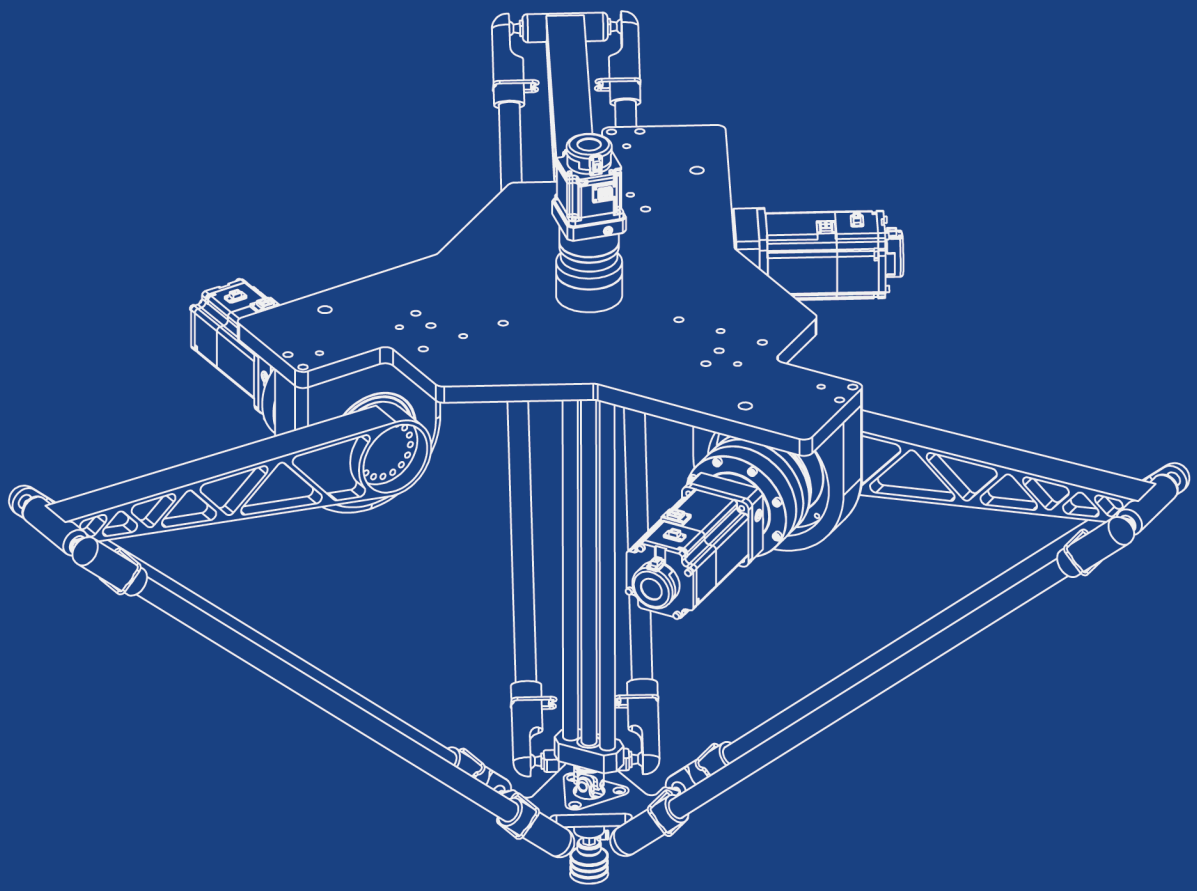
ADEEB ROBOTICS

MOTION CONTROL API DOCUMENTATION

Precision. Scalability. Control



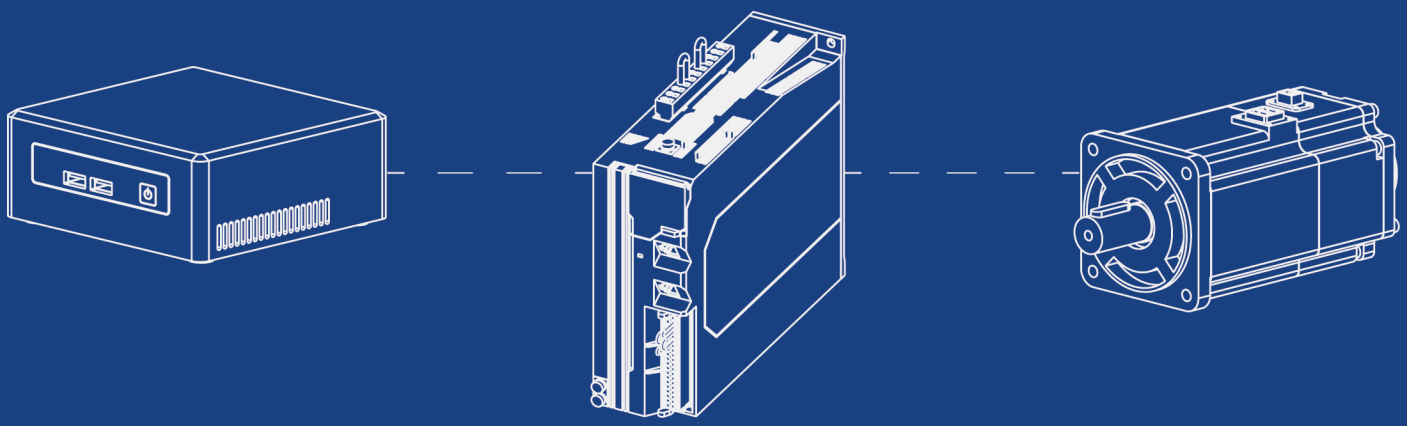


Table of Content

Software Release Version: 0.0.14
Last Updated: 2026-05-05

1. Getting Started with Adeeb Robotics Controller

Thank you for using the Adeeb Robotics Controller software library. To start using the library, open a command prompt or terminal and execute the following command:

```
adeeb_robotics_controller <path_to_config_file>
```

The `config_files` folder contains example configuration files that you may use. For example:

```
adeeb_robotics_controller config_file/delta_config.yml
```

Next, open a separate command prompt or terminal window and run the desired Python motion script:

```
python3 <path_to_python_script>
```

Example motion scripts are available in the `python_scripts` folder. You may use any of these scripts except `arc.py`, which is a helper module. For instance:

```
python3 pack_expo_demo.py
```

Please note that certain Python scripts are intended to be used with specific configuration files. For example:

1. `pack_expo_demo.py` and `autorun.py` should be used with `delta_config.yml` or `demo_stand_config.yml`.
2. `serial_dh_example.py` is intended to be used with `kuka_7_axis_config.yml`.

