EWELLIX

A Schaeffler Company

INSTALLATION, OPERATION AND MAINTENANCE MANUAL

Linear axis for any collaborative robots SLIDEKIT 2.0 0S (Ethernet TCP/IP)

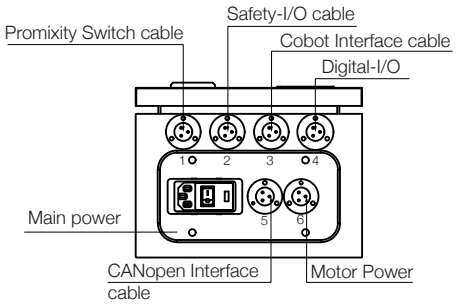
Quick start guide

For more details on each step, read the full instructions

1. Mount LIFTKIT or Robot on SLIDEKIT

Note: The top plate for the robot is not included by default in the package and needs to be ordered separately as an accessory.

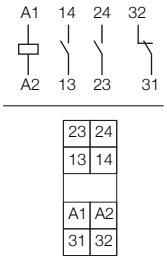
2. Matching number of connecting cables and connectors



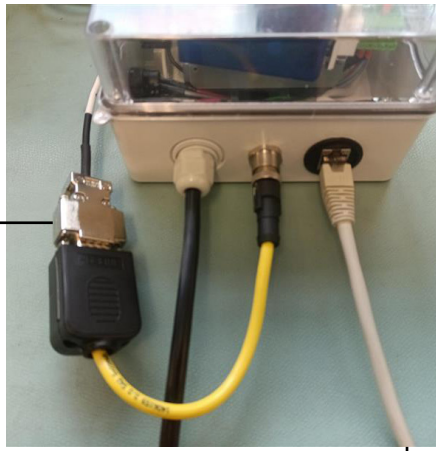
The SLIDEKIT2.0 control box has two integrated safety relay 7S.23 from Finder with forcibly guided contacts.

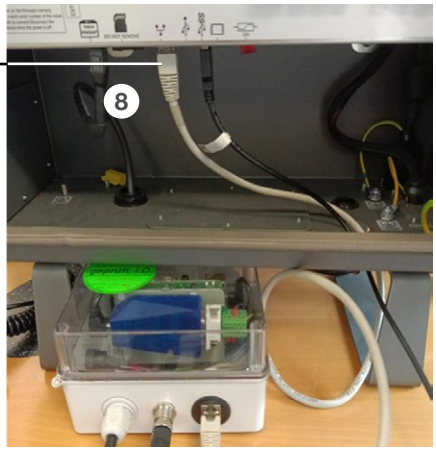
Safety I/O connection

Pin No.	Name	Color
1	SR1 + 24V	Red
2	SR1 GND	White
3	SR2 + 24V	Black
4	SR2 GND	Green
5	N.C.	N.C.



Their coil and feedback contacts are wired directly to the I/O connector.





Contents

1. General information 4
1.1 Information in this manual 4
1.2 Explanation of symbols and signal words 4
1.3 SLIDEKIT designations 5
1.4 Related documents 5
1.5 Target audience 5

2. Safety 6
2.1 Intended use 6
2.2 Functional safety 6
2.3 Safety mechanisms 6
2.4 Application notes 6
2.5 Potential risks 6

3. SLIDEKIT components 7
3.1 Scope of delivery 7
3.2 System requirements 7

4. Mechanical installation 8
4.1 Tools required 8
4.2 Robot installation on the linear module 8
4.3 Lubrication intervals 8
4.4 Dimensional drawing 9

5. Electrical connection 11

6. Software operation PiBox Stargate script commands instruction 12
6.1 SLIDEKIT Installation 12
6.1.1 Safety I/O setup on robot controller 12
6.1.2 Communication 12
6.1.3 SLIDEKIT setup mover Ethernet TCP/IP script commands 12

7. PiBox Ethernet TCP/IP script commend library 13

8. PiBox IP address setting 14

9. Software update PiBox 16

10. PiBox API Programmer's Manual 17
10.1 SLIDEKIT States (Statuses) 17
10.1.1 Initialized 17
10.1.2 Connected 17
10.1.3 Homing active ready 18
10.1.4 Homing active moving 18
10.1.5 Ready 18
10.1.6 Moving 18
10.2 SLIDEKIT command acknowledges 19
10.2.1 OK 19
10.2.2 OOR – Out of range 19
10.2.3 NF – Not found 19
10.2.4 NA – Not allowed 19
10.2.5 WNP – Wrong number of parameters 19
10.2.6 VE – Value Error 19
10.2.7 EF – Execution Failed 19
10.3 SLIDEKIT stargate commands 19
10.3.1 Move to absolute position 19
10.3.2 Get available types 20
10.3.3 Get Type 21
10.3.4 Set Type 21
10.3.5 Get Velocity 21
10.3.6 Set Velocity 22
10.3.7 Get Acceleration 22
10.3.8 Set Acceleration 22
10.3.9 Get Deceleration 23
10.3.10 Set Deceleration 23
10.3.11 Get Position 24
10.3.12 Get Stroke 24
10.3.13 Get Status 24
10.3.14 Stop Moving 25
10.3.15 Get Motion Profile Parameters 25
10.3.16 Set Motion Profile Parameters 26
10.3.17 Start Homing 27
10.3.18 Stop Homing 27
10.3.19 Set Virtual limits 28
10.3.20 Get Virtual Limits 28

11. Appendix 29
11.1 Controller 29
11.2 Limit switch 29

⚠ WARNING

Read this manual before installing, operating or maintaining this device. Failure to follow safety precautions and instructions could cause linear module failure and result in serious injury, death or property damage

1. General information

These installation instructions describe the setup and operation of SLIDEKIT, a horizontal sliding axis for collaborative robots.

1.1 Information in this manual

This manual provides important information on how to work with the product (also called device or drive) safely and efficiently.

The manual is part of the device, must always be kept in the device's direct proximity and should be available for personnel to read at any time. All personnel working with the device must read and understand this manual before starting any work. Strict compliance with all specified safety notes and instructions is a basic requirement for safety at work.

Moreover, the accident prevention guidelines and general safety regulations applicable at the place of use of the device must also be complied with.

For a better representation of the circumstance of use, the illustrations used are not necessarily to scale and may vary from the actual design of the device.

1.2 Explanation of symbols and signal words

Safety precautions

Safety precautions are identified by symbols and signal words as shown to the right. The signal words indicate the severity of the hazard and the chance it could occur.

Follow these safety precautions and act cautiously in order to avoid accidents, personal injury and damage to property.

⚠ DANGER

Indicates a dangerous situation, which will lead to death or serious personal injury, if the precautionary measures are ignored.

⚠ WARNING

Indicates a dangerous situation, which can lead to minor or moderate injury or property damage, if the precautionary measures are ignored.

⚠ CAUTION

Indicates a dangerous situation, which can lead to minor or moderate injury, if the precautionary measures are ignored.

NOTICE

Indicates information considered important, but not hazard-related (e.g. messages relating to property damage).

NOTE

Emphasizes useful hints and recommendations as well as information for efficient and trouble-free operation.

1.3 SLIDEKIT designations

SLIDEKIT contains a Linear module, a controller and additional accessories enabling easy integration with a collaborative robot. Different product configurations are available, according to the product ordering key.

Ordering key

SLIDEKIT - [ ][ ] - [ ]E - [ ][ ][ ][ ] - [ ][ ] - SFM - S[ ][ ]

Robot
- 00 SLIDEKIT mechnical part without motor
- 0S any robots (Ethernet TCP/IP)

Module options

Drive
- B Ball screw (lead 20)
- P Belt (width 40)

E Cover Aluminum and External motor attachment

Stroke

100 ... 3 000
- 1 000 Preferred range Ball screw
- 1 800 Preferred range Ball screw
- 2 500 Preferred range Belt
- 3 000 Preferred range Belt

Electrical options
- 11 120 V AC / US cable
- 22 230 V AC / EU cable
- 23 230 V AC / CN cable
- 24 230 V AC / UK cable
- 25 230 V AC / CH cable

Accessories options

S Limit switch

Cableveyor
- F High Flex cable [1)]

M Standard hole pattern

Customized options

S Option 1 - Safety relay

Robot compatibility
- 00 Small & medium robot size weight less than 50 kg with 20 kg max payload
- 20 Large robot size weight 50 to 75 kg with 30 kg max payload

[1)] The bending radius increased to comply with cobot manufacturers' requirements

1.4 Related documents

This instruction manual does not replace the operating manuals of the included products but adds additional instructions relevant to the setup and operation of the SLIDEKIT system related to collaborative robots.

For general information and safety instructions please refer to installation, operation and maintenance manuals available at www.ewellix.com

- CLSM Linear module
- Linear modules - CLSM Installation, operation and maintenance manual.

1.5 Target audience

This manual is intended for technical personnel and authorized users who use and install SLIDEKIT in their application. This manual and the corresponding operating manuals should always be kept available for reference.

Qualified personnel can carry out assigned work and recognize and prevent possible dangers self-reliantly due to their professional training, knowledge and experience as well as profound knowledge of applicable regulations.

2. Safety

This section provides some safety aspects supplementary to the safety aspects described in the relevant operating manuals of the included devices. Failure to comply with the guidelines and safety instructions contained in this manual may result in serious hazards that could cause possible serious injury or death or damage to the device or equipment.

The listed safety aspects must be reviewed and taken into account in the final application risk assessment prior to the use of SLIDEKIT.

2.1 Intended use

SLIDEKIT has been designed and built for the intended use as described in the operating manual of the linear module, with additional intended use defined as sliding of a robot to extend its operating range in an industrial environment.

Any use that extends beyond the intended use or a use different than the one described above is deemed misuse.

Any type of claims resulting from damage caused by misuse are excluded.

