped with the product in the folder 'LEGAL' in the root directory.

Compliance

The devices are sold as is.

The devices are specifically designed solely for the purposes of research and development only made available on a business-to-business basis.

The devices are not for resale.

Table of Contents

1 Introduction

The PapillArray Tactile Sensor Development Kit (v2.0) is a system of (up to) two PapillArray Tactile Sensor arrays and a Controller. Each PapillArray Tactile Sensor array can measure 3D displacement, 3D force, and vibration on each sensing element, as well as global 3D force, global 3D torque, the onset of slip, and friction. The Controller supplies power for (up to) two sensors and coordinates the simultaneous data acquisition from up to two PapillArray Tactile Sensors; i.e., coordinates sampling of the 9 pillars if one sensor is connected to the Controller, 18 pillars if two sensors are connected to the Controller. The Development Kit is shipped with visualisation software and (optional) C++ libraries for Windows and Linux environments and a ROS node for developing software control algorithms using the sensor signals.

The main components of the PapillArray Tactile Sensor Development Kit (v2.0) are shown in Figure 1.1, connected to a laptop running the visualisation software.

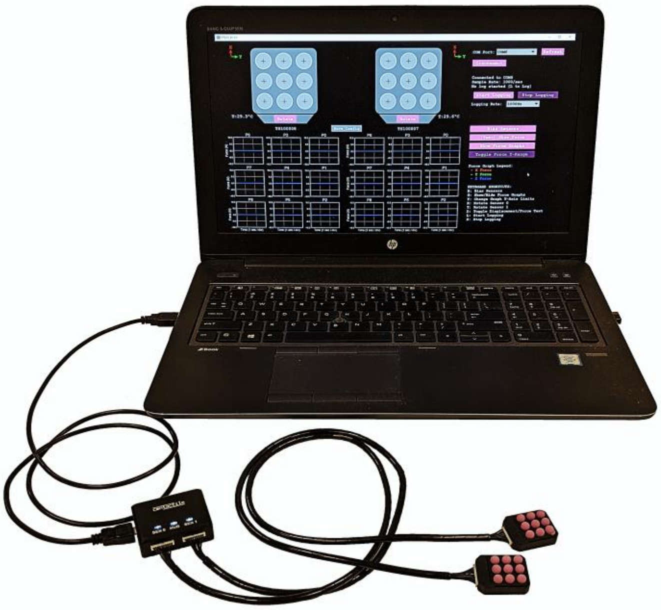


Figure 1.1 – The PapillArray Tactile Sensor Development Kit (v2.0). Laptop not included.

This document is a specification for the Communications Protocol by which data is transmitted from the PTSDK Controller and commands are sent to the PTSDK Controller.

2 Safety

2.1 General

The customer should verify that the maximum loads and moments expected during operation fall within the sensing range of the sensor as outside this range, sensor reading accuracy is not guaranteed (refer to Document #PTS_2.0_SPEC_JUL21). Particular attention should be paid to dynamic loads caused by robot acceleration and deceleration if the sensors are mounted on robotic equipment. These forces can be many multiples of the value of static forces in high acceleration or deceleration situations.

2.2 Explanation of warnings

The warnings included here are specific to the product(s) covered by this manual. It is expected that the user heed all warnings from the manufacturers of other components used in the installation.

⚠ Danger indicates that a situation could result in potentially serious injury or damage to equipment.

⚠ Caution indicates that a situation could result in damage to the product and/or the other system components.

2.3 Precautions

⚠ **DANGER**: Do not attempt to disassemble the sensor. This could damage the sensor and will invalidate the calibration.

⚠ **DANGER**: Do not attempt to drill, tap, machine, or otherwise modify the sensor casing. This could damage the sensor and will void invalidated the calibration.

⚠ **DANGER**: Do not use the sensor on abrasive surfaces or surfaces with sharp points/edges. This could damage the silicone surface of the sensor.

⚠ **CAUTION**: Sensors may exhibit a small offset in readings when exposed to intense light sources.