Configuration files define system settings, including the robot type and workspace layout, such as conveyor and container positions used for pick-and-place operations.

The Python scripts handle motion control and I/O operations.

2. System Information

API Route	Method	Description	Payload
`/get_sw_release_version`	GET	Returns the current software release version	String

2.1. Configuration

API Route	Method	Description	Payload
`/get_config`	GET	Returns the current configuration in JSON format	Config
`/save_config`	POST	Sets the configuration object in JSON format	ConfigSaveData

3. Network and Intialization

3.1. Network Control

API Route	Method	Description	Payload
`/init_network`	POST	Initiates network initialization request	*N/A*
`/init_network/<cmd_idx>`	GET	Checks network initialization response	EcatCmdData

3.1.1. Axis Initialization

API Route	Method	Description	Payload
`/init_axes`	POST	Initiates axes initialization request	*N/A*

`/init_axes/<cmd_idx>`	GET	Checks axes initialization response	MotnCmdData

3.1.2. Operation Mode Control

API Route	Method	Description	Payload
`/req_preop`	POST	Initiates pre-operational mode request	*N/A*
`/req_preop/<cmd_idx>`	GET	Checks pre-operational mode response	EcatCmdData
`/req_op`	POST	Initiates operational mode request	*N/A*
`/req_op/<cmd_idx>`	GET	Checks operational mode response	EcatCmdData

4. Slave Management

API Route	Method	Description	Payload
`/get_number_of_slaves`	POST	Initiates slave count request	*N/A*
`/get_number_of_slaves/<cmd_idx>`	GET	Checks slave count response	NumSlvsData

5. Motion Control

5.1. Axis Control

API Route	Method	Description	Payload
`/axis_servo_on`	POST	Initiates servo on request	AxisData
`/axis_servo_on/<cmd_idx>`	GET	Checks servo on response	AxisData
`/axis_servo_off`	POST	Initiates servo off request	AxisData
`/axis_servo_off/<cmd_idx>`	GET	Checks servo off response	AxisData
`/axis_move_pos`	POST	Initiates absolute position movement request	AxisMovePosData
`/axis_move_pos/<cmd_idx>`	GET	Checks absolute position movement response	AxisMovePosData
`/axis_move_vel`	POST	Initiates velocity movement request	AxisMoveVelData
`/axis_move_vel/<cmd_idx>`	GET	Checks velocity movement response	AxisMoveVelData
`/axis_move_rel`	POST	Initiates relative position movement request	AxisMoveRelData
`/axis_move_rel/<cmd_idx>`	GET	Checks relative position movement response	AxisMoveRelData
`/axis_move_home`	POST	Initiates axis home movement request	AxisData
`/axis_move_home/<cmd_idx>`	GET	Checks axis home movement response	AxisData
`/axis_get_curr_pos`	POST	Initiates getting current position request	AxisCurrPosData
`/axis_get_curr_pos/<cmd_idx>`	GET	Checks getting current position response	AxisCurrPosData
`/axis_write_dig_out`	POST	Append a digital write command to the axis motion queue	AxisDigData
`/axis_write_dig_out/<cmd_idx>`	GET	Checks if a digital write command if appended to the axis motion queue	AxisDigData
`/axis_read_dig_in`	POST	Initiates an immediate (not queued) digital read request	AxisDigData
`/axis_read_dig_in/<cmd_idx>`	GET	Checks digital read response	AxisDigData
`/axis_motion_done`	POST	Initiates axis motion done request	AxisBoolData
`/axis_motion_done/<cmd_idx>`	GET	Checks axis motion done response	AxisBoolData
`/axis_quick_stop`	POST	Initiates axis quick stop request	AxisData
`/axis_quick_stop/<cmd_idx>`	GET	Checks axis quick stop response	AxisData
`/axis_clear_alarms`	POST	Initiates axis clear alarms request	AxisData
`/axis_clear_alarms/<cmd_idx>`	GET	Checks axis clear alarms response	AxisData
`/axis_get_alarms`	POST	Initiates getting axis alarms request	AxisAlarmsData

`/axis_get_alarms/<cmd_idx>`	GET	Checks getting axis get alarms response	AxisAlarmsData
`/axis_zero`	POST	Initiates axis zero request, this alters the enco_offst entry in the config file, thus a server restart is required for it to take effect	AxisData
`/axis_zero/<cmd_idx>`	GET	Checks axis zero response	AxisData
`/axis_sleep`	POST	Initiates axis sleep request, an axis sleep command is inserted into the motion que.	AxisSleepData
`/axis_sleep/<cmd_idx>`	GET	Checks axis sleep response	AxisSleepData
`/axis_set_spd_prcn`	POST	Initiates setting axis speed request, speed is the form of a percent of the maximum possible vel, acc, and jerk motion trajectory limits	AxisSpdPrcnData
`/axis_set_spd_prcn/<cmd_idx>`	GET	Get setting axis speed response	AxisSpdPrcnData
`/axis_get_spd_prcn`	POST	Initiates getting axis speed request, speed is the form of a percent of the maximum possible vel, acc, and jerk motion trajectory limits	AxisSpdPrcnData
`/axis_get_spd_prcn/<cmd_idx>`	GET	Checks getting axis speed response	AxisSpdPrcnData