2.2 Functional safety

The SLIDEKIT system is not a functional safety system compliant with EN ISO 13489-1 or IEC 62061. To integrate the SLIDEKIT into a functional safety chain, external safety devices must be added to the overall system

2.3 Safety mechanisms

The following measure have been integrated in SLIDEKIT to reduce the risk of harm or damage

- Pinching risk between the carriage and the end block of SLIDEKIT is minimized.
- The SLIDEKIT controller must be connected to the robot controller safety IO to operate. Activation of the emergency stop will trigger a stop of the controller via 2 safety relays, certified ISO 13849-1. If the robot system is turned off, SLIDEKIT cannot be operated.
- The SLIDEKIT controller checks the CANopen connection to the robot controller. If this connection is lost, the linear module movement is automatically stopped.
- Stopping or failure of the robot controller software send a stop signal to the SLIDEKIT controller.

2.4 Application notes

- Integration with an emergency-stop is required for its intended use.
- Install emergency stop functions for the linear module and integrate them into the safety chain of the complete system prior to operating SLIDEKIT.
- The emergency stop function must be connected in such a way that a disruption of the power supply or the activation of the power supply after a power disruption cannot cause a hazardous situation for persons and objects.
- The emergency-stop systems must always be freely accessible.
- To integrate SLIDEKIT into a functional safety system with a STO (Safe Torque Off) safe condition, an external safety relay must be connected to the SLIDEKIT controller power supply, triggered by a functional safety function, such as the robot safety IO.

2.5 Potential risks

The following risks during the SLIDEKIT operation must be considered in an application specific risk assessment

- SLIDEKIT does not detect an impact automatically and does not stop movement upon impact.
  This can lead to:
  - Crushing of a person or an object in the path of the linear module, causing significant harm
  - Dynamic impact to a person or an object causing significant harm
- The SLIDEKIT movement does not stop at the desired position and the robot control software does not recognize this
  - Movement of the robot can occur at a different position than intended, causing significant harm or damage.

3. SLIDEKIT components

3.1 Scope of delivery

1. 1 Linear module (with motor)
2. 1 attachment plate compatible with LIFTKIT. The Robot plate can be ordered upon request.
3. 1 SLIDEKIT Control Box
4. PiBox Ethernet TCP/IP module
5. 2 Main power cables (1.5m)
6. 1 Motor Power cable (3 m) [1]
7. 1 CANopen Interface cable (3 m) [1]
8. 1 Promixity Switch #1, #2 cable (3 m)
9. 1 Safety-I/O cable (3 m) [1]
10. 1 Cobot Interface cable (3 m) [1]
11. 1 Digital – I/O Interface cable (3 m) [2]
12. 1 M12-DB9 adapter cable (15 cm)
13. Cableveyor
14. 2 limit switch with sockets

[1] Other cable lengths available on request.
[2] Options

NOTE

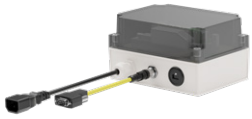
The robot top plate is not included in the package and can be offered on request as an accessory according to the customer's robot

3.2 System requirements

- SLIDEKIT 0S S00: Suitable for small and medium-sized cobots weighing up to 50 kg.
- SLIDEKIT 0S S20: Suitable for large -sized cobots weighing up to 75 kg.
- Robot controller: Supports and is compatible with the Ethernet TCP/IP communication protocol and has an available RJ45 port.
- Power input capacity (at rated load): 120 to 230 V AC, 0.9 K VA.

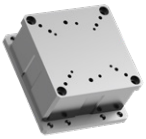
Figure 1

SLIDEKIT 2.0 0S scope of delivery



PiBox including stargate platform

CLSM Linear module



Robot attachment plate available on request, tailored to the customer's robot

SLIDEKIT Control Box

CAN

D-SUB 9Pin

Digital IO

Motor Power

Proximity Limit Switches

4. Mechanical installation

4.1 Tools required

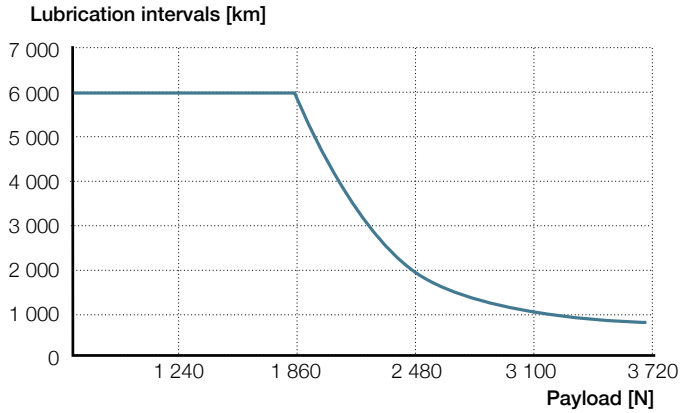
- 5 mm & 6 mm wrench

4.2 Robot installation on the linear module

1. Take SLIDEKIT out of the box
2. For mounting to SLIDEKIT, attach the bottom plate with M6 bolts (not included) to the SLIDEKIT base plate. SLIDEKIT should be supported by its entire length or at least every 300 mm by clamping unit or bolt.
3. Mount the robot on a customized top plate designed specifically for the customer's robot
4. Align the robot or SLIDEKIT with the alignment pins and fix the robot base with the four bolts provided or SLIDEKIT base with the eight bolts provided.

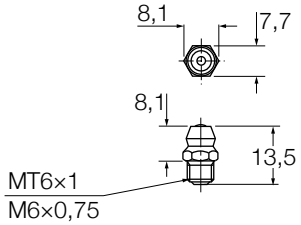
4.3 Lubrication intervals

Under normal operating condition, v≤ 1 m/s, travel under Fm ≤ 0,3 c. Inject grease 4,08 cm2 according to condition as shown in the graph below.



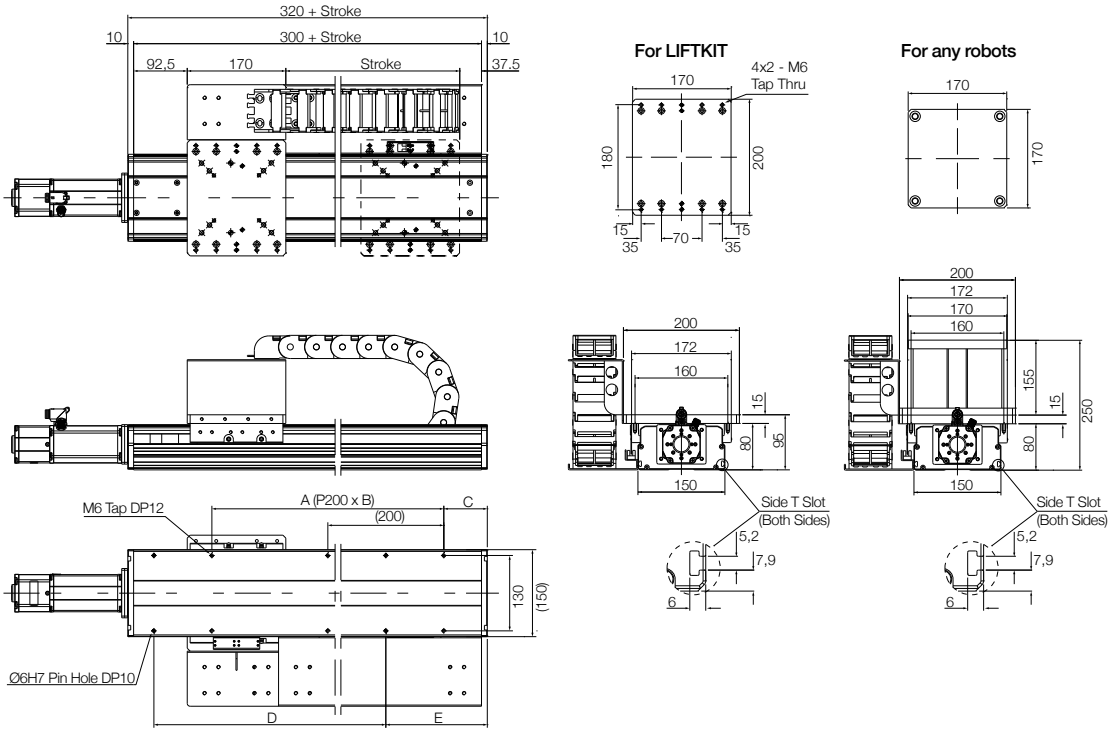
Lubrication can be performed via the dedicated grease nipple (**figure 1**)

Figure 2



4.4 Dimensional drawing

SLIDEKIT 0S BE S00 - Ball Screw version for any robots



	Stroke (mm)	A	B	C	D	E
1	100	200	1	75	200	175
2	200	400	2	25		125
3	300	400	2	75	400	175
4	400	600	3	25		125
5	500	600	3	75	600	175
6	600	800	4	25		125
7	700	800	4	75	800	175
8	800	1 000	5	25		125
9	900	1 000	5	75	1 000	175
10	**1 000**	**1 200**	**6**	**25**		**125**
11	1 100	1 200	6	75	1 200	175
12	1 200	1 400	7	25		125
13	1 300	1 400	7	75	1 400	175
14	1 400	1 600	8	25		125
15	1 500	1 600	8	75	1 600	175
16	1 600	1 800	9	25		125
17	1 700	1 800	9	75	1 800	175
18	**1 800**	**2 000**	**10**	**25**		**125**

