GRIPLINK PLUGIN FOR UNIVERSAL ROBOTS POLYSCOPE X

Version 1.0.0

July 2026

Table of Contents

1 Technical overview and safety

1.1 Introduction

GRIPLINK technology enables automation components from WEISS ROBOTICS to be integrated into robot systems via a standardized communication interface. The GRIPLINK Plugin for Universal Robots PolyScope X is the controller-side interface to WEISS ROBOTICS devices and to compatible third-party devices from the GRIPLINK ecosystem, such as sensors and actuators. It connects and operates these devices on Universal Robots systems running PolyScope X.
The GRIPLINK Plugin supports both devices that use the GRIPLINK protocol natively via TCP/IP (**Ethernet**) and compatible Modbus RTU devices that are connected via the **Tool I/O** interface of the Universal Robots controller.

The term **GRIPLINK controller** is used in this manual as a generic term for all products that integrate GRIPLINK technology. These include both standalone GRIPLINK controllers (e.g., GRIPLINK-ET4) and devices with embedded GRIPLINK functionality, such as WPG series or INTRAPAL series servo grippers from WEISS ROBOTICS. All of these products implement the same GRIPLINK protocol and can therefore be controlled in the same way via the plugin.

This manual describes the functions of the GRIPLINK Plugin. For information on mounting, commissioning, and operation of the GRIPLINK controller, refer to the operating manual of the respective module. These manuals are available online at www.griplink.de.

1.1.1 Notation and Symbols

For clarity, the following symbols are used in this manual:

Function- or safety-related note. Failure to observe these instructions may endanger the safety of personnel and the system, damage the device, or impair its function.

Additional information to aid in understanding the described topic.

Reference to further information.

1.1.2 Intended Use

The "GRIPLINK Plugin" software is intended for communication between the GRIPLINK controller from WEISS ROBOTICS and a robot controller. It enables the integration and control of GRIPLINK-compatible devices via the standardized GRIPLINK communication interface.

The requirements of the applicable guidelines as well as the installation and operating instructions in this manual must be observed and complied with. Any other or additional use is considered improper. The manufacturer is not liable for any resulting damage.

1.1.3 System Requirements

For Ethernet operation, this plugin is compatible with GRIPLINK protocol version 3 and higher. For Tool I/O operation, the device- and firmware-dependent requirements listed below apply. The plugin is a **URCapX** package, the successor format to the original URCap on PolyScope 5.
One of the following Universal Robots robot controllers is required for operation:

- Control Box CB5.6 with software version 10.14 (or higher)

Interface	Supported devices / firmware	License
Ethernet (TCP/IP)	GRIPLINK-ET4 from FW 5.2.0; WPG series from FW 2.4.0; INTRAPAL series from FW 5.2.0	–
Tool I/O (Modbus RTU)	GRIPKIT EASY from FW 2.1.1; GRIPKIT LS from FW 1.1.0; INTRAPAL series from FW 5.3.0	OPT-GKEASY-MB for GRIPKIT EASY; OPT-IPL-MB for INTRAPAL series

OPT-GKEASY-MB and OPT-IPL-MB are device options that enable the Modbus RTU (Tool I/O) interface on the respective device. They are ordered from WEISS ROBOTICS; see the device's operating instructions for details.

The IP address of the GRIPLINK controller must be on the same subnet as that of the robot controller. The GRIPLINK controller manual describes the exact procedure for changing the IP address.

Important Note: Updating the URCapX may affect existing systems, configurations, and robot programs. This may require adjustments to the application. Therefore, carefully review the system requirements as well as the Version History before each update.

It is recommended that you perform a complete functional test of the application after migration.

1.1.4 License Terms

The GRIPLINK Plugin is protected by copyright. The applicable license terms are included with the software package and are also available in the application node under **License Agreement**. By installing the software, you accept these license terms.

1.1.5 Demo Programs

The software package includes a set of example programs that demonstrate the most common integration patterns. They are intended solely for testing and as a starting point for your own application. The descriptions use the GRIPLINK commands and device states documented in the Programming chapters and the Device State appendix; use this list as an overview and return to it after reading those chapters.
Each example ships as a native PolyScope X program (`URProgram/*.urpx`, built from GRIPLINK program nodes) and - except for pick_and_place, whose result branches exist as program nodes only - as an equivalent URScript file (`URScript/*.script`). Both forms perform the same sequence, so the descriptions below apply to both.

simple_grip_release

A single blocking grip and release cycle using a stored preset. It reads the device state, initializes the device (home and enable) if needed, opens the fingers, grips, and classifies the outcome as HOLDING, NO PART, or a failure, then releases and verifies the released state. The recommended starting point for a new integration.

griplink_demo

Runs a configurable number of consecutive blocking grip/release cycles, evaluating the device state after each grip. A cycle that does not reach HOLDING shows a warning with the cycle and state and stops the loop.

flexgrip_cycle

A grip/release cycle in which the motion parameters (position, force, speed, acceleration) are specified explicitly in each call via FLEXGRIP and FLEXRELEASE instead of a stored preset. Useful when the same program must handle different parts without pre-configured presets.

async_preposition

Overlaps finger motion with robot travel to shorten cycle time: a non-blocking FLEXRELEASE starts the fingers moving toward a target position while the robot moves, then WAITVAL waits until the finger position (value index 0) enters the target window. WAITVAL is used instead of WSTR because it waits for a value window rather than a state transition, so it still completes when the device is already at the target.

check_workpiece

Grips a workpiece and uses the finger position as a gauge to accept or reject it. After the grip it aborts on NO PART or FAULT; otherwise it reads the finger position via VALUE index 0 and compares it against a tolerance band, reporting the part as ACCEPTED (with the measured value) or REJECTED before releasing.

multi_grip_release

Grips and releases the devices on a range of ports at the same time, using non-blocking commands with per-port synchronization: all grips are issued non-blocking, then a WSTR on each populated port confirms the HOLDING state; the same pattern is repeated for release. A port that does not reach the expected state is logged.

fault_clearing_via_disable

Detects a device fault and clears it with a DISABLE followed by HOME, then re-reads the state to confirm whether the fault was cleared. It also reports the "not connected" and "not in fault" cases.

pick_and_place

A pick-and-place sequence built with the GRIPLINK **Pick** and **Place** program nodes: Pick grips the part and branches on HOLDING / NO PART, Place releases it and branches on RELEASED / NOT RELEASED. As in the other examples, robot travel is abstracted as a placeholder wait.

1.2 Plugin structure at a glance

Component	Navigation	Main purpose
Application node	Application tab → URCaps → WEISS ROBOTICS GRIPLINK	Plugin settings.
Sidebar	PolyScope X sidebar	Manual live control for commissioning, testing, diagnostics.
Smart Skills	Move screen (Smart Skills tab)	Reusable manual actions: GRIP, RELEASE, HOME.
Program nodes	Program screen → Commands → URCaps group	Graphical programming with the unified GRIPLINK node and Pick/Place nodes.
URScript functions	Script nodes / generated program preamble	Advanced programming through built-in URScript calls.
Demo programs	Software package	Eight example programs, from a basic grip cycle to complete pick-and-place.

1.3 Available communication interfaces

Interface	Devices
Ethernet	Devices with a GRIPLINK controller, such as

	GRIPLINK-ET4, INTRAPAL series, and WPG series.
Tool I/O	Devices with the FLEXGRIP protocol, such as GRIPKIT EASY and GRIPKIT LS-40-050.

1.4 Version and migration notes

Existing PolyScope 5 projects written in URScript can be migrated to PolyScope X, as the GRIPLINK library provides the same calls in PolyScope X as in PolyScope 5. UI-based programs (URProgram) from PolyScope 5 require migration. The functionality of the PolyScope 5 plugin is covered by the PolyScope X plugin as well. The plugin supports both the GRIPLINK and FLEXGRIP protocols on PolyScope X.

1.5 Emergency stop

An emergency stop removes power from the robot arm immediately. Whether the connected device also loses power depends on how it is powered:

Power supply	Behavior on an emergency stop
Powered externally	The device uses its own external power supply. Whether this supply is switched off on an emergency stop or remains powered is the system designer's decision.
Powered over the Tool Connector	The Tool Connector power supply is switched off on an emergency stop, so the device is powered off as well.

The plugin reports status code 18 (`NO_COMM`) for a powered-off device, as for any other interruption of the Tool I/O connection. To restore communication once power returns, press **Rescan** in the Application tab.

2 Installation and commissioning

2.1 Installation

The GRIPLINK Plugin ships as a **URCapX** package (file extension `.urcapx`).

To install the plugin

1. Copy `griplink-1.0.0.urcapx` to the root of a USB flash drive (or otherwise make the file available to PolyScope X).
2. Open **Hamburger menu → System Manager → URCaps**. This section is password-protected; enter the admin password for the installation.
3. Press **+ URCap** and select the `griplink-1.0.0.urcapx` file.

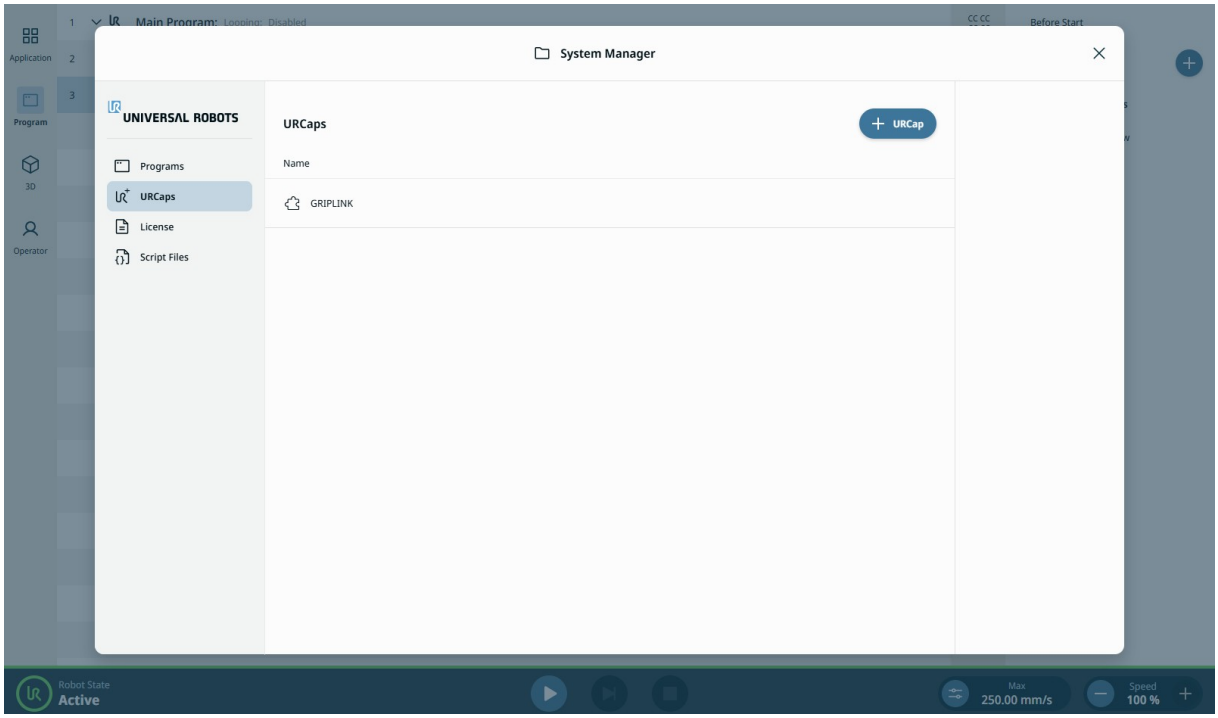

Figure 1: System Manager - URCaps: install the GRIPLINK URCapX.

4. Confirm the installation and restart the robot when prompted.

To uninstall the software

1. Open **Hamburger menu → System Manager → URCaps** and unlock with the admin password.
2. Select GRIPLINK.
3. Press the **remove** button.

4. Restart the robot.

2.2 Commissioning: Ethernet

1. Connect the robot controller and the GRIPLINK controller to the same network (same subnet).
2. Configure the robot's network settings: tap the **Hamburger menu** in the top-left header of the screen and tap **Settings**. Open the sidebar and select the **Connection** tab, then locate **Network**. Change to a static IP address and set the desired IP address of the robot and the subnet mask.
3. Open **Application tab → URCaps → WEISS ROBOTICS GRIPLINK → Setup**.
4. Set **Interface** to **Ethernet**.
5. Enter the controller IP address, default `192.168.1.40` if unchanged.
6. Press **Check GRIPLINK Controller**. Success reports "Controller {ip} is reachable."

The IP address of the GRIPLINK controller must be in the same subnet as the Control Box. The controller's IP address can be changed via its web interface; the factory default is `192.168.1.40`.

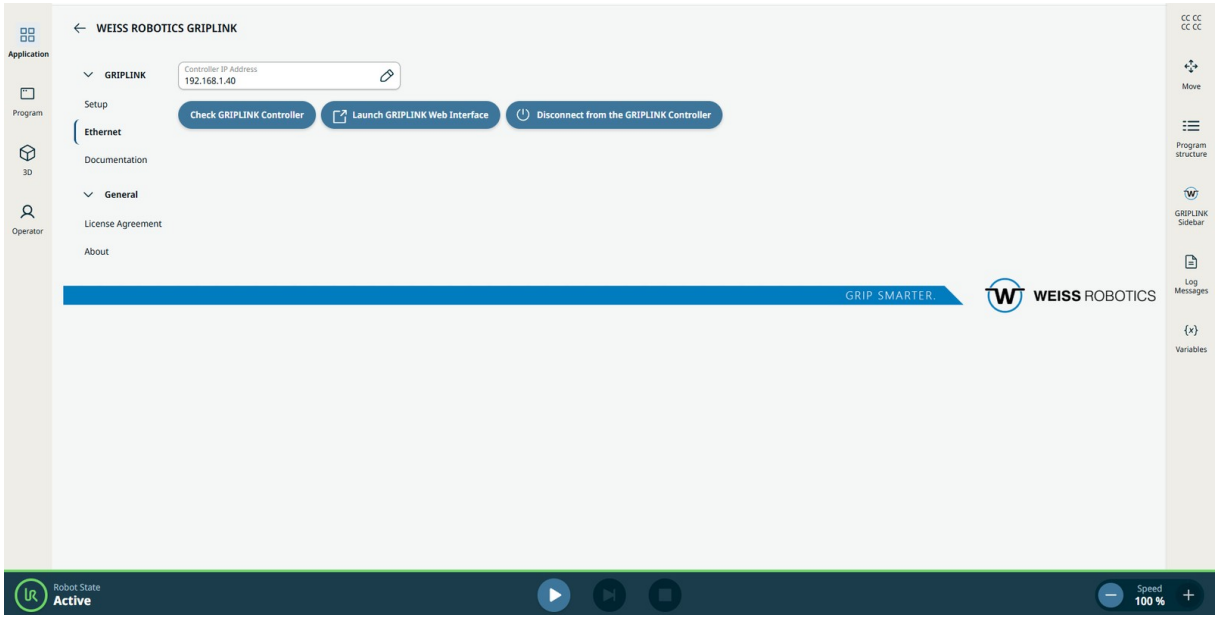

Figure 2: Ethernet commissioning in the GRIPLINK Setup - Controller IP Address and actions.

2.3 Commissioning: Tool I/O

1. De-energize the Tool I/O interface (0 V) before connecting the device (see the hotplug warning at the end of this section).

2. Mount the device and connect the original Tool I/O cable.
3. Open **Application tab → URCaps → WEISS ROBOTICS GRIPLINK → Setup**.
4. Set **Interface** to **Tool I/O**. The plugin configures 24 V and Tool I/O communication automatically.
5. Open **Application tab → Communication → Robot → Tool I/O** and change **Controlled By** from **User** to **GRIPLINK**.
6. Press **Rescan**. The detected devices appear with serial number and firmware.

To recover the Tool I/O interface, use the **Restart Interface** button and rescan again. If no devices are found, repeat steps 1 to 5. Mounting, wiring, and the connect and disconnect procedures are described in detail in the Tool I/O integration chapter.

