

Command List:

Updated

2019/05/01	KPI Measurement Robotic Cell	Initial release
2019/06/05		Record commands
2019/06/20		Update simplify record commands

Implementation Priority:

- 1. Green
- 2. White
- 3. Orange

Common Command List

****: string

#: int or float

x: boolean

Tx end flag: \r\n





Rx end flat: \r\n@_@\r\n


Rx response is returned to acknowledge Tx command is received

[...]: optional parameters

Note: all command responses should be command acknowledge; then the PC will poll the input status to check command completion


Command	Description	Example
cmd_fixture_info()	Get the all information about the fixture.	Tx: cmd_fixture_info() Rx: fixture_name:**** vendor:**** fixture_id:**** firmware_version:**** software_version:**** date:**** @_@
cmd_help()	Get all the TX commands and remarks to reference.	Tx: cmd_help() Rx: cmd_fixture_info() cmd_help() cmd_reset_fixture() ... @_@
cmd_reset_fixture()	Reset the fixture to initial status.	Tx: cmd_reset_fixture() Rx: reset_fixture:True @_@
cmd_move_position_increment()	Move the motion stage according to the axis number and increment position input [mm or deg]. Axis: xp, yp, zp, xr, yr, zr	Tx: cmd_move_position_increment({"xp":10.0,"yp":23,"zp":23,"xr":23,"yr":23,"zr":23}) Rx: move_position_increment:True @_@

cmd_move_position_absolute()	Move the motion stage according to the axis number and absolute position input [mm or deg]. Axis: xp, yp, zp, xr, yr, zr	Tx: cmd_move_position_absolute({"xp":10.0,"yp":23,"zp":23,"xr":23,"yr":23,"zr":23}) Rx: move_position_absolute:True @_@	
cmd_move_joint_increment()	Move the motion stage according to the axis number and increment position input [mm or deg]. Joint: j1, j2, j3, j4, j5, j6	Tx: cmd_move_joint_increment({"j1":10.0,"j2":233,"j3":223,"j4":213,"j5":233,"j6":-23}) Rx: move_joint_increment:True @_@	
cmd_move_joint_absolute()	Move the motion stage according to the axis number and absolute position input [mm or deg]. Joint: j1, j2, j3, j4, j5, j6	Tx: cmd_move_joint_absolute({"j1":10.0,"j2":233,"j3":223,"j4":213,"j5":233,"j6":-23}) Rx: move_joint_absolute:True @_@	
cmd_move_velocity()	Move the motion stage according to the axis number and velocity input [mm/sec or deg/sec].	Tx: cmd_move_velocity({"axis01":10.0,"axis02":20.0}) cmd_move_velocity({"axis02":2.0}) Rx: move_velocity_axis01:True move_velocity_axis02:True @_@ move_velocity_axis02:True @_@	
cmd_wait()	Pause the motion stage according to the time duration input [sec].	Tx: cmd_wait(#) Rx: wait:True @_@	
cmd_home_position()	Search for the home position. When no argument is entered, home all axes.	Tx: cmd_home_position() cmd_home_position("xp","yp") Rx: home_position:{"xp":True,"yp":True, "zp":True,"xr":True, "yr":True,"zr":True } @_@ home_position:{"xp":True,"yp":True} @_@	
cmd_home_joint()	Search for the home position. When no argument is entered, home all axes.	Tx: cmd_home_joint()	