⚠ **CAUTION**: Exceptionally strong and changing electromagnetic fields, such as those produced by magnetic resonance imaging (MRI) machines, constitute a possible source of interference with the operation of the sensor and Controller.

⚠ **CAUTION**: Temperature variations can cause drift in sensor readings. Some temperature compensation is performed. However, bias removal in software prior to operation is necessary, and it is recommended that biasing is performed each time the sensor is known to be unloaded.

3 Data packet

The data streamed through the USB is fully resolved and thus users can read and parse the data using any software programming language they desire.

3.1 Reading the data packet

PTSDK Controller is connected to a computer via a serial connection emulated on the computer's USB port. Data is streamed through the serial port in binary format with 8-bit byte size.

Each sample of data is organised as a data packet beginning with 4 prescribed start bytes (0x55, 0x66, 0x77 and 0x88) and ending with 4 prescribed end bytes (0xAA, 0xBB, 0xCC and 0xDD). Immediately preceding the end bytes are 2 checksum bytes which contain the truncated sum of each byte in the data packet (excluding the start bytes, end bytes and checksum bytes) so that the data integrity can be confirmed.

To read a sample from the COM port:

1. Read one byte at a time until four consecutive bytes matching the start bytes are found.
2. Read one byte at a time and store these data to a buffer until four consecutive bytes matching the end bytes are found

To validate the data integrity:

1. Calculate the sum of the data in the buffer excluding the final two checksum bytes
2. Compare the calculated sum to the checksum bytes

3.2 Parsing the data packet

After reading the data packet into a buffer, the user program should parse the data to extract the data required for the user program.

The data packet is described in Figure **3.1**, Figure **3.2** and Figure **3.3**.