Small & medium size robot compatibility for SLIDEKIT 2.0 0S S00

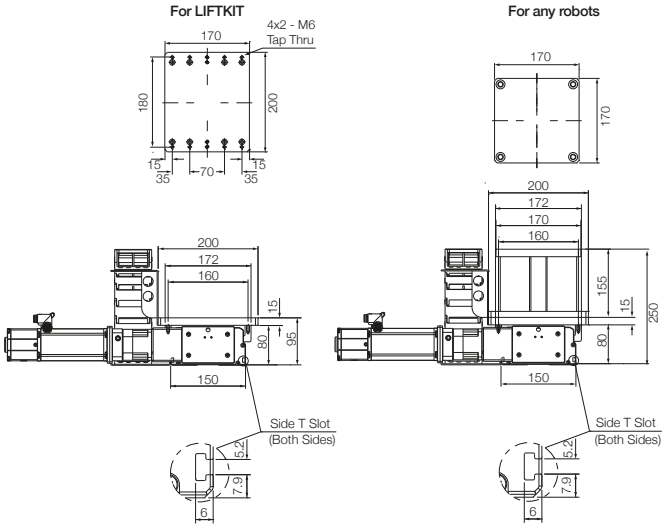
	UR	Fanuc	Yaskawa	TM	Doosan	ABB
Top plate item	on request	on request	on request	on request	on request	on request
Size S00	UR3	CRX5	HC10	TM5	A0509/s	CRB 15000-5 15000-5
	UR5	CRX10		TM12	M0609	4 CRB 15000-10
	UR10	CRX20		TM14	M1509	4 CRB 15000-12
	UR16			TM20	A0912/s M1013 M0617	

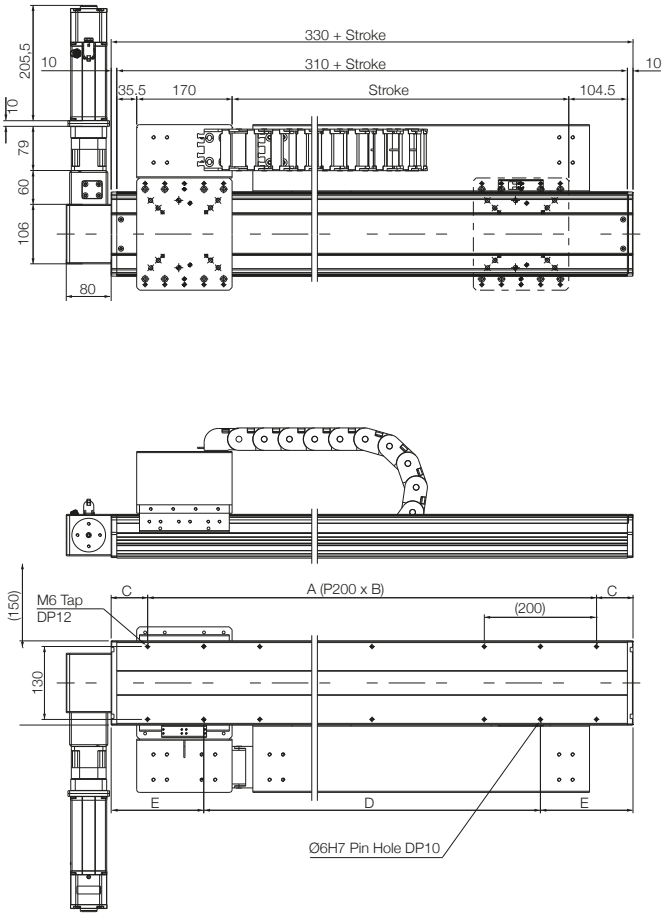
NOTE:

A robot plate to mount the robot onto the SLIDEKIT can be ordered on request as an accessory item; see the robot plate item list.

Standard stroke

SLIDEKIT 0S PE S00 - Belt version for any robots





	Stroke (mm)	A	B	C	D	E
10	1 000	1 200	6	65	1 000	165
11	1 100	1 200	6	115	1 000	215
12	1 200	1 400	7	65	1 200	165
13	1 300	1 400	7	115	1 200	215
14	1 400	1 600	8	65	1 400	165
15	1 500	1 600	8	115	1 400	215
16	1 600	1 800	9	65	1 600	165
17	1 700	1 800	9	115	1 600	215
18	1 800	2 000	10	65	1 800	165
19	1 900	2 000	10	115	1 800	215
20	2 000	2 200	11	65	2 000	165
21	2 100	2 200	11	115	2 000	215
22	2200	2 400	12	65	2 200	165
23	2 300	2 400	12	115	2 200	215
24	2 400	2 600	13	65	2 400	165
25	**2 500**	**2 600**	**13**	**115**	**2 400**	**215**
26	2 600	2 800	14	65	2 600	165
27	2 700	2 800	14	115	2 600	215
28	2 800	3 000	15	65	2 800	165
29	2 900	3 000	15	115	2 800	215
30	**3 000**	**3 200**	**16**	**65**	**3 000**	**165**

NOTE:

A robot plate to mount the robot onto the SLIDEKIT can be ordered on request as an accessory item; see the robot plate item list.

Standard stroke

Small & medium size robot compatibility for SLIDEKIT 2.0 0S S00

	UR	Fanuc	Yaskawa	TM	Doosan	ABB
Top plate item	on request	on request	on request	on request	on request	on request
Size S00	UR3	CRX5	HC10	TM5	A0509/s	CRB 15000-5 15000-5
	UR5	CRX10		TM12	M0609	4 CRB 15000-10
	UR10	CRX20		TM14	M1509	4 CRB 15000-12
	UR16			TM20	A0912/s M1013 M0617	

5. Electrical connection

1. Connect the main power cable (**cable 7**) to the SLIDEKIT controller in connection MAIN POWER (**7**).
2. Connect the two connectors of the motor power (**cable 6**) and CANopen interface (**cable 5**) cables in connector (**6**), (**5**) of the SLIDEKIT controller.
3. Digital-I/O cable (**cable 4**) is a cable required for external I/O control, which is connected to the in connector (**4**) (use only for SLIDEKIT-00 version).
4. Connect the communication interface (**3**) to the PiBox Ethernet TCP/IP module (**8**) via the DB9 connector.
5. Connect the safety-I/O cable (**cable 2**) in connector (**2**).
6. Connect the promiity switch cable (**cable 1**) in connection (**1**).
7. Attach the safety-I/O cable (**cable 2**) to a safety DO on the robot controller.
8. Connect the PiBox Ethernet TCP/IP module (**8**) to the robot controller (**9**) using an RJ45 cable.
9. Connect the main power cable (**cable 10**) of the PiBox.

NOTE

The safety I/O has to be configured in the Safety menu of the robot controller or PLC. Follow the instructions in chapter 6.1.1 Safety I/O setup on robot controller

NOTE

During wiring, please ensure that both the SLIDEKIT control box and the PiBox are not powered.

Figure 3

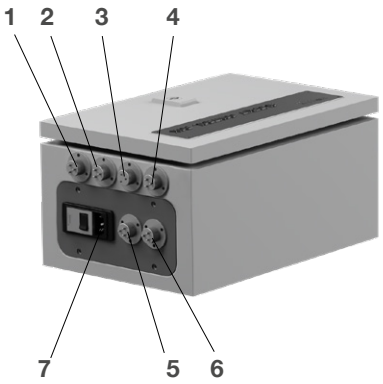


Figure 4

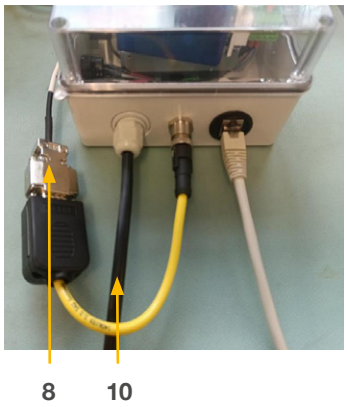


Figure 5

1. Promixity switch #1, #2 cable
2. Safety-I/O cable
3. Communication interface to PiBox
4. Digital-I/O cable
5. CANopen interface cable of the servo-motor
6. Motor power cable
7. Main power cable
8. PiBox Ethernet TCP/IP
9. RJ45 port on robot controller
10. PiBox power cable

6. Software operation PiBox Stargate script commands instruction

The SLIDEKIT 2.0 0S includes a PiBox (Ethernet TCP/IP module). This module enables control of the SLIDEKIT over Ethernet TCP/IP with script commands described in **Chapter 8**. The IP address of the PiBox is 192.168.1.100, and this IP address can be changed using the specific procedure in **Chapter 9**

6.1 SLIDEKIT Installation

The SLIDEKIT installation screen allows to set up three aspects of the SLIDEKIT operation:

1. Safty I/O the Robot safety or PLC I/O
2. Communication
3. SLIDEKIT Setup over Ethernet TCP/IP script commends
4. SLIDEKIT motion (absolute position, velocity, acceleration) over Ethernet TCP/IP commends
5. SLIDEKIT status over Ethernet TCP/IP script commmends

NOTE

Save the installation file to keep the values after restart.

6.1.1 Safety I/O setup on robot controller

To activate the safety I/O of the SLIDEKIT control box, cable the connector of cable n°2 needs to be connected to the Robot safety I/O or PLC safety I/O. This output must be 24 VDC to activate the safety relay, which will power on the SLIDEKIT. If the I/O is not connected to the robot or PLC output and activated, it will not be possible to drive the SLIDEKIT, as the safety relay is normally open. Therefore, you need to activate it with the robot or PLC output using a 24 VDC signal.

6.1.2 Communication

- The communication between the SLIDEKIT and the robot or external PLC is managed by Ethernet TCP/IP communication over the RJ45 cable that need to be connected to the robot RJ45 port and PiBox (Ethernet TCP/IP module) port.
- The communication between the SLIDEKIT and the robot or external PLC is managed via Ethernet TCP/IP over an RJ45 cable, which needs to be connected to the robot's RJ45 port and the PiBox (Ethernet TCP/IP module) port. Please ensure that the PiBox (Ethernet TCP/IP module) is powered and the blue box inside is blinking green. If not, please check the connection.

