WEISS ROBOTICS

GRIPLINK PLUG-IN FOR EPSON

Version ~~2~~4.0.0

April 2023

Table of contents

1 Introduction

With the GRIPLINK technology, servo-electric and smart pneumatic gripping modules from WEISS ROBOTICS as well as sensors/actors from selected third-party vendors can be controlled by robot controllers of all leading robot brands via simple TCP/IP network connections.
The GRIPLINK plug-in for EPSON is the software link between the GRIPLINK-ET4 interface converter and the robot controller and enables the easy integration of WEISS ROBOTICS' GRIPLINK technology into EPSON robot systems.

This manual describes the functions of the GRIPLINK plug-in. For information about installation and operation of the GRIPLINK-ET4 interface converter please refer to the GRIPLINK-ET4 user's manual. The manual can be found online on **www.griplink.de/manuals**

Feldfunktion geändert

1.1 Notation and symbols

For a better understanding, the following symbols are used in this manual:

Functional or safety relevant information. Non-compliance may endanger the safety of personnel and the system, damage the device or impair its function.

Additional information for a better understanding of the described facts.

Reference to further information.

1.2 Intended use

The software "GRIPLINK plug-in" is intended for communication between the GRIPLINK-ET4 interface converter from WEISS ROBOTICS and a robot controller. The requirements of the applicable guidelines as well as the installation and operation instructions in this manual must be noted and adhered to. Any other use or use beyond that is considered improper use. The manufacturer is not liable for any damage resulting from this.

1.3 System requirements

In order to run the GRIPLINK plug-in, the following EPSON products are required:

- EPSON Robot Controller RC700-A or RC90-B
- EPSON RC+ 7.0 programming environment, version 7.4.5 or higher

Please contact EPSON or your EPSON distributor in order to purchase these products.

1.4 License terms

The GRIPLINK plug-in is protected by copyright. The software package includes the applicable license terms. By installing and using the GRIPLINK plug-in, the user accepts these license terms.

2 Installation

2.1 Software installation

To operate the GRIPLINK-ET4, the GRIPLINK plug-in provided by WEISS ROBOTICS is required on the robot controller. To install the GRIPLINK plug-in, please execute the following steps:

Please make sure the most recent version of the GRIPLINK plug-in is used. It can be downloaded from **www.griplink.de/software**.

1. Unzip the previously downloaded ZIP archive with the GRIPLINK plug-in into a directory of your choice
2. Create a new project in the EPSON RC+ programming environment or open an existing project to add the GRIPLINK plug-in.
3. Import the extracted files into your EPSON RC+ project using the menu item "File" -> "Import". Please note that the package contains several files with the extensions *.prg and *.inc. Both file types must be imported separately.
4. The necessary functions for using the GRIPLINK-ET4 in your project are now available.

2.2 Network configuration

The connection between the GRIPLINK-ET4 and the robot controller is established via TCP/IP networking. To use TCP/IP networking, the robot controller must be assigned an IP address in the system settings. Please note that the IP address of the robot controller must be in the same subnet as the IP address of the GRIPLINK-ET4.
The IP address of the GRIPLINK is set to 192.168.1.40 by default and can be adjusted via its web interface. To do this, connect the GRIPLINK-ET4 to a PC or laptop and open the web interface in your favorite browser by entering ***http://192.168.1.40*** into the address bar. You can access the IP settings by pressing the "Config" button.
The GRIPLINK command interface accepts incoming connections on port 10001 (TCP).

Feldfunktion geändert

For more information about the configuration of the GRIPLINK-ET4, please refer to the associated user's manual.

The IP address of the robot controller can be set in the EPSON RC+ programming environment via the menu item "Settings" → "System Settings" in "Controller" → "Configuration" (cf. Figure 1).

The IP addresses of both the robot controller and the GRIPLINK-ET4 interface converter must be located within the same subnet range. Please contact your network administrator if you experience any problems assigning appropriate IP addresses to the devices.

