

igus®



**LOW  
COST  
AUTOMATION**  
by igus®



User Guide  
**roboLink® Robot Arm  
with iRC Control**



Version V1.2-EN, February 2021

© Commonplace Robotics GmbH, 2011 – 2021

igus® and robolink® are registered trademarks of igus GmbH.

igus® GmbH  
Spicher Str. 1a  
51147 Cologne  
Germany  
Phone Support: +49 (0) 2203-96498-255  
[ww-robot-control@igus.net](mailto:ww-robot-control@igus.net)  
[www.igus.de](http://www.igus.de), [www.igus.eu](http://www.igus.eu)

# Table of Contents

1.	Safety Instructions.....	5
2.	Quick Start Guide .....	6
2.1.	Set up and Connections.....	6
2.2.	Switching on .....	6
2.3.	Connect and Move the Robot .....	7
3.	Introduction .....	8
3.1.	System Overview.....	8
3.2.	Glossary and Abbreviations.....	9
3.3.	Specifications.....	11
3.4.	Mechanical Dimensions .....	12
4.	Electrical Connections.....	14
4.1.	Overview.....	14
4.2.	Pinout: Stepper Module .....	15
4.3.	Pinout: Support Module .....	16
4.4.	Pinout: Digital Input/Output Module .....	17
4.5.	Connect Sensors and Actors to the DIO Module .....	18
4.6.	Option: Control Cabinet .....	20
4.7.	Option: Embedded Computer .....	22
4.8.	Option: Operating Panel.....	25
5.	Safety.....	26
5.1.	Safety-Related Features of the Modular Robot Controller .....	26
5.2.	CE Certification .....	26
5.3.	Integration of SIL-Rated Safety Components.....	26
6.	Software Installation .....	28
6.1.	Installation of the iRC - igus® Robot Control.....	28
6.2.	Licensing.....	30
6.3.	Setting up the Ethernet Connection to the Embedded Computer.....	30
6.4.	Installing the CAN-to-USB Driver.....	31
7.	Moving the Robot with iRC .....	32
7.1.	The iRC Graphical User Interface .....	32
7.2.	Connecting the Robot .....	34
7.3.	Referencing the Robot .....	36
7.4.	Moving the Robot with Software Buttons or Gamepad .....	38
7.5.	Starting Robot Programs.....	39
7.6.	Digital Inputs and Outputs.....	40
7.7.	Software Interfaces.....	41

7.8.	Updating the Software.....	41
8.	Programming the Robot with iRC .....	42
8.1.	The Program Editor .....	42
8.2.	Comments and Information in Programs .....	45
8.3.	Variables and Variable Access.....	47
8.4.	Execution Flow.....	51
8.5.	Motion.....	58
8.6.	Gripper and Digital IO.....	64
8.7.	Camera .....	65
9.	Stand-alone Operation with Embedded Computer and the Operating Panel.....	66
9.1.	Reset Errors/Enable Robot .....	67
9.2.	Moving the Robot with the 3-Axis Joystick.....	67
9.3.	Referencing.....	68
9.4.	Starting and Stopping a Program.....	69
9.5.	Manually Setting the Digital Inputs/Outputs.....	69
9.6.	Display of Status Information .....	70
9.7.	Organize Programs on the Embedded Control.....	71
10.	Project Configuration.....	72
10.1.	Program .....	72
10.2.	Tool.....	72
10.3.	Inputs / Outputs.....	73
10.4.	Virtual Box.....	74
11.	Advanced Robot Configuration .....	74
12.	Interfaces Configuration .....	75
12.1.	Cameras .....	75
12.2.	PLC Interface.....	76
12.3.	CRI Ethernet Interface.....	76
13.	Troubleshooting.....	77
13.1.	Support Contacts .....	77
13.2.	Online Tool - Fault Identification and Recovery .....	77
13.3.	Configuration of the Stepper Modules .....	78
13.4.	Calibration of the Robot .....	79
13.5.	Error Codes.....	80

# 1. Safety Instructions



Operate the Robot safely!

**Always ensure personal safety of users and others while operating a Robot Arm or commissioning a robot cell!** In particular, no person or obstructions may be present in the working area of the Robot.

- In its basic version the Robot Controller package does not contain safety related functionality. Depending on the application, these may need to be added.  
See “CE marking” below and Section 5.
- CE marking: Robot Arm and the Robot Controller are a part of one system, which must be risk assessed in its entirety and comply with the current safety regulations to ensure personal safety. Depending on the result of the assessment further safety components must be integrated. These are usually safety relays and door switches. Responsible is the commissioning engineer of the system.
- The Robot Controller contains a 24 V power supply unit that itself requires mains voltage (120 / 240 V), depending on the configuration. Please check the label on the power supply. Only qualified personnel may connect the power supply to the mains and put it into operation.
- Work on the robot electronics should only be carried out by qualified personnel. Check current electrostatic discharge (ESD) guidelines.
- Always disconnect the Robot Controller from the mains (120 / 240 V) when working in the Control Cabinet or any electronics connected to the Robot Controller.
- Do NOT hot-plug! It may cause permanent damage to the motor modules. Do not install or remove any modules or plug/unplug connectors (e.g. Operating Panel, Emergency Stop Button, DIO Modules or external relays, motor connectors) while powered on.
- The Robot Arm must be set up on a sturdy surface and bolted down or otherwise secured.
- Only use and store the system in a dry and clean environment.
- Only use the system at room temperature (15° to 32°C).
- The ventilation of the system must be able to operate without hindrance, to ensure sufficient airflow to cool the Stepper Motor Driver Modules. There must be at least 10 cm of space next to the fan of the Robot Controller. The fan must, ideally, point upwards or to the side (reduced efficiency). The fan must not point down.
- Backup important data prior to installing the igus® Robot Control software.

## 2. Quick Start Guide

### 2.1. Set up and Connections

- Follow Safety Instructions, Section 1.
- Make sure that the on/off switch on the Control Cabinet is set to "off".
- Mount the Robot on a suitable base. Make sure that there is no tension on the cables and that the sheet-metal reference plate of Axis 1 is not bent.
- Feed the Robot cables through the large circular hole in the Control Cabinet and plug them into the Stepper Modules. Each motor is connected to its Stepper Module via 4 connectors.  
All connectors are labeled and coded to aid this process (see Section 4):
  - Motor cable (labelled *Motor*)
  - Encoder cable (2 connectors labelled *ENC-1* and *ENC-2*)
  - Reference sensor (labelled *End-Stop*)
- Secure the robot cables against tension, e.g. with a cable tie to one of the holes in the Control Cabinet. If available, plug in the display cable and secure it via the screw connection.
- After that the Robot can be connected to the mains.