6.1.3 SLIDEKIT setup mover Ethernet TCP/IP script commands

We always recommend starting by checking the status of the SLIDEKIT with the get_status command. If the outcome is not "ready," proceed with the following steps:

Step 1

Check the type using get_typesAvailable.

Then, select the type according to your SLIDEKIT using the script command set_type:

SLIDEKIT 0S BE S00: Please select SK 20x20 BG65S

SLIDEKIT 0S PE S00: Please select SK BELT 40 BG65X

SLIDEKIT 0S PE S20: Please select SK BELT 40 BG65S-C

Step 2

Perform homing by using the start_homing command

Step 3

The virtual limits of the SLIDEKIT need to be set using the command set_virtualLimits,<*Lower limit*>,<*Upper limit*> . The *Lower limit* should generally be 0, and the *Upper limit* should generally be the value returned by the "get_stroke" command. If the value returned by get_stroke is notably different than expected then there may have been a problem during the homing and the homing should be done again

After that, we recommend rechecking the status using the script command get_status and ensuring that the response is "ready".

7. PiBox Ethernet TCP/IP script commend library

Command	Description
moveTo_absolutePosition	Moves SLIDEKIT to specified position
get_typesAvailable	Returns all supported SLIDEKIT types
get_type	Returns current SLIDEKIT type
set_type	Sets new SLIDEKIT type
get_velocity	Returns velocity stored in motion profile parameters
set_velocity	Sets SLIDEKIT velocity
get_acceleration	Returns acceleration stored in motion profile parameters
set_acceleration	Sets SLIDEKIT acceleration
get_deceleration	Returns deceleration stored in motion profile parameters
set_deceleration	Sets SLIDEKIT deceleration
get_position	Returns SLIDEKIT current position
get_stroke	Returns maximum stroke in mm
get_status	Returns SLIDEKIT's current state
stop_moving	Stops SLIDEKIT movement
get_motionProfileParameters	Returns motion profile parameters
set_motionProfileParameters	Sets motion profile parameters
start_homing	Starts SLIDEKIT homing
stop_homing	Stops SLIDEKIT homing
set_virtualLimits	Sets SLIDEKIT virtual limits
get_virtualLimits	Returns SLIDEKIT virtual limits

NOTE:

The SLIDEKIT 2.0 is not a functional safety system compliant with EN ISO 13489-1 or IEC 62061. To integrate the SLIDEKIT 2.0 into a functional safety chain, external safety devices have to be integrated into the overall system.

8. PiBox IP address setting

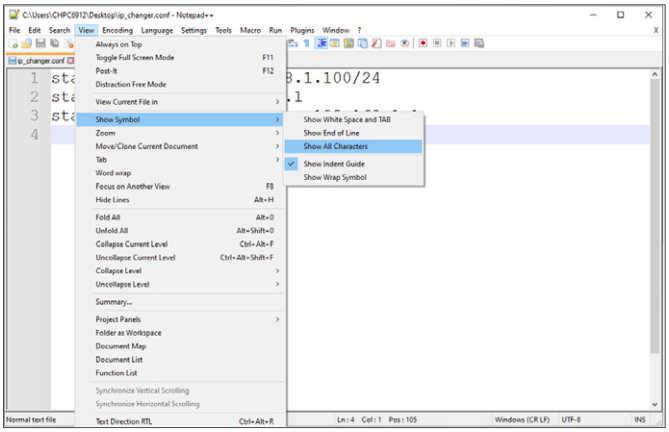
The PiBox uses a static IP address. The default address is 192.168.1.100.

If you need to set a different IP address, please follow the steps listed below:

1. Create a file called *ip_changer.conf* on your PC. We recommend to use the freeware Notepad++ or similar.
2. Insert the following content:
   static ip_address=192.168.1.100/24
   static routers=192.168.1.1
   static domain_name_servers=192.168.1.1
3. Change the addresses to your needs. Make sure that the */24* stays behind the static IP address.
4. Make all characters visible (see **figure 6**).

Figure 6

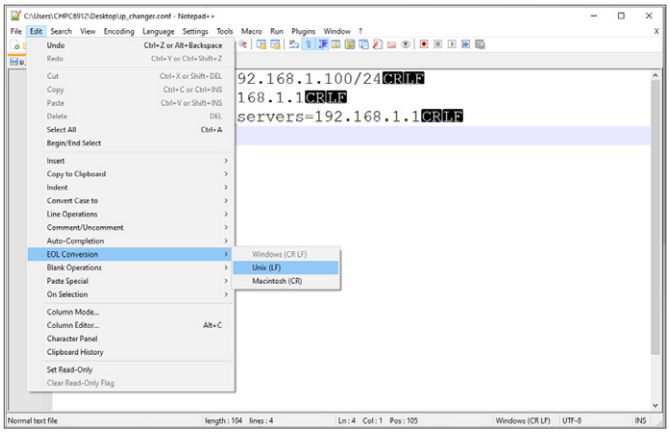
Show all characters



5. Convert the End Of Line into Unix (LF) (see **figure 7**).

Figure 7

EOL conversion to UNIX (LF) format



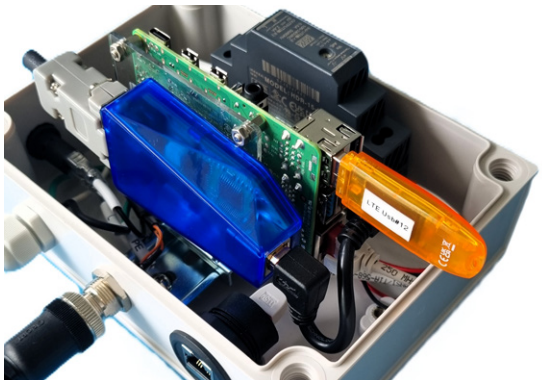
6. Save the file on a USB memory stick previously formatted in FAT32.

Procedure available only for PiBox:

6. Switch off the PiBox with the power bottom remove all the cables attached.
7. open PiBox with a screwdriver.r
8. Insert the USB memory stick into any available USB port of the PiBox controller (see **figure 8**).

Figure 8

Inside view on PiBox



9. Plug in the power cable into the PiBox.
10. Switch on the PiBox with power bottom.
11. Wait for 5 Minutes.
12. Switch off the PiBox .
13. Remove all cables from the PiBox .
14. Remove the USB memory stick.
15. Replace and secure the PiBox cover.
16. The following empty file will be created on the USBmemory stick to confirm the IP address change has been successful:

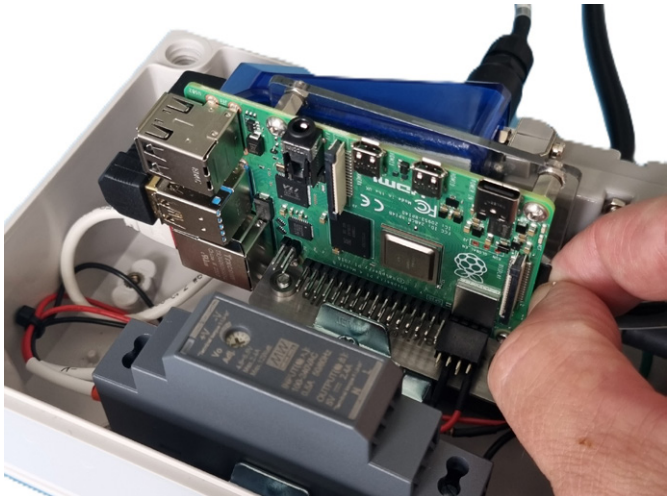
    update_ip_address_successfull_from_"Name of the USB Stick"

9. Software update PiBox

Software updates can be done by flashing a new image to the controller SD card.

1. Switch off the PiBox with the power bottom.
2. Open the PiBox and remove the microSD card (**figure 9**).

Figure 9



3. Download and install one of these tools:
   - Raspberry imager, from https://www.raspberrypi.org/downloads/
   - balenaEtcher, from https://www.balena.io/etcher/
4. Copy Image on SD card:
   - **4.1** Place SD card into your laptop
   - **4.2** Do not format SD card
   - **4.3** Start Raspberry imager or balenaEtcher
   - **4.4** Choose Image
   - **4.5** Select SD Card
   - **4.6** Start writing process
5. Insert the SD card back into the Pi and replace the PiBox cover.

10. PiBox API Programmer's Manual

10.1 SLIDEKIT States (Statuses)

Based on SLIDEKIT state, Stargate returns a status, which describes current SLIDEKIT state. Status can be checked by sending "get_status" command, see below "**10.3.13. Get Status**". Some statuses also have additional info, which gives a better explanation of SLIDEKIT state.

10.1.1 Initialized

Description: Stargate is started and initialized. Additional info describes why SLIDEKIT is not ready for movement.

Additional info:

1. NOT_CONNECTED_TO_SLIDEKIT – SLIDEKIT is not connected to the Stargate.

   Get Status Response: "get_status,OK,INITIALIZED,NOT_CONNECTED_TO_SLIDEKIT

2. CONNECTING – Connection establishing in progress. This info is usually transient and very short with exception of the power cut of SLIDEKIT when it is persistent until some event occurs that causes state/additional info changes.

   Get Status Respons:
   "get_status,OK,INITIALIZED,CONNECTING"

3. KEEP_ALIVE_FAILED_OR_POWER_CUT – Transient state during state change from CONNECTED to INITIALIZED.

   Get Status Response: "get_status,OK,INITIALIZED,-KEEP_ALIVE_FAILED_OR_POWER_CUT

10.1.2 Connected

Description: Connection between Stargate and SLIDEKIT is established.

