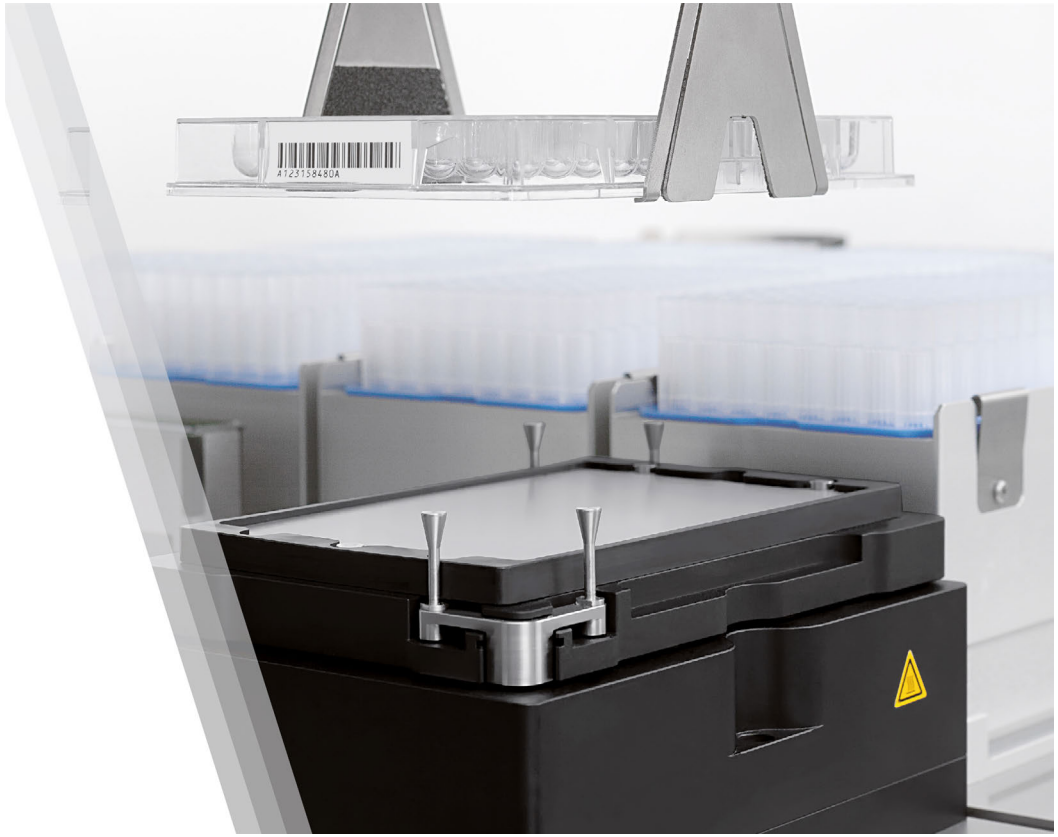
INTEGRATION MANUAL



QINSTRUMENTS
A BICO COMPANY

Version: 010.8

APPLICABLE DEVICES AND FIRMWARE

Device	Firmware	Abbreviation used in this document
BioShake 3000 Series | BioShake 5000 Series | BioShake D-30 Series | HeatPlate	≥ 1.6	BS
BioShake Q1 | BioShake Q1 3mm | BioShake Q2	≥ 1.0	TC*
ColdPlate | ColdPlate slim	≥ 2.4	TC*

* The abbreviation TC stands for Thermo Control. This is related to the ability of all the devices in the two rows (BioShake Q1 ... ColdPlate slim) to heat and cool. All these devices therefore have an adapted set of commands due to the underlying firmware.

TABLE OF CONTENTS

Applicable Devices and Firmware **2**
Table of Contents **3**
1. Introduction **4**
2. Communication Interface **5**
2.1 Hardware 5
2.2 Software 5
3. Software Interface **6**
3.1 General information 6
3.2 Command List 8
3.3 Command details 9
3.4 Error Codes 16
3.5 Example 17
3.6 Extended programming information 18
4. Hardware Interface **19**
4.1 Device Installation 19
4.2 Device feature overview 20
4.3 Device Dimensions and Images 21
BioShake 3000-T elm | BioShake D30-T elm 21
BioShake 3000 elm (DWP) | BioShake 5000 elm | BioShake D30 elm 22
BioShake 3000 | BioShake D30 23
BioShake 3000-T | HeatPlate | BioShake D30-T 24
BioShake Q1 | BioShake Q1 3mm 25
BioShake Q2 26
ColdPlate 27
ColdPlate slim 28
5. Test Software **29**
5.1 QCOM 1 29
5.2 QCOM 3 29
6. Changelog **30**
7. Notes **31**
8. Support **32**

1. INTRODUCTION

QINSTRUMENTS devices are optimized to be integrated seamlessly into automation platforms. The simple and over all our devices standardized command set allows you to easily set and control process parameters and read out sensor values. Through the integrated microelectronics no other external components or control devices are necessary. All units are designed for continuous 24 hour hands-free operation when utilizing sound scientific methods

Providing long term stable hard- and software interfaces and supporting industry standard like SiLA®, paves the way to a superior level of lab automation. Due to the outstanding integration support we are happy to call the leading providers for lab automation our partners.

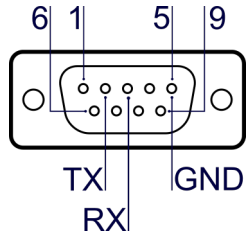
If not stated otherwise, the presented information applies to all applicable QINSTRUMENTS devices (see: "Applicable Devices and Firmware" on page 2). In case there are differences between the devices or the information just applies to a certain type this will be marked.

2. COMMUNICATION INTERFACE

2.1 HARDWARE

The RS232 interface is available through the 9-pin D-subminiature connector. Pins 2 (TX), 3 (RX) and 5 (GND) on the connector are used (see picture below).

Baud rate	9600
Parity	None
Data bits	8
Stop bits	1
Hardware or software handshake (XON/XOFF)	Not supported



The power connection is realized via a barrel connector (ID 2.5 mm x OD 5.5 mm). For all devices the power supply must fulfill the following requirements. Only use the delivered power supply (Mean Well GST120A24) to fully stay within the device certification.

Input	100 - 240 V AC | 50 - 60 Hz
Output	24 V DC | Imax: 5.0 A | Pmax: 120 Watt
Approvals	CE/UL/CS, 85-264 V AC, 47-63 Hz IEC/EN60320-1 C14

2.2 SOFTWARE

The device functions and parameters are controlled via the commands which are provided by QINSTRUMENTS and are explained in detail in chapter: "Software Interface" on the next page. Keep in mind that these commands are just the building blocks for a device driver. It is required to deal with device errors, logging, communication errors and lags, command timing and so on to realize a stable, solid and productive devices control. Additionally specific requirements that are related to the platform in which the unit is integrated might need to be addressed.

Device driver development efforts can be reduced by using the provided SiLA® driver (Order no. 2016-0200).

SiLA Rapid Integration

SiLA® is the global initiative to standardize software interfaces in the field of life science research instrumentation. Instigated by the pharmaceutical industry's need for flexible laboratory automation, the initiative is supported by major device and software suppliers worldwide.

Information is available at: https://sila-standard.com/

3. SOFTWARE INTERFACE

3.1 GENERAL INFORMATION

Information	BS	TC
All commands need to be terminated with: <CR>*	O	O
All return values are terminated with: <CR><LF>**	O	O
Sending an unknown command returns: u->'unknown command'<CR><LF>	O	O
Target speed will be set to 0 rpm after any kind of stop command (soff, seoff, ...) and needs to be set anew before each start of mixing.	O	O
Target acceleration (ssa<value>) needs to be set anew after restarting the device. The value is not saved to EEPROM.	O	O
If the device has an error or the command which is sent interferes with the current status of the device, the device returns: e<CR><LF>***	O	O
If an error occurs (smart LED turns red), the device must be reset, to clear its internal failure memory and continue working normally.****	O	O
When the power supply is connected and active, the system is automatically started (boot process) and all hardware components will be checked. This process takes about 5 seconds for the Q1 and CP devices and about 30 seconds for the BS devices.	O	O
After setting a parameter value or changing the device status with a set command, recheck the parameter value|status with the related get command. This ensures the correct operation of hardware and software.	O	O
The waiting time for status requests with a get command is minimum 100 milliseconds.	O	O
Between two sent signs may be a maximum of 5 seconds. After 5 seconds and without receiving a termination character <CR>, the internal receiver unit is reset and waits for the first character of the next instruction.	-	O
If no communication cable (RS232 cable) is connected, the smart LED turns yellow.	-	O
If the shaker is not used for longer than 15 min, please switch to the ECO mode with the command setEcoMode<CR> to save energy and decrease abrasion. Wake up the shaker with leaveEcoMode<CR>.	O	-

O \| - Applicable function \| Not applicable information

* <CR> is the term for the control character “carriage return” in ASCII code (decimal 13, hexadecimal 0x0D)

** <LF> is the term for the control character “line feed” in ASCII code (decimal 10, hexadecimal 0x0A)

*** Getting e as return value does not necessarily mean that the device has an error. Device errors are written to the error list, which can be accessed with the gel command. An example for status interference is when the command shakeOn is sent while the device is already shaking.