Figure 3.1 Data packet structure

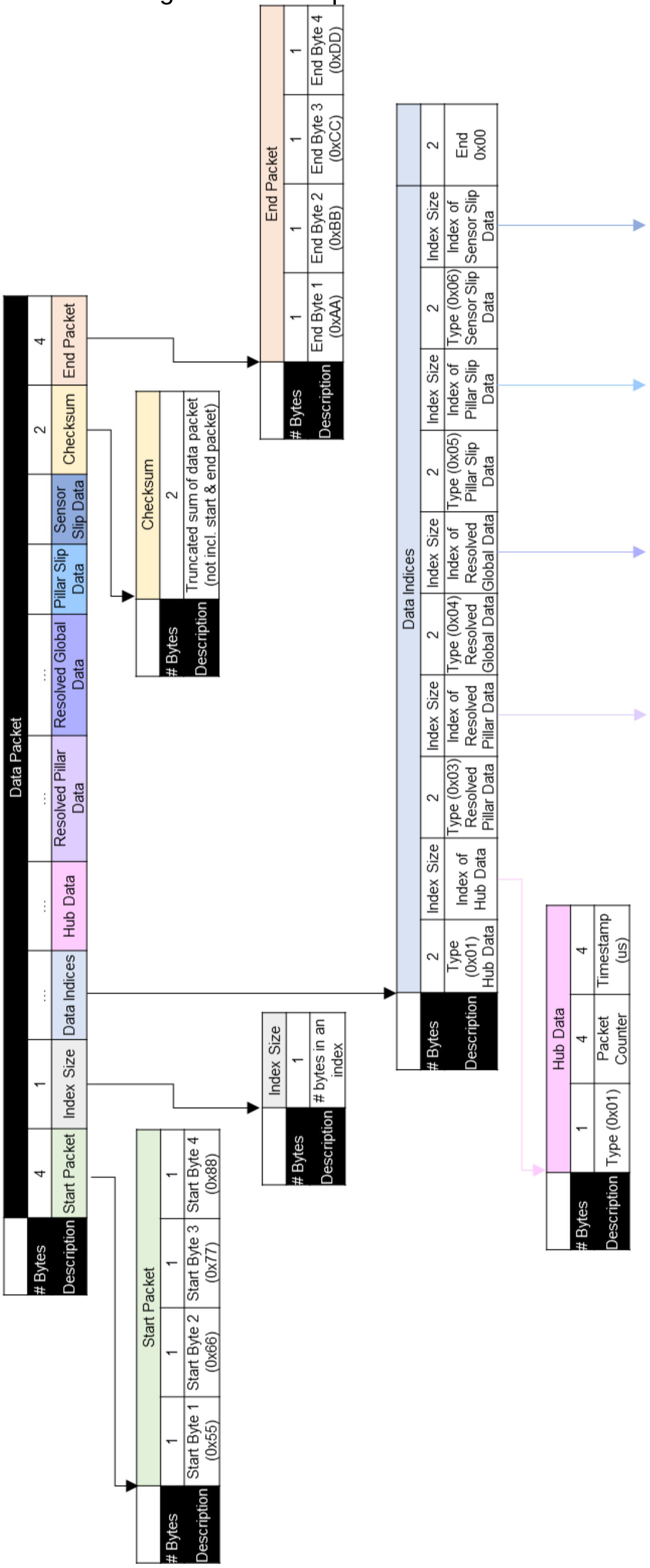
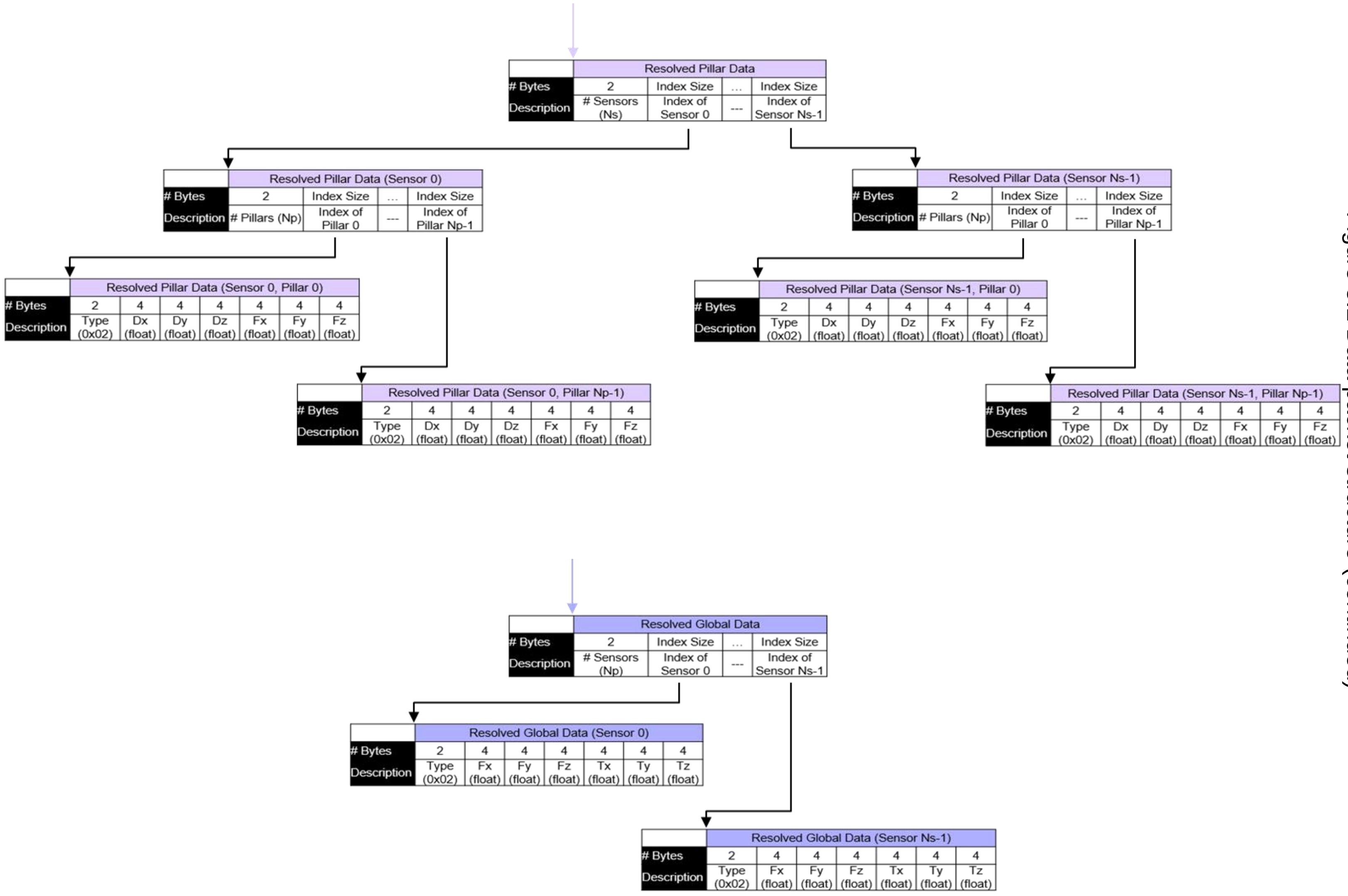