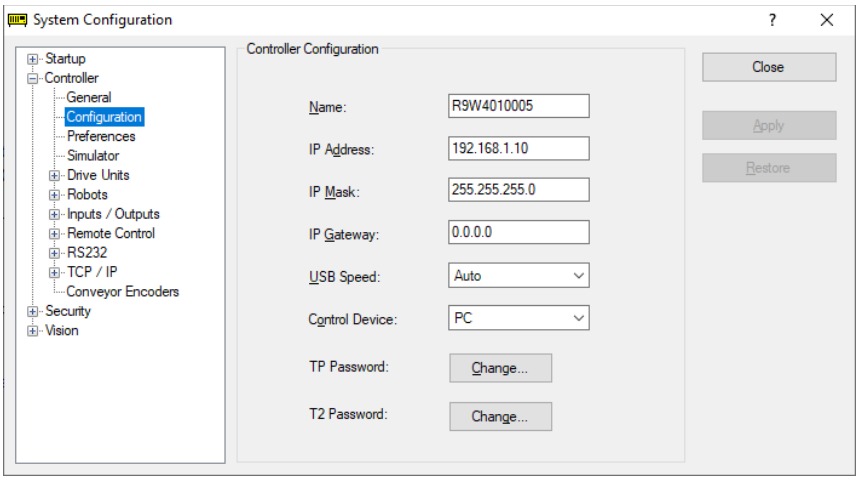

Figure 1: Setting the IP address for the robot controller

3 Command set reference

The GRIPLINK plug-in provides a number of functions to control gripping modules from WEISS ROBOTICS as well as sensors and actors from several third-party vendors. Both single and multi-gripper commands are available.

Multi-gripper commands

With the multi-gripper commands, several actors can be addressed at once. These commands are particularly suitable for handling large or flexible workpieces with multiple gripping modules simultaneously.

The basic program flow with the GRIPLINK plug-in is always as follows:

1. Establish connection with CONNECT
2. Activate gripping module and connection monitoring with ENABLE
3. For servo gripping modules without absolute encoder: Reference gripping module with HOME / MHOME
4. Gripping / releasing with GRIP / MGRIP or RELEASE / MRELEASE

The following chapters describe the available commands provided by the plug-in in detail.

Function	Description
Griplink_Connect()	Open connection to GRIPLINK
Griplink_Disconnect()	Close connection GRIPLINK
Griplink_Enable()	Enable gripping module
Griplink_Disable()	Disable gripping module
Griplink_DevState()	Query device state
Griplink_MultiDevState()	Query device state of all devices
Griplink_Value()	Query sensor value(s)
Griplink_Home()	Reference gripping module
Griplink_Grip()	Grip workpiece
Griplink_MultiGrip()	Grip workpiece with multiple grippers
Griplink_Release()	Release workpiece
Griplink_MultiRelease()	Release workpiece with multiple grippers
Griplink_Hold()	Control gripping force retention PERMAGRIP®
Griplink_LED()	Control LED display

3.1 Open connection - CONNECT

This command establishes the connection between the GRIPLINK-ET4 interface converter and the robot controller. To open a network connection, one of 16 available so-called Ports (range 201 to 216) on the robot controller must be selected.

The IP address of the GRIPLINK-ET4 can be changed via the web interface.

Signature

```
Function Griplink_Connect(strIPAddr$ As String, intControllerPortNo As
Integer)
```

Parameters

strIPAddr$	IP adress of the GRIPLINK interface converter as string
intControllerPortNumber	Port number to be used for the TCP/IP connection. Range between 201 und 216.

Return value

-

Example

Open a connection to the GRIPLINK with IP address 192.168.1.40 by using Port 201 and Timer 0:

```
Griplink_Connect("192.168.1.40", 201, 0)
```

3.2 Close Connection – DISCONNECT

This command closes a connection that has been established previously by the *Griplink_Connect()* function.

Signature

```
Function Griplink_Disconnect()
```

Parameters

-

Return value

-

Example

Open a connection to the GRIPLINK with IP address 192.168.1.40 by using Port 201 and Timer 0:

```
Griplink_Disconnect()
```

3.3 Enable device - ENABLE

This command activates the device connected to the given port of the GRIPLINK. It is intended to enable the device when starting the application or to re-enable it after it has been disabled with *Griplink_Disable()*, e. g. to acknowledge a fault condition.

Please note that for gripping modules by WEISS ROBOTICS, the *Griplink_Enable()* command must only be called after executing *Griplink_Home()*. It is not necessary to enable the grippers before use any more.