**** Exception: The error 33020 required to switch the device on and off (separate from power supply). A software reset is not enough. (see: "Error Codes" on page 16)

Information	BS	TC
The following commands (if applicable to the device) are changing the device parameter permanently. This means the value is saved to EEPROM and kept even if the device is powered off. disableBootScreen| enableBootScreen | disableCLED | enableCLED | setShakeDefaultDirection | setShakeSpeedLimitMax | setShakeSpeedLimitMin | setTemp40Calibr | setTemp90Calibr | setTempLimiterMax | setTempLimiterMin | setElmSelftest | setElmStartupPosition	O	O

3.2 COMMAND LIST

	Long Form<CR>	Short Form<CR>	Return value<CR><LF>	BS	TC
INITIALIZATION	disableBootScreen		ok	O	O
	disableCLED		ok	O	O
	enableBootScreen		ok	O	O
	enableCLED		ok	O	O
	flashLed	fld	ok	-	O
	getCLED		<value>	-	O
	getDescription		<unit type>	O	O
	getErrorList	gel	{'e_1'; ... ;'e_n'}	O	O
	getSerial		<device serial number>	O	O
	getVersion		<fw version>	O	O
	info		<info text>	O	O
	resetDevice	reset	ok + <boot text>	O	O
	setBuzzer<value>		ok	-	O
	version	v	<unit type + fw version>	O	O
ECO	leaveEcoMode	lem	ok	O	-
	setEcoMode	sem	ok	O	-
SHAKING	getShakeAcceleration	gsa	<value>	O	O
	getShakeAccelerationMax	gsamax	<value>	O	O
	getShakeAccelerationMin	gsamin	<value>	O	O
	getShakeActualSpeed	gsas	<value>	O	O
	getShakeDefaultDirection		<value>	-	O
	getShakeDirection	gsd	<value>	-	O
	getShakeMaxRpm	gsmax	<value>	O	O
	getShakeMinRpm	gsmin	<value>	O	O
	getShakeRemainingTime	gsrt	<value>	O	O
	getShakeSpeedLimitMax		<value>	-	O
	getShakeSpeedLimitMin		<value>	-	O
	getShakeState	gsst	<value>	O	O
	getShakeStateAsString	gsstas	<value>	O	O
	getShakeTargetSpeed	gsts	<value>	O	O
	getShakeZPV		<valuex>-<valuey>	O	-
	setShakeAcceleration<value>	ssa<value>	ok	O	O
	setShakeDefaultDirection<value>		ok	-	O
	setShakeDirection<value>	ssd<value>	ok	-	O

	Long Form<CR>	Short Form<CR>	Return value<CR><LF>	BS	TC
SHAKING	setShakeSpeedLimitMax <value>		ok	-	O
	setShakeSpeedLimitMin<value>		ok	-	O
	setShakeTargetSpeed<value>	ssts<value>	ok	O	O
	shakeEmergencyOff	seoff	ok	O	O
	shakeGoHome	sgh	ok	O	O
	shakeOff	soff	ok	O	O
	shakeOffNonZeroPos	soffnzp	ok	O	O
	shakeOffWithDeenergizeSolenoid	soffwds	ok	O	-
	shakeOn	son	ok	O	O
	shakeOnWithRuntime<value>	sonwr<value>	ok	O	O
TEMPERATURE	getTemp40Calibr		<value>	O	-
	getTemp90Calibr		<value>	O	-
	getTempActual	gta	<value>	O	O
	getTempLimiterMax	gtlmax	<value>	-	O
	getTempLimiterMin	gtlmin	<value>	-	O
	getTempMax	gtmax	<value>	O	O
	getTempMin	gtmin	<value>	O	O
	getTempState	gts	<value>	O	O
	getTempStateAsString	gtsas	<value>	O	O
	getTempTarget	gtt	<value>	O	O
	setTemp40Calibr<value>		ok	O	-
	setTemp90Calibr<value>		ok	O	-
	setTempLimiterMax<value>	stlmax<value>	ok	-	O
	setTempLimiterMin<value>	stlmin<value>	ok	-	O
	setTempTarget<value>	stt<value>	ok	O	O
	tempOff	toff	ok	O	O
	tempOn	ton	ok	O	O
ELM	getElmSelftest		<value>	-	O
	getElmStartupPosition		<value>	-	O
	getElmState	ges	<value>	O	O
	getElmStateAsString	gesas	<value>	O	O
	setElmLockPos	selp	ok	O	O
	setElmSelftest<value>		ok	-	O
	setElmStartupPosition<value>		ok	-	O
	setElmUnlockPos	seup	ok	O	O

3.3 COMMAND DETAILS

	Command<CR> | **Short Form**<CR> *Command description*	<Example Execution><CR>		<Example Return Value><CR><LF> *Additional information*	BS	TC
INITIALIZATION	**disableBootScreen**	disableBootScreen	>>	ok	O	O
	Permanent deactivation of the boot screen startup text.					
	disableCLED	disableCLED	>>	ok	O	O
	Permanent deactivation of the LED indication lights. The instrument will reset after this command.					
	enableBootScreen	enableBootScreen	>>	ok	O	O
	Permanent activation of the boot screen startup text.					
	enableCLED	enableCLED	>>	ok	O	O
	Permanent activation of the LED indication lights. The instrument will reset after this command.					
	flashLed | fld	flashLed	>>	ok	-	O
	User device LED flashes five times.					
	getCLED	getCLED	>>	1	-	O
	Returns the status LED state.			*1 LED is enabled* *2 LED is disabled*		
	getDescription	getDescription	>>	Q.MTP-BIOSHAKE 3000	O	O
	Returns model type.					
	getErrorList | gel	getErrorList	>>	{22150; 32022}	O	O
	Returns a semicolon-separated list with errors and warnings which can occur during processing.			*Note: Please see also section "Error control" in this document.*		
	getSerial	getSerial	>>	0000012345	O	O
	Returns the device serial number.					
	getVersion	getVersion	>>	1.8.00	O	O
	Returns the firmware version number.					
	info	info	>>	<boot screen text>	O	O
	Returns the boot screen text.					
	resetDevice | reset	resetDevice	>>	ok	O	O
	Restarts the controller. Returns ok immediately and sends boot text while booting.			*Note: This takes about 30 seconds for BS units and 5 for the Q1, CP models*		
	setBuzzer<value>	setBuzzer500	>>	ok	-	O
	The buzzer beeps for the given time in milliseconds.			*Note: <value> range from 50 to 999*		
	version | v	version	>>	Q.MTP-BIOSHAKE 3000 v1.8.00	O	O
	Returns model type and fw version number.					

	Command<CR> | **Short Form**<CR> *Command description*	<Example Execution><CR>		<Example Return Value><CR><LF> *Additional information*	BS	TC
ECO	**leaveEcoMode | lem**	leaveEcoMode	>>	ok	O	-
	Leaves the economical mode and switches into the normal operating state.					
	setEcoMode | sem	setEcoMode	>>	ok	O	-
	Switches the shaker into economical mode. It will reduce electricity consumption by deactivation the solenoid that locks the home position and deactivation of the ELM function.			*Note: Home position is not locked!* *Note: All commands other than leaveEcoMode will return: e*		
SHAKING	**getShakeAcceleration| gsa**	getShakeAcceleration	>>	5	O	O
	Returns the acceleration/deceleration value.					
	getShakeAccelerationMax | gsamax	getShakeAccelerationMax	>>	30	O	O
	Returns the maximum acceleration/deceleration time in seconds.					
	getShakeAccelerationMin | gsamin	getShakeAccelerationMin	>>	1	O	O
	Returns the minimum acceleration/deceleration time in seconds.					
	getShakeActualSpeed | gsas	getShakeActualSpeed	>>	399.000000	O	O
	Returns the current mixing speed.					
	getShakeDefaultDirection	getShakeDefaultDirection	>>	0	-	O
	Returns the mixing direction when the device starts up.			*0* *Clockwise mixing direction (CW)* *1* *Counter clockwise mixing direction (CCW)*		
	getShakeDirection | gsd	getShakeDirection	>>	0	-	O
	Returns the current mixing direction.			*0* *Sets clockwise mixing direction (CW)* *1* *Sets counterclockwise mixing direction (CCW)*		
	getShakeMaxRpm | gsmax	getShakeMaxRpm	>>	3000	O	O
	Returns the maximum target speed specific to the device.					
	getShakeMinRpm | gsmin	getShakeMinRpm	>>	200	O	O
	Returns the minimal target speed specific to the device.					
	getShakeRemainingTime | gsrt	getShakeRemainingTime	>>	41	O	O
	Returns the remaining time in seconds if device was started with the command: shakeOnWithRuntime					
	getShakeSpeedLimitMax	getShakeSpeedLimitMax	>>	2000	-	O
	Returns the upper limit for the target speed.			*Note: See command: setShakeSpeedLimitMax<value>*		
	getShakeSpeedLimitMin	getShakeSpeedLimitMin	>>	400	-	O
	Returns the lower limit for the target speed.			*Note: See command: setShakeSpeedLimitMin<value>*		