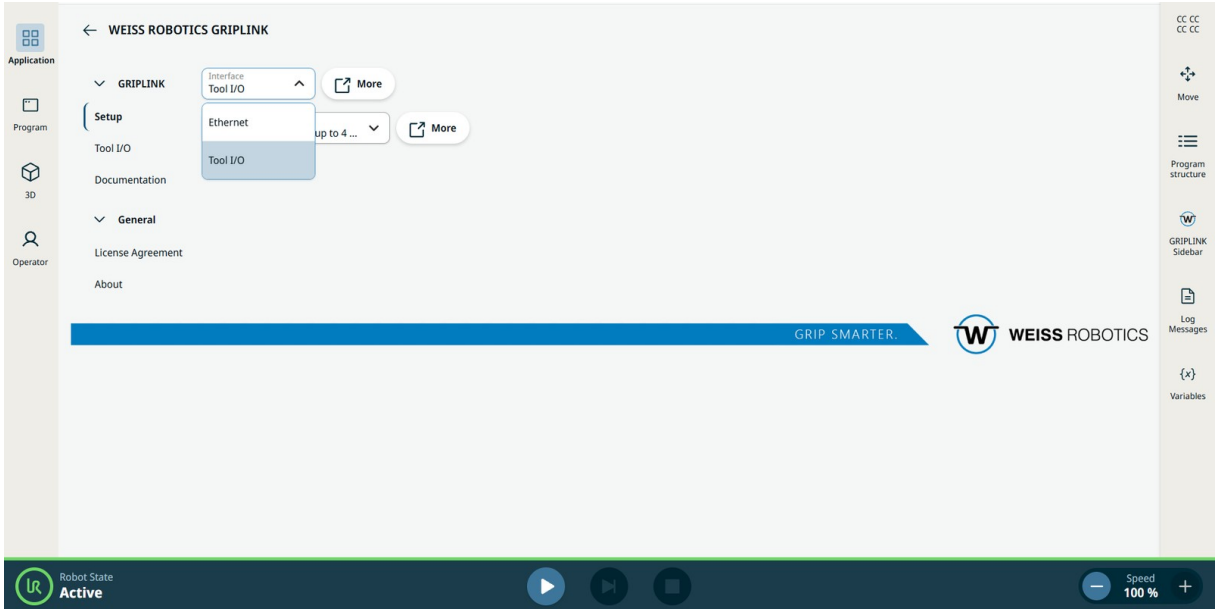

Figure 3: Interface selection in the GRIPLINK Setup - choose Tool I/O.

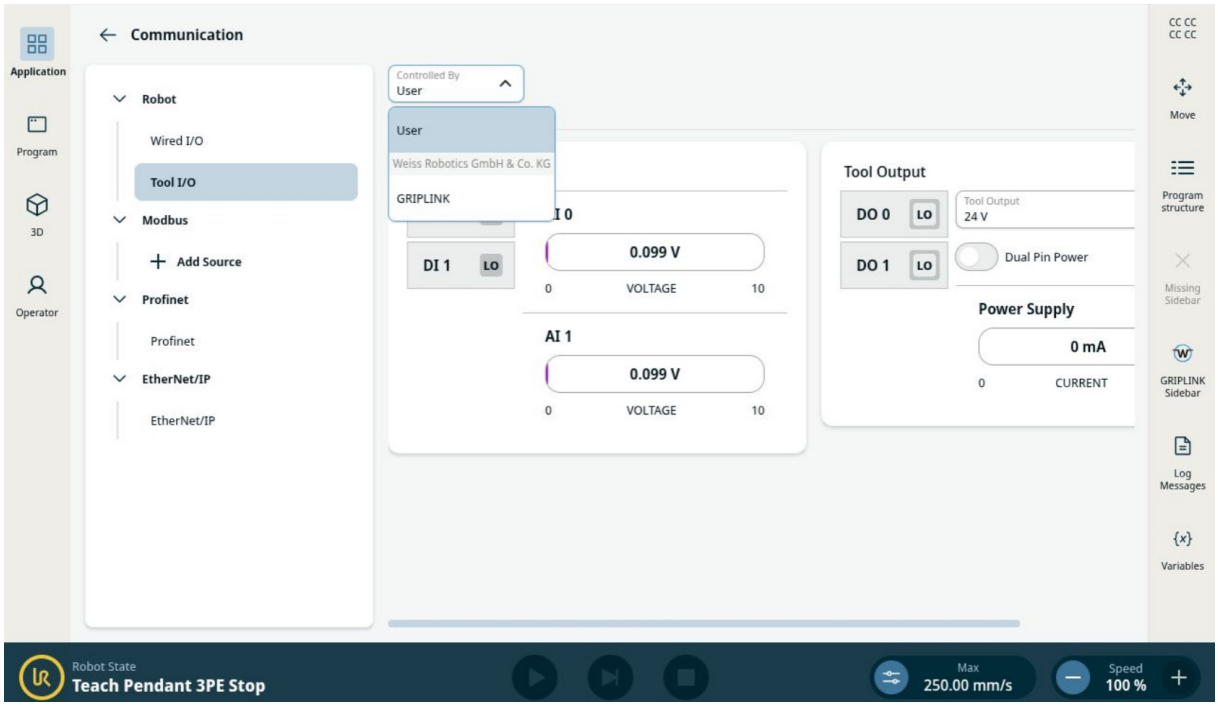

Figure 4: Robot settings Communication → Robot → Tool I/O - set Controlled By to GRIPLINK.

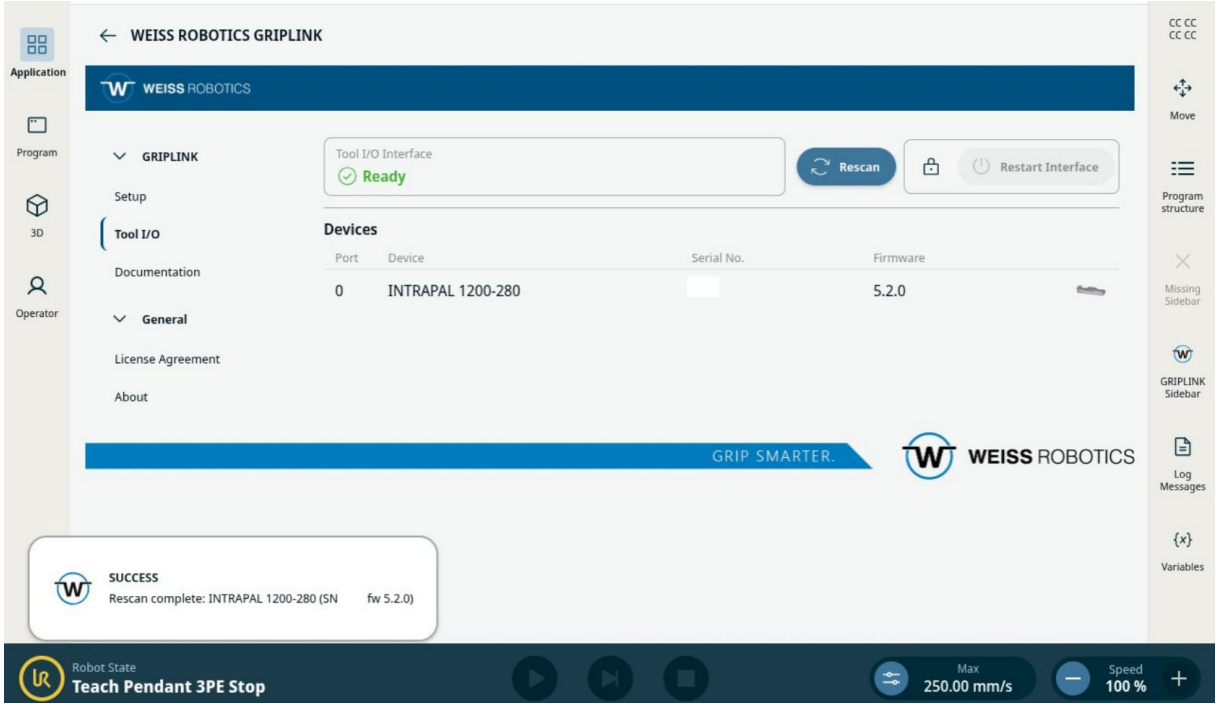

Figure 5: Successful rescan - the detected device appears with serial number and firmware.

⚠ Never "hotplug" (connect or disconnect) tools or Tool Cable Adapters while the Tool Connector is powered. Doing so can permanently damage the robot's internal tool flange connector or the end-effector. Always power off the robot before attaching or

detaching tools.

Behavior in Case of an Error

If the plugin or a device communication attempt errors, the running program stops, any robot motion stops, and an error message appears; the program does not continue while the fault is present. Typical causes: an interrupted controller connection (GRIPLINK controller without power, disconnected Ethernet cable, incorrect network configuration), an invalid parameter, or a device reporting an error, including a device already in or entering FAULT. The plugin shows a popup naming the failed command with a **Retry** option.

3 Application node and setup

Configure the device connection in the **WEISS ROBOTICS GRIPLINK** application node: **Application tab → URCaps → WEISS ROBOTICS GRIPLINK**. The page shows the navigation rail on the left and the configuration content on the right.
GRIPLINK adds one entry under **Application → URCaps**.

Group	Entries
GRIPLINK	**Setup** of the communication interface (**Ethernet** or **Tool I/O**), **Documentation** (opens this manual as a PDF).
General	**License Agreement**, **About** (plugin version, description, and supported commands).

Setup

The Setup page holds only two selectors, **Interface** and **Port Configuration**, and configures how the plugin reaches the device. Select **Interface** first: choosing a mode adds a matching **Ethernet** or **Tool I/O** entry to the navigation rail, with that mode's own fields on its own page.

Interface	Meaning
Ethernet	GRIPLINK communicates with an external GRIPLINK controller over the robot network at the configured IP address.
Tool I/O	The device is connected directly to the robot's Tool I/O interface; no external controller or IP address is needed.

The **Port Configuration** selector (present in both modes) sets how many ports are addressable:

Port Configuration	Meaning
Standalone Controller	The robot communicates with a single GRIPLINK controller; ports **0–3** are selectable in the program nodes.
Proxy Controller	Makes a GRIPLINK controller with up to 32 ports available; ports **0–31** are addressable.

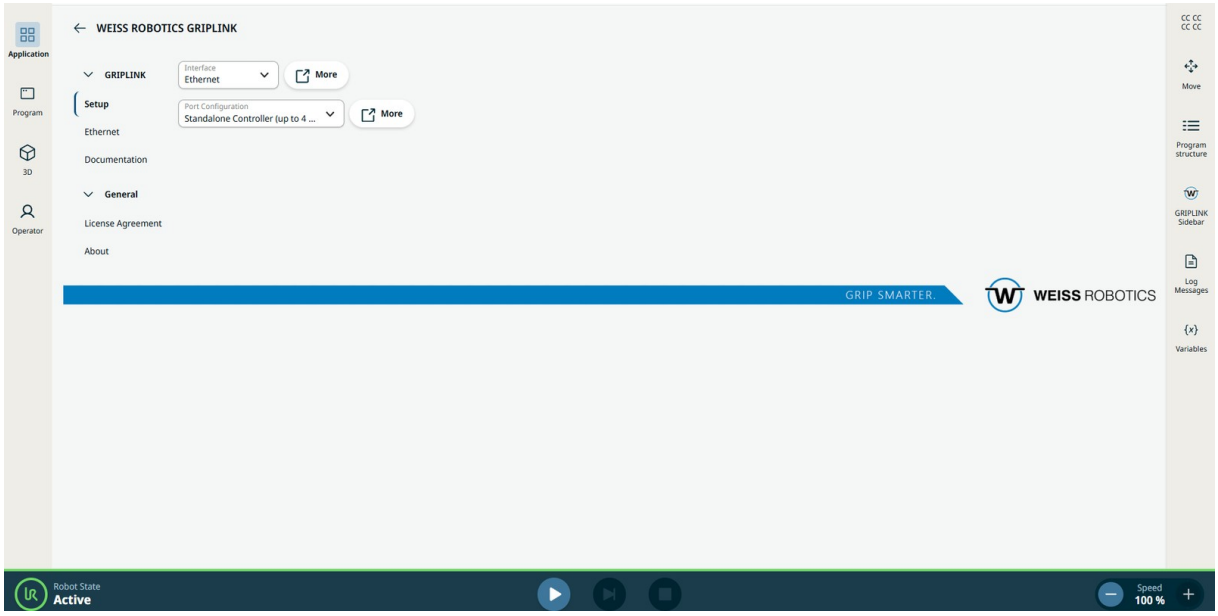

Figure 6: The Setup page - Interface and Port Configuration selectors.

Ethernet Mode

With **Interface** set to **Ethernet**, a separate **Ethernet** page appears in the navigation rail, showing the **Controller IP Address** field (default `192.168.1.40`) and the actions **Check GRIPLINK Controller**, **Launch GRIPLINK Web Interface**, and **Disconnect from the GRIPLINK Controller**.

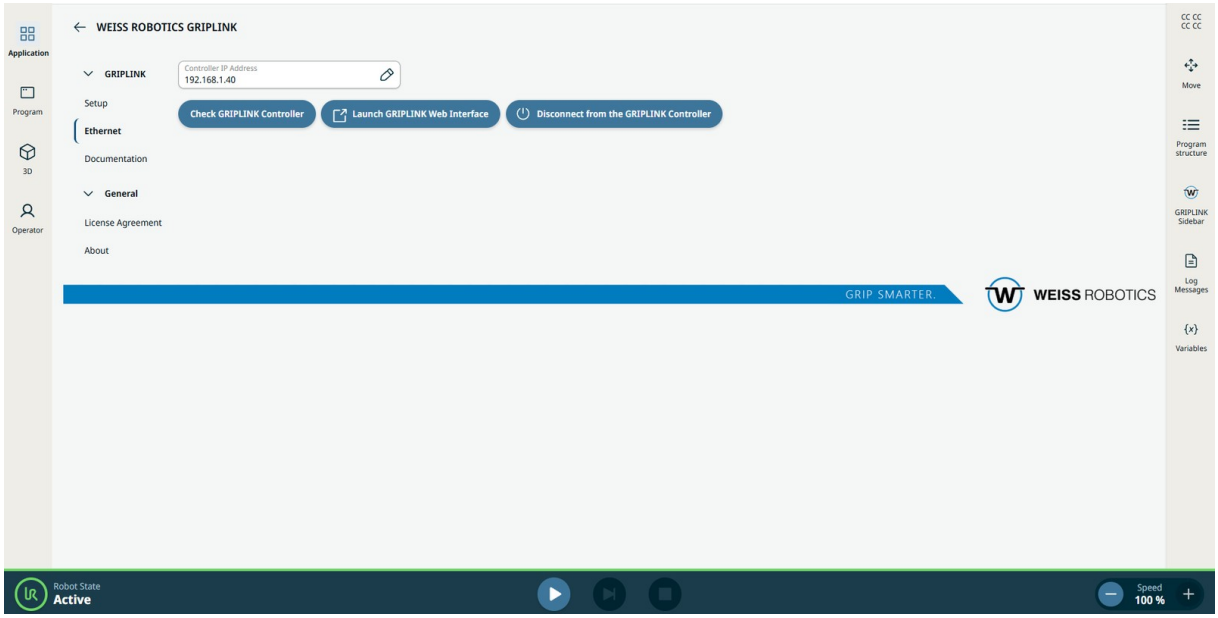

Figure 7: Ethernet page - Controller IP Address and actions.

Action	Description
Check GRIPLINK Controller	Verifies the controller is reachable and reports its type, serial number, and firmware. Success

	shows "Controller {ip} is reachable."
Launch GRIPLINK Web Interface	Opens the controller's web interface in the Chromium browser; available once the controller is reachable.
Disconnect from the GRIPLINK Controller	Closes the open GRIPLINK socket on the next program run, after a "Disconnect from GRIPLINK?" confirmation.

Configure the GRIPLINK controller itself through its web interface, via the **Launch GRIPLINK Web Interface** button on the Ethernet page, or directly in a browser at its IP address (default `192.168.1.40`).

Tool I/O Mode

With **Interface** set to **Tool I/O**, a separate **Tool I/O** page appears in the navigation rail, showing the interface state, the **Rescan** action, and the detected-device table. These are covered together with flange wiring and power requirements in the Tool I/O integration chapter.