Signature

```
Function Griplink_Enable(ubGripperID As UByte)
```

Parameters

ubDevicePort	Port index (0 to 31)

Return value

-

Example

Enable device connected at GRIPLINK port 0:

```
Griplink_Enable(0)
```

3.4 Disable device - DISABLE

This command deactivates the device connected to the given port of the GRIPLINK. This function can be used, for example, when using an automatic tool changer. If a gripping module from WEISS ROBOTICS is connected, it deactivates the drive of the selected gripping module, e.g. when using a tool changer.

Signature

```
Function Griplink_Disable(ubGripperID As UByte)
```

Parameters

ubDevicePort	Port index (0 to 31)

Return value

-

Example

Change gripping module at port 1:

```
'Open connection between robot controller and GRIPLINK
Griplink_Connect("192.168.1.40", 201)
'Enable drive and workpiece monitoring for gripping module at port 1
Griplink_Enable(1, True)
'Execute program (e. g. pick & place)
'…
'Disable gripping module at port 1
Griplink_Disable(1)
'Now the gripping module can be removed.
'…
'Enable new gripping module
Griplink_Enable(1, True)
'Continue with program
'…
```

3.5 Get Device State – DEVSTATE

This function returns the device state of the selected device. The device state is provided as an integer value. To simplify and improve readability, the file *Griplink.inc* defines constants that can be used for processing gripping states.

Signature

```
Function Griplink_DevState(ubDevicePort As UByte) As Integer
```

Parameters

ubDevicePort Port index (0 to 31)

Return value

Current device state (cf. chapter 5)

Example

Wait until the device state of the gripping module at port 2 changes to HOLDING (4):

```
#include "Griplink.inc"
'Query gripping state of gripping module at port 2
Integer intDevState
'The constant DS_HOLDING is defined in Griplink.inc
Do While intDevState <> DS_HOLDING
   intDevState = Griplink_DevState(2)
Loop
```

3.6 Reference gripper - HOME

References the selected servo gripper. The command executes a reference run of the gripping module and waits until it's completed. After the command has been executed, the fingers of the gripping module are without force and must be moved to a defined position by using *Griplink_Grip()* / *Griplink_MultiGrip()* or *Griplink_Release()* / *Griplink_MultiRelease()*.

The direction of the reference run can be configured via the web interface of the GRIPLINK interface converter.

Signature

```
Function Griplink_Home(ubDevicePort As UByte)
```

Parameters

ubDevicePort Port index (0 to 31)

Return value

-

Example

Reference gripping module at port 3:

```
#include "Griplink.inc"
Integer intDevState
'Reference gripping module at port 3
intDevState = Griplink_Home(3)
If intDevState = DS_IDLE Then
    'Gripping module is referenced
Else
    'Gripping module is not referenced
EndIf
```

3.7 Grip part - GRIP

Grips a workpiece with the selected gripping module and grip preset. The command waits until the gripping state changes to either "HOLDING" or "NO PART".

The gripping parameters can be configured via the web interface of the GRIPLINK interface converter.

Signature

```
Function Griplink_Grip(ubDevicePort As UByte, ubGripIndex As UByte) As
Integer
```

Parameters

ubDevicePort	Port index (0 to 31)
ubGripIndex	Grip to be executed (range depends on gripper type)

Return value

Current device state (cf. chapter 5)

Example

Gripping module at port 0 shall execute grip preset 2. If no part is found, display a message:

```
#include Griplink.inc
Integer intDevState
intDevState = Griplink_Grip(0, 2)
'The constants DS_NO_PART and DS_HOLDING are defined in Griplink.inc
If intDevState = DS_NO_PART Then
    Print "No part found"
ElseIf intDevState <> DS_HOLDING Then
    Print "Unexpected grip state"
EndIf
```

3.8 Grip part with multiple grippers – MGRIP

This function executes a grip command with the selected gripping modules. The function waits until all gripping modules have reached either "HOLDING" or "NO PART" state.

The gripping parameters can be configured via the web interface of the GRIPLINK interface converter.