Additional info:

1. TYPE_NOT_SET – SLIDEKIT type is not selected

   Get Status Response:
   "get_status,OK,CONNECTED,TYPE_NOT_SET"

2. HOMING_INTERRUPTED_OR_NOT_STARTED – SLIDEKIT homing was not started

   Get Status Response: "get_status,OK,CONNECTED,HOMING_INTERRUPTED_OR_NOT_STARTED"

3. HOMING_INTERRUPTED_BY_REQUEST – Homing was initiated but not finished successfully because it was stopped by client using "stop_homing"

Get Status Response: "get_status,OK,CONNECTED,HOMING_INTERRUPTED_BY_REQUEST"

If homing is initiated but not finished successfully, the following additional information may be seen:

a. HOMING_ATTAINED_TR_NOT_REACHED

Get Status Response:

"get_status,OK,CONNECTED,HOMING_ATTAINED_TR_NOT_REACHED"

b. HOMING_ERROR_VELOCITY_IS_NOT_ZERO – the velocity during homing was not zero when it should have been stopped

Get Status Response:

"get_status,OK,CONNECTED,HOMING_ERROR_VELOCITY_IS_NOT_ZERO"

c. HOMING_ERROR_VELOCITY_IS_ZERO – the velocity during homing was zero when it should have been moving

Get Status Response:

"get_status,OK,CONNECTED,HOMING_ERROR_VELOCITY_IS _ZERO"

d. HOMING_ERROR_NO_RESPONSE_FROM_MOTOR_NODE

Get Status Response:

"get_status,OK,CONNECTED,HOMING_ERROR_NO_RESPONSE_FROM_MOTOR_NODE"

e. HOMING_ERROR_FAILED_TO_UPDATE_PERSISTENCE – homing values were not able to be written to the motor

Get Status Response:

"get_status,OK,CONNECTED,HOMING_ERROR_FAILED_TO_UPDATE_PERSISTENCE"

4. VIRTUAL_LIMITS_NOT_SET – Homing was finished successfully, but virtual limits are not set

Get Status Response:

"get_status,OK,CONNECTED,VIRTUAL_LIMITS_NOT_SET"

10.1.3 Homing active ready

Description: Transient state which happens when SLIDEKIT is performing homing but is not actively moving.

Additional info: /

10.1.4 Homing active moving

Description: SLIDEKIT is performing homing and is actively moving.

Additional info: /

10.1.5 Ready

Description: SLIDEKIT is ready to perform a movement. Additionally, info shows result of the last movement

Additional info:

1. M2R_NOT_STARTED – No movement executed after homing.

   Get Status Response: "get_status,OK,READY, M2R_NOT_STARTED"

2. POSITION_IS_REACHED – Previously desired position is reached.

   Get Status Response: "get_status,OK,READY, POSITION_IS_REACHED"

3. PROGRAM_STOPPED – Movement stopped by client

   Get Status Response: "get_status,OK,READY,PROGRAM_STOPPED"

If a problem occurred during movement and position is not reached, the following additional information may be seen:

a. SLIDEKIT_RUN_INTO_THE_FAULT_STATE – the SLIDEKIT has run into a FAULT state for some reason (eg. This could be due to a "following error", when the motor cannot keep up with what it is being commanded to do)

   Get Status Response: "get_status,OK,READY,SLIDEKIT_RUN_INTO_THE_FAULT_STATE"

b. M2R_NO_SLIDEKIT_CONNECTION_ERROR – there is some kind of error with SLIDEKIT communication (eg. The communication cable was disconnected between the PiBox and the SLIDEKIT)

   Response Format: "get_status,OK,READY,M2R_NO_SLIDEKIT_CONNECTION_ERROR"

c. M2R_NO_RESPONSE_FROM_MOTOR – the motor has not sent a reply when it should have (eg. Malfunction in the motor's State Machine)

   Response Format: *"get_status,OK,READY,M2R_NO_SLIDEKIT_CONNECTION_ERROR"*

d. POSITION_IS_NOT_REACHED – the commanded position was not reached (eg. The SLIDEKIT may have been blocked)

   Response Format:

   "get_status,OK,READY, POSITION_IS_NOT_REACHED"

If the program is stopped by sending "stop_moving" command in READY state. In that case additional info is a combination of client's command (stop_moving) and previous movement result:

a. M2R_PROGRAM_STOPPED__M2R_POSITION_REACHED – the program was stopped but the last commanded position was reached (eg. It was stopped while performing "get" command)

   Response Format:

   "get_status,OK,READY,M2R_PROGRAM_STOPPED__M2R_POSITION_REACHED"

b. M2R_PROGRAM_STOPPED__M2R_POSITION_NOT_REACHED_ERROR – the program was stopped during a movement and the commanded position was not yet reached

   Response Format:

   "get_status,OK,READY,M2R_PROGRAM_STOPPED__M2R_POSITION_NOT_REACHED_ERROR"

c. M2R_PROGRAM_STOPPED__M2R_MOTOR_RUN_INTO_FAULT_STATE – the program was stopped but also ran into the FAULT state (eg. A "following error")

   Response Format:

   "get_status,OK,READY,M2R_PROGRAM_STOPPED__M2R_MOTOR_RUN_INTO_FAULT_STATE"

10.1.6 Moving

Description: SLIDEKIT is performing a movement

Additional info: /

10.2 SLIDEKIT command acknowledges

Each command returns acknowledge as part of response. Acknowledge gives information if command was successfully executed. Based on the acknowledge, response can have different amount of returned values.

10.2.1 OK

Description: Command is successfully triggered or command is successfully executed.

Response format:

- If command returns a value: <*command_name*>, OK, <*Value 1*>,<*Value 2*>…<*Value n*> Value 1, Value 2… Value n – Values returned by a command. Commands can return different number of values
- If command does not return a value: <*command_name*>, OK

10.2.2 OOR – Out of range

Description: Command cannot be fulfilled because at least one parameter is not in range.

Response format:

<*command_name*>, OOR, <*Index of received param*>, <*Min value*>, <*Max value*>

Index of received param –Index of received parameter which is out of range

Min value – The minimum value for that parameter

Max value – The maximum value for that parameter

10.2.3 NF – Not found

Description: Specified command does not exist

Response format:

unspecified_command,OK

10.2.4 NA – Not allowed

Description: Command cannot be executed in current state

Response format:

<*command_name*>, NA

10.2.5 WNP – Wrong number of parameters

Description: Wrong number of parameters used in the command

Response format:

<*command_name*>, WNP, <*Min value*>, <*Max value*>

Min value - Minimum number of parameters needed

Max value - Maximum number of parameters allowed

10.2.6 VE – Value Error

Description: Wrong data type for provided client parameter

Response format: <*command_name*>, VE, <*Index of received param*>, <*Data type*>

Index of received param – Index of a parameter which has a wrong type

Data type – Type that is required in previously specified index. Types can be: Integer, float (with one decimal), string

10.2.7 EF – Execution Failed

Description: Command and parameters are valid, but execution of command failed

Response format: <*command_name*>, EF, <*Error reason 1*>, <*Error reason 2*>…<*Error reason n*>

Error reason – Additional information on why execution of command failed.

10.3 SLIDEKIT stargate commands

10.3.1 Move to absolute position

Command: “moveTo_absolutePosition”

Description: Command moves SLIDEKIT to specified position. Optionally command can take additional parameters for setting velocity, acceleration, and deceleration. Velocity, acceleration and deceleration values are used only for the current command, and are not stored as motion profile parameters. If only 1, 2 or 3 parameters are provided then the missing values are read from the motion profile parameters. If motion profile parameters are not set (using: “set_motionProfilePramaters”, “set_velocity”, “set_acceleration”, “set_deceleration” commands) and optional parameters are not provided, command returns error. When “moveTo_absolutePosition” command is triggered, SLIDEKIT is locked for manual movement, even when SLIDEKIT reaches desired position, it is still locked.

State Change: Successfully executed command changes SLIDEKIT state from “READY,<*>” to “MOVING”

Client Formats:

1. “moveTo_absolutePosition,<*Position*>”
2. “moveTo_absolutePosition,<*Position*>,<*Velocity*>”
3. “moveTo_absolutePosition,<*Position*>,<*Velocity*>,<*Acceleration*>”
4. “moveTo_absolutePosition,<*Position*>,<*Velocity*>,<*Acceleration*>,<*Deceleration*>”

Parameters:

1. *Position* – FLOAT with one decimal - Position in mm (mandatory)

2. *Velocity* - INTEGER - Velocity in mm/s (optional)
3. *Acceleration* - INTEGER - Acceleration in mm/s² (optional)
4. *Deceleration* - INTEGER - Deceleration in mm/s² (optional)

Command allowed in states:

1. READY

Possible acknowledges:

1. OK" – Movement successfully triggered.

   Response Format: "moveTo_absolutePosition,OK"

2. "OOR" - One of the parameters is out of range.

   Response Format: "moveTo_absolutePosition,OOR,<*Index of received param*>,<*Minimum value*>,<*Maximum value*>"

   Index of received param – Index of first received parameter which is out of range

   Minimum value – The minimum value for that parameter

   Maximum value – The maximum value for that parameter

3. "WNP" – Wrong number of parameters provided. Command should have minimum 1 and maximum 4 parameters

   Response Format: "set_motionProfileParameters, WNP,1,4"

4. "NA" – Not allowed. Movement is not possible in current state

   Response Format: "moveTo_absolutePosition,NA"

5. "VE" – Value error. One of the parameters have invalid type

   Response Format: "moveTo_absolutePosition,VE,<*Index of received param*>,<*Data type*>"

   Index of received param – Index of first parameter which has a wrong type

   Data type – Type that is required in previously specified index

   Example: "moveTo_absolutePosition,VE,1,FLOAT" – Desired position is not of type float

   Example: "moveTo_absolutePosition,VE,2,INTEGER" – Velocity is not of type integer

   Example: "moveTo_absolutePosition,VE,3,INTEGER" – Acceleration is not of type integer

   Example: "moveTo_absolutePosition,VE,4,INTEGER" – Deceleration is not of type integer

6. "EF" – Execution failed. If any of the motion parameters (velocity, acceleration, deceleration) isn't set using command "set_velocity", "set_acceleration", "set_deceleration" or "set_motionProfileParameters" and is not provided in client format then:

   Response format: "moveTo_absolutePosition,EF,Requested data do not exist:<*motion parameter*>".