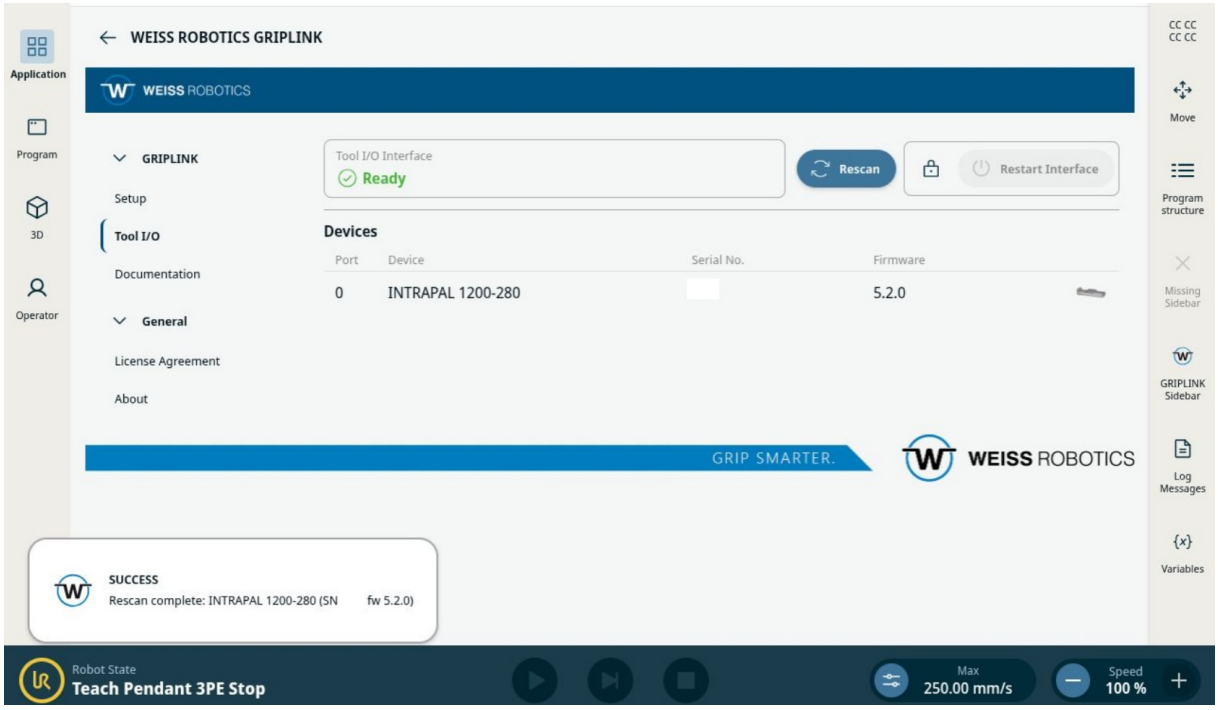

Figure 8: Tool I/O page with a detected device - Ready state, INTRAPAL 1200-280 on port 0 after a successful rescan.

4 Tool I/O integration

This chapter covers Tool I/O mode: mounting and powering the device on the tool flange, how connected devices map to ports, the interface states shown on the Tool I/O page, and when to use Rescan or Restart Interface.

Mounting on the Robot Flange

Mount the device on the robot flange as described in that device's own operating manual.

Connecting and Disconnecting a Device

Proceed as follows to connect or disconnect a device on the tool flange.

To connect a device

1. Mount the device on the tool flange and connect the Tool I/O cable.
2. On the Setup page, set **Interface** to **Tool I/O**. The plugin powers the flange with 24 V and enables Tool I/O communication automatically.
3. Open **Application tab → Communication → Robot → Tool I/O** and change **Controlled By** from **User** to **GRIPLINK**.
4. Press **Rescan**; newly attached devices are detected only by a rescan.
5. Confirm the device appears in the device table with its serial number and firmware version.

To disconnect a device

1. Stop the robot program. If the device is holding a workpiece, release it first.
2. Open **Application tab → Communication → Robot → Tool I/O** and change **Controlled By** from **GRIPLINK** to **User**.
3. De-energize the robot so the Tool I/O interface reaches 0 V (see the warnings below).
4. Disconnect the cable and remove the device from the flange.

Power Supply

The power supply must be rated for the connected devices. See the device's operating manual for the power supply requirements.

Only perform installation, removal, and wiring work while de-energized. Confirm the Tool I/O interface is at 0 V before connecting, disconnecting, or replacing a device.

Ports and Device Assignment

In Tool I/O mode, the device communicates through the robot's Tool I/O interface, connected at the Tool Connector on the tool flange. The plugin assigns every connected device to a logical port, the same port number used in the program nodes and the sidebar. When **Setup → Interface** is Tool I/O, the plugin configures the Tool I/O interface (24 V supply and communication settings) automatically.
In the simplest case - one gripper, no finger modules - the device is **port 0**.
More generally, every device occupies a block of four ports: the gripper itself is the first port of its block, and its finger modules (e.g. a modular INTRAPAL setup) follow on the next ports of the same block. Up to 4 grippers and 16 ports are possible in total (this 16-port layout is specific to the Tool I/O interface):

Port	Device
0	Gripper 1
1–3	Finger modules of gripper 1
4	Gripper 2
5–7	Finger modules of gripper 2
8	Gripper 3
9–11	Finger modules of gripper 3
12	Gripper 4
13–15	Finger modules of gripper 4

Rescan shows which ports are actually populated; the program nodes and the sidebar use exactly these port numbers.

Which block a gripper appears in is decided by the station address configured on the device (address 1 → ports 0–3, address 2 → ports 4–7, ...). With the factory default of 1, a single gripper is always port 0. See the device's operating manual for changing the station address.

Connected devices must not exceed the robot's permitted Tool I/O current draw; exceeding it can cause malfunctions or damage.

Operate the Tool I/O interface only with the original cable for the device. Other cables or wiring can impair the data connection and cause damage.

Only the GRIPLINK Plugin may access the Tool I/O interface during operation. Other URCapXs with access to the Tool I/O interface can cause conflicts, communication errors, or unreliable operation.

Device-specific GRIPKIT EASY parameters can be set as needed using the "GRIPKIT Configurator" application (see <https://weiss-robotics.com/de/gripkit/gripkit-easy/>).

Interface States

The **Tool I/O Interface** field on the Tool I/O page reports the interface state:

State	Meaning
Ready	Operational, at least one device available.
Waiting for Tool I/O access	PolyScope X has not yet granted the plugin Tool I/O control.
Scanning devices…	Searching the Tool I/O interface for devices (typically 2–5 seconds).
Waiting for device	Access granted, no device detected; connect and power a device, then press **Rescan**.
Tool I/O interface error	The interface reported an error; use **Restart Interface** to recover.
Starting…	The interface is starting up.
Interface offline	The interface is temporarily unavailable.

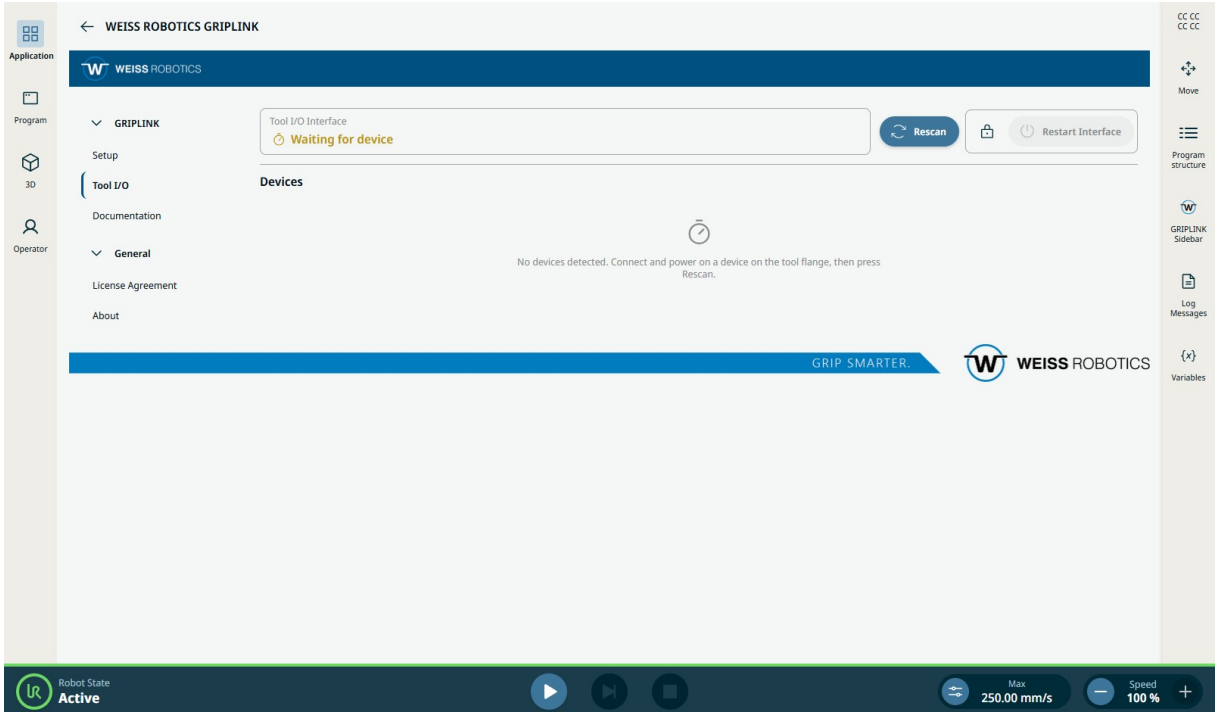

Figure 9: Tool I/O page - interface state, Rescan, and the device table.

Rescan and Restart

Action	Description

Rescan	Searches the Tool I/O interface for devices and refreshes the device table. Press it after attaching or powering a device. Unavailable while a robot program is using the interface.
Restart Interface	Stops and restarts the Tool I/O interface. Shows a confirmation dialog; normally locked, must be unlocked before use. Use it if the interface stays in an error state.

Device Table

For each port, the device table lists **Device**, **Serial No.**, and **Firmware**.

Do not disconnect the connection while a robot program is running: risk of injury or damage from falling parts.

While a connection is active on the Tool I/O interface, **Rescan** is unavailable; **Restart Interface** stays available for recovery.

Tool I/O-specific Status Codes

These status codes describe the state of the Tool I/O interface. They are distinct from the general GRIPLINK protocol status codes in the Status Codes appendix.

Status code	Name	Description
100	TOOL_IO_NOT_GRANTED	Tool I/O access has not been granted by PolyScope X.
101	TOOL_IO_REVOKED	Tool I/O access was revoked by PolyScope X.
102	TOOL_IO_CONNECTING	A device scan on the Tool I/O interface is in progress.
103	DAEMON_BOOTING	The Tool I/O interface is starting up.
104	DAEMON_SHUTDOWN	The Tool I/O interface is shutting down.

5 Manual control via Sidebar and Smart Skills

The plugin provides two surfaces for controlling a device manually, without a robot program: the **GRIPLINK Sidebar** for live control and diagnostics, and **Smart Skills** for preconfigured one-tap actions on the Move screen. Both command the device directly; neither generates URScript nor modifies the robot program.

5.1 Sidebar

The **GRIPLINK Sidebar** provides a user-friendly interface for the connected devices recognized by the current setup. It is opened from the sidebar of the robot screen and supports the following tasks in particular:

- Commissioning
- Diagnostics
- Gripping and releasing during setup and testing
- Viewing presets and their parameters
- Monitoring important device data

The Sidebar displays the data and provides the supported functions for the currently selected port. Controls are enabled or disabled automatically depending on the current device state.

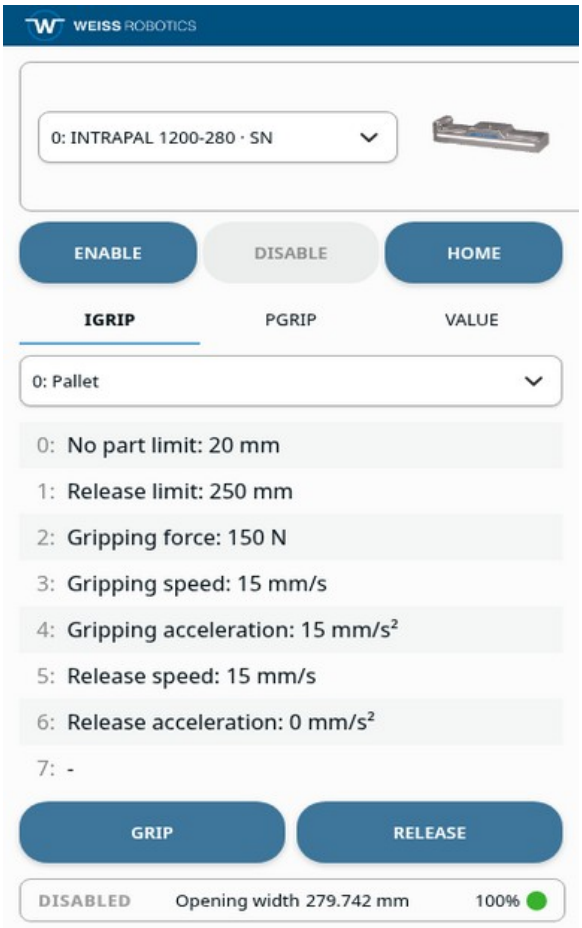

Figure 10: GRIPLINK Sidebar (IGRIP tab).

Sidebar Layout

The upper section of the Sidebar contains the device selection as well as the buttons for controlling the selected device.

Control	Description
Device selection (combo box)	Dropdown menu for selecting the desired device, e.g. 0: INTRAPAL 1200-280 · SN... The port number, device type, and serial number are displayed, together with a picture of the connected device.
ENABLE / DISABLE (buttons)	Enable or disable the selected device. The resulting state is displayed in the status bar, e.g. DISABLED.
HOME (button)	Performs a homing procedure for the selected device.

The status bar at the lower edge of the Sidebar displays cyclically updated data of the selected device:

Display	Description
State	The current device state (e.g. DISABLED, RELEASED, HOLDING, FAULT).
Telemetry data	The device's primary measured value, e.g. the opening width (in mm) for gripper modules.
Condition (status indicator)	The device condition as a percentage with a colored status indicator. Green indicates an operational device; amber and red indicate a faulty condition.

IGRIP Tab - Working with Grip Presets

IGRIP ("Index GRIP") grips or releases with the grip presets stored for the selected device. The tab displays the configured presets and their parameters (see Figure 10).

Control	Description
Preset selection (combo box)	Dropdown menu with the available grip presets

	0–7. Each row displays the index and the tag, e.g. 0: Pallet.
Preset parameter display	Lists the parameters stored in the selected preset. Depending on the device, these include: No part limit (mm), Release limit (mm), gripping force (N), gripping and release speed (mm/s) and acceleration (mm/s²).

To grip or release by preset

1. Select a preset from 0 to 7.
2. Press **GRIP** or **RELEASE**.

Typical use

- Testing a newly configured preset.
- Viewing presets and their configured parameters.
- Quick teach-in: select a preset, execute several grip cycles via the Sidebar, and observe the behavior.
- Checking whether a part is reliably detected with the set limits.

PGRIP Tab - Flexible Grip / Release

PGRIP ("Parameter GRIP") enables parameterized, flexible gripping and releasing with explicit values instead of a stored preset (analogous to the FLEXGRIP and FLEXRELEASE commands).

Control	Description
Grip - Position (slider)	Target position for gripping in mm.
Grip - Force (slider)	Gripping force in N.
Grip - Speed (slider)	Gripping speed; in Ethernet mode in mm/s, in Tool I/O mode as a percentage (%) of the maximum device speed. With **Auto speed** enabled, the device's default speed is used.
Release - Position (slider)	Target position for release in mm.
Release - Speed (slider)	Release speed, with its own **Auto speed** checkbox. Releasing uses position and speed only.

To grip or release with explicit parameters

1. Adjust position, force, and speed with the sliders (or enable **Auto speed**).

2. Press **GRIP** or **RELEASE**.

Applications

- Determining a suitable gripping position and gripping force for a new part.
- Testing how gently or quickly a part can be gripped or released.
- Checking tolerances (e.g. varying workpiece widths) before storing a fixed preset.

The **PGRIP** and **VALUE** tabs require a connected and enabled device; without one, both tabs are disabled.