		<p>cmd_home_joint("j1","j2")</p> <p>Rx: home_joint:{"j1":True,"j2":True, "j3":True,"j4":True , "j5":True,"j6":True } @_ home_joint:{"j1":True,"j2":True} @_@</p>	
cmd_stop()	Stop the motion stage according to the axis number. When no argument is entered, stop all axes.	<p>Tx: cmd_stop ()</p> <p>Rx: stop:True @_@</p>	
cmd_check_position()	Report the motion stage position value according to the axis number. When no argument is entered, return all status.	<p>Tx: cmd_check_position()</p> <p>Rx: check_position:{"xp":10.0,"yp":23,"zp":23,"xr":23,"yr":23,"zr":23} @_@</p>	
cmd_check_joint()	Report the motion stage position value according to the axis number. When no argument is entered, return all status.	<p>Tx: cmd_check_joint()</p> <p>Rx: check_joint:{"j1":10.0,"j2":233,"j3":223,"j4":213,"j5":233,"j6":-23} @_@</p>	
cmd_check_velocity()	Report the motion stage velocity value according to the axis number. When no argument is entered, return all status.	<p>Tx: cmd_check_velocity("axis##":["axis##"]) cmd_check_velocity()</p> <p>Rx: check_velocity_axis##: [check_velocity_axis##:] @_ check_velocity_axis##: check_velocity_axis##: @_@</p>	
cmd_check_input()	Query digital input status. When no argument is entered, return all status.	<p>Tx: cmd_check_input() cmd_check_input("input03","input04")</p> <p>Rx: check_input:{"input01":False,"input02":True,"input03":False,"input04":True } @_ check_input:{"input03":False,"input04":True} @_@</p>	

cmd_check_output()	Query digital output status. When no argument is entered, return all status.	Tx: cmd_check_output() cmd_check_output("output03","output04") Rx: check_output:{"output01":False,"output02":True,"output03":False,"output04":True } @_@ check_output:{"output03":False,"output04":True} @_@
cmd_set_input() [OPTIONAL]	Set station input status.	Tx: cmd_set_input({"input01":True}) Rx: input01:x @_@
cmd_set_output()	Set station output status.	Tx: cmd_set_output({"output01":True}) Rx: set_output:{"output01":True} @_@
cmd_move_position_set_velocity()		Tx: cmd_move_position_set_velocity({"xp":360}) Rx: move_position_set_velocity:True @_@
cmd_move_joint_set_velocity()		Tx: cmd_move_joint_set_acceleration({"j1":360,"j2":320}) Rx: move_joint_set_acceleration:True @_@
cmd_move_position_set_acceleration()		Tx: cmd_move_position_set_acceleration({"xp":360}) Rx: move_position_set_velocity:True @_@
cmd_move_joint_set_acceleration ()		Tx: cmd_move_joint_set_acceleration({"j1":360,"j2":320}) Rx: move_joint_set_acceleration:True @_@

cmd_abort()		Tx: cmd_abort() Rx: abort:True @_@
Cmd_record_start()	Record xp, yp, zp, xr, yr, zr, j1, j2, ... j6 for the duration as given in the numeric parameter [sec] or until stop command is issued when no parameter is available.	Tx: cmd_record_start(("position")) Rx: record_start:True @_@
Cmd_record_stop()	Stop recording robot position and joint data	Tx: Cmd_record_stop() Rx: Cmd_record_stop: True @_@
cmd_record_position_start()	Record xp, yp, zp, xr, yr, zr for the duration as given in the numeric parameter [sec] or until stop command is issued when no parameter is available.	Tx: cmd_record_position_start() Rx: record_position_start:True @_@
cmd_record_position_stop()	TBD	Tx: cmd_record_position_stop() Rx: record_position_stop:True @_@
cmd_record_joint_start()	Record j1, j2, j3, j4, j5, j6 for the duration as given in the numeric parameter [sec] or until stop command is issued when no parameter is available.	Tx: cmd_record_joint_start() Rx: record_joint_start:True @_@
cmd_record_joint_stop()	TBD	Tx: cmd_record_joint_stop() Rx: record_joint_stop:True @_@
cmd_get_record()	Load the previous position or joint recording	Tx: cmd_get_record()

	Return False if no recording is available. TBC	Rx: get_record: True t0, j1, j2, j3, j4, j5, j6 t1, j1, j2, j3, j4, j5, j6 t2, j1, j2, j3, j4, j5, j6 t3, j1, j2, j3, j4, j5, j6 ... tn, j1, j2, j3, j4, j5, j6 @_@
cmd_run_job()	Run the program by its number from the robot controller	Tx: cmd_run_job(1) Rx: run_job: True @_@ 
	Need a command to check robot status. Use output status?	