   For Example:

   6.1 If client format is "moveTo_absolutePosition,<*Position*>" and velocity, acceleration and deceleration are not stored as motion profile parameters (using "set_motionProfileParameters" or "set_velocity", "set_acceleration", "set_deceleration") then:

   Response format:

   "moveTo_absolutePosition,EF,Requested data do not exist:velocity,Requested data do not exist:acceleration,Requested data do not exist:deceleration"

   6.2 If client format is "moveTo_absolutePosition,<*Position*>,<*Velocity*>" but acceleration and deceleration are not stored as motion profile parameters (using "set_motionProfileParameters" or "set_acceleration", "set_deceleration") then:

   Response format:

   "set_motionProfileParameters,EF,Requested data do not exist:acceleration,Requested data do not exist:deceleration"

   6.3 If client format is "moveTo_absolutePosition,<*Position*>,<*Velocity*>,<*Acceleration*>" but deceleration is not stored as motion profile parameters (using "set_motionProfileParameters" or "set_deceleration") then:

   Response format: "moveTo_absolutePosition,EF,Requested data do not exist:deceleration"

10.3.2 Get available types

Command: "get_typesAvailable"

Description: Command returns all supported SLIDEKIT types. Types are separated by "," (comma)

State Change: /

Client Format: "get_typesAvilable"

Parameters: /

Command allowed in states:

1. INITIALIZED
2. CONNECTED
3. READY
4. MOVING
5. HOMING_ACTIVE_READY
6. HOMING_ACTIVE_MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. Types are succesfully returned in response

   Response format: "get_typesAvailable,OK,<*SLIDEKIT Type 1*>,<*SLIDEKIT Type 2*>…<*SLIDEKIT Type n*>"

2. "WNP" – Wrong number of parameters provided. Client command should not have any parameters.

   Response format: "get_typesAvailable,WNP,0,0"

10.3.3 Get Type

Command: "get_type"

Description: Command returns current SLIDEKIT type.

State Change: /

Client Format: "get_type"

Parameters: /

Command allowed in states:

1. INITIALIZED
2. CONNECTED
3. READY
4. MOVING
5. HOMING_ACTIVE_READY
6. HOMING_ACTIVE_MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. SLIDEKIT type is returned.

   Response Format: "get_type,OK,<*SLIDEKIT type*>"

2. "WNP" – Wrong number of parameters provided. Command should not have any parameters.

   Response format: "get_type,WNP,0,0"

3. "EF" – Execution Failed (e.g. SLIDEKIT type cannot be obtained because type is not set).

   Response format: "get_type,EF,Requested data do not exist:config_type"

   Get Status response: "get_status,OK,CONNECTED,-TYPE IS NOT SET"

10.3.4 Set Type

Command: "set_type"

Description: Command sets new SLIDEKIT type. SLIDEKIT type must be one of the types returned by get_typesAvailable command

State Change: Successfully executed command changes SLIDEKIT state to "HOMING_INTERRUPTED_OR_NOT_STARTED"

Client Format: "set_type,<*Type*>"

Parameters:

1. *Type* - STRING – Desired type (mandatory)

Command allowed in states:

1. INITIALIZED
2. CONNECTED
3. READY

Possible acknowledges:

1. "OK" – Command successfully executed. Type is successfully set

   Response Format: "set_type,OK"

2. "WNP" – Wrong number of parameters provided. Command should have exactly one parameter

   Response format: "set_type,WNP,1,1"

3. "OOR" – Selected type is out of range

   Response format: "set_type,OOR,ConfigurationHolder: There is no configuration named <*Type*>"

4. NA" – Not allowed. Set type cannot be executed in current state

   Response Format: "set_type,NA"

5. "VE" – Value error (e.g. empty string provided as a type)

   Response Format: "set_type,VE,1,STRING"

6. "EF" – Execution Failed (e.g. trying to set SLIDEKIT type that is already set).

   Response Format: "set_type,EF,Model named <*Type*> already selected"

10.3.5 Get Velocity

Command: "get_velocity"

Description: Command returns velocity which is stored in motion profile parameters. Velocity unit of measurement is mm/s.

State Change: /

Client Format: "get_velocity"

Parameters: /

Command allowed in states:

1. CONNECTED
2. READY
3. MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. Velocity is returned in response.

Response Format: "get_velocity,OK,<*Velocity*>"

2. "WNP" – Wrong number of parameters provided. Command should not have any parameters

Response format: "get_velocity,WNP,0,0"

3. "NA" – Not allowed. Velocity cannot be obtained in current state

Response Format: "get_ velocity,NA"

4. "EF" – Execution Failed (e.g. velocity cannot be obtained because it is not set)

Response Format: "get_velocity,EF,Requested data do not exist:velocity"

10.3.6 Set Velocity

Command: "set_velocity"

Description: Command sets SLIDEKIT velocity. Velocity is stored as motion profile parameter. If SLIDEKIT is performing movement, "set_velocity" command will not impact the current movement.

State Change: /

Client Format: "set_velocity,<*Velocity*>"

Parameters:

1. Velocity - INTEGER – Desired velocity in mm/s (mandatory).

Command allowed in states:

1. CONNECTED

2. READY

3. MOVING – "set_velocity" is not applied in current movement. It can be applied in further commands

Possible acknowledges:

1. "OK" – Command successfully executed. Velocity successfully set.

Response Format: "set_velocity,OK"

2. "WNP" – Wrong number of parameters provided. Command should have exactly one parameter

Response Format: "set_velocity,WNP,1,1"

3. "OOR" – Out of range. Provided velocity is not within defined velocity limits.

Response Format: "set_velocity,OOR,1,<*Minimum value*>,<*Maximum value*>"

Minimum value – Minimum supported velocity

Maximum value – Maximum supported velocity

4. "NA" – Not allowed. Set velocity cannot be set in current state

Response Format: "set_velocity,NA"

5. "VE" – Value error. Velocity must be INTEGER

Response Format: "set_velocity,VE,1,INTEGER"

6. "EF" – Execution failed (e.g. velocity cannot be set because SLIDEKIT type is not set)

Response Formatt: "set_velocity,EF,Requested data do not exist:min_ velocity,Request data do not exist:max_ velocity"

Get Status Response:
"get_status,OK,CONNECTED,TYPE_NOT_SET"

10.3.7 Get Acceleration

Command: "get_acceleration"

Description: Command returns acceleration. Acceleration is stored as motion profile parameter. Acceleration unit of measurement is mm/s^2.

State Change: /

Client Format: "get_acceleration"

Parameters: /

Command allowed in states:

1. CONNECTED

2. READY

3. MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. Acceleration is returned in response

Response Format:
"get_acceleration,OK,<*Acceleration*>"

2. "WNP" – Wrong number of parameters provided. Command should not have any parameters

Response format: "get_acceleration,WNP,0,0"

3. "NA" – Not allowed. Acceleration cannot be obtained in current state

Response Format: "get_acceleration,NA"

4. "EF" – Execution Failed (e.g. acceleration cannot be obtained because it is not set)

Response Format: "get_acceleration,EF,Requested data do not exist: acceleration"

10.3.8 Set Acceleration

Command: "set_acceleration"

Description: Command sets SLIDEKIT acceleration. Acceleration is stored as motion profile parameter.

State Change: /

Client Format: "set_acceleration,<*Acceleration*>

Parameters:

1. *Acceleration* - INTEGER – Desired acceleration in mm/s^2 (mandatory)

Command allowed in states:

1. CONNECTED

2. READY

3. MOVING – "set_acceleration" is not applied in current movement. It can be applied in further commands

Possible acknowledges:

1. "OK" – Command successfully executed. Acceleration successfully set

Response Format: "set_acceleration,OK"

2. "WNP" – Wrong number of parameters provided. Command should have exactly one parameter

Response Format: "set_ acceleration,WNP,1,1"

3. "OOR" – Out of range. Provided acceleration is not within defined acceleration limits

Response Format: "set_ acceleration,OOR,1,<*Minimum value*>,<*Maximum value*>"

Minimum value – Minimum supported acceleration

Maximum value – Maximum supported acceleration

4. "NA" – Not allowed. Set acceleration cannot be executed in current state

Response Format: "set_ acceleration,NA"

5. "VE" – Value error. Acceleration must be INTEGER

Response Format: "set_ acceleration,VE,1,INTEGER"

6. "EF" – Execution failed (e.g. acceleration cannot be set because SLIDEKIT type is not set)

Response Format: "set_acceleration,EF,Requested data do not exist:min_ acceleration,Request data do not exist:max_acceleration"

Get Status Response: "get_status,OK,CONNECTED,TYPE_NOT_SET"

10.3.9 Get Deceleration

Command: "get_ deceleration"

Description: Command returns deceleration. Deceleration is stored as motion profile parameter. Deceleration unit of measurement is mm/s^2.

State Change: /

Client Format: "get_deceleration"

Parameters: /

Command allowed in states:

1. CONNECTED

2. READY

3. MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. Deceleration is returned in response

Response Format
"get_deceleration,OK,<*Deceleration*>"

2. "WNP" – Wrong number of parameters provided. Command should not have any parameters

Response Format: "get_ deceleration,WNP,0,0"

3. "NA" – Not allowed. Deceleration cannot be obtained in current state

Response Format: "get_ deceleration,NA"

4. "EF" – Execution Failed (e.g. deceleration cannot be obtained because it is not set)

Response Format: "get_deceleration,EF,Requested data do not exist: deceleration"

10.3.10 Set Deceleration

Command: "set_deceleration"

Description: Command sets SLIDEKIT deceleration. Deceleration is stored as motion profile parameter.