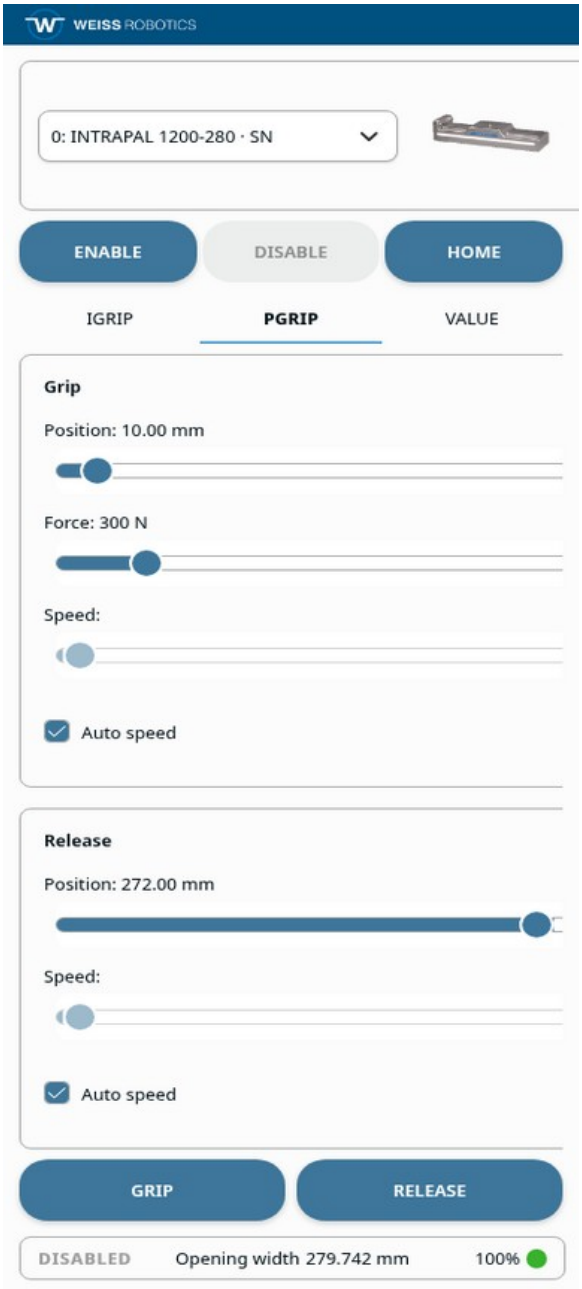

Figure 11: GRIPLINK Sidebar (PGRIP tab) - flexible grip and release parameters.

VALUE Tab - Monitor Device Data

The **VALUE** tab displays the current values of the selected device, read-only. The available indices depend on the device type; the values are updated cyclically.

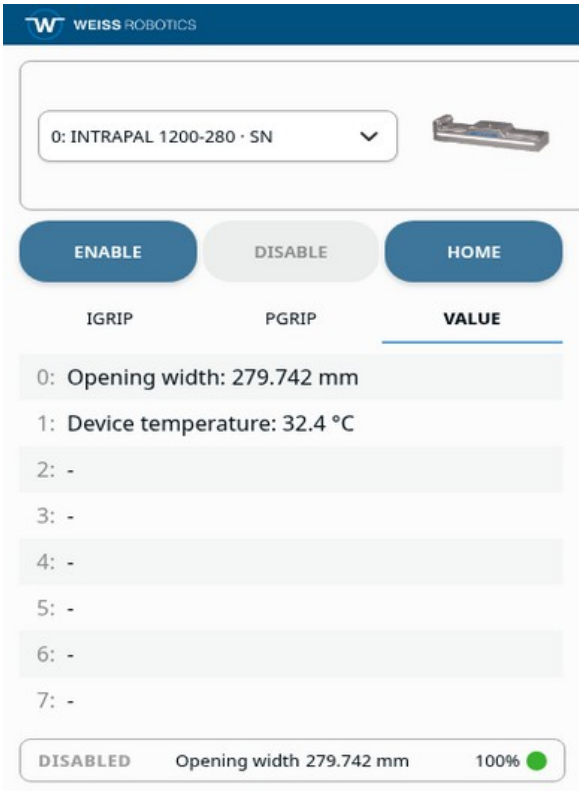

Figure 12: GRIPLINK Sidebar (VALUE tab) - live device telemetry (opening width, temperature).

Example for the INTRAPAL series:

Index	Value
0	Opening width (mm)
1	Device temperature (°C)
2–7	Not used in the INTRAPAL series

Typical use

- Checking whether a part has been correctly detected.
- Monitoring the opening width during commissioning and testing.
- Checking the device temperature during long cycles.
- Assistance in selecting appropriate limit values (No part limit, Release limit, etc.).

Notes on Using the Sidebar

⚠ Sidebar and Smart Skill actions are mutually exclusive with an open URScript connection (GL_CONNECT), even while the program is only paused. Sidebar and Smart Skill actions become available again once the connection is closed (program ended or DISCONNECT executed).

After repeated rapid presses, the buttons are temporarily disabled, then reactivated automatically, to prevent communication and robot-control overload.

The Sidebar does not modify existing program nodes; use it only for manual operation, diagnostics, and commissioning. Make permanent process changes in the robot program.

5.2 Smart Skills

Smart Skills are preconfigured manual actions triggered with a single tap. A skill is configured once under **Application → Smart Skills**; the operator then triggers it from the Smart Skills tab of the **Move** screen, without opening a program or the Sidebar. A skill acts on the device directly, over the same live channel as the Sidebar; it neither generates URScript nor modifies the robot program.
GRIPLINK adds three Smart Skills:

Icon	Title	Description
	**GRIPLINK Grip**	Grip with the configured preset or parameters on the selected port.
	**GRIPLINK Release**	Release with the configured preset or parameters on the selected port.
	**GRIPLINK Home**	Reference (home) the gripper on the selected port.

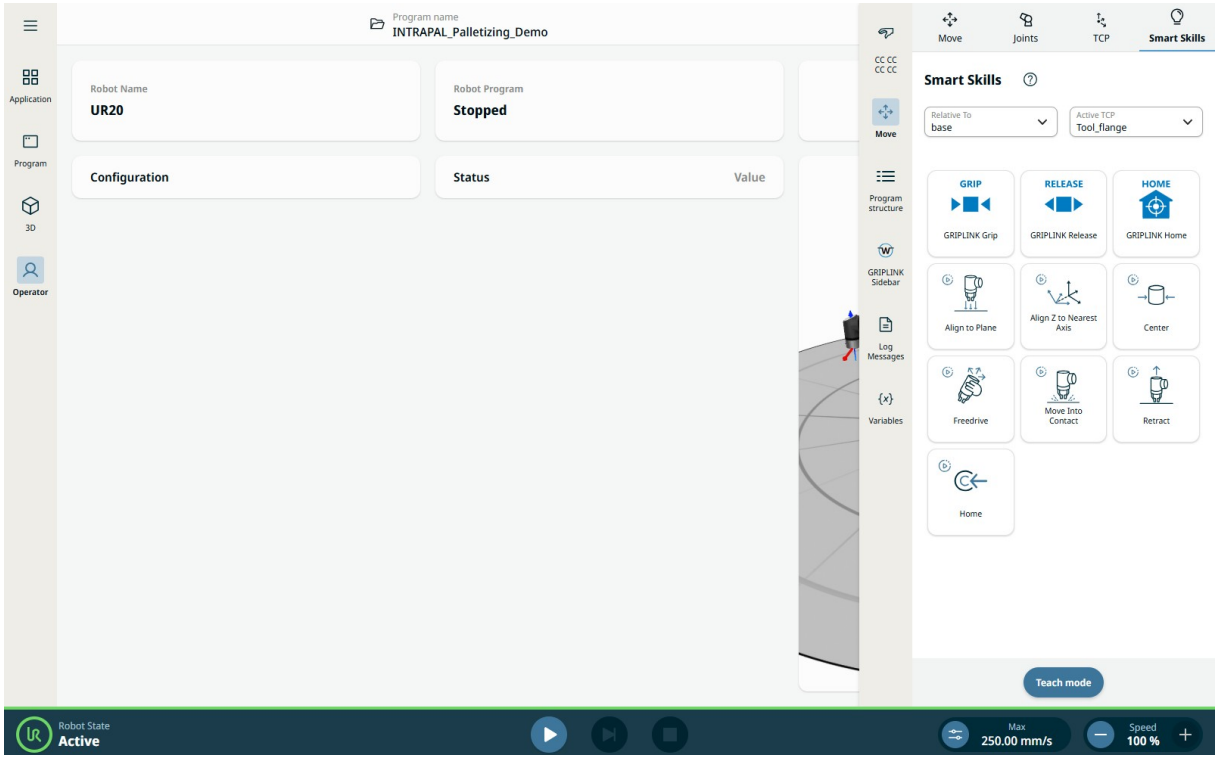

Figure 13: The Smart Skills tab in the Move screen - GRIPLINK's three skills alongside PolyScope X's built-in skills.

Use Smart Skills for manual operation and commissioning, like the Sidebar; they do not modify the robot program. To make a gripper action part of the program, use a GRIPLINK program node instead.

Managing Skill Instances

Configure each Smart Skill under **Application → Smart Skills**. The left-hand list separates **Activated** skills (shown in the Move screen's Smart Skills tab) from **Deactivated** ones (configured but hidden). **Create Smart Skill** duplicates an existing skill, or starts a new one from scratch.

Action	Description
Rename	Change the instance's display name.
Duplicate	Create a second, independently configured instance of the same skill, e.g. a "Grip" instance for port 0 and a separate one for port 1.
Deactivate / Delete	Remove it from the Move screen. GRIPLINK's three built-in skills remain in the catalog and are deactivated; a duplicated instance can also be deleted.

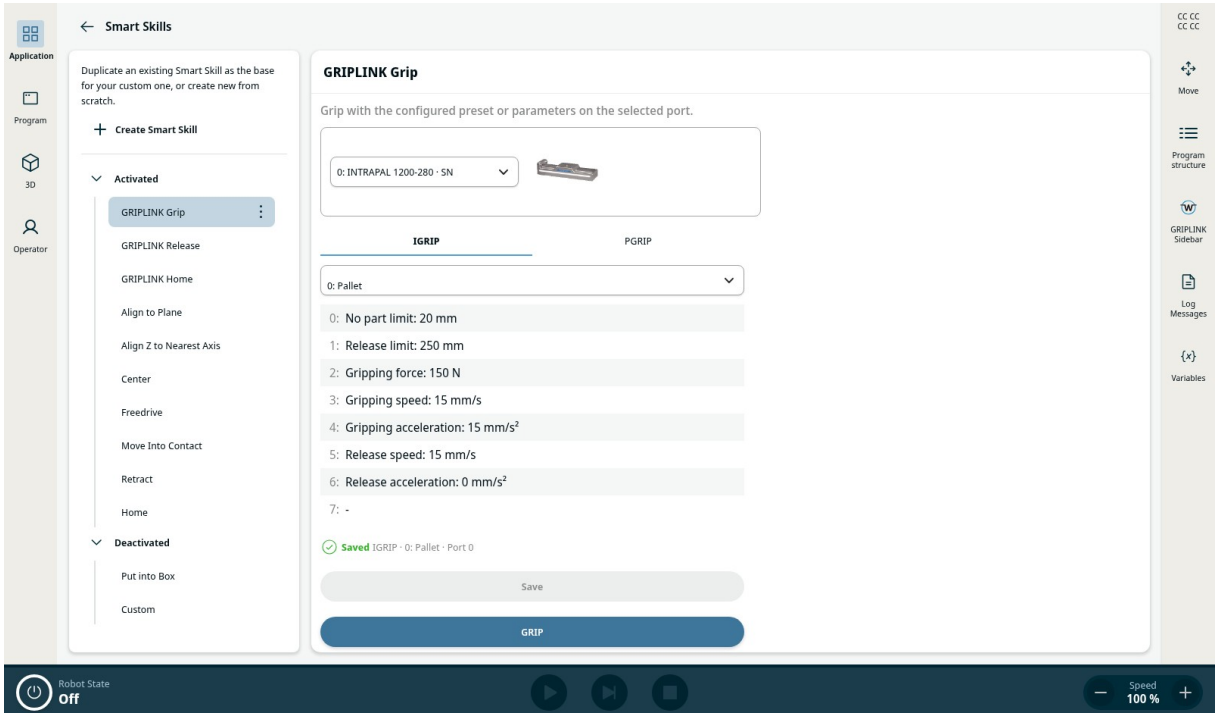

Figure 14: GRIPLINK Grip skill configuration, opened from the Smart Skills catalog.

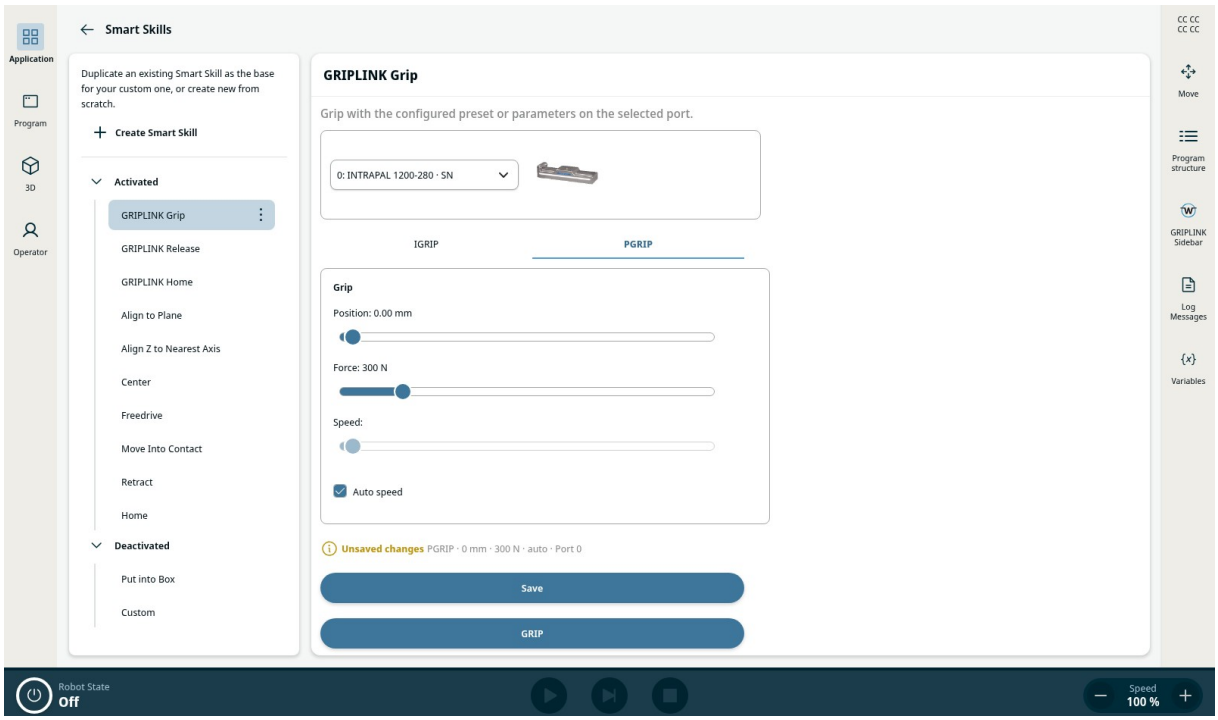

Figure 15: GRIPLINK Grip skill (PGRIP tab) - grip with explicit position, force, and speed.

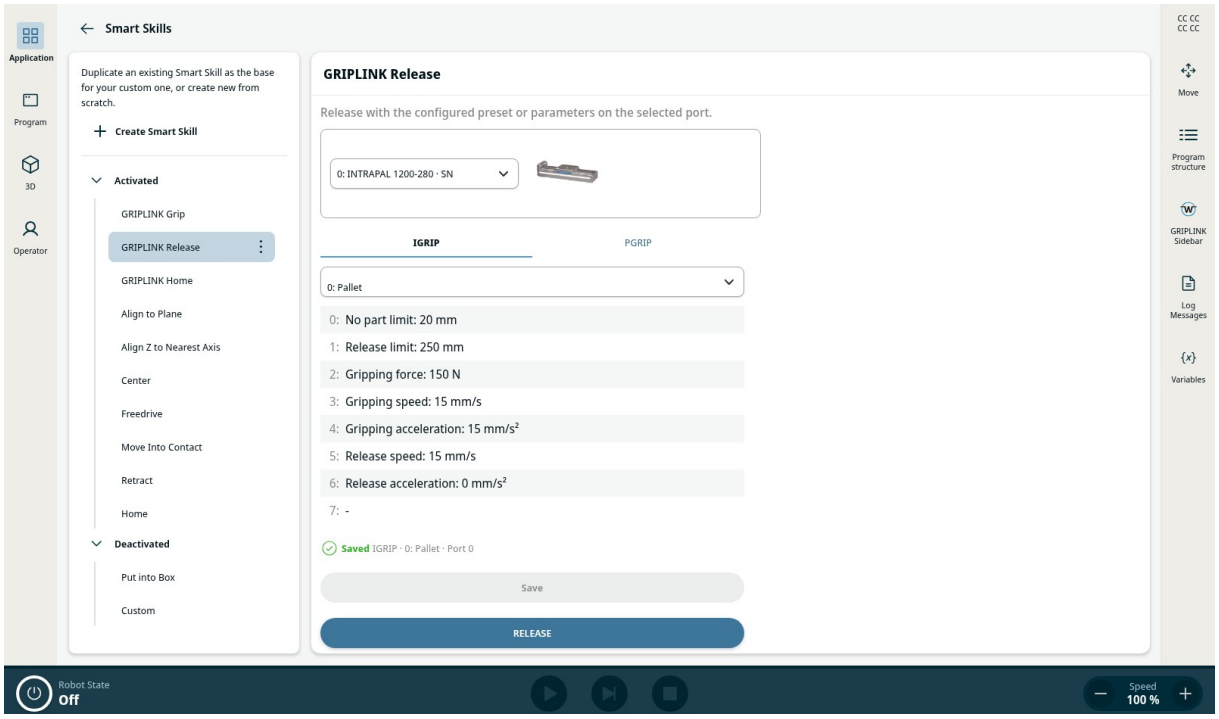

Figure 16: GRIPLINK Release skill (IGRIP tab) - release by stored preset.