Signature

```
Function Griplink_MultiGrip(boolGripper0 As Boolean, boolGripper1 As
Boolean, boolGripper2 As Boolean, boolGripper3 As Boolean, ubGripIndex
As UByte)
```

Parameters

boolGripper0	*True*	Gripping module at port 0 is selected, workpiece will be gripped
	False	Gripping module at port 0 is not selected
boolGripper1	*True*	Gripping module at port 1 is selected, workpiece will be gripped
	False	Gripping module at port 1 is not selected
boolGripper2	*True*	Gripping module at port 2 is selected, workpiece will be gripped
	False	Gripping module at port 2 is not selected
boolGripper3	*True*	Gripping module at port 3 is selected, workpiece will be gripped
	False	Gripping module at port 3 is not selected
ubGripIndex	Selected grip preset (range depends on gripper type)	

Return value

-

Example

Gripping modules at port 1 and 2 shall grip a workpiece. Display a message if both workpieces were gripped correctly:

```
#include Griplink.inc
Griplink_MultiGrip(False, True, True, False, 2)
'The constants DS_NO_PART and DS_HOLDING are defined in Griplink.inc
If Griplink_GetState(1) = DS_HOLDING And Griplink_GetState(2) =
DS_HOLDING Then
    Print "Holding parts"
EndIf
```

3.9 Release part - RELEASE

Releases the workpiece previously gripped with the selected gripping module. The command waits until the workpiece has been released.

The gripping parameters can be configured via the web interface of the GRIPLINK interface converter.

Signature

```
Function Griplink_Release(ubDevicePort As UByte, ubGripIndex As UByte)
```

Parameters

ubDevicePort	Port index (0 to 31)
ubGripIndex	Selected grip preset (range depends on gripper type)

Return value

-

Example

Gripping module at port 0 shall release a workpiece previously gripped with grip preset 2:

```
#include Griplink.inc
Integer intDevState
intDevState = Griplink_Release(0, 2)
'The constant DS_RELEASED is defined in Griplink.inc
If intDevState = DS_RELEASED Then
    Print "Part released"
EndIf
```

3.10 Release part with multiple grippers – MRELEASE

Releases the workpiece(s) gripped with the selected gripping modules. The function waits until all gripping modules have reached the RELEASED state.

The gripping parameters can be configured via the web interface of the GRIPLINK interface converter.

Signature

```
Function Griplink_MultiRelease(boolGripper0 As Boolean, boolGripper1 As
Boolean, boolGripper2 As Boolean, boolGripper3 As Boolean, ubGripIndex
As UByte)
```

Parameters

boolGripper0	*True*	Gripping module at port 0 is selected, workpiece will be released
	False	Gripping module at port 0 is not selected
boolGripper1	*True*	Gripping module at port 1 is selected, workpiece will be released
	False	Gripping module at port 1 is not selected
boolGripper2	*True*	Gripping module at port 2 is selected, workpiece will be released
	False	Gripping module at port 2 is not selected
boolGripper3	*True*	Gripping module at port 3 is selected, workpiece will be released
	False	Gripping module at port 3 is not selected
ubGripIndex	Selected grip preset (range depends on gripper type)	

Return value

-

Example

Gripping module at ports 1, 2 and 3 release a workpiece previously gripped with grip preset 3:

```
#include Griplink.inc
Griplink_MultiRelease(True, True, True, False, 3)
```

3.11 Get sensor value – VALUE

This command returns current measured value of a connected sensor. If a WEISS ROBOTICS gripping module is connected, the command can be used to read the current finger position in micrometers. If more than one sensor value per device is available, the desired value can be selected by using the parameter *ubIndex*.
For Weiss Robotics grippers, value index 0 returns the finger position in micrometers (µm).

Signature

```
Function Griplink_Value(ubDevicePort As UByte, ubIndex As UByte) As Long
```

Parameters

ubDevicePort	Port index (0 to 31)
ubIndex	Value index (range depends on device)

Return value

Sensor value (e. g. Finger position in µm)

Example

Read the current finger position of the gripper at port 0:

```
Long lngPos
lngPos = Griplink_Value(0, 0)
If lngPos > 80000 Then
    Print "Finger position is 80 mm"
EndIf
```

3.12 Set control value – SETVAL

This command sets a control value in the selected device. It is non-blocking, i. e. the command returns immediately without waiting for the control value to become active. To make sure the control value is active before proceeding with the robot program, please use the WAITVAL command.