### 2.2. Switching on

- Switch on the Robot using the on/off switch on the Control Cabinet.
- The green light-emitting diodes (LEDs) on the modules are now on, also most red LEDs and potentially some of the yellow LEDs.
- Optional: When shipped with an Embedded Computer, the green LEDs will start flashing after approx. 20 s. This shows communication, now the Robot Controller is up and running. If available, you can now move the Robot using the Operating Panel (optional). Details can be found in Section 8.



## 2.3. Connect and Move the Robot

Continue with point 2.3.1 or 2.3.2 depending on whether you have an Embedded Computer or USB-Can adapter supplied with the robot

### 2.3.1. Preparation using the Embedded Computer

- Connect your PC to the Robot Controller via an Ethernet cable. Use the Ethernet port located directly next to the USB socket on the embedded computer of the Robot Controller.
- Set the IP address of the PC to: static and 192.168.3.1 with subnet mask of 255.255.255.0

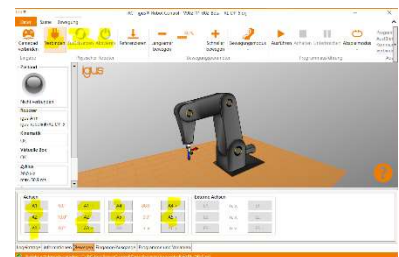
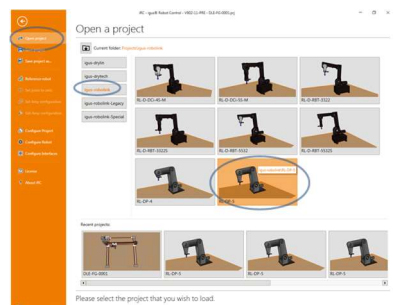


### 2.3.2. Preparation using the USB-CAN adapter (if supplied)

- Connect the USB-CAN-Adapter to a computer (USB port) and the Robot Controller (D-Sub-9 port with "CAN" marking). Install the necessary driver from the USB memory stick, see Section 6.2.

### 2.3.3. Connect and Move the Robot

- Install the igus® Robot Control (iRC) software on your PC, see Section 6.1.
  - Start the iRC software. On Start-up you can choose the project suitable for your Robot. Please refer to the igus product number, the project names are based on these.
  - You can now activate the Robot by pressing:
    - "Connect",
    - "Reset" and
    - "Enable" in sequence.
  - Now the "Status" indicator on the left should be green, status "No Error".
  - You can now move the joints of the robot using the buttons in the "Jog" tab.
  - Referencing is mandatory prior to moving in cartesian coordinates or executing a program.
- For details, please refer to Section 7.





# 3. Introduction

## 3.1. System Overview

The robot system consists of four basic components:

1. **Robot Arm**: the mechanical robolink DP;
2. **Robot Controller**: Support Module, Steppermotor Driver and DIO Modules;
3. **Robot Control Software**: control software to execute robot programs;
4. **Programming Environment**: graphical software to set up robot programs.

There are two basic system configurations available:

**Configuration A** uses an Embedded Computer with Robot Control Software to run the Modular Robot Controller. This set up can execute robot programs without an external Windows PC connected. For programming an external PC can be connected via Ethernet.

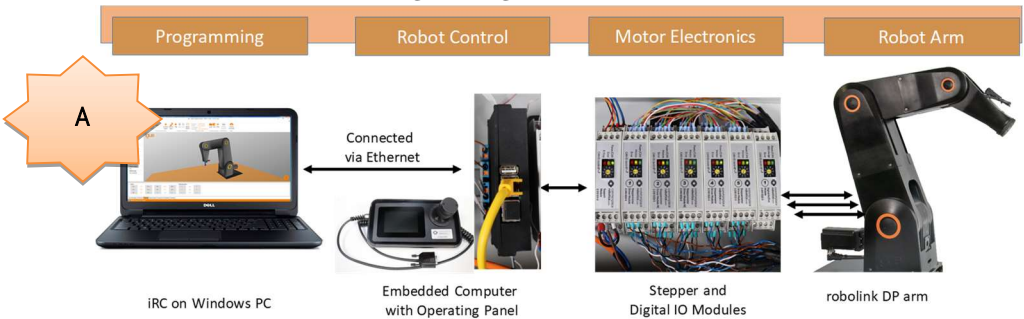


Figure 3.1: robolink DP with Embedded Computer running the Robot Control Software.

**In configuration B** a Windows PC serves both as programming environment and to command the Modular Robot Controller via a USB-CAN adapter. This setup is often used in educational settings.

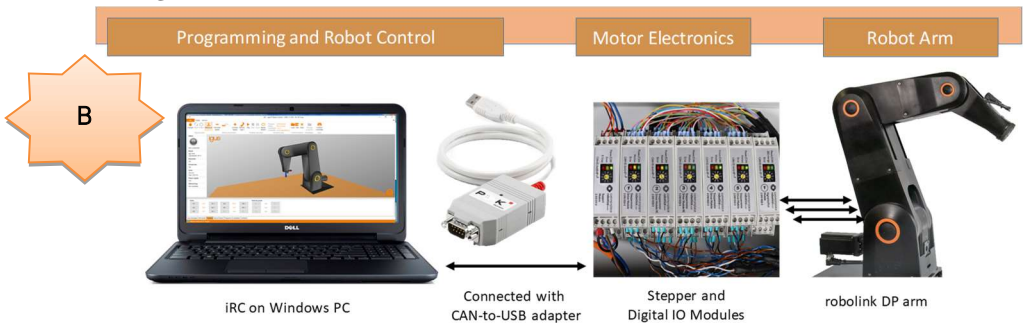






Figure 3.2: robolink DP with Robot Control Software and Programming Environment.