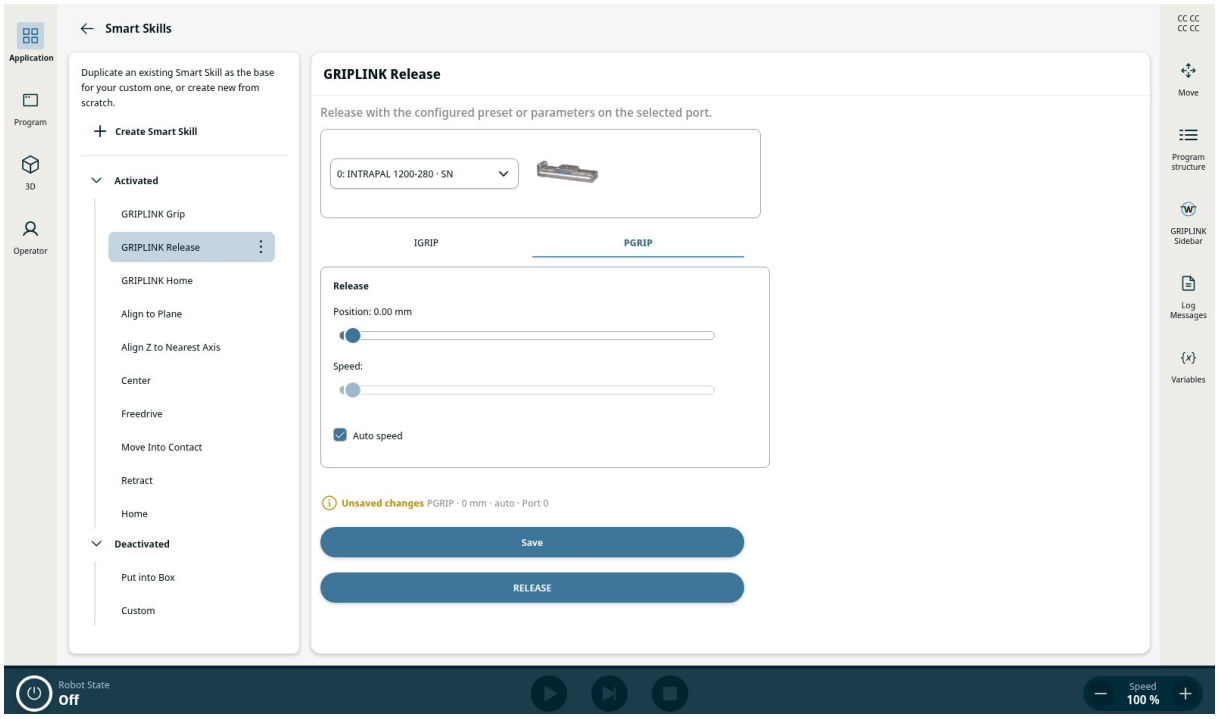

Figure 17: GRIPLINK Release skill (PGRIP tab) - release with explicit position and speed.

Configuring a Skill - Port, Mode, and Preset

Opening a skill shows the same controls as the matching Sidebar tab, scoped to that one instance:

Control	Description
Port	The controller port this skill instance acts on, with a live picture of the connected device.
IGRIP / PGRIP (Grip and Release only)	Choose a stored preset (the same eight-slot table as the Sidebar IGRIP tab), or set position, force, and speed directly (the same sliders and **Auto speed** checkbox as the Sidebar PGRIP tab, including the mode-dependent speed unit).

GRIPLINK Home requires only the port.
Each skill instance keeps its own port, mode, and values, independent of the Sidebar and every other instance.

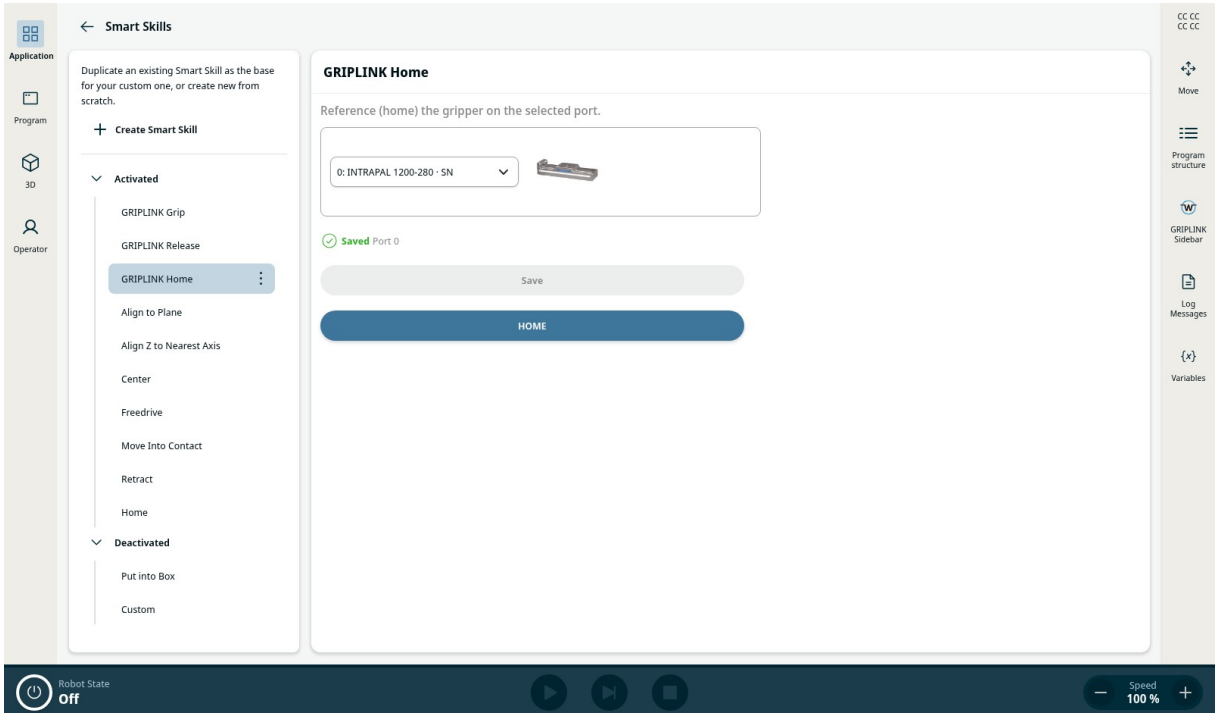

Figure 18: GRIPLINK Home skill configuration - the simplest skill, port only.

Draft, Save, and the Unsaved-Changes Indicator

Changing a port, mode, preset, or slider value first edits a **draft**; it takes effect only when saved. A status line reports which state the panel is in:

Status	Meaning
Unsaved changes	The draft has edits not yet stored. A one-line summary always shows the current draft (e.g. "IGRIP · 0: Pallet · Port 0" or "PGRIP · 40 mm · 25 N · auto · Port 0"), so the action that will run is visible at a glance.

Saved	The draft matches the stored configuration and is retained when navigating away.

Save commits the draft and is enabled only while there are unsaved changes.

Availability and Validation

A skill's action button is enabled only when the matching Sidebar button would also be enabled for the current device state, and only when no other control channel (such as an open connection from a robot program) is currently commanding the device.

6 Programming with program nodes

6.1 How to read the command reference

Basic Command Sequence for Programs

Use this sequence as the default pattern for a robot program; skip optional steps only when the device type and application design allow it.

Step	Command / action	Purpose	Application decision
1	CONNECT / GL_CONNECT	Open communication to the GRIPLINK controller or the Tool I/O interface.	Run once before device commands.
2	ASSERTDEVICE / GL_DEVASSERT	Confirm the expected device is present on the selected port.	Use at startup or after a tool change.
3	HOME / GL_HOME	Reference servo grippers that require homing.	Skip only if the device needs no referencing, or was already referenced.
4	ENABLE / GL_ENABLE	Enable the device for operation.	Run before gripping, releasing, or parameterized motion.
5	GRIP / RELEASE / FLEXGRIP / FLEXRELEASE	Execute the process action.	Preset commands for simple cycles; FLEX commands with position, force, speed.
6	DEVICESTATE / WSTR / VALUE / WAITVAL	Evaluate the result or monitor states and values.	Branch on HOLDING, NO PART, RELEASED, FAULT, or timeout.
7	DISABLE / DISCONNECT	Disable the device or close the connection.	Tool change.

Layer	Use it when	Documented in
Program nodes	The application is built visually in the program tree.	This chapter.
Pick / Place nodes	The program should branch	Pick / Place, this chapter.

	automatically on HOLDING or NO PART.	
URScript functions	The program needs script-level logic, loops, variables, or generated code.	URScript reference chapter.
Appendices	The program evaluates state values or diagnostics.	Device State and Status Codes.

Example: Minimal Program

The shortest complete gripper program - connect, prepare the device, then grip and release:

```
# Ethernet: controller IP address; Tool I/O: 127.0.0.1
GL_CONNECT("192.168.1.40", "sock_griplink")
GL_HOME(0, "sock_griplink")
GL_ENABLE(0, True, "sock_griplink")
GL_GRIP(0, 0, 1, "sock_griplink")
# ... robot motion ...
GL_RELEASE(0, 0, 1, "sock_griplink")
```

The same sequence applies to program nodes: CONNECT, HOME, and ENABLE in the initialization section (e.g. BeforeStart), GRIP and RELEASE in the main program.

6.2 Program-node command reference

GRIPLINK commands are inserted into the program tree via the GRIPLINK program nodes. The core node is the unified **GRIPLINK** node, exposing every command through the **Action** selector. The structural **GRIPLINK Pick** and **GRIPLINK Place** nodes (see Pick / Place in this chapter) build on the same GRIP/RELEASE actions and add automatic result branches.

Every node has a **More** dialog with a command description and a URScript preview of the generated call. The URScript reference chapter documents the underlying `GL_*` function for each action, including exact parameter types and ranges.

Basic Program Flow

To set up the initialization section (e.g. BeforeStart)

1. Add a **GRIPLINK CONNECT** node to open the connection to the controller.
2. Add a **GRIPLINK HOME** node for each device that requires referencing.
3. Add a **GRIPLINK ENABLE** node for each device that requires it.

Figure 19: GRIPLINK program nodes in the initialization section (BeforeStart).

In the main program, use **GRIPLINK GRIP** / **RELEASE** (or the **GRIPLINK Pick** / **Place** nodes) to handle parts, and evaluate the result to branch the flow.

CONNECT, ENABLE, and HOME in the initialization section ensure that every program run starts from a defined device state.

The Action Selector

Insert a **GRIPLINK** node and choose the command from the **Action** selector:

Action (UI label)	What it does
GRIPLINK CONNECT	Open the connection to the controller.
GRIPLINK DISCONNECT	Close the connection.
GRIPLINK ASSERTDEVICE	Verify the device type on a port (Vendor / Product ID).
GRIPLINK HOME	Reference (home) the device.
GRIPLINK ENABLE	Enable the device (optionally wait for the state transition).
GRIPLINK DISABLE	Disable the device (optionally wait).
GRIPLINK GRIP	Grip workpiece via grip preset.
GRIPLINK RELEASE	Release workpiece via grip preset.
GRIPLINK FLEXGRIP	Grip workpiece via parameters (position, force, speed, acceleration).
GRIPLINK FLEXRELEASE	Release workpiece via parameters (position, speed, acceleration).
GRIPLINK CLAMP	Turn force retention on/off.
GRIPLINK LED	Set the indicator LED to a preset.
GRIPLINK DEVICESTATE	Query the current device state (into a variable).
GRIPLINK VALUE	Query a device value (into a variable).
GRIPLINK WSTR	Wait for a device state change.
GRIPLINK WAITVAL	Wait until a device value reaches a target.
GRIPLINK SETVALUE	Set a device value.

GRIPLINK SGRIPCFG	Configure a grip preset.

Common parameter ranges: **Port** 0–3 (Standalone) or 0–31 (Proxy); **Preset** 0–7.

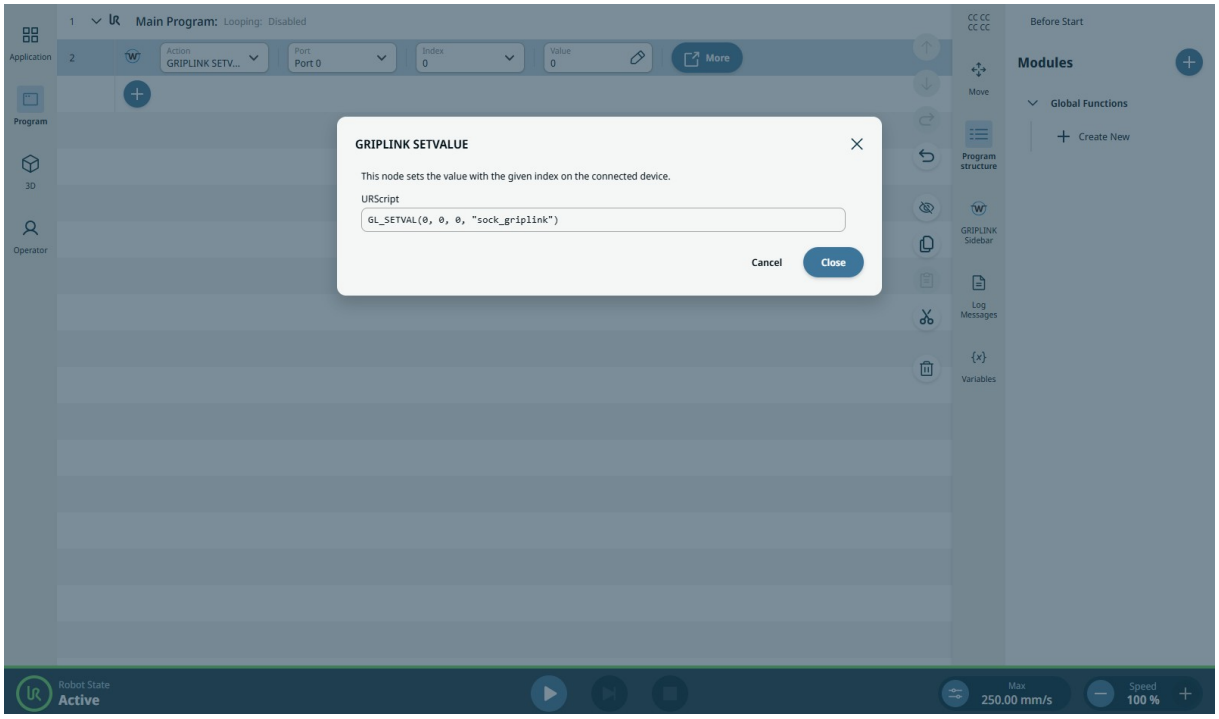

Figure 20: The More dialog (here for GRIPLINK SETVALUE) - description and a URScript preview of the generated call.

6.3 Establish Connection – CONNECT

The CONNECT command establishes the connection to the configured GRIPLINK controller. The IP address and socket name come from the application node; the node has no further fields.

Figure 21: GRIPLINK CONNECT - no fields beyond the Action selector.

6.4 Close Connection – DISCONNECT

The DISCONNECT command closes the connection CONNECT opened, for when a program needs to release the controller before ending or reconnecting. The node has no further fields.

Figure 22: GRIPLINK DISCONNECT - no fields beyond the Action selector.

6.5 Check a Connected Device – ASSERTDEVICE

The ASSERTDEVICE command checks whether the device on the selected port matches the expected **Vendor ID** and **Product ID**. Both IDs come from the device's IODD file (IO-Link Device Description) and are editable; press **Apply** to read them directly from the connected device instead of typing them.

Figure 23: GRIPLINK ASSERTDEVICE - Port, Vendor ID, Product ID, and Apply.

See the device's own operating instructions for its Vendor and Product IDs.

6.6 Reference the Device – HOME

The HOME command references the device on the selected port. **Port** is the only field.

Figure 24: GRIPLINK HOME - Port is the only field.

6.7 Enable the Device – ENABLE

The ENABLE command enables the device on the selected port. With **Wait for state transition** turned on, the node waits until the device reports the ENABLED state before continuing.