5.2. Group Control

API Route	Method	Description	Payload
`/group_servo_on`	POST	Initiates group servo on request	GroupData
`/group_servo_on/<cmd_idx>`	GET	Checks group servo on response	GroupData
`/group_servo_off`	POST	Initiates group servo off request	GroupData
`/group_servo_off/<cmd_idx>`	GET	Checks group servo off response	GroupData
`/group_move_pos`	POST	Initiates group absolute position movement request	GroupMovePosData
`/group_move_pos/<cmd_idx>`	GET	Checks group absolute position movement response	GroupMovePosData
`/group_move_rel`	POST	Initiates group relative position movement request	GroupMoveRelData
`/group_move_rel/<cmd_idx>`	GET	Checks group relative position movement response	GroupMoveRelData
`/group_move_home`	POST	Initiates group home movement request	GroupData
`/group_move_home/<cmd_idx>`	GET	Checks group home movement response	GroupData
`/group_end_eff_engage`	POST	Initiates group end effector engagement request	GroupData
`/group_end_eff_engage/<cmd_idx>`	GET	Checks group end effector engagement response	GroupData
`/group_end_eff_eject`	POST	Initiates group end effector eject request	GroupData
`/group_end_eff_eject/<cmd_idx>`	GET	Checks group end effector eject response	GroupData
`/group_end_eff_clear`	POST	Initiates group enffector clear request; end effector is neither engaged nor ejected	GroupData
`/group_end_eff_clear/<cmd_idx>`	GET	Checks group end effector clear response	GroupData
`/group_get_bgn_pos`	POST	Intiates next motion queue begin position request	GroupBgnPosData
`/group_get_bgn_pos/<cmd_idx>`	GET	Checks next motion queue begin position response	GroupBgnPosData
`/group_get_curr_pos`	POST	Intiates getting group current position request	GroupCurrPosData

`/group_get_curr_pos/<cmd_idx>`	GET	Checks getting group current position response	GroupCurrPosData
`/group_write_dig_out`	POST	Append a digital write command to the group motion queue	GroupDigData
`/group_write_dig_out/<cmd_idx>`	GET	Checks if a digital write command if appended to the group motion queue	GroupDigData
`/group_motion_done`	POST	Initiates group motion completion request	GroupBoolData
`/group_motion_done/<cmd_idx>`	GET	Checks group motion completion response	GroupBoolData
`/group_quick_stop`	POST	Intiates group quick stop request	GroupData
`/group_quick_stop/<cmd_idx>`	GET	Checks group quick stop response	GroupData
`/group_clear_alarms`	POST	Intiates group clear alarms request	GroupData
`/group_clear_alarms/<cmd_idx>`	GET	Checks group clear alarms response	GroupData
`/group_get_alarms`	POST	Intiates group get alarms request	GroupAlarmsData
`/group_get_alarms/<cmd_idx>`	GET	Checks group get alarms request response	GroupAlarmsData
`/group_zero`	POST	Intiates group zero request	GroupData
`/group_zero/<cmd_idx>`	GET	Checks group zero response	GroupData
`/group_sleep`	POST	Intiates group sleep request	GroupSleepData
`/group_sleep/<cmd_idx>`	GET	Checks group sleep response	GroupSleepData
`/group_set_spd_prcn`	POST	Initiates setting group move profile velocity, acceleration and jerk as % of max supported value	GroupSpdPrcnData
`/group_set_spd_prcn/<cmd_idx>`	GET	Checks setting group move profile velocity, acceleration, and jerk, as % of max supported value response	GroupSpdPrcnData
`/group_get_spd_prcn`	POST	Initiates getting group move profile velocity, acceleration and jerk as % of max supported value	GroupSpdPrcnData
`/group_get_spd_prcn/<cmd_idx>`	GET	Checks getting group move profile velocity, acceleration, and jerk, as % of max supported value response	GroupSpdPrcnData
`/clear_alarms`	POST	Intiates clear all alarms request	MotnCmdData
`/clear_alarms/<cmd_idx>`	GET	Checks clear all alarms response	MotnCmdData
`/emergency_stop`	POST	Intiates emergency stop request	MotnCmdData
`/emergency_stop/<cmd_idx>`	GET	Checks emergency stop response	MotnCmdData

5.3. Conveyor Operations

API Route	Method	Description	Payload
`/conv_set_spd`	POST	Initiates setting conveyor speed request	ConvSpdData
`/conv_set_spd/<cmd_idx>`	GET	Checks setting conveyor speed response	ConvSpdData
`/group_conv_intercept_pickup`	POST	Initiates conveyor intercept for pick-up request	GroupConvPickupData
`/group_conv_intercept_pickup/<cmd_idx>`	GET	Checks conveyor intercept for pick-up response	GroupConvPickupData
`/group_conv_intercept_dropoff`	POST	Initiates conveyor intercept for drop-off request	GroupConvDropoffData