### 3.2. Glossary and Abbreviations

Image	Name	Abbrev.	Description
	igus® robotlink® Robot Arm	Robot	Mechanical robot arm including structure, motors and cables
	Modular Robot Controller	Robot Controller	Consists of: <ul style="list-style-type: none"> <li>• 1 Support Module</li> <li>• 3 or more Stepper Modules</li> <li>• 1-3 DIO Modules</li> </ul>
	Control Cabinet	-	Steel cabinet that the Robot Controller can be mounted in. Includes a 24 V / 10 A power supply.
	Embedded Computer	-	Optional Embedded Computer with Linux operating system and the TinyCtrl Robot Control Software
	Operating Panel	-	Optional operating hand-held device with 3.5" capacitive touchscreen display and 3-axis joystick

	Emergency Stop Button	-	Single-channel emergency stop
<b>Electronic Control Modules</b>			
	Digital Input/Output Module	DIO Module	Reads 7 digital inputs on 24 V level. Sets 7 digital output channels based on Solid State Relays.
	Stepper Driver Module	Stepper Module	<p>Versions:</p> <ul style="list-style-type: none"> <li>• Motor Encoder (ME): <ul style="list-style-type: none"> <li>- high current (HC)</li> <li>- low current (LC)</li> </ul> </li> <li>• Output Encoder (AE): <ul style="list-style-type: none"> <li>- high current (HC)</li> <li>- low current (LC)</li> </ul> </li> </ul>
	Support Module	-	Provides 5 V logic voltage, max. 2 A. Interface for single-channel Emergency Stop Button. Interface for additional e.g. dual channel safety relays.

### 3.3. Specifications

#### Robot Arm (if included)

Type	igus® robotlink®
Number of axes	Depending on version: 4 or 5
Payload	Depending on version: 2.5 to 3 kg

#### Modular Robot Controller

Power supply	24 V at 10 A
Type	DIN rail modules ME format with 5-pin bus connector
Communication	Controller Area Network (CAN) Fieldbus 500 kBit USB-to-CAN Adapter PCAN-USB from Peak Systems Ethernet
Support Modules	Provision of 5 V logic voltage: max. 2 A SoftStart to prevent overloading the power supply 1-channel emergency stop function without safety classification Provision to connect external safety relays
Stepper motor Driver Modules (Stepper Modules)	For operating a bipolar stepper motor with quadrature encoder RS422 24 V reference switch input
Digital In/Out Modules (DIO Module)	7 digital inputs, 12 – 24 V, optocoupled 7 digital outputs, solid state relays, max. 500 mA

#### Embedded Computer (if included)

Platform	Phytec Regor or comparable CPU e.g. Texas Instruments AM3352
Operating system	Linux
Software	TinyCtrl Robot Control Software
Interfaces	Control of the drives and DIO Modules via the CAN bus Connection to igus® Robot Control via Ethernet RS232 display connection

#### Programming and Robot Control Software

igus® Robot Control (iRC)	
Recommended System requirements	PC with e.g. Intel i5 CPU (minimum i3) and Windows 10, free USB 2.0 port, ethernet port, 500 MB disk space

### 3.4. Mechanical Dimensions

For more information see the igus® manual “Technical Documentation robolink® DP Version”.

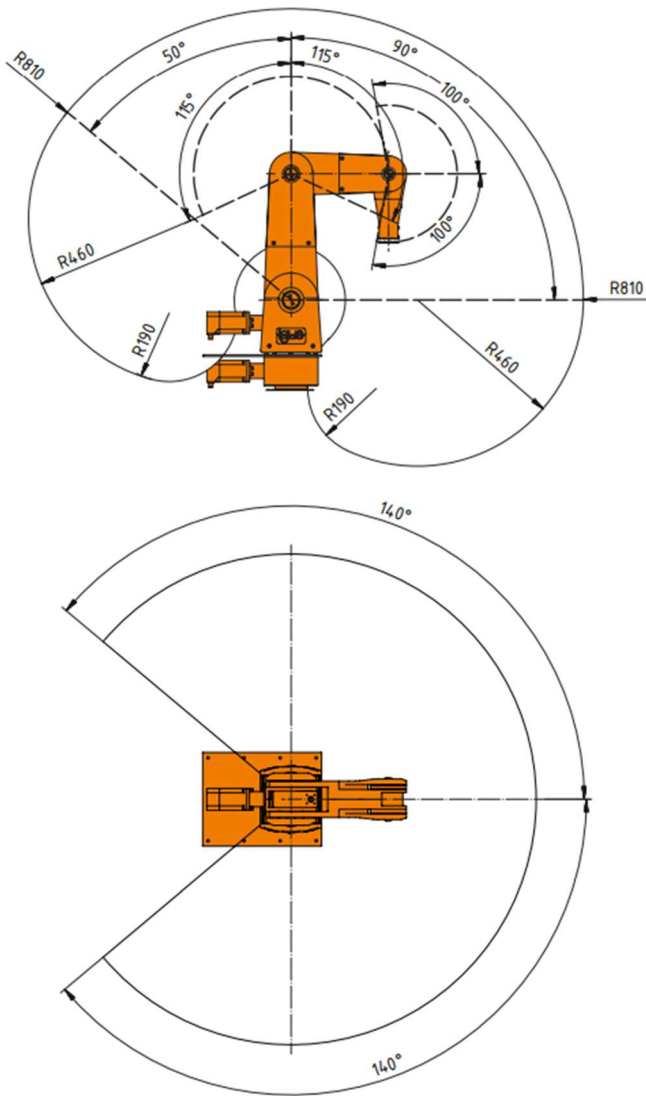


Figure 3.3: Schematic of operating range of igus® robolink® DP, 5-axis version.