	Command<CR> | **Short Form**<CR> *Command description*	<Example Execution><CR>		<Example Return Value><CR><LF> *Additional information*		BS	TC
SHAKING	**getShakeState | gsst**	getShakeState	>>	0			
	Returns Shaker state.			*0*	*Running*	*O*	*O*
				1	*Detected a stop command*	*O*	*-*
				2	*Breaking mode*	*O*	*-*
				3	*Stopped | TC only: and is locked at home position*	*O*	*O*
				4	*Manual mode for external control*	*O*	*-*
				5	*Accelerates*	*O*	*O*
				6	*Decelerates*	*O*	*O*
				7	*Decelerates to stop*	*O*	*O*
				8	*Decelerates to stop at home position*	*-*	*O*
				9	*Stopped and is **not** locked | ex. stopping with seoff*	*-*	*O*
				90	*ECO mode*	*O*	*-*
				99	*Boot process running*	*O*	*-*
	getShakeStateAsString | gsstas	getShakeStateAsString	>>	RUN			
	Returns Shaker state as string.			*RUN*	*Running*	*O*	*O*
				STOP	*Stopped and is locked at home pos*	*O*	*O*
				ESTOP	*Emergency Stop*	*O*	*-*
				RAMP+	*Accelerates*	*O*	*O*
				RAMP-	*Decelerates*	*O*	*O*
				dec-stop	*Decelerates to stop*	*-*	*O*
				dec-stop-home	*Decelerates to stop at home pos*	*-*	*O*
				stopped	*Stopped and is **not** locked*	*-*	*O*
	getShakeTargetSpeed | gsts	getShakeTargetSpeed	>>	1800.000000		O	O
	Returns the target mixing speed.						
	getShakeZPV	getShakeZPV	>>	0-0012		O	-
	Returns the state of zero position and the coded voltage.			*0-xxxx*	*Not defined and not locked zero position*		
				1-xxxx	*Correct zero position*		
				Note: This command is only used for service.			
	setShakeAcceleration<value> | **ssa**<value>	setShakeAcceleration12	>>	ok		O	O
	Sets the acceleration/deceleration value in seconds.			*Note: <value> range: [AccMin] – [AccMax] (1 or 2-digit value without comma)*			

	Command<CR> | **Short Form**<CR> *Command description*	<Example Execution><CR>	<Example Return Value><CR><LF> *Additional information*	BS	TC
SHAKING	**setShakeDefaultDirection**<value>	setShakeDefaultDirection0 >>	ok	-	O
	Sets the default mixing direction after device start up.		*0* Sets clockwise mixing direction (CW) *1* Sets counter clockwise mixing direction (CCW)		
	setShakeDirection<value> | **ssd**<value>	setShakeDirection0 >>	ok	-	O
	Sets the mixing direction.		*0* Sets clockwise mixing direction (CW) *1* Sets counter clockwise mixing direction (CCW)		
	setShakeSpeedLimitMax<value>	setShakeSpeedLimitMax1900 >>	ok	-	O
	Set upper limit for the target speed.		*Note: <value> range: 0 to 3000 (1 to 4 digit value without comma)* *Note: <value> must not be lower than the minimum limit*		
	setShakeSpeedLimitMin<value>	setShakeSpeedLimitMin500 >>	ok	-	O
	Set lower limit for the target speed.		*Note: <value> range: 0 to 3000 (1 to 4 digit value without comma)* *Note: <value> must not be greater than the maximum limit* *Note: Speed values below 200 RPM are possible, but not recommended*		
	setShakeTargetSpeed<value> | **ssts**<value>	setShakeTargetSpeed1400 >>	ok	O	O
	Sets the target mixing speed.		*Note: <value> range: [MinRpm] – [MaxRpm] (3 to 4-digit without comma)*		
	shakeEmergencyOff | seoff	shakeEmergencyOff >>	ok	O	O
	Shaker stops immediately at an undefined position ignoring the defined deceleration time.				
	shakeGoHome | sgh	shakeGoHome >>	ok	O	O
	Shaker moves to the home position and locks in place.		*Note: Minimum response time is less than 4 sec (internal failure timeout).*		
	shakeOff | soff	shakeOff >>	ok	O	O
	Stops shaking within the defined deceleration time, go to the home position and locks in place.				
	shakeOffNonZeroPos | soffnzp	shakeOffNonZeroPos >>	ok	O	O
	*Stops shaking within the defined deceleration time, do **not** go to home position and do **not** lock in home position.*				
	shakeOffWithDeenergizeSolenoid | soffwds	shakeOffWithDeenergizeSolenoid >>	ok	O	-
	Stops shaking within the defined deceleration time, go to the home position and locks in place for 1 second, then unlock home position		*Note: Prevents unwanted temperature raise caused by an energized solenoid*		
	shakeOn | son	shakeOn >>	ok	O	O
	Starts shaking with defined speed with defined acceleration time.				
	shakeOnWithRuntime<value> | **sonwr**<value>	shakeOnWithRuntime30 >>	ok	O	O
	Starts shaking with defined speed with defined acceleration time for given time value in seconds.		*Note: <value> range: 0 – 999999 (1 to 6-digits, without comma)*		

	Command<CR> | **Short Form**<CR> *Command description*	<Example Execution><CR>		<Example Return Value><CR><LF> *Additional information*	BS	TC
TEMPERATURE	**getTemp40Calibr**	getTemp40Calibr	>>	40.100000	O	-
	Returns the offset value at the 40 °C calibration point.					
	getTemp90Calibr	getTemp90Calibr	>>	89.800000	O	-
	Returns the offset value at the 90 °C calibration point.					
	getTempActual	getTempActual	>>	74.400000	O	O
	Returns the current temperature.			*Note: Deprecated command: getActualTemp, gat*		
	getTempLimiterMax | gtlmax	getTempLimiterMax	>>	70.000000	-	O
	Returns the upper limit for the target temperature.			*Note: Can be adjusted with setTempLimiterMax*		
	getTempLimiterMin | gtlmin	getTempLimiterMin	>>	4.000000	-	O
	Returns the lower limit for the target temperature.			*Note: Can be adjusted with setTempLimiterMin*		
	getTempMax | gtmax	getTempMax	>>	99.999999	O	O
	Returns the device specific maximum target temperature.					
	getTempMin | gtmin	getTempMin	>>	-20.999999	O	O
	Returns the device specific minimum target temperature.					
	getTempState| gts	getTempState	>>	0	O	O
	Returns the state of the temperature control feature.			*0* *Temperature control is disabled* *1* *Temperature control is enabled*		
	getTempStateAsString| gtsas	getTempStateAsString	>>	on	O	O
	Returns the state of the temperature control feature as as string.			*off* *Temperature control is disabled* *on* *Temperature control is enabled*		
	getTempTarget | gtt	getTempTarget	>>	64.000000	O	O
	Returns the target temperature.			*Note: Deprecated command: getTargetTemp*		
	setTemp40Calibr<value>	setTemp40Calibr401	>>	ok	O	-
	Sets the offset value at the 40 °C calibration point in 1/10 °C increments.			*Note: <value> range: 000 – 990 (3-digits, without comma)*		
	setTemp90Calibr<value>	setTemp40Calibr892	>>	ok	O	-
	Sets the offset value at the 90 °C calibration point in 1/10 °C increments.			*Note: <value> range: 000 – 990 (3-digits, without comma)*		
	setTempLimiterMax<value> | **stlmax**<value>	setTempLimiterMax700	>>	ok	-	O
	Sets the upper limit for the target temperature in 1/10 °C increments.			*Note: <value> range: -200 to 999 (3-digits, without comma)* *Note: <value> must not be lower than the device specific TempMin*		

	Command<CR> | **Short Form**<CR> *Command description*	<Example Execution><CR>		<Example Return Value><CR><LF> *Additional information*	BS	TC
TEMPERATURE	**setTempLimiterMin**<value> | **stlmin**<value>	setTempLimiterMin40	>>	ok	-	O
	Sets the lower limit for the target temperature in 1/10 °C increments.			*Note: <value> range: -200 to 999 (3-digits, without comma)* *Note: <value> must not be higher than the device specific TempMax*		
	setTempTarget<value> | **stt**<value>	setTempTarget640	>>	ok	O	O
	Sets the target temperature between TempMin and TempMax in 1/10 °C increments.			*Note: <value> range: TempMin to TempMax (3-digits, without comma)* *Note: Temperature value is automatically limited to the min/max value* *Note: Best cooling is achieved if the set temperature is within the boundaries of the technical specifications and not if it is set to the minimal value.*		
	tempOff | toff	tempOff	>>	ok	O	O
	Switches off the temperature control feature and stops heating/cooling.					
	tempOn | ton	tempOn	>>	ok	O	O
	Switches on the temperature control feature and starts heating/cooling.			*Note: It is strongly recommended to control if the desired temperature is reached (see command getTempActual)*		
ELM	**getElmSelftest**	getElmSelftest	>>	0	-	O
	Returns whether the ELM self-test is enabled or disabled at device startup.			*0 ELM self-test is disabled at startup* *1 ELM self-test is enabled at startup*		
	getElmStartupPosition	getElmStartupPosition	>>	0	-	O
	Returns the ELM position after device startup.			*0 ELM is locked after the device startup* *1 ELM is unlocked after the device startup*		
	getElmState | ges	getElmState	>>	1		
	Returns the ELM status.			*0 ELM position is undefined* - O *1 ELM is in locked position* O O *3 ELM is in unlocked position* O O *9 ELM error occurred* O O		
	getElmStateAsString | gesas	getElmStateAsString	>>	1		
	Returns the ELM status as a string.			*ELMUndefined ELM position is undefined* - O *ELMLocked ELM is in locked position* O O *ELMUnlocked ELM is in unlocked position* O O *ELMError ELM error occurred* O O		
	setElmLockPos | selp	setElmLockPos	>>	ok	O	O
	Closes the ELM.			*Note: The runtime is less than 3 seconds.* *Note: This position is a current-free static state.* *Note: Deprecated command: setElmShakePos, sesp*		

	Command<CR> | **Short Form**<CR> *Command description*	<Example Execution><CR>		<Example Return Value><CR><LF> *Additional information*	BS	TC
ELM	**setElmSelftest**<value>	setElmSelftest0	>>	ok	-	O
	Set whether the ELM self-test is enabled or disabled at device startup.			*0* ..... *ELM self-test is disabled at startup* *1* ..... *ELM self-test is enabled at startup*		
	setElmStartupPosition<value>	setElmStartupPosition0	>>	ok	-	O
	Set the ELM position after device startup.			*0* ..... *ELM is locked after device startup* *1* ..... *ELM is unlocked after device startup*		
	setElmUnlockPos | seup	setElmUnlockPos	>>	ok	O	O
	Opens the ELM			*Note: The runtime is less than 3 seconds.* *Note: This position is a current-free static state.* *Note: The ELM should only be used when the Tablar is in the home position. Otherwise, it does not open as wide as possible, and it is likely that the position of the Tablar will change, while opening.*		