`/group_conv_intercept_dropoff/<cmd_idx>`	GET	Checks conveyor intercept for drop-off response	GroupConvDropoffData
`/group_conv_descend`	POST	Initiates conveyor descend request	GroupConvData
`/group_conv_descend/<cmd_idx>`	GET	Checks conveyor descend response	GroupConvData
`/group_conv_track_pickup`	POST	Initiates tracking conveyor pick up request	GroupConvData
`/group_conv_track_pickup/<cmd_idx>`	GET	Checks tracking conveyor pick up response	GroupConvData
`/group_conv_track_dropoff`	POST	Initiates tracking conveyor drop-off request	GroupConvData
`/group_conv_track_dropoff/<cmd_idx>`	GET	Checks tracking conveyor drop-off response	GroupConvData
`/group_conv_ascend`	POST	Initiates conveyor ascend request	GroupConvData
`/group_conv_ascend/<cmd_idx>`	GET	Checks conveyor ascend response	GroupConvData
`/group_conv_pickup`	POST	Initiates group conveyor pickup request	GroupConvPickupData
`/group_conv_pickup/<cmd_idx>`	GET	Checks group conveyor pickup response	GroupConvPickupData
`/group_conv_dropoff`	POST	Initiates group conveyor drop-off request	GroupConvDropoffData
`/group_conv_dropoff/<cmd_idx>`	GET	Checks group conveyor drop-off response	GroupConvDropoffData

5.4. Container Operations

API Route	Method	Description	Payload
`/group_cont_intercept_dropoff`	POST	Initiates container intercept at drop-off request	GroupContDropoffData
`/group_cont_intercept_dropoff/<cmd_idx>`	GET	Checks container intercept at drop-off response	GroupContDropoffData
`/group_cont_intercept_pickup`	POST	Initiates container intercept at pick-up request	GroupContPickupData
`/group_cont_intercept_pickup/<cmd_idx>`	GET	Checks container intercept at pick-up response	GroupContPickupData
`/group_cont_intercept_pickup_idx`	POST	Initiates queing a group move to a position directly above the indexed part in the cantainer	GroupContPickupIdxData
`/group_cont_intercept_pickup_idx/<cmd_idx>`	GET	Checks response of queing a group move to a position directly above the indexed part in the cantainer	GroupContPickupIdxData
`/group_cont_descend`	POST	Initiates container descend request	GroupContData
`/group_cont_descend/<cmd_idx>`	GET	Checks container descend response	GroupContData
`/group_cont_track_dropoff`	POST	Initiates tracking container drop-off request	GroupContData
`/group_cont_track_dropoff/<cmd_idx>`	GET	Checks tracking container drop-off response	GroupContData
`/group_cont_track_pickup`	POST	Initiates tracking container pick-up request	GroupContData
`/group_cont_track_pickup/<cmd_idx>`	GET	Checks tracking container pick-up response	GroupContData

`/group_cont_ascend`	POST	Initiates container ascend request	GroupContData
`/group_cont_ascend/<cmd_idx>`	GET	Checks container ascend response	GroupContData
`/group_cont_dropoff`	POST	Initiates container drop-off request	GroupContDropoffData
`/group_cont_dropoff/<cmd_idx>`	GET	Checks container drop-off response	GroupContDropoffData
`/group_cont_pickup`	POST	Initiates container pick-up request	GroupContPickupData
`/group_cont_pickup/<cmd_idx>`	GET	Checks container pick-up response	GroupContPickupData
`/group_cont_pickup_idx`	POST	Intiates container part pick-up in motion queue using auto part indexing	GroupContPickupIdxData
`/group_cont_pickup_idx/<cmd_idx>`	GET	Checks response of container part pick-up in motion queue using auto part indexing	GroupContPickupIdxData
`/cont_set_empty`	POST	Initiates setting internal container part count to zero request	ContData
`/cont_set_empty/<cmd_idx>`	GET	Checks setting internal container part count to zero response	ContData
`/cont_set_full`	POST	Initiates setting internal container part count to full request	ContPartData
`/cont_set_full/<cmd_idx>`	GET	Checks setting internal container part count to full response	ContPartData
`/cont_not_empty`	POST	Initiates checking container is not empty request	ContBoolData
`/cont_not_empty/<cmd_idx>`	GET	Checks container is not empty response	ContBoolData
`/cont_not_full`	POST	Initiates checking container is not full request	ContPartBoolData
`/cont_not_full/<cmd_idx>`	GET	Checks container is not full response	ContPartBoolData

6. EtherCAT Commands

6.1. COE (CANopen over EtherCAT) Operations

API Route	Method	Description	Payload
`/coe_wr_u8`	POST	Initiates U8 write request	CoeU8Data
`/coe_wr_u8/<cmd_idx>`	GET	Checks U8 write response	CoeU8Data
`/coe_wr_u16`	POST	Initiates U16 write request	CoeU16Data
`/coe_wr_u16/<cmd_idx>`	GET	Checks U16 write response	CoeU16Data
`/coe_wr_u32`	POST	Initiates U32 write request	CoeU32Data
`/coe_wr_u32/<cmd_idx>`	GET	Checks U32 write response	CoeU32Data
`/coe_rd_u8`	POST	Initiates U8 read request	CoeU8Data
`/coe_rd_u8/<cmd_idx>`	GET	Checks U8 read response	CoeU8Data
`/coe_rd_u16`	POST	Initiates U16 read request	CoeU16Data
`/coe_rd_u16/<cmd_idx>`	GET	Checks U16 read response	CoeU16Data
`/coe_rd_u32`	POST	Initiates U32 read request	CoeU32Data
`/coe_rd_u32/<cmd_idx>`	GET	Checks U32 read response	CoeU32Data

6.2. CIA402 Op mode Control

API Route	Method	Description	Payload

`/set_cia402_op_mode`	POST	Initiates CIA402 operation mode setting request	Cia402OpModeData
`/set_cia402_op_mode/<cmd_idx>`	GET	Checks CIA402 operation mode setting response	Cia402OpModeData

Note: All endpoints that have POST/GET pairs follow an asynchronous pattern where the operation is initiated with the POST endpoint and its completion can be checked with the GET endpoint.