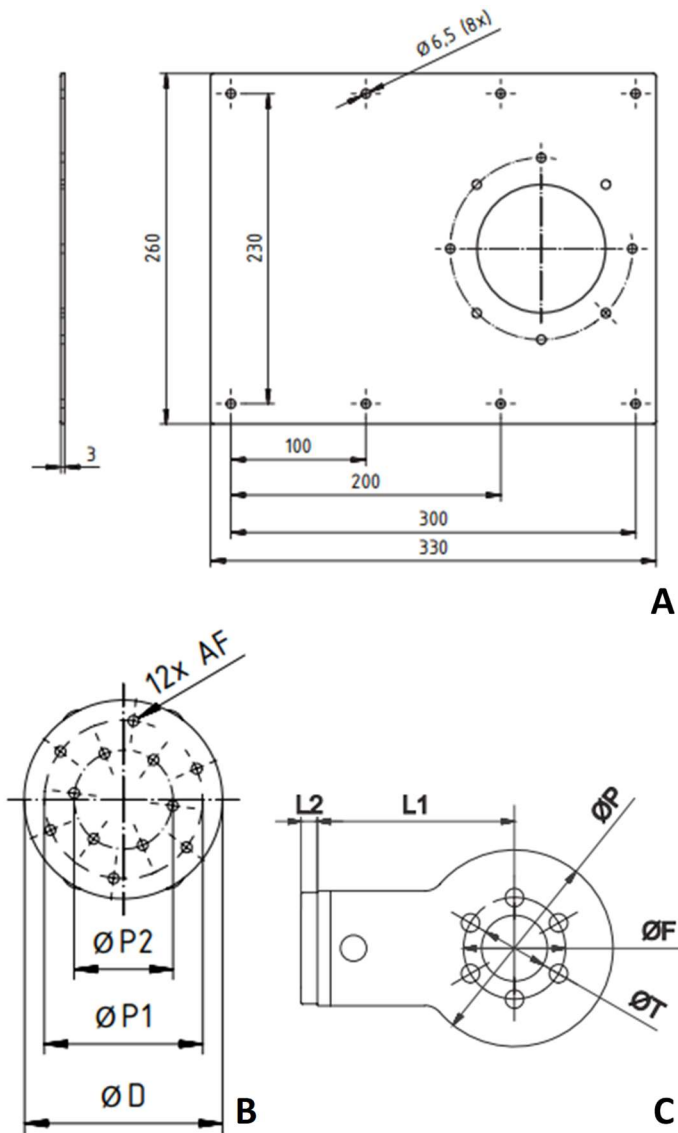


Figure 3.4: igus® robolink® DP of: A) Robot base; B) hole pattern flange of the 5<sup>th</sup> axis (five axes version); C) Gripper adapter (four axes version). Dimensions are depicted in mm.

# 4. Electrical Connections

## 4.1. Overview

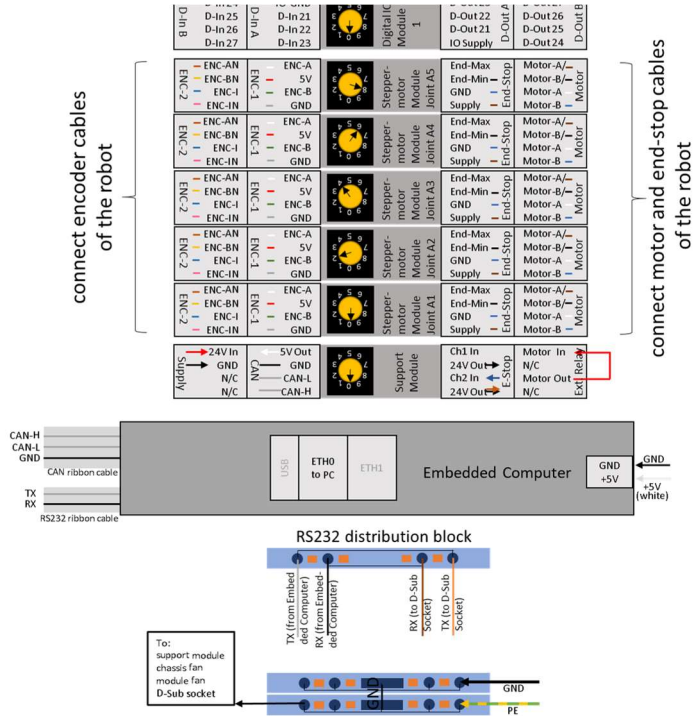


Figure 4.1: Overview of electrical connections.

## 4.2. Pinout: Stepper Module

Each Stepper Module drives a bipolar stepper motor with Motor Encoders. The encoder signals are evaluated by a line driver (RS422).

Signals to each axis travel via three cables: the motor cable, encoder cable and reference switch cable.

The motor cable is connected to a plug labelled "Motor", the encoder cable is connected to two plugs labelled "ENC-1" and "ENC-2" and the reference switch cable is connected to the plug labelled "End-Stop".



### Motor connector:

Connects a bipolar stepper motor.

Pin 1 (left):	blue	B
Pin 2:	white	A
Pin 3:	black	B/
Pin 4:	brown	A/



### Encoder connector1 (ENC-1):

Connects a quadrature encoder to a line driver.

Pin 1 (left):	white	A
Pin 2:	red	5 V DC
Pin 3:	green	B
Pin 4:	grey	0 V

### Encoder connector 2 (ENC-2):

Connects a quadrature encoder to a line driver.

Pin 1 (left):	brown	A-N
Pin 2:	yellow	B-N
Pin 3:	blue	index
Pin 4:	pink	index-N

All eight wires (encoder connectors 1 and 2) must be connected to read the encoder.



### End-Stop connector:

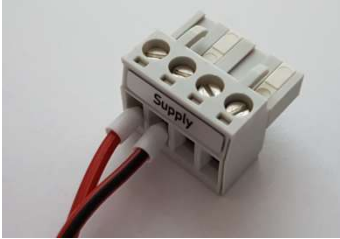
Connects to an end-stop or reference switch.

Pin 1 (left):	brown	24 V
Pin 2:	blue	ground (GND)
Pin 3:	black	signal
Pin 4:		do not connect



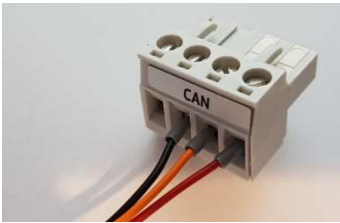
### 4.3. Pinout: Support Module

The Support Module provides 5 V logic voltage, a single-channel emergency stop relay, a SoftStart relay. It feeds the signals into the DIN rail bus system.



#### Supply voltage connector:

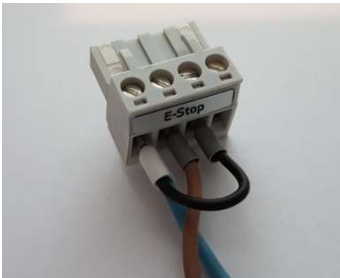
Pin 1 (left):	red	24 V
Pin 2:	black	GND
Pin 3:	do not connect	-
Pin 4:	do not connect	-



#### Controller Area Network (CAN) connector:

Connects controller and CAN-interface of Embedded Computer.