Figure 3.2 Data packet structure (continued)



	Resolved Pillar Data			
# Bytes	2	Index Size	...	Index Size
Description	# Sensors (Ns)	Index of Sensor 0	---	Index of Sensor Ns-1

	Resolved Pillar Data (Sensor 0)			
# Bytes	2	Index Size	...	Index Size
Description	# Pillars (Np)	Index of Pillar 0	---	Index of Pillar Np-1

	Resolved Pillar Data (Sensor Ns-1)			
# Bytes	2	Index Size	...	Index Size
Description	# Pillars (Np)	Index of Pillar 0	---	Index of Pillar Np-1

	Resolved Pillar Data (Sensor 0, Pillar 0)						
# Bytes	2	4	4	4	4	4	4
Description	Type (0x02)	Dx (float)	Dy (float)	Dz (float)	Fx (float)	Fy (float)	Fz (float)

	Resolved Pillar Data (Sensor Ns-1, Pillar 0)						
# Bytes	2	4	4	4	4	4	4
Description	Type (0x02)	Dx (float)	Dy (float)	Dz (float)	Fx (float)	Fy (float)	Fz (float)

	Resolved Pillar Data (Sensor 0, Pillar Np-1)						
# Bytes	2	4	4	4	4	4	4
Description	Type (0x02)	Dx (float)	Dy (float)	Dz (float)	Fx (float)	Fy (float)	Fz (float)