7. Configuration File

7.1. Ethernet

Field	Type	Description
interface_name	String	Ethernet interface name
prioritize_usr_space_eth_drv	Boolean	If true, prioritize obtaining slave configuration from ESI file; otherwise use slave EEPROM SSI data

7.2. EtherCAT

Field	Type	Description	Unit
esi_file_path	String	ESI file path	*N/A*
prioritize_esi_config	Boolean	If true, prioritize obtaining slave configuration from ESI file; otherwise use slave EEPROM SSI data	*N/A*
cycle_time	32-bit unsigned integer	EtherCAT cycle time, only multiples of 250us is supported	[us]
shift_time	32-bit unsigned integer	DC master frame send to sync0 offset	% of cycle_time
rx_frame_timeout	32-bit unsigned integer	Receive frame timeout, only supported on Linux and MacOS	[us]

7.3. Move Parameters

Value	Field	Type	Unit
move_params	rotational	MoveProfile	deg
	linear	MoveProfile	mm

7.3.1. Move Profile Type Details

Value	Fields	Data Type	Unit
move_profile	max_vel	64-bit float	[unit/s]
	max_acc	64-bit float	[unit/s^2]
	max_dec	64-bit float	[unit/s^2]
	max_jrk	64-bit float	[unit/s^3]

7.4. Axes: Single axis example

Field	Type	Description	Unit
enco_offst	32-bit integer	Encoder offset	[Encoder tics]
gear_ratio	64-bit float	Gear Ratio	*N/A*
slv_idx	Unsigned integer	Slave index: zero indexed position of axis within the network	*N/A*
min_pos	64-bit float	Minimum angular position of the rotational motor	[Degrees]
max_pos	64-bit float	Maximum angular position of the rotational motor	[Degrees]

7.5. Groups

Field	Type	Description	Units
axs_idx_vec	Vector of unsigned integers of length 4	Axes indices	*N/A*
pos	Vector of 64-bit floating points of length 6	robot frame position relative to world frame, rot convension: alpha beta gamma: x-y′-z″ (intrinsic rotations) or z-y-x (extrinsic rotations)	[mm, mm, mm, deg, deg, deg]
world_limits_type	WorldLimitType	World Limit Enum	*N/A*
group_type	GroupType	Group Type Enum	*N/A*
end_eff	EndEffConfig	End effector digital output configuration	*N/A*

7.5.1. World Limit Enum Details

Value	Fields	Data Type	Unit
CylindricalZConfig	z_min	64-bit floating point	[mm]
	z_max	64-bit floating point	[mm]
	r_xy_max	64-bit floating point	[mm]

7.5.2. Group Type Enum Details

Value	Fields	Data Type	Unit
DeltaConfig	base_radius	64-bit floating point	[mm]
	base_offset	64-bit floating point	[mm]
	upper_arm_length	64-bit floating point	[mm]
	lower_arm_length	64-bit floating point	[mm]
	platform_offset	64-bit floating point	[mm]
	platform_radius	64-bit floating point	[mm]
	motor_angles	64-bit floating vector of length 3	[degrees]
	upper_arm_mass	64-bit floating point	[kg]
	lower_arm_mass	64-bit floating point	[kg]
	lower_plate_mass	64-bit floating point	[kg]
	upper_arm_cm_radius	64-bit floating point	[mm]
	upper_arm_cm_inertia	64-bit floating point	[Kg.mm^2]
SerialDHConfig	theta	Vector of size 7	[degrees]
	a	Vector of size 7	[degrees]
	d	Vector of size 7	[degrees]
	alpha	Vector of size 7	[degrees]
	joint_type	Vector of size 7	*N/A*
GantryConfig	*N/A*	*N/A*	*N/A*

7.5.3. End Effector Details

Value	Fields	Data Type
end_eff	engage	DigSignal
	eject	DigSignal

7.5.3.1. DigSignal Details

Value	Fields	Data Type	Unit
DigSignal	axs_idx	Unsigned Integer	*N/A*
	bit_off	8-bit unsigned integer	[bits]
	delay	64-bit floating point	[s]

7.6. Cameras

Field	Type	Description	Unit
id	String	camera id number	*N/A*
exposure_time	64-bit floating point	camera exposure time	[us]
pixel_size	64-bit floating point	camera pixel size	[um]
focal_length	64-bit floating point	lens focal length	[mm]
pos	64-bit float vector of size 6	camera frame position relative to world frame, rot convension: alpha beta gamma: x-y′-z″ (intrinsic rotations) or z-y-x (extrinsic rotations)	[mm, mm, mm, deg, deg, deg]

7.7. Conveyors

Field	Type	Description	Unit
width	64-bit floating point	conveyor width	[mm]
length	64-bit floating point	conveyor length	[mm]
cam_idx_vec	vector of unsigned integers	cameras used for tracking objects on this conveyor	*N/A*
pos	64-bit float vector of size 6	conveyor frame position relative to world frame, rot convension: alpha beta gamma: x-y′-z″ (intrinsic rotations) or z-y-x (extrinsic rotations)	[mm, mm, mm, deg, deg, deg]
descend_dist	64-bit floating point	descend/ascend distance in z direction in the conv frame during pickup	[mm]
end_eff_delay	64-bit floating point	end effector delay to fully engage	[s]

7.8. Parts

Field	Type	Description	Unit
width	64-bit floating point	width of the part	[mm]
length	64-bit floating point	length of the part	[mm]
height	64-bit floating point	height of the part	[mm]
mass	64-bit floating point	mass of the part	[kg]
inertia_zz	64-bit floating point	inertia about the center of mass in the z direction	[Kg.mm^2]
label	String	string, either “”, “alphabetic”, “alphanumeric”, or “text”, used to detect if 180 degree rotation is needed or not	*N/A*
max_linear_error	64-bit floating point	percentage of acceptable error in width or length when identifying parts	*N/A*