Pin 1 (left)		5 V to Embedded PC
Pin 2:	black	CAN-GND
Pin 3:	orange	CAN-L
Pin 4:	red	CAN-H



#### Emergency stop (E-Stop) connector:

Connects the Emergency Stop Button.

Pin 1 (left):	blue	E-Stop channel 1
Pin 2:	brown	24 V output signal
Pin 3:	black	E-Stop channel 2
Pin 4:	do not connect	-

This is a single channel setup, adapt this to your safety requirements! The options for connecting a safety relay are described below and in Section 5.



#### Motor power bridge:

Allows the motor current to be interrupted by external safety switches, see Section 5.

Pin 1 (left):	motor power-out
Pin 2:	do not connect
Pin 3:	motor-power in
Pin 4:	do not connect

## 4.4. Pinout: Digital Input/Output Module

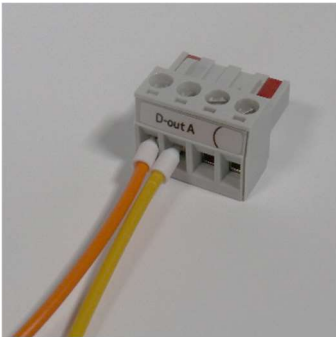
The DIO Module provides input and output channels, e.g. to operate a gripper valve. The outputs can switch up to 500 mA. The inputs use optocouplers and are compatible to input Voltages between 12 and 24 V.



A circuit switched by the output relays must not contain any larger capacitors. If the current exceeds 500 mA, the solid-state relays can suffer damage.

For safety reasons, the inputs and outputs of the DIO Module are electrically isolated. This means that the circuits of the inputs and outputs are not connected to the internal circuits of the controller.

It is, therefore, necessary to connect a supply voltage for the outputs and a ground line for the inputs. For this purpose, the 24 V available in the robot controller can be used, but also an external independent voltage source. In Software the inputs and outputs of the first DIO Module are numbered 21-27; the second DIO Module (if supplied) has numbers 31-37; and the third 41-47.



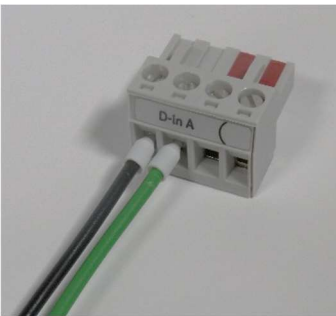
### Digital Out connector A (D-out A):

The output relays connect the pin of the power supply with the corresponding output pins.

- Pin 1 (left): Input voltage (for all seven channels)
- Pin 2: D-Out channel 1 (in software: Dout21)
- Pin 3: D-Out channel 2 (in software: Dout22)
- Pin 4: D-Out channel 3 (in software: Dout23)

### Digital Out connector B (D-out B):

The D-out B pins are (from left to right) the Digital Out channels 4-7 (picture not shown).



### Digital In connector A (D-in A):

Pin 1 of D-In A is the corresponding GND pin for all input pins.

- Pin 1 (left): Signal GND (for all seven channels)
- Pin 2: D-In channel 1 (in software: DIn21)
- Pin 3: D-In channel 2 (in software: DIn22)
- Pin 4: D-In channel 3 (in software: DIn23)

### Digital In connector B (D-in B):

The D-in B pins are (from left to right) the Digital In channels 4-7 (picture not shown).

## 4.5. Connect Sensors and Actuators to the DIO Module

The easiest way to connect a programmable logic controller (PLC) is via digital inputs and outputs. Each Robot Controller comes with one DIO Module. This provides 7 inputs and 7 outputs (see Section 4.4). If additional inputs and outputs are required, up to two additional DIO Modules can be integrated, see Section 10.1. A total of up to 3 DIO Modules can be controlled.

The outputs are controlled via solid state relays and can switch up to 500 mA. This value must not be exceeded during the switching process (e.g. by charging currents of capacitors) to prevent damage to the relays.

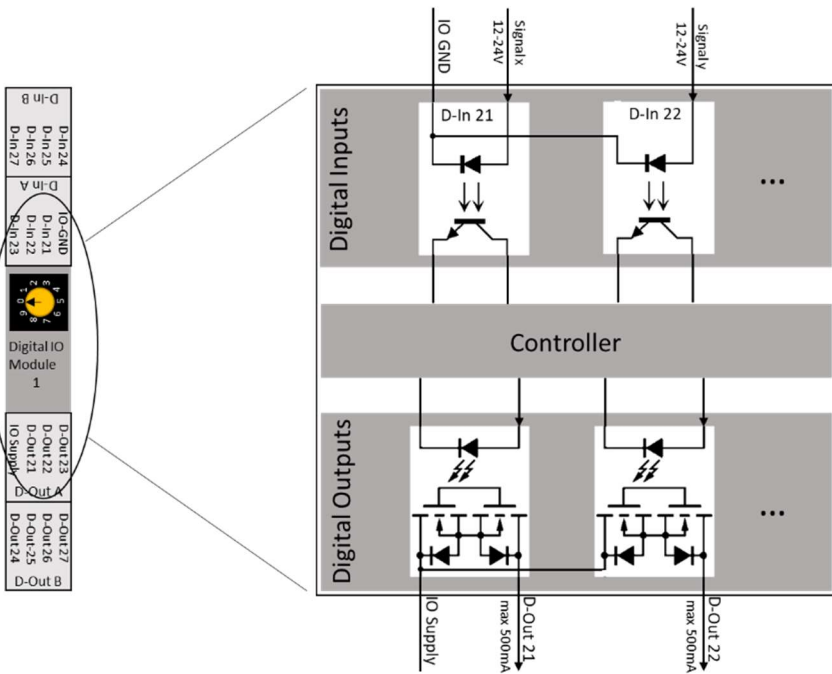


Figure 4.2: Internal set up of a DIO Module.

The inputs and outputs are galvanically separated from the robot control. A power supply (labelled “IO Supply” in the Figure above) must be connected. The integrated 24 V supply may also be used.

### 4.5.1.1. To Connect a Sensor

- Pin 1 (GND) of the D-In 1 connector must be connected to GND of the power supply of the sensor.
- The sensor signal (positive) must be connected to an input pin D-In 1 connector pins 2-4 or D-In 2 connector pins 1-4. the positive side of the sensor has to be connected to VDD of the power supply.
- The status of the inputs can be monitored in the "Input/Output" tab at the bottom of iRC, see Section 7.1.
- In a robot program inputs can be queried and reacted upon e.g. in an if-then-else command, see Section 0.