3.4 ERROR CODES

In case of failure the device smart LED turns red and the device error code is accessible via the command: `getErrorList`

If an error occurred and not stated otherwise the device must be reset to clear its internal failure memory and continue working normally. If the error cannot be solved by restarting the device, please get in contact with our "Support" on page 32

BS DEVICES

	Code	Description
SHAKING	101[1]	Error by the DC motor controller.
	102	Error due to speed failure, for example happens through mechanical locking.
	103	Errors caused by an uninitialized shaker or incorrect initialization parameters after switching on.
	104[1]	Errors caused by unsuccessful initialization routine.
	105[1]	Errors caused by not achieving the home position at stop command.
	106[1]	Errors caused by over speed.
TEMPERATURE	201[1]	Error due failed answers from temperature sensors or incorrect internal settings of temperature sensors.
	202[1]	Error due to temperature communication bus system.
	203	Sensor with the requested ID is not found while working.
	204	Errors caused by a faulty temperature measurement while working.
	206[1]	Error caused by checksum of the internal temperature sensor.
	207[1]	Error caused by checksum of the main temperature sensor.
	208[1]	Error caused by general checksum.
	209[1]	Error caused by unknown temperature method.
	210[1]	Error caused by overheating.
ELM	300[1]	General error.
	301[1]	IC-Driver error.
	303	Verification error by the unlock position.
	304	Error caused by unsuccessful reach the lock position (timeout).
	305	Error caused by unsuccessful reach the unlock position (timeout).
	306	Error caused by unsuccessful reach the lock position (over current).
	307	Error caused by unsuccessful reach the unlock position (over current).

TC DEVICES

	Code	Description
BASIC	10002 or 10003	Instruction sent with an invalid parameter.
	100xx[2]	Internal firmware sequence failure.
	2xxxx[2]	Internal MCU periphery error.
	310xx[2]	EEPROM data verification failed.
TEMPERATURE	320xx[2]	Communication with internal temperature sensors failed.
	33010[3]	Device internal temperature too hot.
	33020[3, 4]	Emergency shutdown of the temperature fuse has triggered.
	33030	Emergency temperature sensor validation has failed.
	34010 or 34110	FAN-1 or FAN-2 power supply invalid.
	34020 or 34120	FAN-1 or FAN-2 has stalled.
	34030 or 34130	FAN-1 or FAN-2 airway clogged.
	35010	TEC power supply invalid.
	35020	TEC power supply short circuit detected.
	35030	TEC power supply open circuit detected.
	360xx[2]	Internal temperature controller failure.
SHAKING	37030	Shaker has stalled.
	37040	Shaker cannot move because the solenoid does not unlock.
	37060	Unable to lock Shaker in the home position.
	37070	“Shaker find home position” timeout occurred.
	370xx[2]	Internal shake controller failure.
	39030	Solenoid motion timeout occurred.
	390xx[2]	Internal solenoid controller failure.
ELM	38030	ELM motion timeout occurred.
	38090	ELM self-test has failed.
	380xx[2]	Internal ELM controller failure.

[1] - Please get in contact with the QINSTRUMENTS service team.
[2] - Value for x can be ignored.
[3] - Please let the device cool down, before performing a reset.
[4] - The power of the device must be turned off to clear the error.

3.5 EXAMPLE

The following example shows a simple process routine, where the device is brought into home position, the ELM is opened and closed to grab a microplate, shaking parameters are set and the device is started and stopped after a defined amount of time.

This is a working but simplified example (no error handling, communication control, ...) and is hence, not suitable for productive use. In this example Python 3.6.1 and PySerial 3.3 are used.

	Python Code	Print Output	Description
INIT	`import serial; import time`		Import of required python modules
	`ser = serial.Serial('COM3', timeout=1)`		Open the connection to COM-Port 3 where the device is connected to
	`ser.write(b'`**`getShakeState`**`\r'); time.sleep(0.2)`		Get device status to check if device reached home position; Wait 0.2 seconds before reading the cmd response.
	`print(ser.read(ser.in_waiting))`	**3**<CR><LF>	Device is in home position
ELM	`ser.write(b'`**`setElmUnlockPos`**`\r'); time.sleep(3)`		Open ELM; Wait at least 3 seconds to execute ELM movement
	`print(ser.read(ser.in_waiting))`	**ok**<CR><LF>	Command was executed successfully
	`ser.write(b'`**`getElmState`**`\r'); time.sleep(0.2)`		Get ELM status; Wait 0.2 seconds before reading the command response
	`print(ser.read(ser.in_waiting))`	**3**<CR><LF>	ELM is open
	Place the microplate on the device		
ELM	`ser.write(b'`**`setElmLockPos`**`\r'); time.sleep(3)`		Close ELM; Wait at least 3 seconds to execute ELM movement
	`print(ser.read(ser.in_waiting))`	**ok**<CR><LF>	Command was executed successfully
	`ser.write(b'`**`getElmState`**`\r'); time.sleep(0.2)`		Get ELM status; Wait 0.2 seconds before reading the command response
	`print(ser.read(ser.in_waiting))`	**1**<CR><LF>	ELM is closed
SHAKING	`ser.write(b'`**`setShakeTargetSpeed1500`**`\r'); time.sleep(0.2)`		Set mix velocity to 1500 rpm; Wait 0.2 seconds before reading the command response
	`print(ser.read(ser.in_waiting))`	**ok**<CR><LF>	Command was executed successfully
	`ser.write(b'`**`setShakeAcceleration5`**`\r'); time.sleep(0.2)`		Set time for acceleration/deceleration to 5 seconds; Wait 0.2 seconds before reading the command response
	`print(ser.read(ser.in_waiting))`	**ok**<CR><LF>	Command was executed successfully
	`ser.write(b'`**`shakeOn`**`\r'); time.sleep(7)`		Start mixing; Wait 7 seconds (5 sec. acceleration + extra time) before reading the command response
	`print(ser.read(ser.in_waiting))`	**ok**<CR><LF>	Command was executed successfully
	`ser.write(b'`**`getShakeState`**`\r'); time.sleep(0.2)`		Get device status to check if device is shaking; Wait 0.2 seconds before reading the command response
	`print(ser.read(ser.in_waiting))`	**0**<CR><LF>	Devise is running
	`ser.write(b'`**`getShakeActualSpeed`**`\r'); time.sleep(0.2)`		Get device speed as additional check of status; ; Wait 0.2 seconds before reading the command response
	`print(ser.read(ser.in_waiting))`	**1490.000000**<CR><LF>	Set speed is reached
	`time.sleep(60)`		Keep mixing for 60 seconds
	`ser.write(b'`**`shakeOff`**`\r'); time.sleep(7)`		Stop mixing; Wait 7 seconds (5 sec. deceleration + extra time) before reading the command response
	`print(ser.read(ser.in_waiting))`	**ok**<CR><LF>	Command was executed successfully
	`ser.write(b'`**`getShakeState`**`\r'); time.sleep(0.2)`		Get device status to check if device stopped; Wait 0.2 seconds before reading the command response
	`print(ser.read(ser.in_waiting))`	**3**<CR><LF>	Devise is stopped and is locked at home position

3.6 EXTENDED PROGRAMMING INFORMATION

This chapter presents additional information on programming while using the device commands.

Return value 'e'

Information: It is important to know that there are several situations in which the return value of a command is `e`, although there is no error in the device error list. This is due to point that the return value `e` is also returned if the command does not fit to the current status of the device or other criteria.

Examples are:

- Sending `son` or `sonwr` if speed is not set
- Sending `son` or `sonwr` command while Shaker is already running.
- Sending `son` or `sonwr` command while the ELM is open
- Most commands while in ECO mode
- `seup` while ELM is already open and the other way around
- `ton` while temperature control is already running
- ...

Tip: Use the get commands (`getShakeState, getTempState, getElmState`) to evaluate the current device status before sending commands to prevent erroneous usage of commands.

setElmLockPos | setElmUnlockPos | setEcoMode

Information: After sending the command `setElmLockPos, setElmUnlockPos, setEcoMode` the device does not send an immediate response. The `ok` value is written to the serial input buffer after the ELM reached the (un)lock position, ECO mode. In the meantime, all commands that are sent to the device will be buffered and executed after the device reached the new status.

Tip: To determine the end of the `setElmLockPos, setElmUnlockPos, setEcoMode` command, read from the serial input buffer until the ok value is received. Do not send any commands in the meantime.

leaveEcoMode | resetDevice

Information: After sending the command `leaveEcoMode, resetDevice` the device immediately writes `ok` to the serial input buffer although the change of the status has not finished and commands that are sent in the meantime will not be executed or return `e`. Additionally it is to be kept in mind, that the reset command will write the boot text to the serial buffer while booting.

Tip: To determine the end of the `leaveEcoMode, resetDevice` command poll for the device status with the `getShakeState` command until it reaches status `3` (Shaker stopped and is locked in home position).

4. HARDWARE INTERFACE

4.1 DEVICE INSTALLATION

All QINSTRUMENTS devices are mounted via the same physical interface and have one corner for the cable outlets, as can be seen in the schematic top view on the device, below.