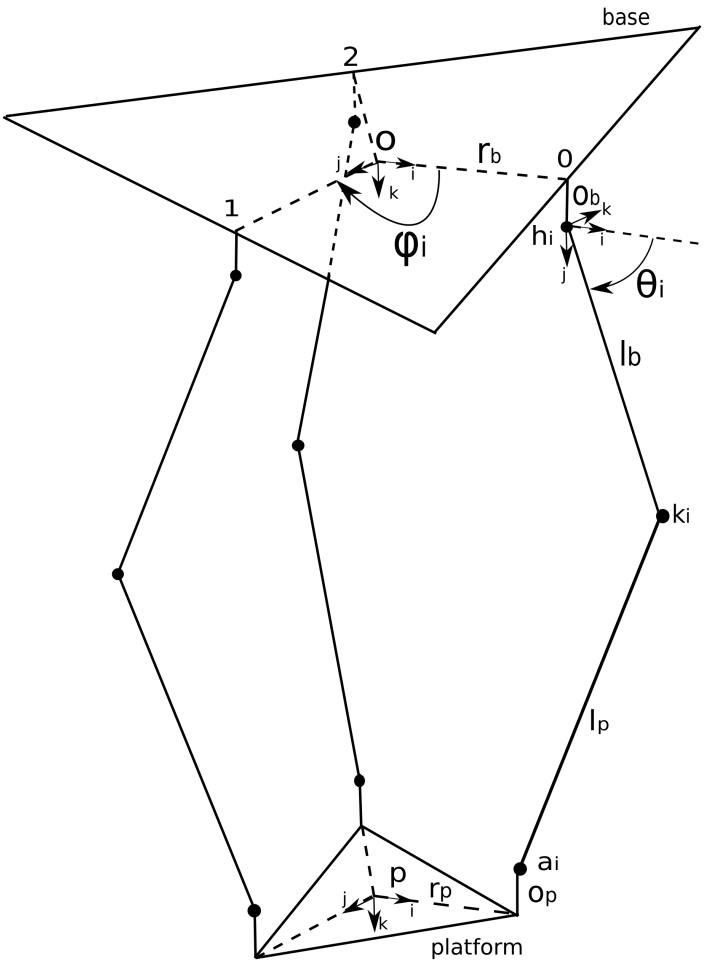
7.9. Cycles

Field	Type	Description	Unit
grp_idx	Unsigned Integer	Group index	*N/A*
cnv_idx	Unsigned Integer	Conveyor index	*N/A*

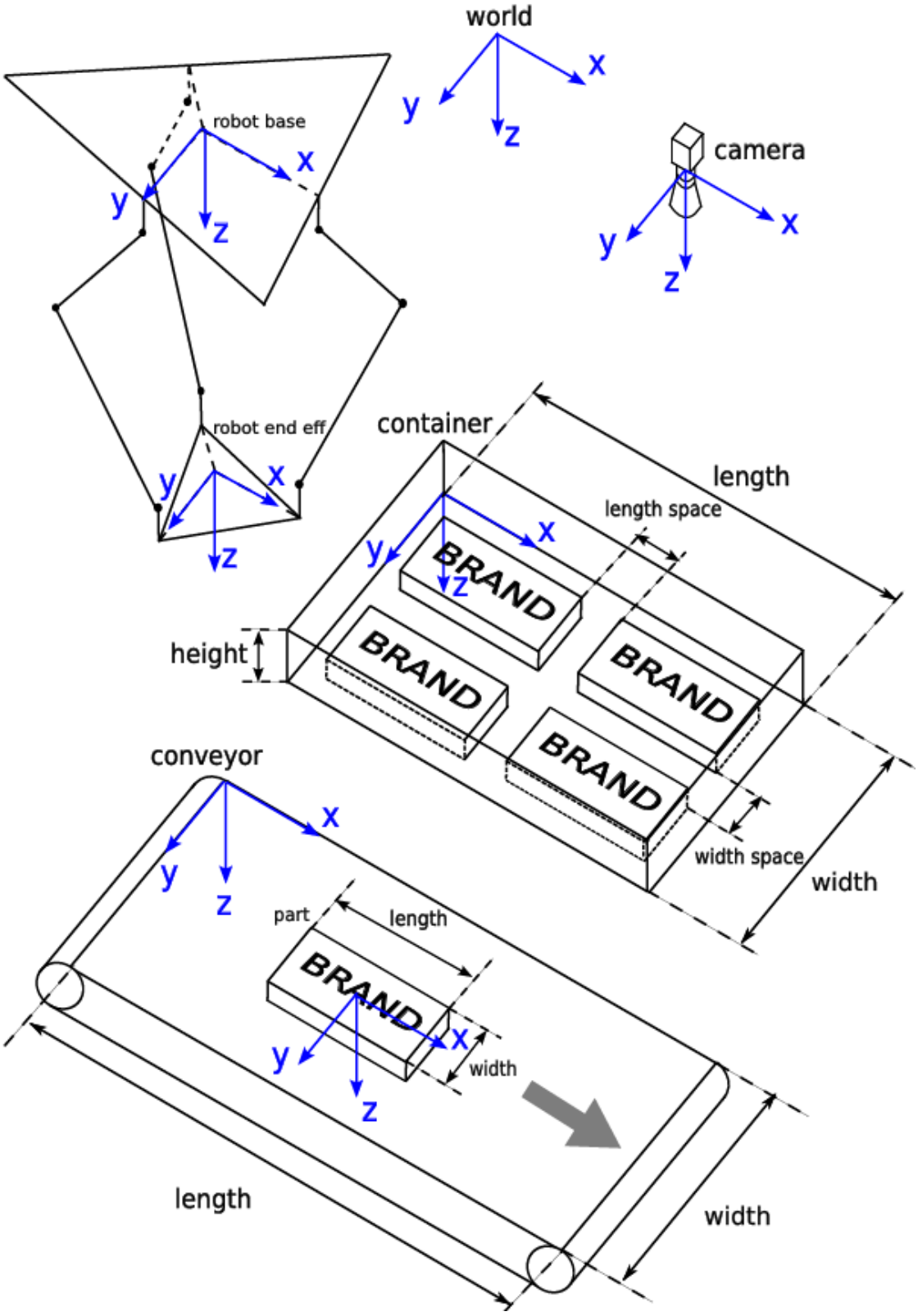
cnt_idx	Unsigned Integer	Container index	*N/A*
prt_idx	Unsigned Integer	Part index	*N/A*
prcn	8-bit unsigned integer	Motion speed, max value is 255	% max speed
cnv_pickup_timeout	64-bit floating point	How long a conveyor intercept or pickup call keeps attempting to find a part using the camera	[s]
cnv_spd	64-bit floating point	Conveyor speed	[mm/s]
cycle_type	!CnvToCnt OR !CntToCnv	Conveyor to container or container to conveyor	*N/A*
start_signal	DigSignal	Digital input signal for cycle to start	*N/A*
done_signal	DigSignal	Digital output signal when cycle is done	*N/A*