### 4.5.1.2. To Connect an Actor (LED, pneumatic valve, relay...)

- Pin 1 (Supply) of the D-Out 1 connector must be connected to a power supply (e.g. 24 V)
- The actor (relay, etc.) is then powered from a free pin of the D-Out connectors (D-Out 1 pin 2-4 and D-Out 2 pin 1-4).
- You can set the outputs manually in the "Input / Output" tab on the bottom of iRC, see Section 7.1.
- In a robot program you can set the state of the outputs using the digital-out command, see Section 8.6.

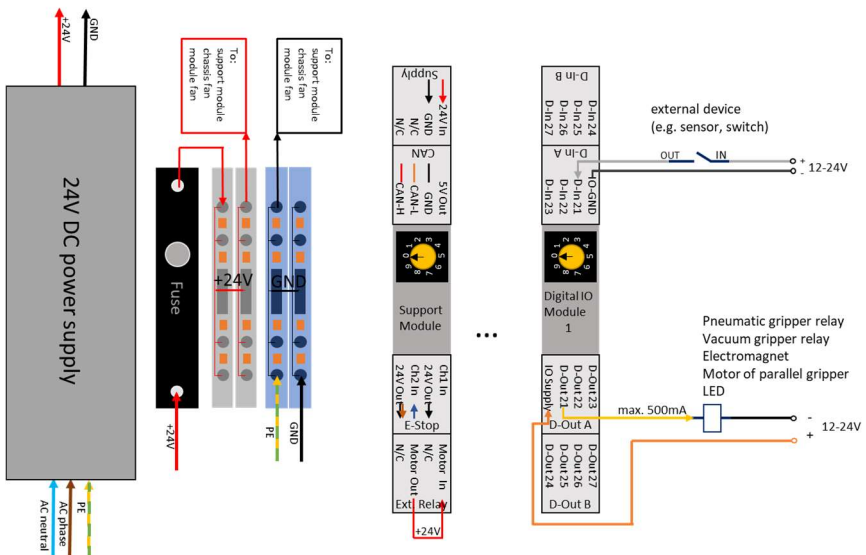


Figure 4.3: Connection of an input and output to the DIO Module.

## 4.6. Option: Control Cabinet

The Modular Robot Controller can be ordered in a steel cabinet. The cabinet shields the control from dust, humidity and accidental access.

The dimensions of the control cabinet are W x L x H: 600 x 200 x 125 mm



Figure 4.4: Control Cabinet, closed. On the lower left the mains IEC connector / switch / fuse and the cable fittings can be seen.

The cabinet includes a 24 V / 10 A power supply for 220 V / 110 V input. A fuse (8 A). Distribution blocks are included. A small fan is mounted on the left side of the cabinet.

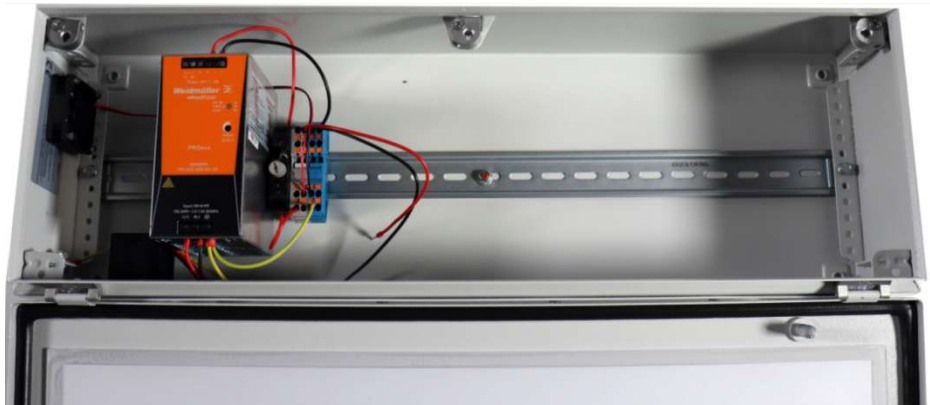


Figure 4.5: Control Cabinet with Power Supply, without Modular Control Electronics.



## 4.7. Option: Embedded Computer

An Embedded Computer can be combined with Modular Robot Controller, so that an external computer is just required for programming, but not for day to day operation. This resembles “Configuration A” in the introduction, Section 3.1.

The Embedded Computer is a single board computer with Linux operating system, running the TinyCtrl robot control software. It is used to:

- Operate the Robot Arm
- Replay programs
- Connect with iRC on a Windows computer via Ethernet for programming.

The Embedded Computer communicates with the Modular Control Electronics on the DIN rails.



Figure 4.6: Embedded Computer with Ethernet cable connected to the ETH0 socket.

### 4.7.1. Ethernet Connection

ETH0 Primary Ethernet Port (next to the USB port)	Connection to PC for programming via igus® Robot Control	Standard Ethernet IP 192.168.3.11
ETH1 Secondary Ethernet Port	Usually none. Can be used to connect a camera	Standard Ethernet IP 192.168.4.11

### 4.7.2. Operating Panel Connection

An Operating Panel can be connected, see Section 4.8

RS232 connection	RS232 connection to the Operating Panel	Lead 1: UART2-TX Lead 2: UART2-RX
------------------	---	--------------------------------------



The 2 lead ribbon cable of the Embedded Computer is the RS232 connection for the Operating Panel. One lead is marked black (Lead 1).

It is connected via a distribution block to a 9-pin D-Sub connector.

Lead 1 (black)	D-Sub conn. Orange
Lead 2	D-Sub conn. Brown

To provide power to the display, the black and red lead of the D-Sub connector are connected to GND and 24 V respectively.

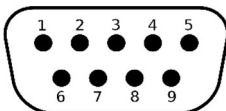


Figure 4.7: D-Sub connector to the Operating Panel.

The pinout of the connector is as follows:



Caution! The connection is proprietary.  
It is not suitable for a null modem cable or the like.