Signature

```
Function Griplink_SetVal(ubDevicePort As UByte, ubIndex As UByte,
lngValue As Long)
```

Parameters

ubDevicePort	Port index (0 to 31)
ubIndex	Value index (range depends on device)
lngValue	Value to be set

Return value

-

Example

On a WPG gripping module, set the finger position to pre-position the fingers to a position of 100 mm (ie. 100.000 µm). Then use the WAITVAL command to wait until this position has been reached:

```
Griplink_SetVal(0, 0, 100000)
Griplink_WaitVal(0, 0)
```

3.13 Wait for control value to become active – WAITVAL

This command is intended to wait for a control value to become active that has been previously set with the SETVAL command.

Signature

```
Function Griplink_WaitVal(ubDevicePort As UByte, ubIndex As UByte) As
Long
```

Parameters

ubDevicePort	Port index (0 to 31)
ubIndex	Value index (range depends on device)

Return value

Returns the control value set or its current value

Example

See SETVAL command.

3.14 Control gripping force retention - CLAMP

The innovative gripping force retention PERMAGRIP® developed by WEISS ROBOTICS keeps up the gripping force on the workpiece, even if the power supply to the gripping module is interrupted unexpectedly. PERMAGRIP® also enables permanent gripping without the gripping module overheating.
This command activates or deactivates the PERMAGRIP® gripping force retention for the selected gripping module.

This command is not necessary to keep up the gripping force in case the power supply gets interrupted. The gripping force retention in case of an interrupted power supply is always active.

PERMAGRIP® is not available on all gripping modules.

Signature

```
Function Griplink_HoldClamp(ubDevicePort As UByte, boolEnable As
Boolean)
```

Parameters

ubDevicePort	Port index (0 to 31)
boolEnable	Enable (*True*) oder disable (*False*) PERMAGRIP

Return value

-

Example

Grip workpiece with grip preset 0 on the gripper connected at port 2. Activate PERMAGRIP if a workpiece has been detected:

```
#include Griplink.inc
Integer intDevState
'Grip workpiece
intDevState = Griplink_Grip(2, 0)
'The constant DS_HOLDING is defined in Griplink.inc
If intDevState = DS_HOLDING Then
    'Activate PERMAGRIP
    Griplink_HoldClamp(2, True)
EndIf
```

3.15 Control of the LED display –LED

This command changes the color and the pattern of the illuminated ring of a selected WEISS ROBOTICS CRG gripping module.

Light patterns and colors can be configured via the web interface of the GRIPLINK interface converter.

This function is only available for CRG gripping modules by WEISS ROBOTICS.

Signature

```
Function Griplink_LED(ubDevicePort As UByte, ubLEDIndex As UByte)
```

Parameters

ubDevicePort	Port index (0 to 31)
ubLEDIndex	Index of the pre-configured light pattern (range 0 to 7)

Return value

-

Example

Use the gripping module at port 3 and change the color of the light ring according to the finger position. Display light pattern 0 if the finger position is greater than or equal to 8.1 mm, otherwise display light pattern 1:

```
#include "Griplink.inc"
Integer intDevState
Long lngPos
intDevState = Griplink_Grip(3, 0)
If intDevState = DS_HOLDING Then
    lngPos = Griplink_Value(3, 0)
    If lngPos > 8100 Then
        'Set light pattern 0
        Griplink_LED(0)
    Else
        'Set light pattern 1
        Griplink_LED(1)
    EndIf
Else
    Print "Failed to grip part"
EndIf
```

3.16 Configure grip settings - GRIPCFG

Configure grip settings for the given device. Each preset consists of a tag string and 3 parameters that configure the grip. The tag string can be used to give the preset a meaningful name for later identification.

The parameters may have specific limits depending on the connected device. See the device manual for details

Signature

```
Function Griplink_GripCfg(ubDevicePort As UByte, ubGripIndex As UByte,
label$ As String, ByRef lngParams() As Long)
```

Parameters

ubDevicePort	Port index (0 to 31)
ubGripIndex	Grip to be executed (range depends on gripper type)
label$	Label string
intParams	Array of 8 configuration parameters

For servo-electric gripping modules from Weiss Robotics, parameter 0 represents the no part limit in micrometers (µm), parameter 1 represents the release limit in micrometers (µm) and parameter 2 represents the gripping force in percent, multiplied by a factor of 1000.