8. Diagram Quantities

Symbol	Description
φ_i	Motor Angle
θ_i	Joint Angle
r_b	Base Radius
O_b	Base Offset
l_b	Base Leg Length
l_p	Platform Leg Length
O_p	Platform Offset
r_p	Platform Radius



9. Workspace Coordinate System Defines



10. Appendix A: Payload Details

10.1. Config Save Payload Details

Payload	Fields	Description	Types	Units
ConfigSaveData	`config`	Configuration	Config	*N/A*
	`file_name`	Config file name	String	*N/A*

10.2. Slave Management Payload Details

Payload	Fields	Description	Types	Units

NumSlvsData	`num_slaves`	Number of slaves	Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

10.3. Axis Payload Details

Payload	Fields	Description	Types	Units
AxisData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisCurrPosData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`curr_pos`	Current position	64-bit floating point	[mm] or [deg]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer *See Appendix B: Motion result error codes.*	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisMovePosData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`end_pos`	End position	64-bit Floating Point	[mm] or [deg]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisMoveVelData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`end_vel`	End velocity	64-bit Floating Point	[mm/s] or [deg/s]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisMoveRelData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`rel_pos`	Relative position	64-bit Floating Point	[mm] or [deg]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisDigData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`bit_off`	Bit Offset	8-bit Unsigned Integer	[Bits]
	`val`	Value	Boolean	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisBoolData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`val`	Value	Boolean	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*

	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisAlarmsData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`alarms`	*See Appendix D: Motion alarm codes.*	Vector of 16-bit Unsigned Integers	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisSleepData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`t_tot`	Total time	64-bit float	[s]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisSpdPrcnData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`prcn`	Motion speed, max value is 255	8-bit Unsigned Integer	% max speed
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
AxisSpdData	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

10.4. Group Payload Details

Payload	Fields	Description	Types	Units
GroupData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupMovePosData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`end_pos`	End position	64-bit Floating Vector	[mm] or [deg]
	`coord_type`	Coordinate type: “Joint or World”	String	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupMoveRelData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`rel_pos`	Relative position	64-bit Floating Vector	[mm] or [deg]

	`coord_type`	Coordinate type: “Joint or World”	String	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupDigData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`axs_idx`	Axis index	Unsigned Integer	*N/A*
	`bit_off`	Bit offset	8-bit Unsigned Integer	[Bits]
	`val`	value	Boolean	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupBoolData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`val`	Value	Boolean	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupBgnPosData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`coord_type`	Coordinate type: “Joint or World”	String	*N/A*
	`bgn_pos`	Begin position	64-bit Floating Vector	[mm] or [deg]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupCurrPosData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`joint_pos`	Joint position	64-bit Floating Vector	[mm] or [deg]
	`world_pos`	World position	64-bit Floating Vector	[mm] or [deg]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupAlarmsData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`alarms`	*See Appendix D: Motion alarm codes.*	Vector of 16-bit Unsigned Integers	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupSleepData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`t_tot`	Total time	64-bit Floating Vector	[s]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*

	`done`	Request response	Boolean	*N/A*
GroupSpdPrcnData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`prcn`	Motion speed, max value is 255	8-bit Unsigned Integer	% max speed
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
MotnCmdData	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

10.5. Conveyor Payload Details

Payload	Fields	Description	Types	Units
ConvSpdData	`cnv_idx`	Conveyor index	Unsigned Integer	*N/A*
	`cnv_spd`	Conveyor speed	64-bit floating point	[m/s]
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupConvInterceptData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cnv_idx`	Conveyor index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`label`	Label	String	*N/A*
	`drop_rz`	`drop_rz` is in group frame and is only applicable to XYZRz groups	64-bit Floating Point	*N/A*
	`found`	Part foound on conveyor	Boolean	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupConvData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cnv_idx`	Conveyor index	Unsigned Integer	*N/A*
	`coord_type`	Coordinate type	String	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupConvPickupData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cnv_idx`	Conveyor index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`label`	Label	String	*N/A*
	`drop_rz`	`drop_rz` is in group frame and is only applicable to XYZRz groups	64-bit Floating Point	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupConvDropoffData	`grp_idx`	Group index	Unsigned Integer	*N/A*

	`cnv_idx`	Conveyor index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

10.6. Container Payload Details

Payload	Fields	Description	Types	Units
GroupContDropoffData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupContPickupData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`drop_rz`	`drop_rz` is in group frame and is only applicable to XYZRz groups	64-bit Floating Point	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupContPickupIdxData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`drop_rz`	`drop_rz` is in group frame and is only applicable to XYZRz groups	64-bit Floating Point	*N/A*
	`i_idx`	“i” index	Unsigned Integer	*N/A*
	`j_idx`	“j” index	Unsigned Integer	*N/A*
	`k_idx`	“j” index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
GroupContData	`grp_idx`	Group index	Unsigned Integer	*N/A*
	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
ContData	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
ContPartData	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*