Figure 25: GRIPLINK ENABLE - Port and the Wait for state transition switch.

Some devices must be enabled before they can be used.

6.8 Disable the Device – DISABLE

The DISABLE command disables the device on the selected port, putting it into a safe, powered-down state. The node has the same fields as ENABLE: **Port** and **Wait for state transition**.

Figure 26: GRIPLINK DISABLE - Port and the Wait for state transition switch.

If the device is already DISABLED when this runs with Wait for state transition selected, a timeout error may result.

The demo program fault_clearing_via_disable clears a device fault with DISABLE followed by HOME.

6.9 Grip a Workpiece – GRIP

The GRIP command performs a grip with the selected grip preset (**Preset**, 0–7) on the selected port. **Blocking** is on by default, so the node waits until the grip finishes before continuing; turned off, the node returns immediately while the motion continues.

Figure 27: GRIPLINK GRIP - Port, Preset, and Blocking (on by default).

See the gripper module's own operating instructions for gripping parameters.

The demo programs simple_grip_release and griplink_demo demonstrate complete grip/release cycles with a stored grip preset (see Demo Programs).

6.10 Release a Workpiece – RELEASE

The RELEASE command releases the workpiece with the selected grip preset (**Preset**, 0–7) on the selected port. The node has the same fields as GRIP: Port, Preset, and Blocking.

Figure 28: GRIPLINK RELEASE - Port, Preset, and Blocking.

6.11 Grip Workpiece via Parameters – FLEXGRIP

The FLEXGRIP command performs a grip with the parameters **Position**, **Force**, **Speed**, and **Acceleration** instead of a stored grip preset. The parameters are set in a separate **Configure** dialog ("Parameters"); the node row itself shows only **Port**, **Configure**, and **Blocking**.

Figure 29: GRIPLINK FLEXGRIP - Port, Configure, and Blocking.

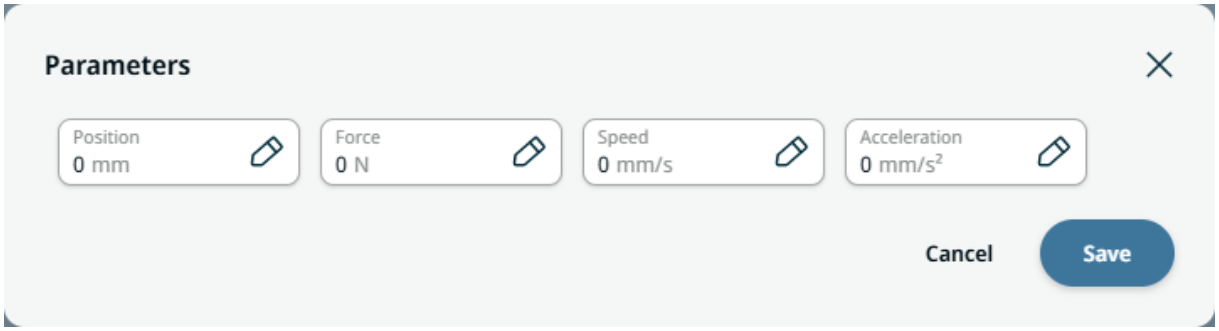

Figure 30: The FLEXGRIP Configure dialog - Position, Force, Speed, Acceleration.

See the gripper module's own operating instructions for motion parameters.

The meaning of **Speed** and **Acceleration** depends on the connection mode: physical mm/s and mm/s² in Ethernet mode; override percentages in Tool I/O mode (Speed 0–200 %, Acceleration 0–100 %, 100 % = nominal).

The demo program flexgrip_cycle demonstrates a grip/release cycle with explicit motion parameters.

6.12 Release Workpiece via Parameters – FLEXRELEASE

The FLEXRELEASE command releases the workpiece with the parameters **Position**, **Speed**, and **Acceleration**. In Tool I/O mode, **Speed** is a 10–100 % override and **Acceleration** a 0–100 % override; in Ethernet mode both are physical (mm/s, mm/s²).

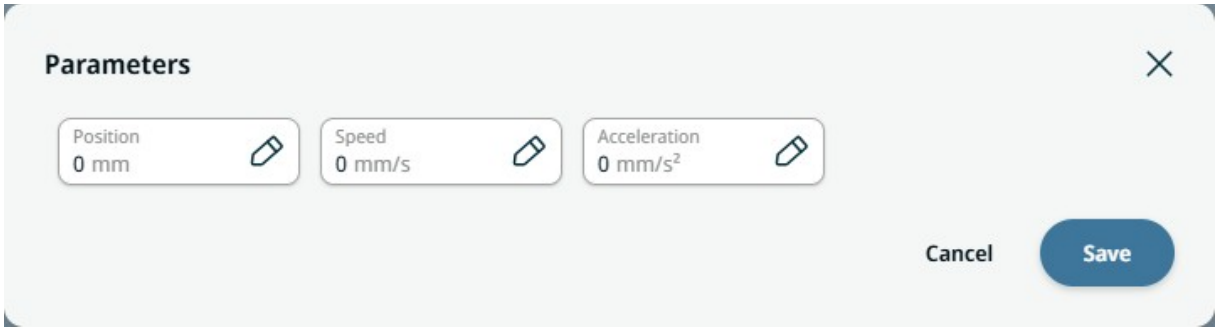

Figure 31: The FLEXRELEASE Configure dialog - Position, Speed, Acceleration.

The demo program async_preposition uses a non-blocking FLEXRELEASE to pre-position the fingers while the robot moves.

6.13 Pick / Place

For part handling, GRIPLINK provides two structural program nodes that combine a grip or release with automatic outcome branches, so the program does not need a separate DEVICESTATE query and If node.

GRIPLINK Pick

Performs a grip (GRIP or FLEXGRIP, per configuration) and creates two fixed result branches:

Branch	Runs when
HOLDING	The gripper is holding a part after the grip.
NO PART	The gripper closed without a part.

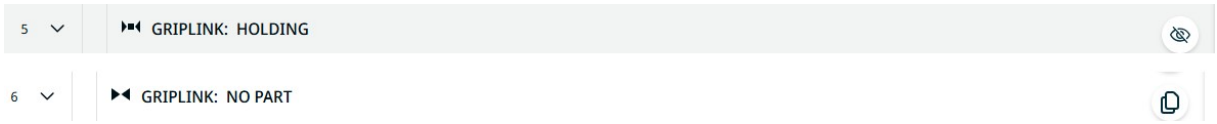

Figure 32: GRIPLINK Pick with its two result branches - HOLDING (part gripped) and NO PART (no part gripped).

GRIPLINK Place

Performs a release (RELEASE or FLEXRELEASE) and creates:

Branch	Runs when
RELEASED	The gripper successfully released the part.
NOT RELEASED	The gripper did not reach the RELEASED state (part still held, or an error occurred).

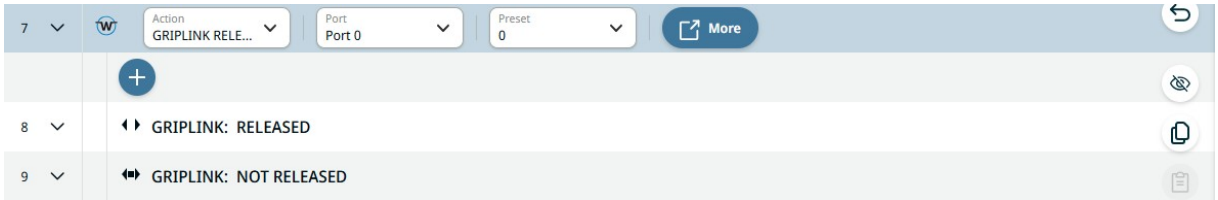

Figure 33: GRIPLINK Place with RELEASED / NOT RELEASED branches.

Pick and Place always wait (block) for the operation to finish before evaluating the result branch; only a completed operation yields a defined outcome.

The demo program pick_and_place is a complete pick-and-place sequence built with

these nodes (see Demo Programs).

6.14 Control Force Retention – CLAMP

The CLAMP command turns force retention (PERMAGRIP) on or off for the device on the selected port; the node has a single **Clamp** switch for this. Available on compatible devices (e.g. CRG series).

Figure 34: GRIPLINK CLAMP - Port and the Clamp switch.

6.15 Control the Status LED – LED

The LED command sets the device's status LED on the selected port to one of eight preset patterns. Available on compatible devices (e.g. CRG series).

Figure 35: GRIPLINK LED - Port and Preset only.

6.16 Query the Current Device State – DEVICESTATE

The DEVICESTATE command queries the device's current state on the selected port and stores it in a **Return** variable, to check afterward with an If node; see the Device State appendix for possible values. **Return** has no default; the node shows a validation warning until a variable is picked.

Figure 36: GRIPLINK DEVICESTATE - Port and Return (a variable must be selected).

Return lists application variables of type **Integer**. Create one on the **Application Variables** page of the **Application** view: press **Create Variable**, enter a name (e.g. DeviceState), and set **Type** to **Integer**. The variable persists between sessions; name and type are fixed at creation.

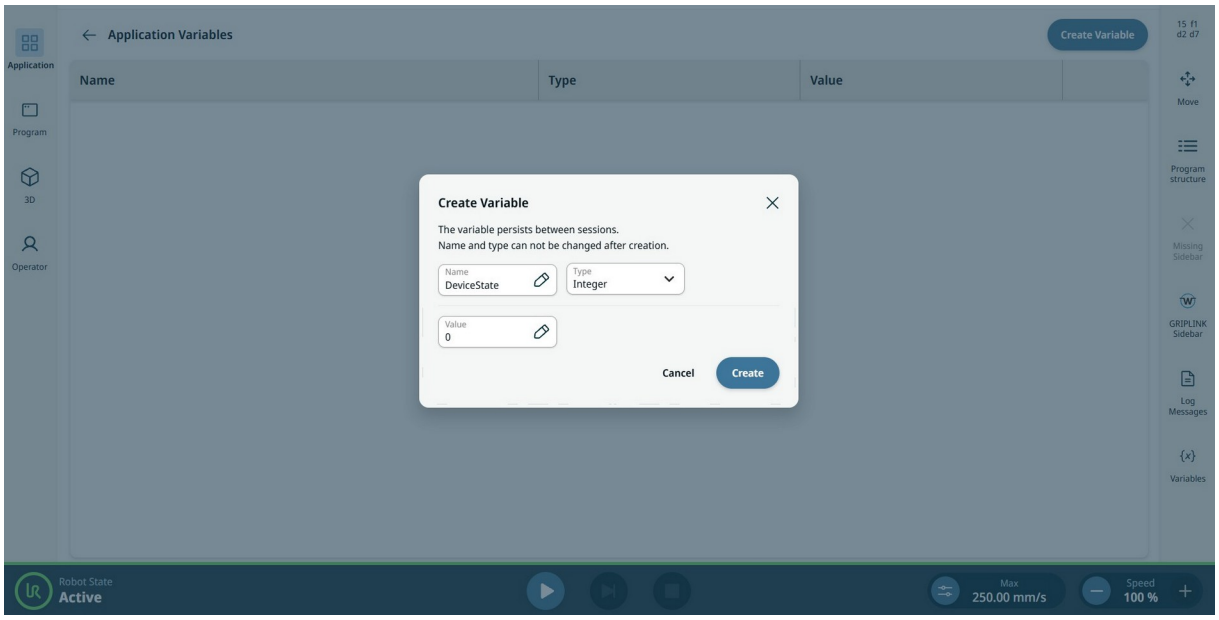

Figure 37: The Create Variable dialog - an Integer variable for the DEVICESTATE Return field.

This node reads whatever state the device is in right now. To wait for the next change first, use a WSTR node immediately before it.

6.17 Query a Device Value – VALUE

The VALUE command queries one numeric value (**Index** 0–7) of the device on the selected port and stores the result in a **Return** variable. The variable picker works like DEVICESTATE's, with one difference: it lists variables of type **Float**, matching the value's data type.

Figure 38: GRIPLINK VALUE - Port, Index, and Return.

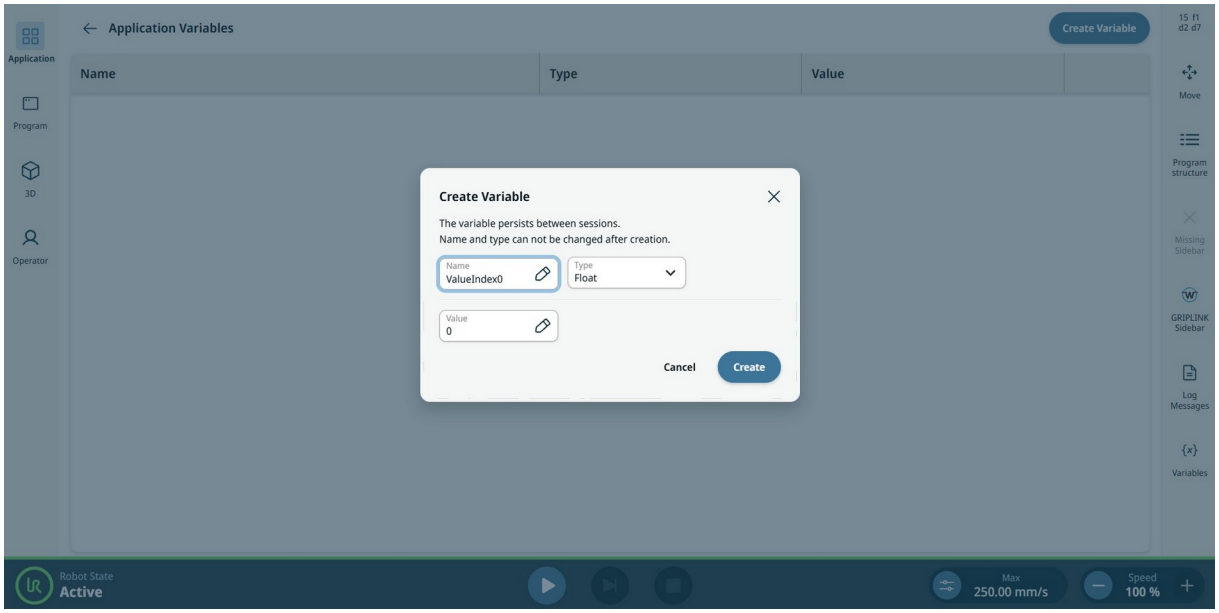

Figure 39: The Create Variable dialog - a Float variable for the VALUE Return field.

The demo program check_workpiece uses VALUE to gauge the gripped part by its finger position.

6.18 Wait for a Device State Change – WSTR

The WSTR command ("Wait for State Transition") waits until the device on the selected port next changes state, with no other action. **Port** is the only field, the same layout as HOME. Useful after starting a non-blocking GRIP/RELEASE/ENABLE/DISABLE elsewhere, or right before a DEVICESTATE query that must see a fresh state.

Figure 40: GRIPLINK WSTR - Port is the only field.

The demo program multi_grip_release uses WSTR to synchronize non-blocking grips across several ports.

6.19 Wait for a Device Value – WAITVAL

The WAITVAL command waits until a value reaches a target value, or a timeout elapses. The node row shows only **Port** and **Configure**; the parameters live in a separate dialog.

Figure 41: GRIPLINK WAITVAL - Port and Configure.

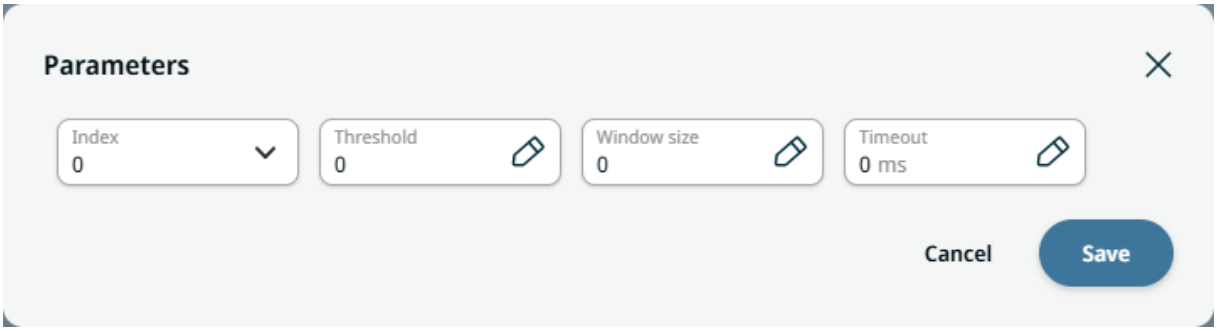

Figure 42: The WAITVAL Configure dialog - Index, Threshold, Window size, Timeout.