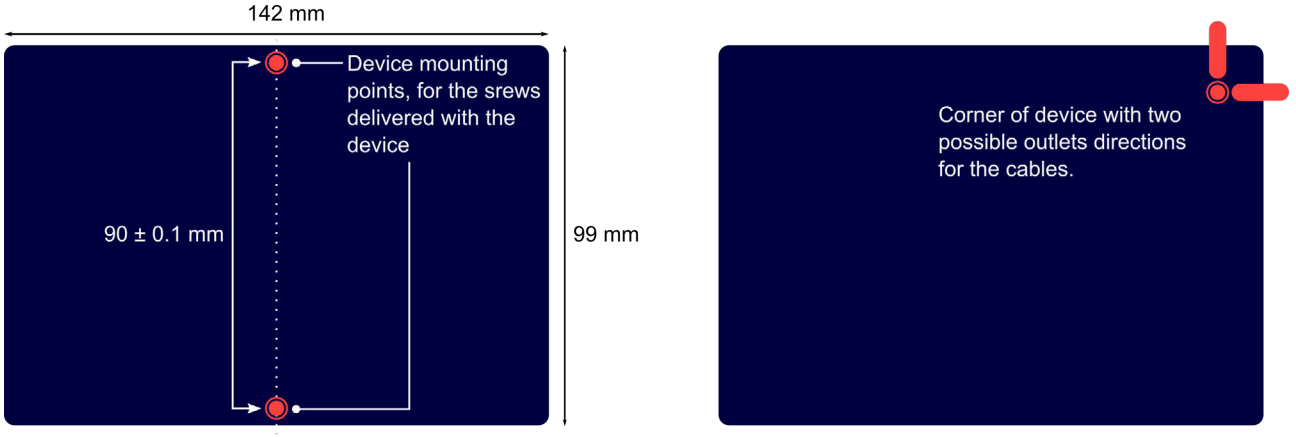


Figure 1 Schematic device top view showing the mounting points and cable outlet options

Besides mounting, other important aspects, for example

- cable length (2 m by default)
- sufficient venting for devices with heat/cool feature
- deal with condensation water for devices with cooling feature
- space between Shakers if multiple Shakers are integrated (space for ELM movement, air inlet and outlet, other Labware nearby, ...)
- height of adapters
- robot specific requirements (device height, clearance, collision areas, ...)

need to be addressed when integrating one or multiple devices. A detailed description about the installation procedure, in depth device specifications and further device specific information is available in the device specific Operation manual.

Required device installation information for each device is available online at: https://www.qinstruments.com/service/downloads/
This information is mandatory required to successfully integrate a device.

Pulling at the upper part (Tablar) can damage the instrument and result in malfunction.
ONLY handle and lift the instrument on the lower section.

4.2 DEVICE FEATURE OVERVIEW

In the table below an overview of the most important device features is presented. These features and the related requirements must be taken care of, for a successful integration approach. Especially for devices that can **heat/cool** or are equipped with an **adapter** the impact for additional space and venting must be considered. Please use the Operation Manual of the device for further information.

Detailed device specification are available for each product on our homepage and as a technical data sheet and Operation manual

https://www.qinstruments.com/service/downloads/

	Article name	Part number	Link to Manual	ELM	Orbit* [mm]	Speed** [rpm]	Heat	Cool	Adapter is usable	Width [mm]	Length [mm]	Height*** [mm]
BS	BioShake 3000	2016-0016	🔗	-	2	3000	-	-	O	99	142	60.75
	BioShake 3000 elm	2016-0017	🔗	O	2	3000	-	-	-	99	142	55.35
	BioShake 3000 elm DWP	2016-0018	🔗	O	2	3000	-	-	-	99	142	55.35
	BioShake 3000-T	2016-0516	🔗	-	2	3000	O	-	O	99	142	62.7
	BioShake 3000-T elm	2016-0517	🔗	O	2	3000	O	-	O	99	142	60.55
	BioShake 5000 elm	2016-0022	🔗	O	1.2	5000	-	-	-	99	142	55.35
	BioShake D30	2016-0015	🔗	-	3	2000	-	-	O	99	142	60.75
	BioShake D30 elm	2016-0025	🔗	O	3	2000	-	-	-	99	142	55.35
	BioShake D30-T	2016-0519	🔗	-	3	2000	O	-	O	99	142	62,7
	BioShake D30-T elm	2016-0518	🔗	O	3	2000	O	-	O	99	142	60.55
	HeatPlate	2016-0100	🔗	-	-	-	O	-	O	99	142	62.7
TC	ColdPlate	2016-0110	🔗	-	-	-	O	O	O	99	142	79
	ColdPlate slim	2016-0111	🔗	-	-	-	O	O	O	99	235	45.5
	BioShake Q1	2016-0600	🔗	O	2	3000	O	O	O	99	142	97.7
	BioShake Q1 3mm	2016-0601	🔗	O	3	2000	O	O	O	99	142	97.7
	BioShake Q2	2016-0620	🔗	-	3	2000	O	O	O	99	142	90.2

* Amplitude = 0.5 x Orbit

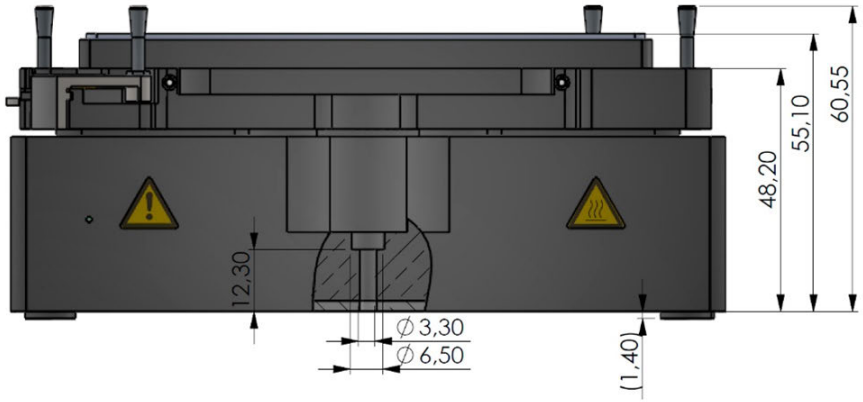
** The given value is the maximum speed of the device The minimum speed for all devices is 200 rpm. The actual speed that should be used, strongly depends on several parameters like the adapter and microplate that is used, the amount of liquid and much more. It will therefore be potentially lower. Please refer to the Operation Manual of your device for further information.

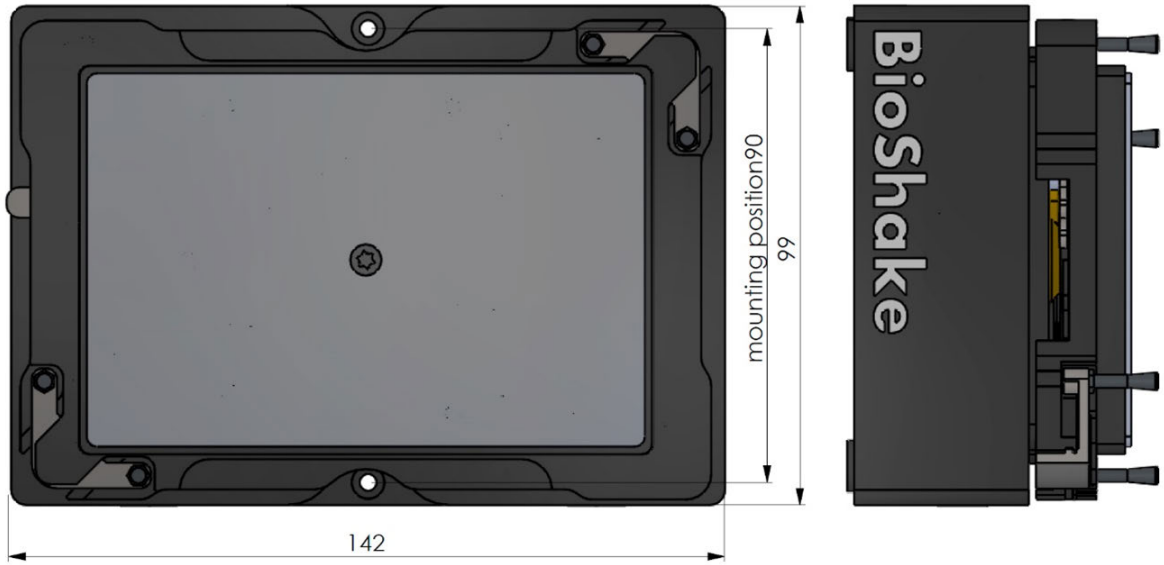
*** The given value is the maximum height of the device. This encompasses the standard pins but **not** the adapter that might be mounted and **not** the rubber feet (height of 1.4 mm)