Return value

-

Example

Set the configuration for grip preset 2 of the gripper connected to port 1:

```
#include "Griplink.inc"
Long lngParams(8)
lngParams(0) = 2000    // No Part Limit 2.0 mm
lngParams(1) = 10000   // Release Limit 10.0 mm
lngParams(2) = 80000   // Gripping force 80%
lngParams(3) = 0
…
lngParams(7) = 0
Griplink_GripCfg(0, 1, "Workpiece 1", ByRef lngParams())
```

hat formatiert: Englisch (Vereinigte Staaten)

hat formatiert: Englisch (Vereinigte Staaten)

3.17 Assert device type - DEVASSERT

Assert that a device with the specified Vendor and Device ID is connected to the selected port.

The necessary parameters IO-Link Vendor ID and Device ID can be found in the device's manual.

Signature

```
Function Griplink_DevAssert(ubDevicePort As UByte, intVendorID As
Integer, intDeviceID As Integer)
```

Parameters

ubDevicePort Port index (0 to 3)
intVendorID Device's IO-Link Vendor ID
intDeviceID Device's IO-Link Device ID

Return value

-

Example

Check, if a gripper of type IEG55-020 (Device ID 20) by Weiss Robotics (Vendor ID 815) is connected to port 2:

```
#include "Griplink.inc"
Griplink_DevAssert(2, 815, 20)
```

4 Error Handling

If a problem occurs within the GRIPLINK plug-in, an error will be raised on the robot controller. If this error won't be caught by the user, the robot program will stop. Any running robot motion will be completed before the robot stops.

It is the responsibility of the programmer to handle these errors and to bring the system into a safe state after an error occured. The EPSON programming environment provides appropriate commands to catch errors, for example *OnErr* (see documentation of the EPSON robot controller).

In order to differentiate between different types of errors, the GRIPLINK plug-in uses several different error numbers from the robot controller's range of user-defined error numbers. Table 1 shows the predefined error numbers and their meaning. In order to avoid conflicts with existing robot programs, these error numbers can be adjusted in the *GriplinkConfig.inc* file.

Error	Error Number	Description
GRIPLINK_CONNECTION_ERROR	8100	Connection error between GRIPLINK interface converter and gripping module
GRIPLINK_COMMAND_ERROR	8101	Failed to execute command
GRIPLINK_DEVICE_ERROR	8102	Device error on connected gripping module (e.g. overtemperature)
GRIPLINK_PARAM_ERROR	8103	Parameter error when calling a Griplink function (e. g. value out of range)
GRIPLINK_TIMEOUT_ERROR	8104	Timeout error when executing a Griplink command.

Table 1: Error numbers defined by the GRIPLINK plug-in

Errors that might be raised while executing the GRIPLINK functions described in chapter 3 can be caught using the usual error handling of SPEL+. The following code snippet gives an example:

```
#include Griplink.inc
Integer intDevState
OnErr GoTo errorHandler
intDevState = Griplink_Grip(0, 2)
'The constants DS_NO_PART and DS_HOLDING are defined in Griplink.inc
If intDevState = DS_NO_PART Then
    Print "No part found"
ElseIf intDevState <> DS_HOLDING Then
    Print "Unexpected grip state"
EndIf
errorHandler:
    MsgBox "Failed to execute grip command"
```

For more information on error handling in SPEL +, please refer to the documentation of your EPSON robot controller.

5 Device States

Device State	Constant	Code	Description
NOT CONNECTED	DS_NOT_CONNECTED	0	Port not connected
NOT INITIALIZED	DS_NOT_INITIALIZED	1	Device not initialized
DISABLED	DS_DISABLED	2	Ready for operation, not active
RELEASED	DS_RELEASED	3	Workpiece released
NO PART	DS_NO_PART	4	Workpiece not found
HOLDING	DS_HOLDING	5	Holding workpiece
OPERATING	DS_OPERATING	6	Ready for operation, active
FAULT	DS_FAULT	7	Error