State Change: /

Client Format: "set_deceleration,<*Deceleration*>"

Parameters:

1. *Deceleration* - INTEGER – Desired deceleration in mm/s^2 (mandatory)

Command allowed in states:

1. CONNECTED

2. READY

3. MOVING – "set_deceleration" is not applied in current movement. It can be applied in further commands

Possible acknowledges:

1. "OK" – Command successfully executed. Deceleration successfully set

Response Format: "set_deceleration,OK"

2. "WNP" – Wrong number of parameters provided. Command should have exactly one parameter

Response Format: "set_ deceleration,WNP,1,1"

3. "OOR" – Out of range. Provided deceleration is not within defined deceleration limits

Response Format: "set_ deceleration,OOR,1,<*Minimum value*>,<*Maximum value*>"

Minimum value – Minimum supported deceleration

Maximum value – Maximum supported deceleration

4. "NA" – Not allowed. Set deceleration cannot be executed in current state

Response Format: "set_ deceleration,NA"

5. "VE" – Value error. Deceleration must be INTEGER.

Response Format: "set_deceleration,VE,1,INTEGER"

6. "EF" – Execution failed (e.g. deceleration cannot be set because SLIDEKIT type is not set)

Response Format: "set_deceleration,EF,Requested data do not exist:min_deceleration,Request data do not exist:max_deceleration"

Get Status Response:
"get_status,OK,CONNECTED,TYPE_NOT_SET"

10.3.11 Get Position

Command: "get_position"

Description: Command returns SLIDEKIT current position in mm.

State Change: /

Client Format: "get_position"

Parameters: /

Command allowed in states:

1. CONNECTED
2. READY
3. MOVING
4. HOMING_ACTIVE_READY
5. HOMING_ACTIVE_MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. Position is returned in response "OK" acknowledge is expected in states:
   b. "CONNECTED,VIRTUAL_LIMITS_NOT_SET",
   c. "READY,<*Additional info*>",
   d. "MOVING"

Response Format: "get_position,OK,<*Position*>"

5. "WNP" – Wrong number of parameters provided. Command should not have any parameters

Response Format "get_ position,WNP,0,0"

6. "NA" – Not allowed. Position cannot be obtained in current state

Response Format: "get_ position,NA"

7. "EF" – Execution failed (e.g. position cannot be obtained because homing is not done)

Response Format: "get_position,EF,Homing is not done" "EF" is always expected in states:
   a. "HOMING_ACTIVE_READY",
   b. "HOMING_ACTIVE_MOVING",
   c. "CONNECTED,TYPE_NOT_SET",
   d. "CONNECTED,HOMING_INTERRUPTED_OR_NOT_STARTED",
   e. "CONNECTED,HOMING_INTERRUPTED_BY_REQUEST",
   f. "CONNECTED,<*Homing error*>"

10.3.12 Get Stroke

Command: "get_stroke"

Description: Command returns maximum stroke in mm. Maximum stroke is calculated during SLIDEKIT homing.

State Change: /

Client Format: "get_stroke"

Parameters: /

Command allowed in states:

1. CONNECTED
2. READY
3. MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. Stroke is returned in response

Response Format: "get_stroke,OK,<*Stroke*>"

2. "WNP" – Wrong number of parameters provided. Command should not have any parameters

Response Format:"get_ stroke,WNP,0,0"

3. "NA" – Not allowed. Stroke cannot be obtained in current state

Response Format:"get_ position,NA"

4. "EF" – Execution failed (e.g. max stroke cannot be obtained because homing is not done)

Response Format:"get_position,EF, Requested data do not exist:max_stroke"

10.3.13 Get Status

Command: "get_status"

Description: Command returns SLIDEKITs current state (status) or current state and additional info. For all SLIDEKIT states and additional infos refer to: "1. SLIDEKIT states (statuses)"

State Change: /

Client Format: “get_status”

Parameters: /

Command allowed in states:

1. CONNECTED
2. READY
3. MOVING
4. HOMING_ACTIVE_READY
5. HOMING_ACTIVE_MOVING

Possible acknowledges:

1. “OK” – Command successfully executed. Status and optionally additional info are returned in response

   a. If returned status have additional info:

   Response Format: “get_status,OK,<*State*>,<*Additional info*>”

   b. If returned status does not have additional info:

   Response Format: “get_status,OK,<*State*>”

3. “WNP” – Wrong number of parameters provided. Command should not have any parameters

   Response format: “get_ status,WNP,0,0”

10.3.14 Stop Moving

Command: “stop_moving”

Description: Command stops SLIDEKIT movement. When SLIDEKIT is stopped using “stop_moving” command, it is unlocked and can be moved manually.

State Change: Successfully executed command changes SLIDEKIT state:

a. If previous state was “MOVING”, new state is “READY,PROGRAM_STOPPED”

b. If previous state was “READY,POSITION_ IS_REACHED”, new state is “READY,M2R_ PROGRAM_STOPPED_M2R_POSITION_ REACHED”

c. If previous state was “READY,SLIDEKIT_ RUN_INTO_THE_FAULT_STATE” , new state is “READY,M2R_PROGRAM_STOPPED__M2R_ MOTOR_RUN_INTO_FAULT_STATE”

d. If previous state was “READY,POSITION_IS_ NOT_REACHED”, new state is “READY,M2R_ PROGRAM_STOPPED__M2R_POSITION_NOT_ REACHED_ERROR”

Client Format: “stop_moving”

Parameters: /

Command allowed in states:

1. READY
2. MOVING

Possible acknowledges:

1. “OK” – Command successfully executed. SLIDEKIT successfully stopped.

   Response format: “stop_moving,OK”

2. “WNP” – Wrong number of parameters provided. Command should not have any parameters

   Response format: “stop_moving,WNP,0,0”

3. “NA” – Not allowed. Stop moving cannot be executed in current state

   Response format: “stop_moving,NA”

10.3.15 Get Motion Profile Parameters

Command: “get_motionProfileParameters”

Description: Command returns moiton profile parameters. Motion profile parameters consist of velocity, acceleration and deceleration. Each motion profile parameter can be obtained by using “get_velocity”, “get_acceleration”, “get_deceleration”

State Change: /

Client Format: “get_motionProfileParameters”

Parameters: /

Command allowed in states:

1. READY
2. MOVING
3. CONNECTED

Possible acknowledges:

1. “OK” – Command successfully executed. All profile parameters (velocity, acceleration, deceleration) are returned in response.

   Response Format: “get_motionProfileParameters,OK,<*Velocity*>,<*Acceleration*>,<*Deceleration*>”

2. “WNP” – Wrong number of parameters provided. Command should have any parameters.

   Response Format: “get_motionProfileParameters,WNP,0,0”

3. “NA” – Not allowed. Profile parameters cannot be obtained in current state

   Response Format: “get_motionProfileParameters,NA”

4. "EF" – Execution Failed (e.g. one or more profile parameters are not set). Depending on how many profile parameters are not set, command can return multiple error messages.

   a. If all profile parameters are not set:

Response Format: "get_motionProfileParameters,EF,Requested data do not exist:velocity, Requested data do not exist:acceleration, Requested data do not exist:deceleration"

   b. If velocity is not set:

Response Format: "get_motionProfileParameters,EF,Requested data do not exist:velocity"

   c. If acceleration is not set:

Response Format: "get_motionProfileParameters,EF,Requested data do not exist: acceleration"

   d. If deceleration is not set:

Response Format: "get_motionProfileParameters,EF,Requested data do not exist: deceleration"

   e. If velocity and acceleration are not set:

Response Format: "get_motionProfileParameters,EF,Requested data do not exist: velocity,Requested data do not exist: acceleration"

   f. If velocity and deceleration are not set:

Response Format: "get_motionProfileParameters,EF,Requested data do not exist: velocity,Requested data do not exist: deceleration"

   g. If acceleration and deceleration are not set:

Response Format: "get_motionProfileParameters,EF,Requested data do not exist: acceleration,Requested data do not exist: deceleration"

10.3.16 Set Motion Profile Parameters

Command: "set_motionProfileParameters"

Description: Command sets motion profile parameters. Values stored in profile parameters (velocity, acceleration, deceleration) are used as a default values when move to absolute position is executed. Each motion parameter can be also set separately by using "set_velocity", "set_acceleration" and "set_deceleration"

State Change: /

Client Format(s):

1. "set_motionProfileParameters,<*Velocity*>
2. "set_motionProfileParameters,<*Velocity*>, <Acceleration>"
3. "set_motionProfileParameters,<*Velocity*>, <*Acceleration*>,<*Deceleration*>"

Parameters:

1. Velocity - INTEGER – Desired velocity mm/s (mandatory).
2. Acceleration – INTEGER – Desired acceleration mm/s^2 (optional).
3. Deceleration – INTEGER - Desired deceleration mm/s^2 (optional).