4.3 DEVICE DIMENSIONS AND IMAGES

BioShake 3000-T elm | BioShake D30-T elm

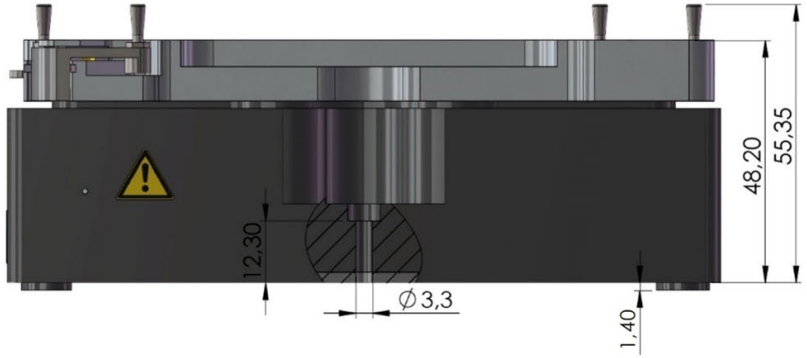
2016-0517 2016-0518

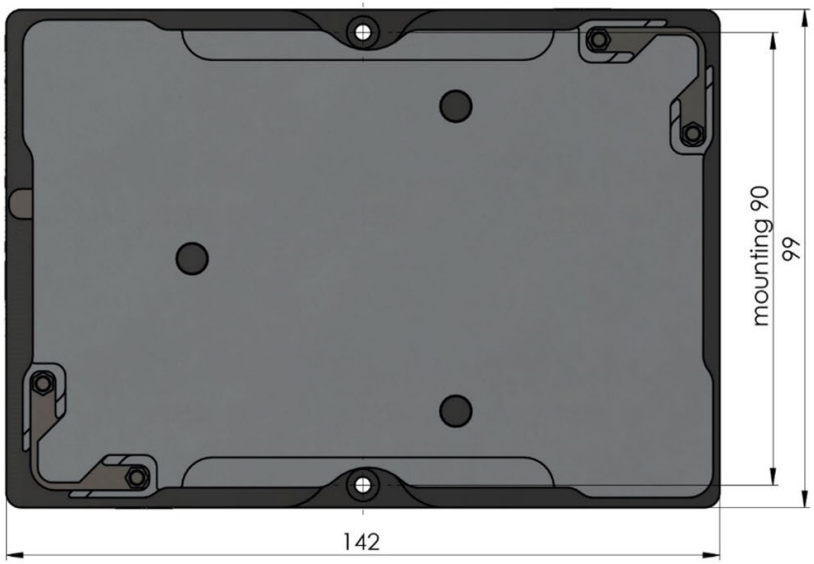




BioShake 3000 elm (DWP) | BioShake 5000 elm | BioShake D30 elm

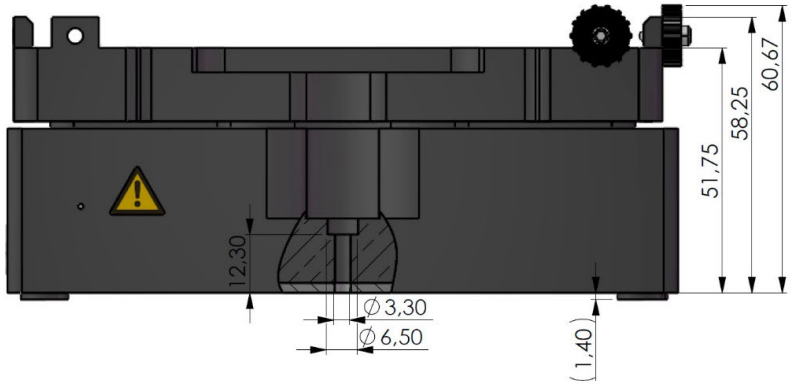
2016-0017 (2016-0018) 2016-0022 2016-0025