Parameters

`Index`	The value to watch (0–7).
`Threshold`	The target value.
`Window size`	The tolerance band around Threshold counted as "reached."
`Timeout`	How long to wait (ms) before giving up.

The demo program async_preposition uses WAITVAL to wait until the finger position reaches the target window.

6.20 Set a Device Value – SETVALUE

The SETVALUE command sets a value on one value channel (**Index**) of the device on the selected port. The fields are Port, Index, and an editable **Value** field. Support for this command depends on the device type.

Figure 43: GRIPLINK SETVALUE - Port, Index, and Value.

Setting device values can cause actuators to move.

6.21 Configure a Grip Preset – SGRIPCFG

The SGRIPCFG command configures a grip preset. The node row shows Port, a **Preset** selector (0–7), and **Configure**, which opens the dialog that programs that preset: a **Tag** naming the configuration,

and a **Value interpretation** template that relabels **P0–P7** for the selected device class, showing only the parameters that class uses.

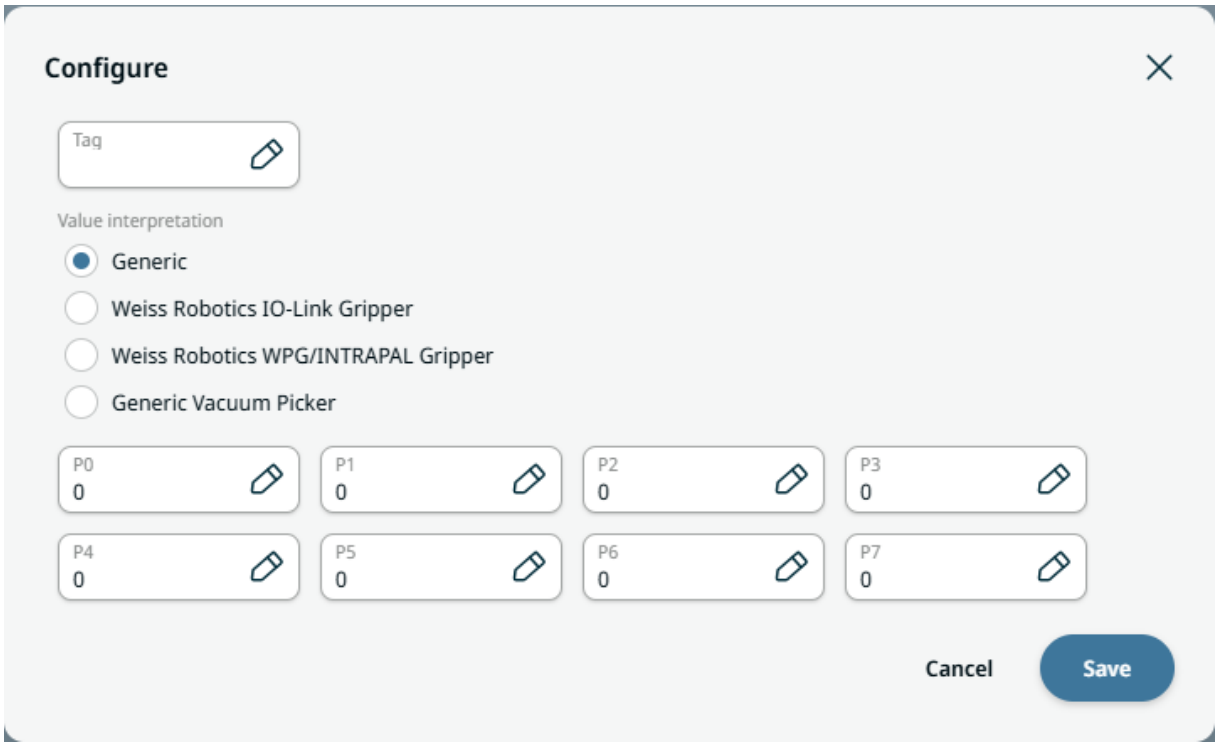

Figure 44: The SGRIPCFG Configure dialog - Tag, Value interpretation template, and P0–P7.

Template	P0	P1	P2	P3	P4	P5	P6	P7
Generic	P0 (no unit)	P1	P2	P3	P4	P5	P6	P7
Weiss Robotics IO-Link Gripper	No Part Limit (mm)	Release Limit (mm)	Force Factor (%)	–	–	–	–	–
Weiss Robotics WPG/INTRAPAL Gripper	No Part Limit (mm)	Release Limit (mm)	Force Factor (%)	Grip Velocity (mm/s)	Grip Acceleration (mm/s²)	Release Velocity (mm/s)	Release Acceleration (mm/s²)	–
Generic Vacuum Picker	Part Present Setpoint (kPa)	Energy Saving Setpoint (kPa)	Blow-off Duration (ms)	–	–	–	–	–

⚠ Switching the Value interpretation template and saving resets every parameter the new

template does not use back to 0. Switching from WPG/INTRAPAL (7 parameters) to IO-Link (3 parameters), for example, clears P3–P6.

7 URScript reference

Every GRIPLINK command is also a plain URScript function (`GL_*`) callable directly from a script node, the same functions the graphical nodes call internally. This chapter is the reference for advanced programming.
The application node generates a `GL_*` helper preamble once per program; script nodes call these functions directly.

How errors are reported

If a call fails, the plugin shows a popup naming the failed command with a **Retry** option, and the program pauses until the operator retries or stops it; see Installation → Behavior in Case of an Error.

Example

> A program calls `GL_GRIP(0, 2, 1)`, but the GRIPLINK controller has just lost power. The program pauses at that line; a popup names `GL_GRIP` as the failed command and the status code.

7.1 Establish Connection – GL_CONNECT

This command establishes the connection between the GRIPLINK controller and the robot controller. It must be the first GRIPLINK call in a program; every other `GL_*` function assumes the connection is already established. The command also checks whether the connected GRIPLINK controller meets the minimum requirements of the GRIPLINK protocol (see System Requirements).
The connection always uses TCP port 10001 on the GRIPLINK controller. The command always closes any existing connection under `<SOCKET_NAME>` before opening a new one, so calling it again mid-program (e.g. to reconnect with a different IP) is safe without a prior `GL_DISCONNECT`.

The Tool I/O interface is addressed via the IP address `127.0.0.1`. When calling `GL_CONNECT` in Tool I/O mode, the IP address `127.0.0.1` must therefore be used. The GRIPLINK socket is identical for both interfaces and defaults to `"sock_griplink"`.

Syntax

```
GL_CONNECT(<IP>, <SOCKET_NAME>)
```

Parameters

`string <IP>`	IP address of the GRIPLINK controller (Ethernet) or `127.0.0.1` (Tool I/O).

`string <SOCKET_NAME>` Name of the socket, e.g. `"sock_griplink"`. Used by all subsequent calls.

Common causes of a failed connection

- A wrong IP address or mismatched subnet.
- An unpowered GRIPLINK controller or unplugged cable.

Example

Open the connection once at the start of the program, before any other GRIPLINK call:

```
GL_CONNECT("192.168.1.40", "sock_griplink")
```

Every other GL_ function accepts <SOCKET_NAME> as an optional last parameter with the default `"sock_griplink"`. Passing it is only required when GL_CONNECT was called with a different socket name.

7.2 Close Connection – GL_DISCONNECT

This command closes the connection between the robot controller and the GRIPLINK controller. This may be necessary if the network connection to the GRIPLINK controller is interrupted by a tool changer, or when a program needs to release the controller before ending or before reconnecting with different settings.

Syntax

```
GL_DISCONNECT(<SOCKET_NAME>)
```

Parameters

`string <SOCKET_NAME>` Name of the socket to close.

Example

```
GL_DISCONNECT("sock_griplink")
# ... tool change here ...
GL_CONNECT("192.168.1.40", "sock_griplink")
```

7.3 Check a Connected Device – GL_DEVASSERT

This command checks whether the intended device is connected to the selected port. If the connected device does not have the specified **Vendor ID** and **Product ID** (from the device's IODD file), the robot program is stopped. Use it at the start of a program to confirm the correct device before running an application built for it.

Syntax

```
GL_DEVASSERT(<PORT>, <VENDOR_ID>, <PRODUCT_ID>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index (0...3 Standalone, 0...31 Proxy).
`int <VENDOR_ID>`	Expected Vendor ID (device's IODD file).
`int <PRODUCT_ID>`	Expected Product ID (device's IODD file).

Find the Vendor and Product IDs in the device's own description or IODD file.

Example

Confirm a WEISS ROBOTICS device (Vendor ID 815, Product ID 52) is connected on port 0:

```
GL_DEVASSERT(0, 815, 52, "sock_griplink")
```

7.4 Reference the Device – GL_HOME

Performs a homing cycle on the selected device. The command executes the device's built-in home-referencing function and waits until it is complete.

Syntax

```
GL_HOME(<PORT>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.

Behavior varies by device - see the connected device's operating instructions.

Example

Home the device on port 0:

```
GL_HOME(0, "sock_griplink")
```

7.5 Enable the Device – GL_ENABLE

This command activates the device connected to the selected port. IO-Link gripper modules then switch to the RELEASED state and execute the release action of the currently selected grip preset.

The <WSTR> parameter can be used to enable waiting for a state transition (WSTR). If WSTR is enabled, command execution waits until the device state on the selected port changes to ENABLED or FAULT. If no state change occurs, command execution ends with a TIMEOUT.

Syntax

```
GL_ENABLE(<PORT>, <WSTR>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`bool <WSTR>`	True waits for ENABLED before returning.

Use <WSTR> = True only when the device is currently DISABLED - calling ENABLE on an already-ENABLED device with <WSTR> = True can time out, since no transition occurs.

Example

Enable the device on port 0 and wait for it to become ready:

```
GL_ENABLE(0, True, "sock_griplink")
```

7.6 Disable the Device – GL_DISABLE

This command disables the device connected to the selected port. The command can be used, for example, to release the fingers of a gripper module or to temporarily disable the laser beam of a distance sensor.
DISABLE can also be used to acknowledge device errors, for example, if the connected device is in the FAULT state. With <WSTR> = True, command execution waits until the device state changes to DISABLED or FAULT; if this does not happen, execution ends with a TIMEOUT error.

Syntax

```
GL_DISABLE(<PORT>, <WSTR>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`bool <WSTR>`	True waits for DISABLED before returning.

Use <WSTR> = True only when the device is currently ENABLED (or FAULT) - calling DISABLE on an already-DISABLED device with <WSTR> = True can time out.

Example

On startup, check whether the device on port 0 is in FAULT and clear it before homing:

```
state = GL_DEVSTATE(0, "sock_griplink")
if state == 7:
GL_DISABLE(0, True, "sock_griplink")
GL_ENABLE(0, True, "sock_griplink")
end
GL_HOME(0, "sock_griplink")
```

7.7 Grip a Workpiece – GL_GRIP

The GRIP command performs a grip using the specified grip preset (0–7). The <BLOCKING> parameter specifies whether the command returns immediately after the movement starts or only after it is complete: with <BLOCKING> = 1, the call waits until the gripping process is complete; with 0, it returns immediately after the movement starts, while the motion continues in the background. In non-blocking mode, GL_WSTR can also be used to wait for a state change, and the result can then be checked with GL_DEVSTATE.

If a device is already in the HOLDING or NO PART state, the GRIPLINK protocol rejects a further GRIP command with the message "Service unavailable within current state."

Syntax

```
GL_GRIP(<PORT>, <PRESET>, <BLOCKING>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`int <PRESET>`	Preset index (0…7).
`int <BLOCKING>`	1 waits for the grip to complete before returning; 0 returns immediately.

The grip parameters can be configured via the GRIPLINK controller's web interface or via the GL_SETGRIPCFG command.

Example

Grip with preset 2 on port 0; retry once with a release if no part was detected:

```
GL_GRIP(0, 2, 1, "sock_griplink")
state = GL_DEVSTATE(0, "sock_griplink")
if state == 4:
# NO_PART - release and try again
```

```
GL_RELEASE(0, 2, 1, "sock_griplink")
GL_GRIP(0, 2, 1, "sock_griplink")
end
```

7.8 Release a Workpiece – GL_RELEASE

The RELEASE command releases a previously gripped workpiece using the parameters of the specified grip preset (0–7). With <BLOCKING> = 1, the call waits until the release is complete; with 0, it returns immediately after the movement starts.

Syntax

```
GL_RELEASE(<PORT>, <PRESET>, <BLOCKING>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`int <PRESET>`	Preset index (0…7).
`int <BLOCKING>`	1 waits for the release to complete before returning; 0 returns immediately.

Example

Release the part held on port 0 using preset 2:

```
GL_RELEASE(0, 2, 1, "sock_griplink")
```

7.9 Grip Workpiece via Parameters – GL_FLEXGRIP

The FLEXGRIP command performs a gripping operation with freely selectable motion parameters. Unlike the GRIP command, no stored grip preset is used here; instead, the motion sequence is determined entirely by the command parameters. This allows individual target positions, forces, speeds, and accelerations to be specified directly in the command, useful when the target changes at runtime.
With <BLOCKING> = 1, the call waits until the process is complete (workpiece gripped or no workpiece detected); with 0, it returns immediately while the motion is executed in the background.

Syntax

```
GL_FLEXGRIP(<PORT>, <POSITION>, <FORCE>, <SPEED>, <ACCELERATION>, <BLOCKING>,
                <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`float <POSITION>`	Target position (mm).
`float <FORCE>`	Target gripping force (N).

`float <SPEED>` Target speed – physical mm/s in Ethernet mode; a 0–200 % override in Tool I/O mode.

`float <ACCELERATION>` Target acceleration – physical mm/s² in Ethernet mode; a 0–100 % override in Tool I/O mode. 0 uses the device's default.

`int <BLOCKING>` 1 waits for the motion to complete before returning; 0 returns immediately.

Example

Grip on port 0 with 60 N force, a no-part limit of 15 mm, and 45.5 mm/s, blocking:

```
GL_FLEXGRIP(0, 15.0, 60, 45.5, 0, 1, "sock_griplink")
```

7.10 Release Workpiece via Parameters – GL_FLEXRELEASE

The FLEXRELEASE command releases a previously gripped workpiece using freely selectable motion parameters. It can be used both for a simple release and for targeted pre-positioning of a servo gripper. After the command is called, the gripper module opens the fingers to the specified target position; the state typically changes to RELEASED upon completion.
In non-blocking execution (`<BLOCKING>` = 0), the command returns immediately while the movement runs in the background; with `<BLOCKING>` = 1, it waits until the movement is complete.

Syntax

```
GL_FLEXRELEASE(<PORT>, <POSITION>, <SPEED>, <ACCELERATION>, <BLOCKING>,
                <SOCKET_NAME>)
```

Parameters

`int <PORT>` Port index.

`float <POSITION>` Target position (mm).

`float <SPEED>` Target speed – physical mm/s in Ethernet mode; a 10–100 % override in Tool I/O mode.

`float <ACCELERATION>` Target acceleration – physical mm/s² in Ethernet mode; a 0–100 % override in Tool I/O mode. 0 uses the device's default.

`int <BLOCKING>` 1 waits for the motion to complete before returning; 0 returns immediately.

Example

Pre-position the fingers on port 0 to 19.5 mm while the robot is still moving to the pick point (non-blocking):

```
GL_FLEXRELEASE(0, 19.5, 0, 0, 0, "sock_griplink")
movej(pick_approach)
GL_WSTR(0, "sock_griplink")
```

In Tool I/O mode the speed override range is 10–100 % and the acceleration override

range 0–100 % (100 % = nominal).

7.11 Control Force Retention – GL_CLAMP

With the CLAMP command, gripper modules with integrated grip-force retention (PERMAGRIP) can de-energize the drive in the HOLDING state to reduce the gripper module's power consumption. The workpiece remains held by the PERMAGRIP grip-force retention.
When CLAMP is activated (<STATE> = True), the drive switches off in the HOLDING state. When CLAMP is deactivated (<STATE> = False), grip-force control remains active and the gripper module's controller continuously adjusts the gripping force. Available on compatible devices (e.g. CRG series).

Syntax

```
GL_CLAMP(<PORT>, <STATE>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`bool <STATE>`	True engages the clamp; False releases it (a boolean state, not a force value).

Requires a device with integrated grip-force retention (e.g. CRG series).

Example

Engage the clamp on port 1 after gripping:

```
GL_GRIP(1, 0, 1, "sock_griplink")
GL_CLAMP(1, True, "sock_griplink")
```

7.12 Control the Status LED – GL_LED

This command activates a predefined LED preset for the status LED of a connected device. This allows the operating status or progress of a process to be visually indicated directly on the end-effector. Available on compatible devices (e.g. CRG series).

LED presets can be configured via the GRIPLINK controller's web interface.