Command allowed in states:

1. CONNECTED
2. READY
3. MOVING – "set_motionProfileParameters" is not applied in current movement. It can be applied in further commands

Possible acknowledges:

1. "OK" – Command successfully executed. Profile parameter(s) successfully set

   Response Format: "set_motionProfileParameters,OK"

2. "WNP" – Wrong number of parameters provided. Command should have minimum 1 and maximum 3 parameters

   Response Format: "set_motionProfileParameters,WNP,1,3"

3. "OOR" – Out of range. One of the provided parameters are not within defined limits

   Response Format: "set_motionProfileParameters,OOR,<*Index of received param*>,<*Minimum value*>,<*Maximum value*>"

   Index of received param – Index of first received parameter which is out of range

   Minimum value – The minimum value for that parameter

   Maximum value – The maximum value for that parameter

4. "NA" – Not allowed. Profile parameters cannot be set in current state

   Response Format: "set_motionProfileParameters,NA"

5. "VE" – Value error. All profile parameters should be INTEGER type

   Response Format: "set_motionProfileParameters,VE,<*Index of received param*>,INTEGER"

   I*ndex of received param* – Index of first parameter which has a wrong type

6. "EF" – Execution failed (e.g. one or more profile parameters are provided, but SLIDEKIT type is not set). Depending on how many profile parameters are provided, command can return multiple error messages:

a. If velocity is provided in client command ("set_motionProfileParameters,<*Velocity*>")

Response formats: "set_motionProfileParameters,EF,Requested data do not exist:min_velocity,Requested data do not exist:max_velocity"

b. If velocity and acceleration are provided in client command ("set_motionProfileParameters,<*Velocity*>,<*Acceleration*>")

Response formats: "set_motionProfileParameters,EF,Requested data do not exist:min_velocity,Requested data do not exist:max_velocity, Requested data do not exist:min_acceleration,Requested data do not exist:max_acceleration"

c. If velocity, acceleration and deceleration are provided in client command ("set_motionProfileParameters,<*Velocity*>,<*Acceleration*>,<*Deceleration*>")

Response formats: "set_motionProfileParameters,EF,Requested data do not exist:min_velocity,Requested data do not exist:max_velocity,Requested data do not exist:min_acceleration,Requested data do not exist:max_acceleration,Requested data do not exist:min_deceleration,Requested data do not exist: max_deceleration"

10.3.17 Start Homing

Command: "start_homing"

Description: Command starts SLIDEKIT homing. While SLIDEKIT is executing homing it is retracting to its minimum position and extending to its maximum. When homing process is done SLIDEKIT will have stored information of maximum stroke.

State Change: Successfully triggered command changes SLIDEKIT state:

1. f previous state was "READY,<*>", new state is "HOMING_ACTIVE_READY" for short period of time, then it becomes "HOMING_ACTIVE_MOVING"
2. If previous state was "CONNECTED,<*>", new state is "HOMING_ACTIVE_READY" for short period of time, then it becomes "HOMING_ACTIVE_MOVING"
3. If error occurs during homing, state is changed to "CONNECTED,<*Homing error*>"

   Homing error – refer to chapter "1.2. Connected" for more information about homing errors

Client Format: "start_homing"

Parameters: /

Command allowed in states:

1. CONNECTED
2. READY

Possible acknowledges:

1. "OK" – Homing successfully triggered

   Response Format: "start_homing,OK"

2. "WNP" – Wrong number of parameters provided. Command should not have any parameters

   Response Format: "start_homing,WNP,0,0"

3. "NA" – Not allowed. Homing cannot be started in current state.

   Response Format: "start_homing,NA"

4. "EF" – Execution failed (e.g. homing cannot be started because type is not set)

   Response Format: "start_homing,EF,Requested data do not exist:config_type"

   Get Status Response: "get_status,OK,CONNECTED,-TYPE _NOT_SET"

10.3.18 Stop Homing

Command: "stop_homing"

Description: Command stops SLIDEKIT homing. When homing is stopped by client command "stop_homing", status is always set to CONNECTED,HOMING_INTERRUPTED_BY_REQUEST

State Change: Successfully executed command changes SLIDEKIT state from "HOMING_ACTIVE_READY" or "HOMING_ACTIVE_MOVING" to "CONNECTED,HOMING_INTERRUPTED_BY_REQUEST"

Client Format: "stop_homing"

Parameters: /

Command allowed in states:

1. HOMING_ACTIVE_READY
2. HOMING_ACTIVE_MOVING

Possible acknowledges:

1. "OK" – Command successfully executed. Homing successfully stopped

   Response Format: "stop_homing,OK"

2. "WNP" – Wrong number of parameters provided. Command should not have any parameters

   Response Format: "stop_homing,WNP,0,0"

3. "NA" – Not allowed. Stop homing cannot be executed in current state

   Response Format: "stop_homing,NA"

10.3.19 Set Virtual limits

Command: “set_virtualLimits”

Description: Command sets SLIDEKIT virtual limits.

State Change: Successfully executed command changes SLIDEKIT state:

1. If previous state was “CONNECTED,VIRTUAL_LIMITS_NOT_SET”, new state is “READY,M2R_NOT_STARTED”

Client Format: “set_virtualLimits,<*Lower limit*>,<*Upper limit*>”

Parameters:

1. Lower limit – FLOAT with one decimal – Minimum position that SLIDEKIT can retract to in mm (mandatory)
2. Upper limit – FLOAT with one decimal - Maximum position that SLIDEKIT can extend to in mm(mandatory)

Command allowed in states:

1. CONNECTED
2. READY

Possible acknowledges:

1. “OK” – Command successfully executed. Limits successfully set

   Response Format: “set_virtualLimits,OK”

2. “WNP” – Wrong number of parameters provided. Command accepts exactly 2 parameters

   Response Format: “set_virtualLimits,WNP,2,2”

3. “OOR” – Out of range. One of the provided parameters are not within limits. Lower limit cannot be lower than 0 and upper limit cannot be greater than maximum stroke

   Response Format: “set_virtualLimits,OOR,<*Index of received param*>,<*Minimum value*>,<*Maximum value*>”

   Index of received param – Index of first received parameter which is out of range

   Minimum value – The minimum value for that parameter

   Maximum value – The maximum value for that parameter

4. “NA” – Not allowed. Set limits cannot be executed in current state

   Response Format: “set_virtualLimits, NA”

5. “VE” – Value error. Limits should be of type FLOAT

   Response Format: “set_virtualLimits,VE,<*Index of received param*>,FLOAT”

   Index of received param – Index of first parameter which has a wrong type

6. “EF” - Execution failed (e.g. limits cannot be set because homing is not done)

   Response Format: “set_virtualLimits,EF,Requested data do not exist:max_stroke”

Limits cannot be set if max stroke is not known. Max stroke is calculated during homing.

10.3.20 Get Virtual Limits

Command: “get_virtualLimits”

Description: Command returns SLIDEKIT virtual limits. Virtual limits unit of measurement is mm

State Change: /

Client Format: “get_virtualLimits”

Parameters: /

Command allowed in states:

1. CONNECTED
2. READY
3. MOVING

Possible acknowledges:

1. “OK” – Command successfully executed. Virtual limits are returned in response

   Response Format: “get_virtualLimits,OK,<*Lower limit*>,<*Upper limit*>”

   Lower limit – Virtual minimum that SLIDEKIT can retract to

   Upper limit – Virtual maximum that SLIDEKIT can extend to

2. “WNP” – Wrong number of parameters provided. Command should not have any parameters

   Response Format: “get_virtualLimits,WNP,0,0”

3. “NA” – Not allowed. Virtual limits cannot be obtained in current state

   Response Format: “get_virtualLimits,NA”

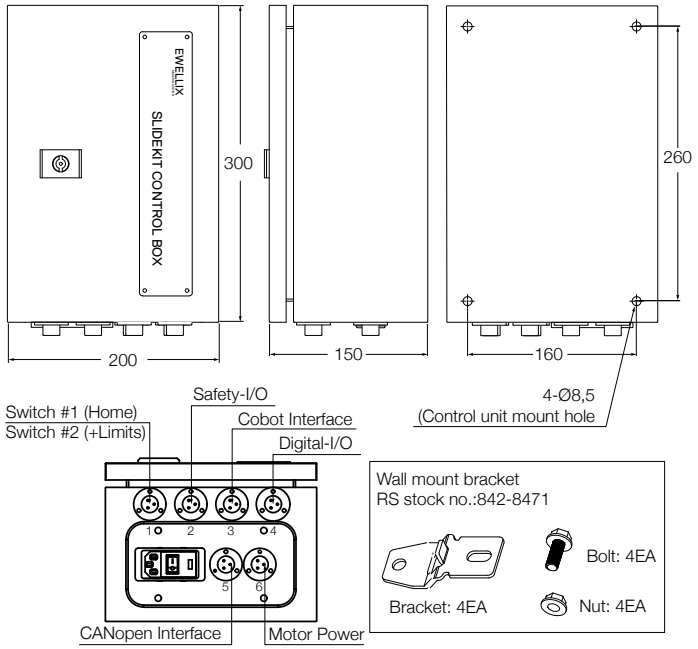
4. “EF” - Execution failed (e.g. limits cannot be obtained because they are not set)

   Response Format: “get_virtualLimits,EF,Requested data do not exist:virtual_minimum,Request data do not exist:virtual_maximum”

   Get Status Response: “get_status,OK,CONNECTED,VIRTUAL_LIMITS_NOT_SET”

11. Appendix

11.1 Controller



11.2 Limit switch

- EE-SX674P-WR 1M

PNP Output

Model	Output configuration	Timing charts	Terminal connections	Output circuit
EE-SX67□P EE-SX67□P-WR	Light-ON	Incident / Interrupted Light indicator (red) ON / OFF Output transistor ON / OFF Load (relay) Operates / Releases	Short-circuited between ⊖ terminal and positive ⊕ terminal	Light indicator (red), Main circuit, OUT, IC, Load, 5 to 24 VDC *The terminal arrangement depends on the model. Check the dimensional diagrams.
	Dark-ON	Incident / Interrupted Light indicator (red) ON / OFF Output transistor ON / OFF Load (relay) Operates / Releases	Open between ⊖ terminal and positive ⊕ terminal *1 *2	

Terminal Arrangement

(1)	⊕	Vcc
(2)	L	L*
(3)	OUT	OUTPUT
(4)	⊖	GND (0 V)

* Pin 2 is not used for the EE-SX474.

ewellix.com

PUB NUM TC-08062-EN-June 2025