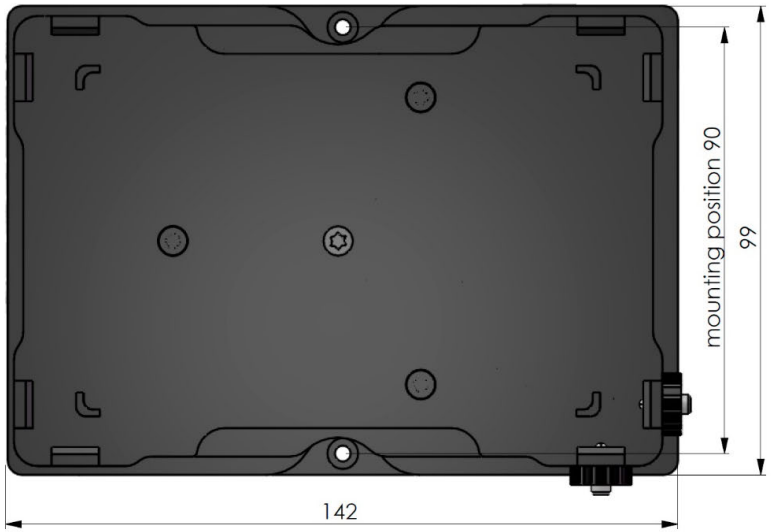


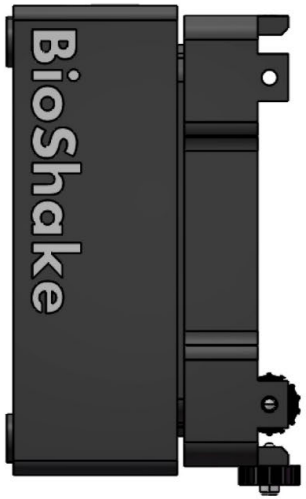


BioShake 3000 | BioShake D30

2016-0016 2016-0015

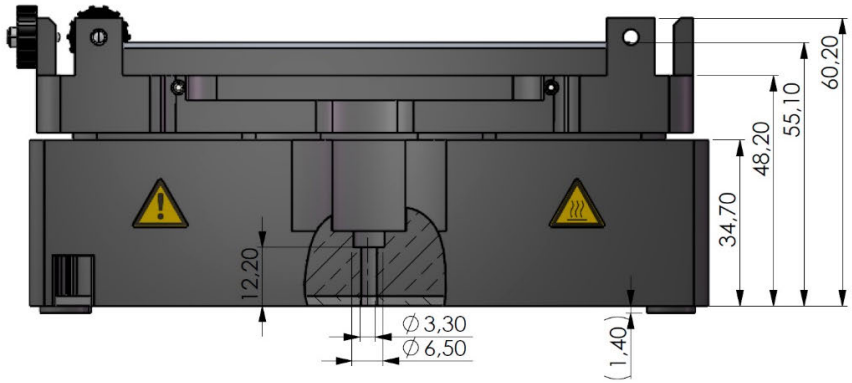


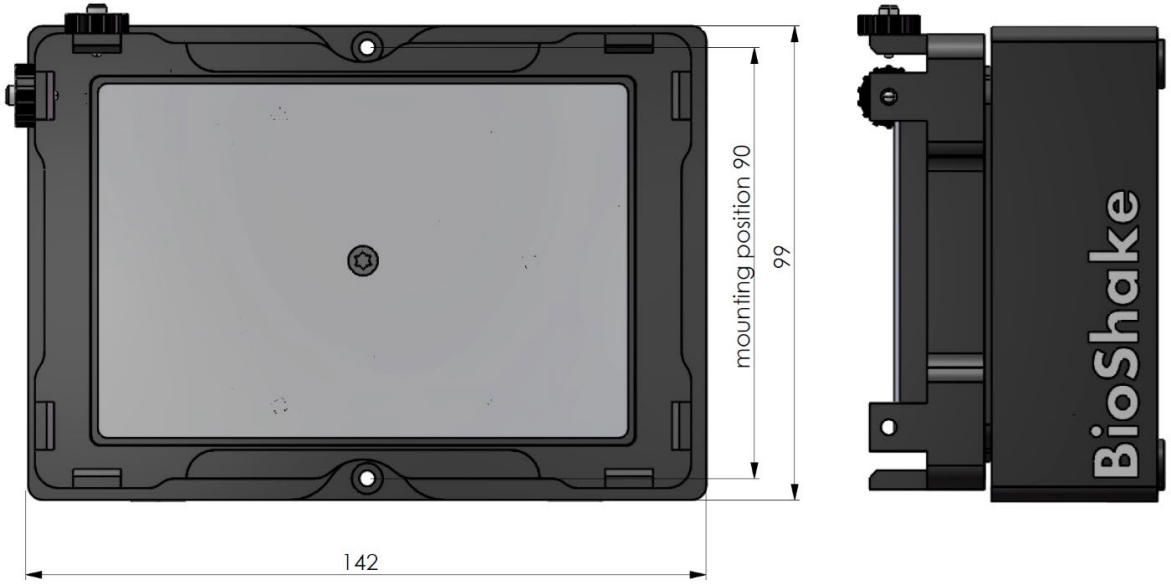




BioShake 3000-T | HeatPlate | BioShake D30-T

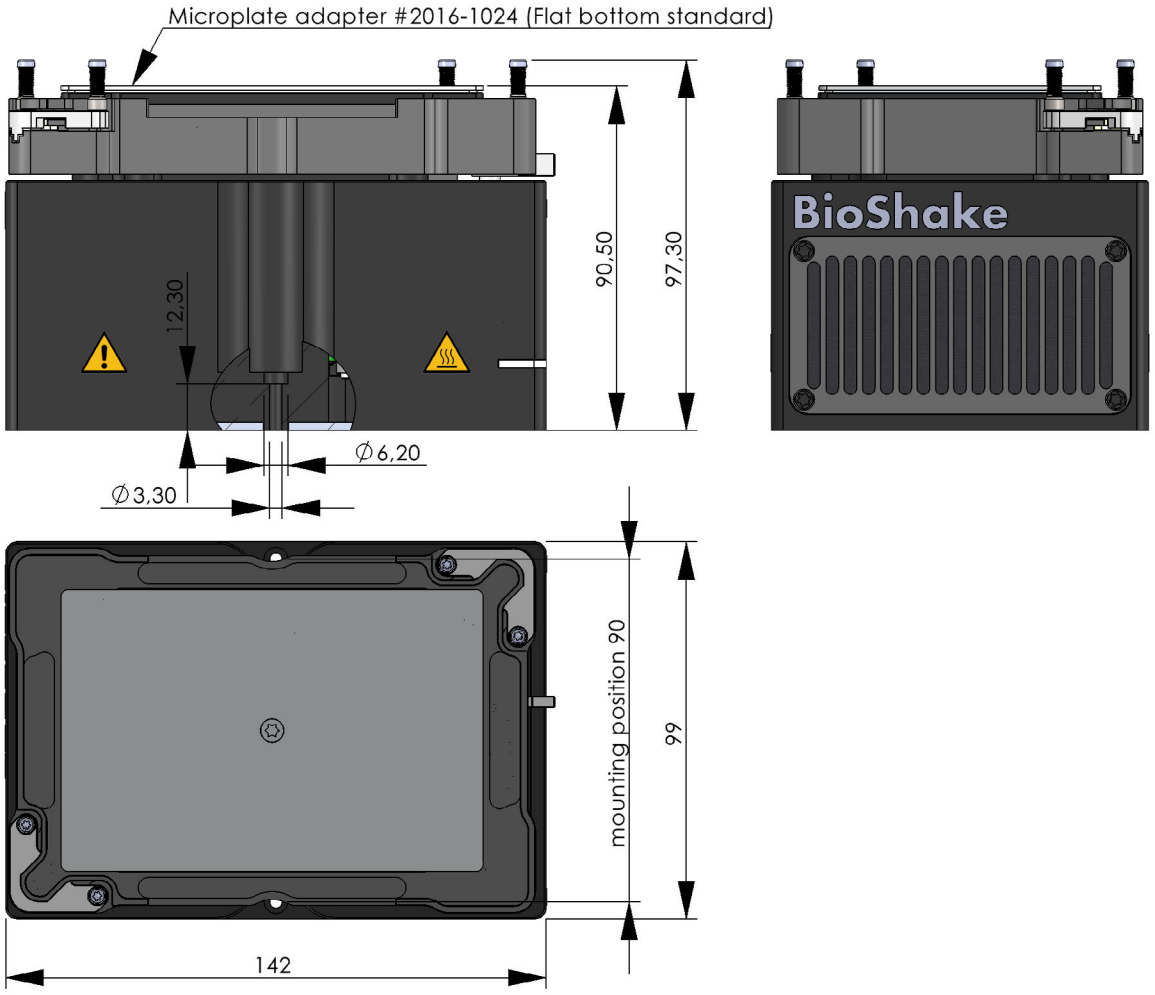
2016-0516 2016-0100 2016-0519





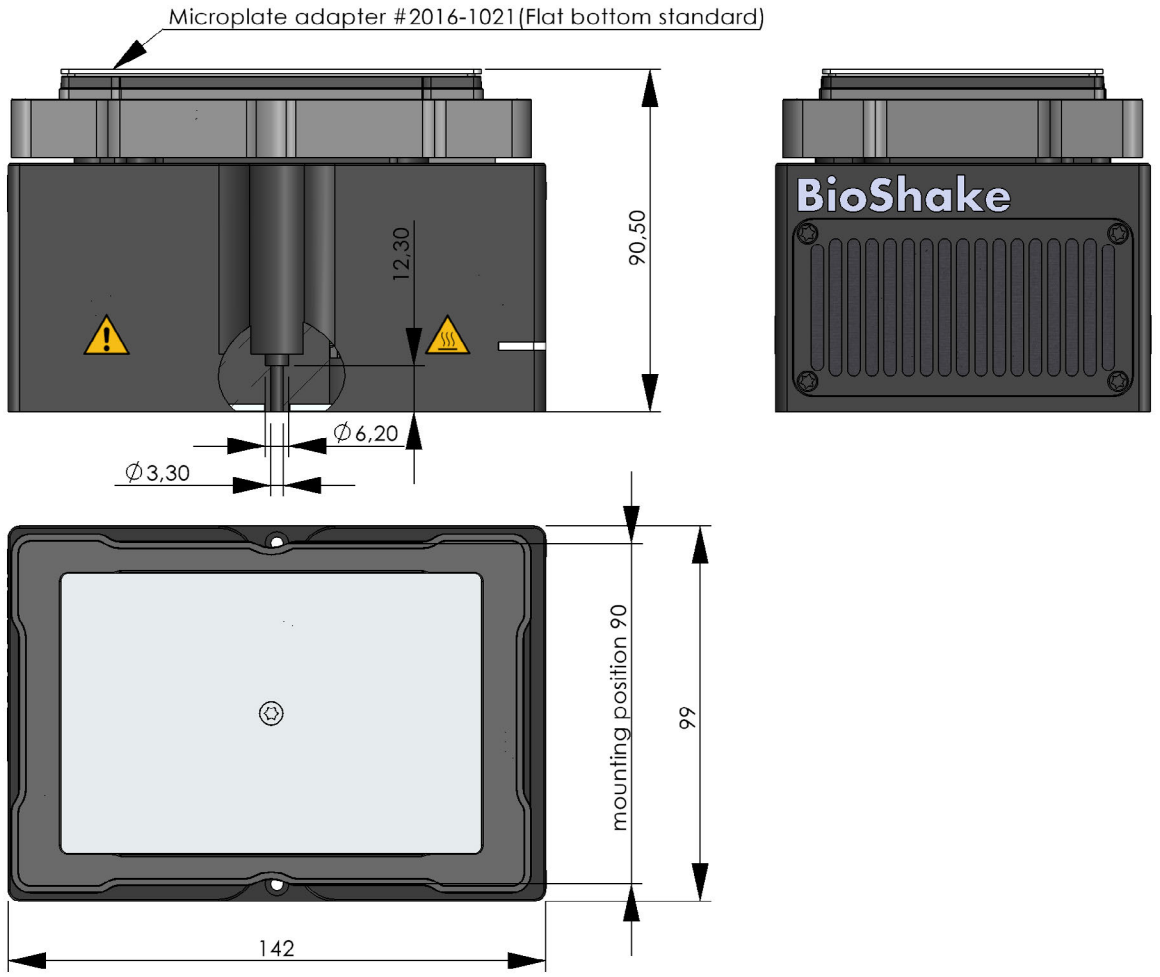
BioShake Q1 | BioShake Q1 3mm

2016-0600 2016-0601



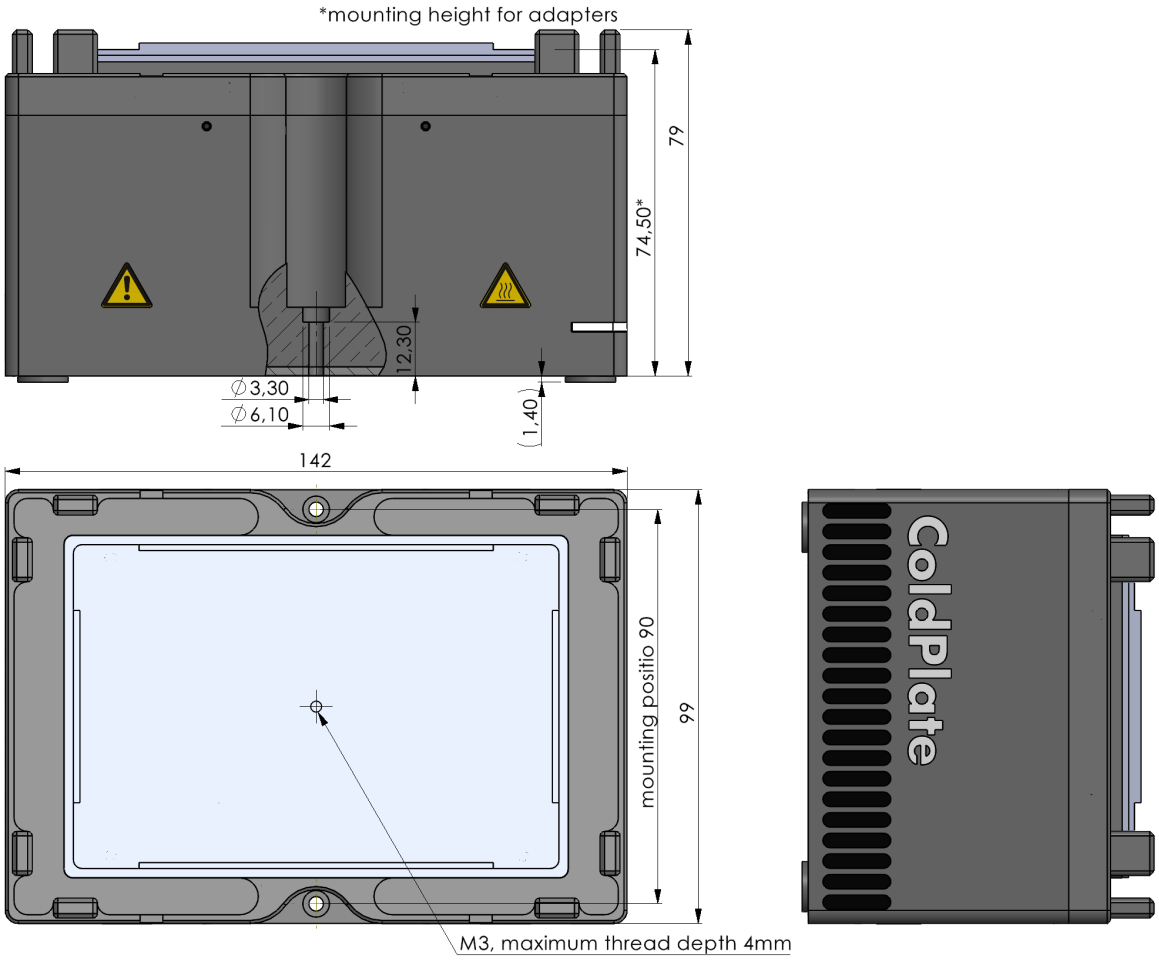
BioShake Q2

2016-0620



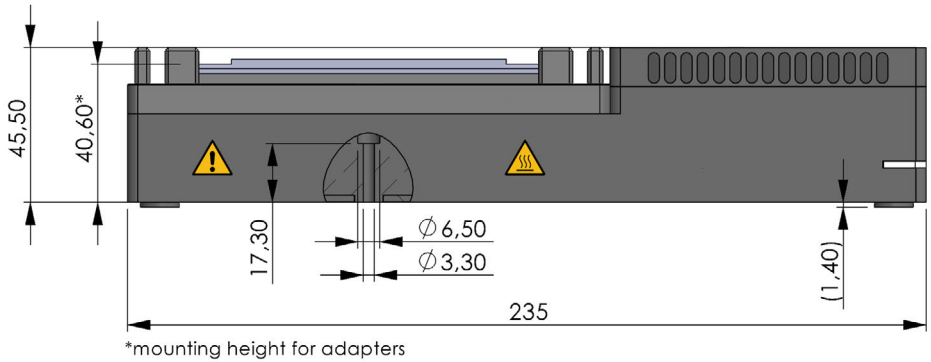
ColdPlate

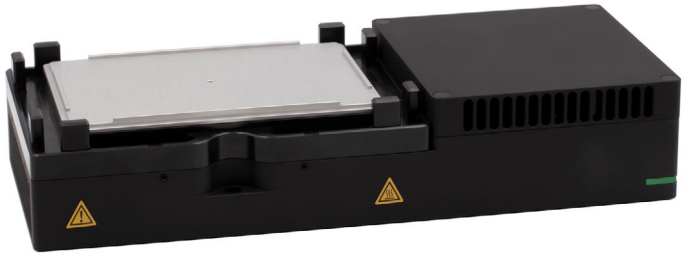
2016-0110

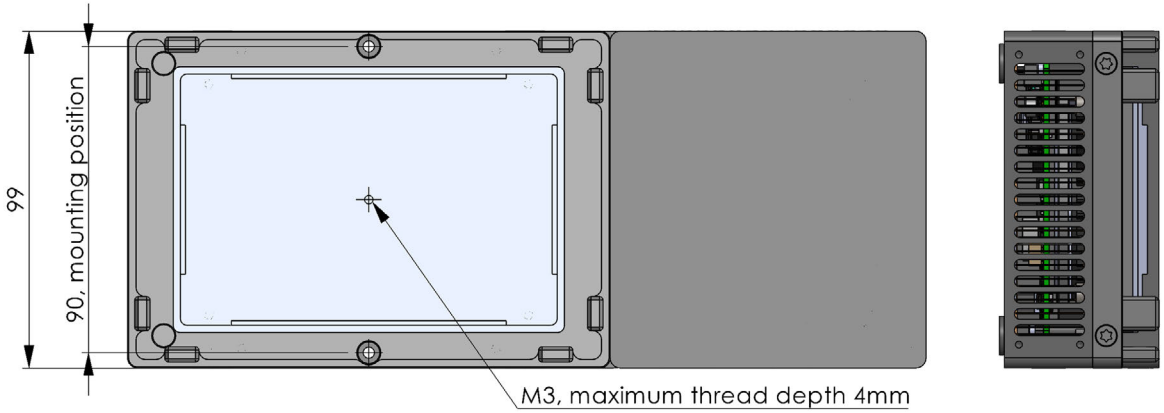


ColdPlate slim

2016-0111



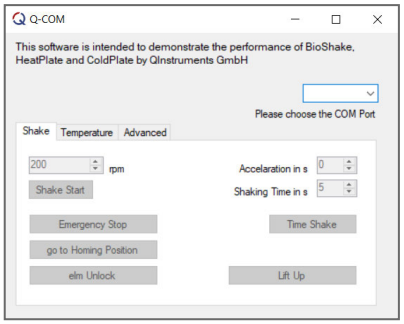




5. TEST SOFTWARE

5.1 QCOM 1

QCOM1 is a simple test tool for Windows to start using the shaker in moments and to exercise all shaker features. Plug in the RS232 cable from the single BioShake module into a free port of your computer. If it's necessary, please use a USB/RS232 converter.

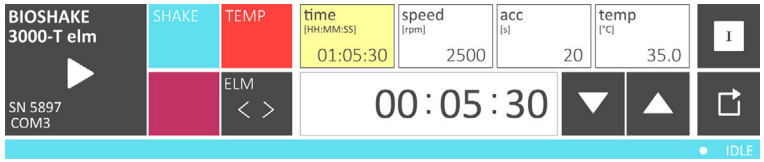


Download is available at

https://www.qinstruments.com/service/downloads/

5.2 QCOM 3

QCOM3 is a small test software with a graphical user interface (GUI) to control lab automation instruments from QINSTRUMENTS, eg. BioShake, ColdPlate, HeatPlate, TiltStation.



The main purpose of QCOM3 is to get easy access to the unit to execute initial testing.

Download is available at

https://www.qinstruments.com/service/downloads/

To start the program, execute QCOM3.exe. By default, the program scans through the available COM ports and detects if a supported QINSTRUMENTS device is connected to that port. The first valid device that is found is used, the scanning process is stopped, and the program starts using the identified device. Device features will be detected at start-up, and the GUI will be adapted to them.

For more details on how to use the software, please refer to the manual that is part of the QCOM3 download.

6. CHANGELOG

Version	Information
010.8	Added installation note | Fixed error in command spelling | Corrected ELM status description
010.7	Extend resetDevice command return value description
010.6	Change in table layout | Removed inaccessible shaker state | Fixed shaker state 3 description | Switch from QCOM 2 to QCOM 3
010.5	Fixed error in command spelling and added note to shaker state
010.4	Added Q1 3mm, Q2 | CP, Q1, Q1 3mm, Q2 forming a new group: TC (thermo control) | Update QCOM 2 pic | Prog. example fixed
010.3	Fixed error in command list and description <getSerial> | Fixed error in <setElmStartupPosition>, <getCLED> description
010.2	Fixed error in error number | Fixed links to Operational Manuals | Added Information for constant change of parameters
010.1	Improvement of spelling mistakes
010.0	Initial Creation • Merge of two distinct Integration Manuals for BioShake devices with cooling feature and without • New design integrating BICO branding

7. NOTES

8. SUPPORT

We provide a range of technical material (e.g., application notes, poster, bulletin, data sheets) that describes our products and key applications in detail. All of our technical documents are available on our web page in the corresponding product pages. Technical Data Sheet, Operation manual, 2D and 3D drawings and Software can be found in the download area of each product.