	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
ContBoolData	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`val`	*N/A*	Boolean	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
ContPartBoolData	`cnt_idx`	Container index	Unsigned Integer	*N/A*
	`prt_idx`	Part index	Unsigned Integer	*N/A*
	`val`	*N/A*	Boolean	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix B: Motion result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

10.7. EtherCAT Command Payload Details

10.7.1. EtherCAT Command Payload Details

Payload	Fields	Description	Types	Units
EcatCmdData	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix C: ECAT result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

10.7.2. COE Payload Details

Payload	Fields	Description	Types	Units
CoeU8Data	`slv_idx`	Slave index	Unsigned Integer	*N/A*
	`index`	COE index value	16-bit Unsigned Integer	*N/A*
	`sub_index`	COE sub-index value	8-bit Unsigned Integer	*N/A*
	`val`	Read or write COE object value	8-bit Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix C: ECAT result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
CoeU16Data	`slv_idx`	Slave index	Unsigned Integer	*N/A*
	`index`	COE index value	16-bit Unsigned Integer	*N/A*
	`sub_index`	COE sub-index value	8-bit Unsigned Integer	*N/A*
	`val`	Read or write COE object value	16-bit Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix C: ECAT result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*
CoeU32Data	`slv_idx`	Slave index	Unsigned Integer	*N/A*
	`index`	COE index value	16-bit Unsigned Integer	*N/A*
	`sub_index`	COE sub-index value	8-bit Unsigned Integer	*N/A*
	`val`	Read or write COE object value	32-bit Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix C: ECAT result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

10.7.3. Cia402 Op mode Payload Details

Payload	Fields	Description	Types	Units
Cia402OpModeData	`slv_idx`	Slave index	Unsigned Integer	*N/A*
	`op_mode`	*N/A*	8-bit Unsigned Integer	*N/A*
	`cmd_idx`	Unique command index	64-bit Unsigned Integer	*N/A*
	`rslt`	*See Appendix C: ECAT result error codes.*	16-bit Unsigned Integer	*N/A*
	`done`	Request response	Boolean	*N/A*

11. Appendix B: Motion result error codes

Value	Description
0x0000	Success
0x1001	Initialization axes network error
0x1002	Invalid axis index
0x1003	Invalid group index
0x1004	Invalid conveyor index
0x1005	Invalid container index
0x1006	Invalid part index
0x1007	Invalid digital io bit offset
0x1008	Axis index not in group
0x1009	Motion commanded while not servoed on
0x100A	Joint position limits reached
0x100B	Joint velocity limits reached
0x100C	Joint torque limits reached
0x100D	World position limits reached
0x100E	World velocity limits reached
0x100F	World torque limits reached
0x1010	Small or zero length trajectory append dropped
0x1011	Servo off commanded while motion running
0x1012	Axis and group command traj attempting to move the same axis
0x1013	Conveyor part not found
0x1014	Conveyor speed not initialized
0x1015	Conveyor speed exceeds group speed, it is not possible for the robot to catch up with or to track the conveyor
0x1016	Conveyor intercept solution did not converge, often due to conveyor running too fast relative to the robot
0x1017	Conveyor vert traj failed, max x vel/acc required during move exceeded
0x1018	Move append while an alarm present
0x1019	Move append while a vel move is running
0x101A	Move append joint traj to a world trajectory on a group that has non unique inv kin solution, is not allowed
0x101B	Quick stop append traj failed, quick stop already running
0x101C	Motion command mismatch
0x101D	Forward kinematics unreachable
0x101E	Inverse kinematics non convergent
0x101F	Inverse kinematics unreachable
0x1020	Inverse kinematics singularity
0x1021	No end effector config available

0x1022	Container was being filled by parts of one type, and then switched to another part type before getting full
0x1023	A dropoff with auto fill was made while the container is already full
0x1024	A pickup with auto empty was made while the container is already empty
0x1025	Container fill index invalid
0x1026	Speed percent can't be zero
0x1027	Emergency stop pressed
0x1FFF	Unspecified motion error

12. Appendix C: ECAT result error codes

Value	Description
0x0000	Success
0x0001	Invalid slave index
0x0002	Uninitialized slave index
0x0003	Uninitialized station address
0x0004	Working counter not incremental
0x0005	Frame receive time out
0x0006	Frame command mismatch
0x0007	ESI file not found
0x0008	SM data cat section not found in SSI
0x0009	Register response mismatch
0x000A	SSI maximum byte length reached
0x000B	Max retries reached, while polling coe sm1 response mbox ready status
0x000C	CoE mbox emergency
0x000D	CoE mbox abort code
0x000E	Cia operational mode maximum retries reached
0x000F	Ecat command mismatch

13. Appendix D: Motion alarm codes

Alarms should normally not occur. If they do occure, they are typically because serial DH chain inverse kinematics either encountered an error or the robot joint pos/vel limits were reached during motion. When an alarm occures, a quick stop motion command is automatically generated, in order to safely stop all motion. Additionally, no motion or io commands may be done before an alarm clear command is made. Note that alarm clear commands clear all queued motion and io commands. Any of the error codes in appendix B, my correspond to an alarm. Alarm codes equal 0x1000 plus the value of the corresponding error code. For example, if joint velocity limits are reached, while a 7 axis robotic arm is running motion, motion will stop, and an alarm code 0x200B will be generated. Note that axis and group alarms are treated entirely separately. For example, an axis will not necessarily have an alarm, even if it belongs to a group of axes and that group does have an alarm.