Syntax

```
GL_LED(<PORT>, <PRESET_INDEX>, <SOCKET_NAME>)
```

Parameters

`int <PORT>` Port index.

`int <PRESET_INDEX>` LED preset index (0…7).

Available only on devices with a status LED.

Example

Grip on port 2, then set the LED to preset 1 if the part is wide enough, otherwise preset 2:

```
GL_GRIP(2, 0, 1, "sock_griplink")
width = GL_VALUE(2, 0, "sock_griplink")
if width >= 8.1:
GL_LED(2, 1, "sock_griplink")
else:
GL_LED(2, 2, "sock_griplink")
end
```

7.13 Query the Current Device State – GL_DEVSTATE

This command reads the current device state on the specified port and returns it directly; assign the call to a variable. See the Device State appendix for possible values and the corresponding URScript constants (e.g. `S_HOLDING` = 5, `S_NO_PART` = 4).

Syntax

```
<VAR> = GL_DEVSTATE(<PORT>, <SOCKET_NAME>)
```

Parameters

`int <PORT>` Port index.

Return value

`int` The current device state; see the Device State appendix (e.g. 5 = HOLDING, 4 = NO_PART).

Example

Wait until the device on port 2 reports HOLDING (5) by polling:

```
GL_GRIP(2, 0, 0, "sock_griplink")
state = 0
while state != 5:
state = GL_DEVSTATE(2, "sock_griplink")
```

```
sleep(0.1)
end
```

7.14 Wait for a Device State Change – GL_WSTR

The WSTR command waits for a change in the device state at the specified port. It can be used, for example, to check whether an asynchronously executed motion (GL_GRIP/GL_RELEASE/GL_FLEXGRIP/GL_FLEXRELEASE with <BLOCKING> = 0) has been completed. The command blocks until a state change occurs on the specified device. Read the result afterwards with GL_DEVSTATE.

Syntax

```
GL_WSTR(<PORT>, <SOCKET_NAME>)
```

Parameters

`int <PORT>` Port index.

If no state change occurs, status code 5 (TIMEOUT) is raised.

Example

Pre-position the gripper while the robot moves, then wait for the transition before gripping:

```
GL_RELEASE(0, 2, 0, "sock_griplink")
movej(approach)
GL_WSTR(0, "sock_griplink")
state = GL_DEVSTATE(0, "sock_griplink")
if state != 3:
popup("Gripper not RELEASED")
halt
end
movej(pick_point)
GL_GRIP(0, 2, 1, "sock_griplink")
```

7.15 Query a Device Value – GL_VALUE

This command reads an indexed device value (value channel 0–7) and returns it directly. The meaning of each index is device-specific; the valid range for the value index can be found in the manual of the respective device.

Syntax

```
<VAR> = GL_VALUE(<PORT>, <VALUE_INDEX>, <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`int <VALUE_INDEX>`	Index of the device value (0…7); which indices are available depends on the device type.

Return value

`float`	The value read. A sentinel value if the reading is unusable; see the note below.

`GL_VALUE` returns a sentinel number if the reading is unusable: `2147483647.0` for positive infinity, `-2147483648.0` for negative infinity, `-1.0` for NaN. Treat these as invalid data.

Example

After gripping on port 3, read the part width (index 0) and branch on an OK/NOK check:

```
GL_GRIP(3, 0, 1, "sock_griplink")
part_width = GL_VALUE(3, 0, "sock_griplink")
if part_width < 29.9 or part_width > 30.1:
# NOK - move to reject position
movej(reject_pose)
GL_RELEASE(3, 0, 1, "sock_griplink")
else:
# OK - move to use position
movej(use_pose)
GL_RELEASE(3, 0, 1, "sock_griplink")
end
```

7.16 Wait for a Device Value – GL_WAITVAL

The WAITVAL command waits until a specific device value reaches the previously set target value or falls within a defined tolerance range; this avoids polling `GL_VALUE` in a loop. The monitored value corresponds to the index value readable via `GL_VALUE`. The command blocks until one of the following conditions is met:

- The monitored value is within the specified tolerance window, or
- the timeout has been exceeded.

Syntax

```
GL_WAITVAL(<PORT>, <VALUE_INDEX>, <THRESHOLD>, <WINDOW_SIZE>, <TIMEOUT>,
           <SOCKET_NAME>)
```

Parameters

`int <PORT>` Port index.
`int <VALUE_INDEX>` Index of the device value (0…7).
`float <THRESHOLD>` Target value to wait for.
`float <WINDOW_SIZE>` Tolerance window around the threshold.
`int <TIMEOUT>` Maximum time to wait, in milliseconds (ms).

If the value does not reach the window within `<TIMEOUT>`, a timeout error (status code 5) is raised and the program is interrupted.

Example

Wait up to 5 seconds (5000 ms) for a distance sensor on port 2 to reach 100.0 ± 0.5:

```
GL_WAITVAL(2, 0, 100.0, 0.5, 5000, "sock_griplink")
```

7.17 Set a Device Value – GL_SETVAL

This command can be used to set a device-specific value. The value is addressed via an index whose valid range depends on the connected device; the meaning of the written value is also device-specific. Support for this command depends on the device type.

For the valid ranges of the value index and the value to be written, refer to the manual of the respective device.

Syntax

```
GL_SETVAL(<PORT>, <VALUE_INDEX>, <VALUE>, <SOCKET_NAME>)
```

Parameters

`int <PORT>` Port index.
`int <VALUE_INDEX>` Index of the device value (0…7).
`float <VALUE>` Value to write.

Setting device values can cause actuators to move.

Example

Set value channel 2 on port 3 to 10.5:

```
GL_SETVAL(3, 2, 10.5, "sock_griplink")
```

7.18 Configure a Grip Preset – GL_SETGRIPCFG

The SETGRIPCFG command is used to adjust the parameters of a grip preset. A grip preset defines the motion and force parameters that a gripper module uses when executing a GRIP or RELEASE command. Up to eight parameters (<P0>…<P7>) can be configured for each grip preset; their meaning and valid ranges are device-specific.

Syntax

```
GL_SETGRIPCFG(<PORT>, <PRESET>, <TAG>, [<P0>, <P1>, <P2>, <P3>, <P4>, <P5>,
                <P6>, <P7>], <SOCKET_NAME>)
```

Parameters

`int <PORT>`	Port index.
`int <PRESET>`	Preset index (0…7) to configure.
`string <TAG>`	Free-text tag identifying the configuration (letters, digits, and `- _ * + # % / < > :`, max 32 characters).
`float <P0> … <P7>`	Up to eight device-specific configuration parameters, as a URScript list.

Example

For a WEISS ROBOTICS IO-Link gripper on port 3, set preset 1 with a 31 mm no-part limit, 41 mm release limit, and 59 % force factor:

```
GL_SETGRIPCFG(3, 1, "bearing", [31, 41, 59, 0, 0, 0, 0, 0],
     "sock_griplink")
```

8 Troubleshooting and diagnostics

The GRIPLINK Plugin outputs error messages during operation. Typical messages, possible causes, and recommended actions are described below. The information provided here supports commissioning, diagnostics, and service.

Error-handling model

Identify the layer first: a communication problem, a device-state problem, and a Tool I/O access problem require different checks.

Layer	Typical symptom	First check	Recommended next action
Ethernet communication	Check GRIPLINK Controller fails, or commands cannot connect.	IP address, subnet, controller power, cable, web interface.	Correct network settings and run the check again.
Tool I/O access	Waiting for Tool I/O access, or status code 100/101.	Tool I/O access state, competing URCapXs/functions.	Let GRIPLINK control Tool I/O; press Rescan once the device is powered.
Device state	Command stops: DISABLED, FAULT, or not initialized.	Device State appendix and Sidebar status bar.	Enable, home, clear the fault, or adapt the program sequence.
Process result	NO PART or timeout although communication works.	Grip parameters, preset, workpiece, stroke, force, target state.	Use DEVICESTATE, WSTR, VALUE, or WAITVAL to branch safely.

8.1 Troubleshooting matrix

Check this matrix first; the individual topics below add detail for the most common commissioning/operation situations.

Symptom / message	Likely cause	Check	Action	Related section
Tool I/O Interface shows Waiting for Tool I/O access	PolyScope X has not granted Tool I/O control, or another function controls it.	Only one plugin may control Tool I/O.	Ensure only GRIPLINK controls Tool I/O; restart the program or	Tool I/O-specific Status Codes

			interface if access was revoked.	
Tool I/O Interface shows Waiting for device	No compatible device detected on the Tool I/O interface.	Cable, power, supported device type.	Press **Rescan** after restoring power/wiring.	Connecting and Disconnecting a Device
Scanning devices... stays visible too long	The plugin is scanning for devices, or the interface is stuck.	Wait 2–5 seconds; check if the device table updates.	Use **Restart Interface**; if that fails, power-cycle the robot.	Rescan and Restart
Rescan is unavailable	A robot program holds the interface.	Running/paused program.	Stop the program and close active GRIPLINK connections before rescanning.	Notes on Using the Sidebar
Ethernet **Check GRIPLINK Controller** fails	Wrong IP, different subnet, unpowered controller, or cable problem.	IP address, subnet, cable, power, default `192.168.1.40`.	Correct network settings and press **Check GRIPLINK Controller** again.	Ethernet mode
Command popup reports `NO_COMM` / status code 18	Communication to the controller or Tool I/O device was lost.	Ethernet link, power, Tool I/O interface, controller state.	Restore power/communication and retry; in Tool I/O mode press **Rescan** if the device was powered off.	Status Codes
Command reports `TIMEOUT` / status code 5	Device did not reach the expected state/value in time.	Target state, grip parameters, blocking option.	Start from a valid state, check grip parameters, or disable blocking.	Timeout when "Wait for state transition" is enabled
Non-blocking command does not behave asynchronously	Device firmware does not support the required async behavior.	Device type and firmware version.	For GRIPKIT EASY firmware below 3.0.0, update firmware or use blocking execution.	Non-blocking execution on GRIPKIT EASY with firmware versions below 3.0.0
Sidebar or Smart Skill commands are blocked	The robot program is active and the connection is open.	Robot program.	Close the running script connection, or end the program, before using manual controls.	Notes on Using the Sidebar

Device enters **FAULT**	Device reported a fault, or a command was invalid for its state.	Device state, device manual, error popup.	Use **DISABLE**; check the device manual and status codes.	Device State / Status Codes
Pick/Place branch returns **NO PART**	The gripper detected no part after the motion.	Preset, position, force, part geometry, sensor state.	Adjust the preset, or use FLEXGRIP/FLEXRELEASE with correct parameters.	Pick / Place
`INVALID_PARAMETER` / status code 10	At least one parameter is outside its accepted range.	Port, preset, position, force, speed, index.	Compare parameters against the command reference and device capabilities.	Program-node command reference / URScript reference

8.2 No access to the Tool I/O interface

Possible cause

Tool I/O access has not been granted to GRIPLINK by PolyScope X; the **Tool I/O Interface** field shows Waiting for Tool I/O access (status code 100/101). Another function or URCapX is controlling the Tool I/O interface.

Recommended action

In the robot settings, check under **Communication → Robot → Tool I/O** that **Controlled By** is set to **GRIPLINK**, and ensure that GRIPLINK is the only URCapX or function controlling the Tool I/O interface. Restart the program or the Tool I/O interface if previously granted access was revoked. Then check in the GRIPLINK Plugin's application node whether Tool I/O access has been granted again.

8.3 Device is not recognized in Tool I/O mode

Possible cause

The connected device is not compatible, not wired correctly, not connected with the original Tool I/O cable, or the Tool I/O interface has not yet been rescanned. The Tool I/O interface has no hot-plug detection; the **Tool I/O Interface** field shows Waiting for device.

Recommended action

Check the wiring, power supply, and device compatibility. Ensure that the original Tool I/O cable is being used. Then, in the application node of the GRIPLINK Plugin, perform the **Rescan** action and check whether the device appears in the device table.

8.4 Timeout when "Wait for state transition" is enabled

Possible cause

The expected state transition did not occur because the device was already in the target state, the preceding command did not trigger a state change, or the device was not in a valid initial state. After the internal timeout, a timeout error (status code 5) is raised.

Recommended action

Check the program flow and the device state before the command. Enable the wait function only if a state transition is actually expected. If necessary, first check the current device state using **GRIPLINK DEVICESTATE** / GL_DEVSTATE.

8.5 Non-blocking execution on GRIPKIT EASY with firmware versions below 3.0.0

Possible cause

GRIPKIT EASY devices with a firmware version below **3.0.0** support only blocking execution of gripping and release commands.

Recommended action

Use only blocking execution for these devices, or update the device firmware. Check the firmware version in use and adapt existing programs accordingly.

9 Appendices

9.1 Device State

The following table lists the possible state values for connected devices. The device state is returned by GRIPLINK DEVICESTATE / GL_DEVSTATE. The URScript constants listed in the right-hand column are defined by the plugin and can be used in the robot program.

Device state	Value	Meaning	Name of the URScript constant
NOT CONNECTED	0	Gripper module not connected	S_NOT_CONNECTED
NOT INITIALIZED	1	Gripper module not initialized	S_NOT_INITIALIZED
DISABLED	2	Drive inactive. Fingers can be moved manually; device is disabled.	S_DISABLED
RELEASED	3	Workpiece released	S_RELEASED
NO PART	4	No workpiece found	S_NO_PART
HOLDING	5	Workpiece is held	S_HOLDING
ENABLED	6	Drive active. Finger position is held; device is enabled.	S_ENABLED
FAULT	7	Fault condition	S_FAULT

⚠ Do not assign values other than those listed in the table to the URScript constants, as this will impair the function of the URCapX and may result in malfunction!

9.2 Status Codes

If an error occurs during the execution of a GRIPLINK command, a corresponding status code is output. This describes the cause of the error and can be used for diagnosis in the robot program or during commissioning. The following table provides an overview of the possible status codes and their meanings. Tool I/O-specific codes (100+) are in the Tool I/O integration chapter.

Status code	Identifier	Description

0	E_SUCCESS	No error. Command successfully executed.
1	E_OVERRUN	Data overrun
2	E_RANGE_ERROR	Value out of range
3	E_NOT_AVAILABLE	Function or data not available
4	E_NOT_INITIALIZED	Device not initialized
5	E_TIMEOUT	Timeout
6	E_INSUFFICIENT_RESOURCES	Not enough memory available
7	E_CHECKSUM_ERROR	Checksum error
8	E_ACCESS_DENIED	Access denied
9	E_INVALID_HANDLE	Invalid handle
10	E_INVALID_PARAMETER	Invalid parameter
11	E_INDEX_OUT_OF_BOUNDS	Index out of bounds
12	E_IO_ERROR	Generic I/O error
13	E_READ_ERROR	Read error
14	E_WRITE_ERROR	Write error
15	E_NOT_FOUND	Resource not found
16	E_NOT_OPEN	File or device not open
17	E_EXISTS	Resource already exists
18	E_NO_COMM	Connection error
19	E_STATE_CONFLICT	Invalid state
20	E_NOT_SUPPORTED	Command or function not supported
21	E_INCONSISTENT_DATA	Data inconsistent
22	E_CMD_SYNTAX	Syntax error
23	E_CMD_UNKNOWN	Unknown command
24	E_CMD_ABORTED	Command aborted
25	E_CMD_FAILED	Command failed
26	E_AXIS_BLOCKED	Axis is blocked
27	E_PENDING	Pending action / Not yet finished

9.3 Tool I/O-specific status code actions

These codes describe the state of the Tool I/O interface; they are separate from the GRIPLINK protocol status codes above.

Code	Name	Meaning	First check / recommended action
100	TOOL_IO_NOT_GRANTED	PolyScope X has not granted Tool I/O access.	Ensure the GRIPLINK Plugin is allowed to control Tool I/O and no other function is using it.
101	TOOL_IO_REVOKED	Tool I/O access was revoked by PolyScope X.	Stop the program, restore access, and restart the interface if required.
102	TOOL_IO_CONNECTING	A device scan on the Tool I/O interface is in progress.	Wait until scanning finishes; do not issue commands during scanning.
103	DAEMON_BOOTING	Tool I/O backend is starting.	Wait a few seconds after installation/restart.
104	DAEMON_SHUTDOWN	Tool I/O backend is shutting down.	Wait for shutdown/restart to complete before issuing commands.

9.4 Version History

Date	Version	Type	Description	Author
07/2026	1.0.0	Initial release	GRIPLINK Plugin for Universal Robots PolyScope X. Supports the GRIPLINK protocol over TCP/IP (Ethernet) and the FLEXGRIP protocol over Modbus RTU (Tool I/O).	H. Efkere

GRIP SMARTER.

WEISS ROBOTICS

.