Please use the following link to our support form (https://www.qinstruments.com/service/support/) in case service or support for your product is needed. Please ensure to provide the serial number, as it is an important information for our support team. Direct contact via phone or email is also possible.

Online	QINSTRUMENTS.com
E-Mail	support@qinstruments.com
Phone	+49 3641 55430
Address	QINSTRUMENTS GmbH **A BICO COMPANY** Loebstedter Strasse 101 07749 Jena Germany

DISCLAIMER, LEGAL NOTICES AND TRADEMARKS

All document design, text, graphics, the selection and arrangement thereof and all other materials in this document are copyright by QINSTRUMENTS. QINSTRUMENTS GmbH reserves the right to modify their products for quality improvement and such modifications may not be documented in this manual. This manual and the information herein have been assembled with due diligence. QINSTRUMENTS does not assume liability for misprints or cases of damage resulting from misprints in this manual. If there are any uncertainties, please contact info@qinstruments.com.

QINSTRUMENTS is owner of numerous patents worldwide. Please respect our intellectual property.

WO2008135565, US8323588, EP2144716: Sample handling device for and methods of handling a sample | **WO2011113858, US9126162, EP2547431**: Positioning unit for a functional unit
WO2013113847, US10052598, EP2809436: Cog-based mechanism for generating an orbital shaking motion
WO2013113849, US9371889, EP2809435: Mechanism for generating an orbital motion or a rotation motion by inversing a drive direction of a drive unit
WO2014207243, US20160368003, EP3013480: Application-specific sample processing by modules surrounding a rotor mechanism for sample mixing and sample separation
WO0020221281 4A1: Laboratory apparatus comprising a fixing mechanism for fixing a slide | **WO0020221281 28809A2**: Laboratory apparatus comprising a mixing mechanism for mixing a medium of a slide

Please notify us in writing, by email or mail to our designated agent, if you believe that a user has infringed our intellectual property rights. QINSTRUMENTS trademarks are recognised worldwide. Please respect our trademarks as we will vigorously protect their proper usage.

QINSTRUMENTS®, BioShake®, ColdPlate®, HeatPlate® (QINSTRUMENTS GmbH)

Trademarks of third parties may appear on this site when referring to those entities or their products or services. All registered names, trademarks, etc. used on this site, even when not specifically marked as such, are not to be considered unprotected by law. Any names and trademarks not specifically marked or listed are property of the respective owner.

Further trademarks used in this website and catalogs: Brand® (BRAND GmbH + Co KG), Corning® (Corning, Inc.), Eppendorf® (Eppendorf AG), Thermomixer® (Eppendorf AG), Eppendorf Tubes® (Eppendorf AG), Eppendorf twin.tec® (Eppendorf AG), Falcon® (Becton, Dickinson And Company), Greiner® (Greiner Labortechnik GmbH), MOXA® (Moxa, Inc.), NUNC® (Nunc NS Corporation), SILA Rapid Integration® (Association Consortium Standardization in Lab Automation), TECAN® (TECAN Group AG), Windows® (Microsoft Corporation).

Technical specifications are subject to change without notice. All rights reserved.