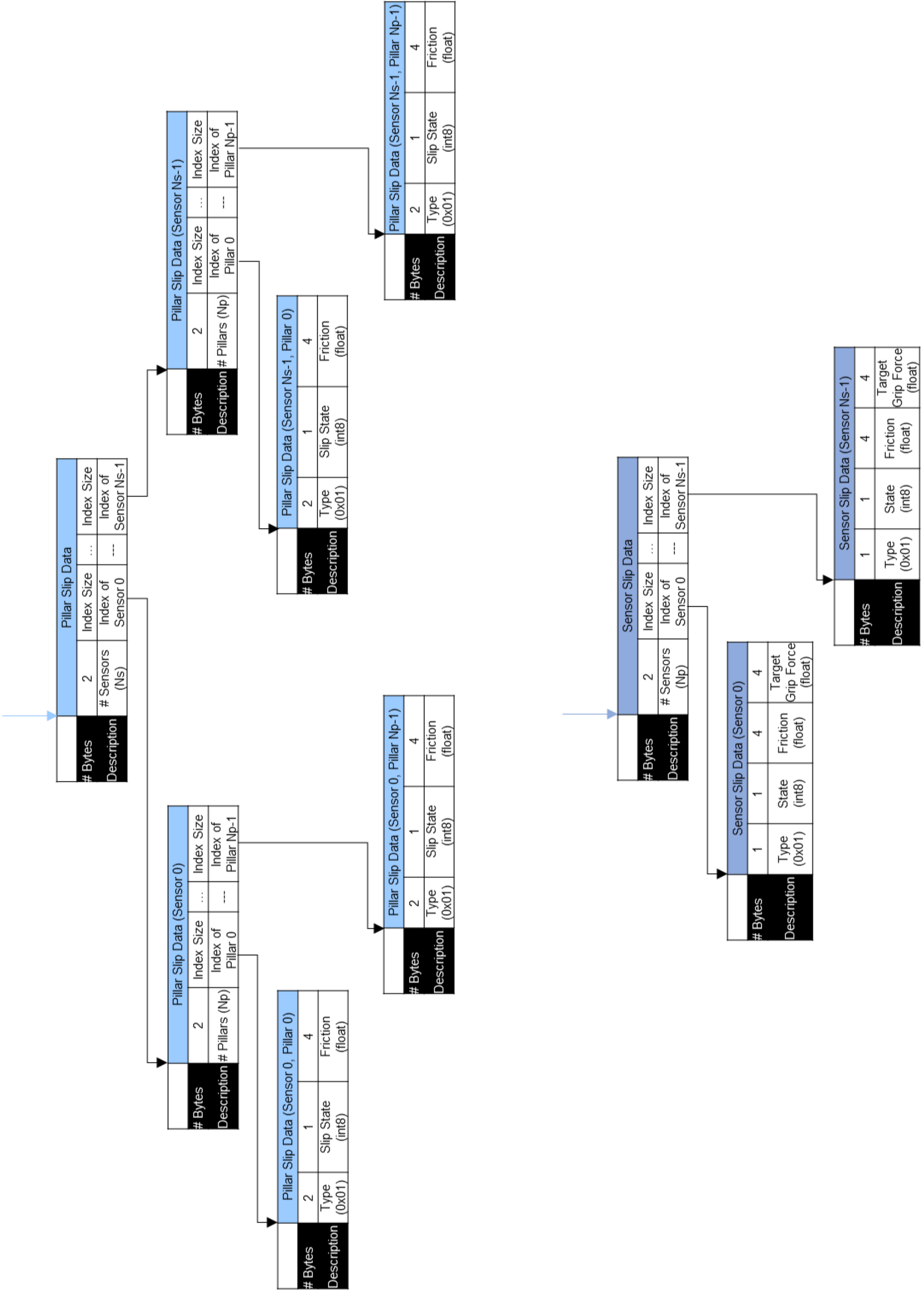
	Resolved Pillar Data (Sensor Ns-1, Pillar Np-1)						
# Bytes	2	4	4	4	4	4	4
Description	Type (0x02)	Dx (float)	Dy (float)	Dz (float)	Fx (float)	Fy (float)	Fz (float)

	Resolved Global Data			
# Bytes	2	Index Size	...	Index Size
Description	# Sensors (Np)	Index of Sensor 0	---	Index of Sensor Ns-1

	Resolved Global Data (Sensor 0)						
# Bytes	2	4	4	4	4	4	4
Description	Type (0x02)	Fx (float)	Fy (float)	Fz (float)	Tx (float)	Ty (float)	Tz (float)

	Resolved Global Data (Sensor Ns-1)						
# Bytes	2	4	4	4	4	4	4
Description	Type (0x02)	Fx (float)	Fy (float)	Fz (float)	Tx (float)	Ty (float)	Tz (float)

Figure 3.3 Data packet structure (continued)



4 Commands

The PTSDK Controller accepts commands via the COM port. The commands are sent as ASCII characters. These are listed in Table 4.1.

Table 4.1 – Commands accepted by the PTSDK Controller

ASCII Characters		Command Description
z\n	Bias sensors	Sends a bias request to the Controller. A bias should be performed after connecting to the serial port and starting to stream data with the sensor unloaded. A bias should be performed each time the sensor is known to be unloaded. A bias operation can take up to 2 s. Ensure that the sensor remains unloaded throughout this time.
S\n	Start slip detection	Starts the slip detection algorithms on the Controller. This should be called after a number of pillars of the sensor are already in contact, before tangential loading of the sensor.
s\n	Stop slip detection	Stops and resets the slip detection algorithms on the Controller.
f100\n	Set sampling rate to 100 Hz	
f250\n	Set sampling rate to 250 Hz	
f500\n	Set sampling rate to 500 Hz	
f1000\n	Set sampling rate to 1000 Hz	