Pin 1:	24 V
Pin 3:	RS232-TX
Pin 6:	GND
Pin 7:	RS232-RX

All remaining pins are not connected



### 4.7.3. CAN Connection

CAN ribbon cable (three conductors)	Establishes the CAN connection to the Stepper Modules. To be connected to the Support Module	Lead 1: GND (black) Lead 2: CAN-L Lead 3: CAN-H
-------------------------------------	--	---



The 3 leaded ribbon cable of the Embedded Computer is the CAN connection. It has one lead marked in black (Lead 1, GND). Leads 2 and 3 are CAN-L and CAN-H respectively.

The leads are connected to the support board as follows:

- Lead 1 (black)      Support Board CAN Port Pin 2
- Lead 2              Support Board CAN Port Pin 3
- Lead 3              Support Board CAN Port Pin 4

### 4.7.4. Power Supply

Power supply 5 V	Supplies the Linux Board with 5 V DC	Pin 1: 5 V Pin 2: GND
------------------	--------------------------------------	--------------------------



The connector has a white (5 V) and a blue (GND) wire. When the module is mounted on the DIN rail, white (5 V) is towards the bottom of the cabinet. The 5 V (white) cable is connected to the Support Module 5 V port (Pin 1 of the CAN connector). The black cable is connected to the power common GND distribution blocks of the DIN rail.

Double check the polarity of the connection before switching on.

## 4.8. Option: Operating Panel

The Operating Panel is used to control the Robot via the Embedded Computer. For operating instructions see Section 9.

The connection of the D-Sub-9 connector is mating to the D-Sub-9 socket of the Control Cabinet Section 4.7.2.



Figure 4.8: Operating Panel with touch screen and 3-axis joystick.

## 5. Safety

### 5.1. Safety-Related Features of the Modular Robot Controller

The Modular Control Electronics have provision for the connection of external safety components. However, external safety relays/switches can be connected to interrupt motor power. The emergency stop is a 1 channel device and does not comply to a safety integrity level (SIL).

### 5.2. CE Certification

The components of this robot system are provided with CE marking as requested by the European machinery directive. Responsible for the components are:

- igus GmbH: robolink DP Robot Arm
- Commonplace Robotics GmbH: Modular Robot Controller.

These CE markings confirm the adherence of these parts of the final machine to the necessary standards. The final machine, i.e. the complete robot cell, is not covered by these CE markings.

### 5.3. Integration of SIL-Rated Safety Components

The Modular Robot Controller does not provide any safety-relevant functions. The integrated emergency stop function is single channel.



In order to operate the complete, customer-specific robot system, the commissioning engineer must carry out a risk assessment as part of the CE certification and, depending on the result, integrate additional safety components. These are usually safety relays, fence and door switches.

A safety relay can be integrated as shown below in the 24V motor supply. In case of an emergency stop this supply is suspended with the safety rating of the the safety relay. An active motion of the motors is not possible anymore, but the motor modules are still on logic supply. After a release of the emergency stop and pressing the reset and enable buttons the robot can move on in his operation.

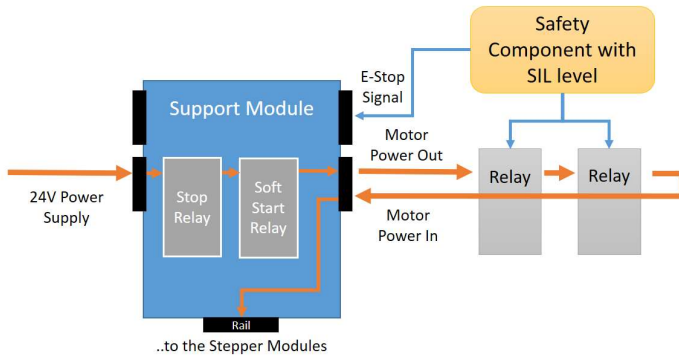


Figure 5.1: Safety relay integrated using the “Ext Rel” plug of the Support Module

Figure 5.1 shows the safety relay in series with the motor power bridge plug „Ext Rel” of the support module, see section 4.3. This plug routes the motor power supply out of the Support Module and in again. By breaking this connection an active motion of the motors is not possible anymore.

Figure 5.2 below shows an alternative integration of the safety relay, directly in the bus connections, breaking the 24V supply line. This way the power is not routed through the support module, avoiding further sources of failure. Connectors: Phoenix Contact 1803604, 1918751.



Figure 5.2: Safety relay included in between the Support Module and the Motor Modules

## 6. Software Installation

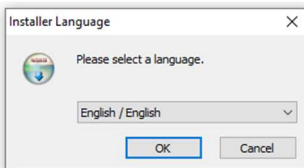


- Backup important data before installing the igus® Robot Control software.
- Before updating the igus® Robot Control software, create a backup of the current version, e.g. by renaming the folder C:\iRC-igusRobotControl\ to C:\iRC-igusRobotControl\_BAK\.

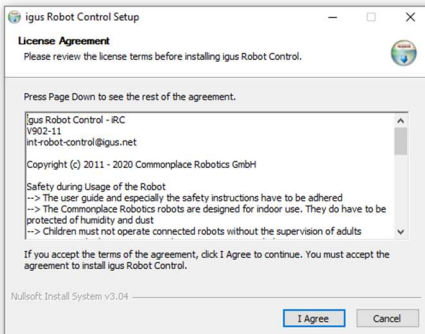
### 6.1. Installation of the iRC - igus ® Robot Control



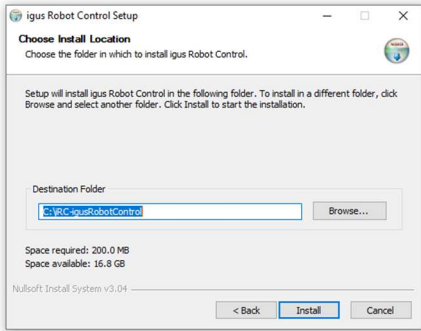
Insert the igus® Robot Control USB memory stick. Start the installer: D:\Installer\_IRC\_V11-nnn.exe  
You may have to allow changes on your system.



After starting the installation, you have to choose between German and English language.

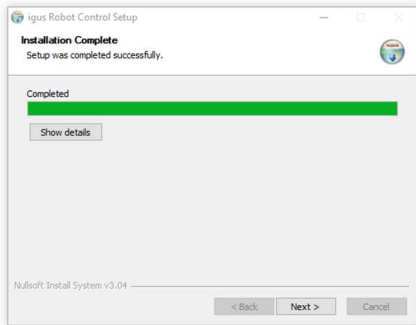


Then confirm the license agreement.



In the next step you can choose where igus® Robot Control should be installed. The recommended directory is  
C:\IRC-igusRobotControl

When installing igus® Robot Control in a Windows program directory like C:\Programs it is possible that igus® Robot Control can only be started as administrator.



The installation usually only takes a few seconds.



After finishing the installation, you can start igus® Robot Control.

Now you can start igus® Robot Control via the link on the desktop or via the start menu.

### Installation error:

The installation wizard checks whether all necessary extensions are available, especially the .NET framework. If this is not the case, an error message appears. The .NET framework must be installed manually:

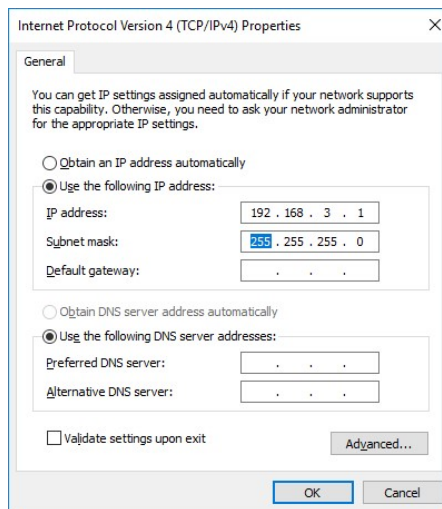
- Search the network for "Microsoft .NET download" and install it.

## 6.2. Licensing

- The igus® Robot Control software requires a license key to be started.
- This key is installed automatically during the installation of igus® Robot Control.
- Please do not change the content of the license file, otherwise it will become invalid!
- The included standard license allows the installation and use of igus® Robot Control on any number of computers in the company or organization of the licensee.

## 6.3. Setting up the Ethernet Connection to the Embedded Computer

- Connect the Ethernet port (ETH0, adjacent to the USB socket) of the Robot to a Windows PC using a standard LAN cable.
- Set the IP of the PC to 192.168.3.1 (the Robot has IP 192.168.3.11). Further assistance on how to change the IP address can be found online.



From this point on the igus® Robot Control programming environment can be used to operate the connected Robot.

## 6.4. Installing the CAN-to-USB Driver

When no Embedded Computer has been supplied, a CAN to USB adapter is supplied that connects the Modular Robot Controller and a computer. It requires the appropriate driver. The Robot Control Software is delivered with the PCAN-USB driver from [www.peak-system.com](http://www.peak-system.com), which can also be found igus® Robot Control installation USB memory stick (directory PCAN-USB-Adapter) or the installation CD of the manufacturer.

After starting the installation, you must:

- accept the license agreement,  
and
- specify the installation folder.

In the next step, please check whether the PCAN-USB device and the PCAN-View CAN-Bus Monitor are selected for installation as shown below.



Choose both for the installation:

- PCAN-USB
- PCAN-View

The PCAN-View CAN-Bus Monitor allows you to check whether the adapter is correctly connected but is not required in day to day operation.



## 7. Moving the Robot with iRC

igus® Robot Control (iRC) is a control and programming environment for robots. The 3D user interface should help to get the robot up and running quickly. Due to its modular design different kinematics and motor drivers can be controlled.

These operating instructions are supplemented with the respective robot-specific operating instructions.

### 7.1. The iRC Graphical User Interface

This section explains the igus® Robot Control software. Without the Robot connected all steps can be simulated. In Section 7.2 the real Robot is then connected and moved.

The iRC programming environment allows control and programming of the Robot. iRC is a Windows software. You can work both online and offline, i.e. with the robot or in simulation (with the robot switched off or disconnected).

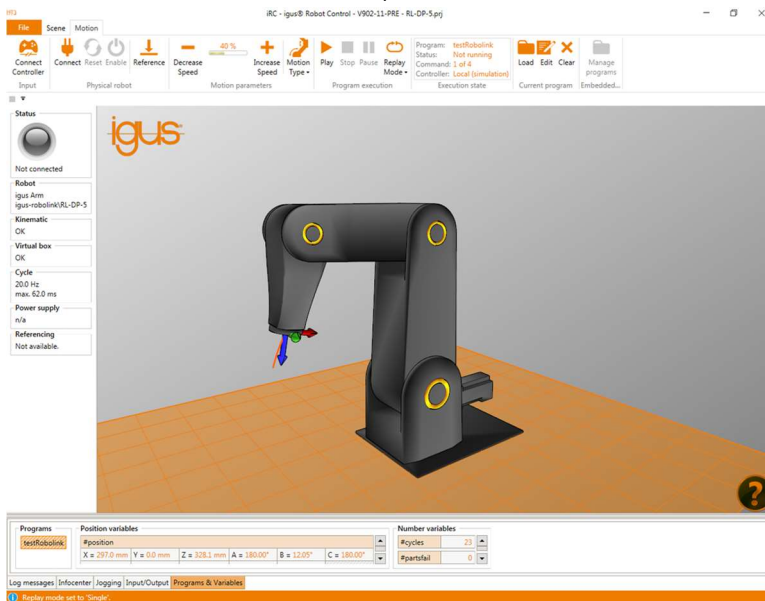


Figure 7.1: igus® Robot Control user interface.

In the upper left corner, the three tabs "File", "Scene" and "Motion" provide access to the main function-sets. In the left corner, information about the current state of the physical Robot is displayed. Additional functions such as loading another project ("Open project") or "Reference robot" can be found in the "File" tab (Figure 7.1).

There are six tabs at the bottom of the window:

- "Log messages": messages from the program about status or errors.
- "Infocenter": shows axis values, the cartesian position and further information.
- "Jogging": keys to move the Robot.
- "Input/Output": display and set the DIO of the Robot Controller.
- "Programs & Variables": displays the current state of program variables.
- "Cameras": provides information on the cameras



The "Help" icon can be found in the bottom right corner and contains links to the Wiki pages (related to "Online documentation", "Software updates", "Examples", "Troubleshooting") and link to "Contact support".

## 7.1.1. Selecting the Robot Type

igus® Robot Control offers project-related settings for different types of Robots, such as 4- or 5-axis Robots. Click on the "File" tab in the upper left corner and select "Open project" (Figure 7.2). Select Robot group e.g. "robotlink". Now click at the named image that fits your specific Robot, e.g. RL-DP-5.

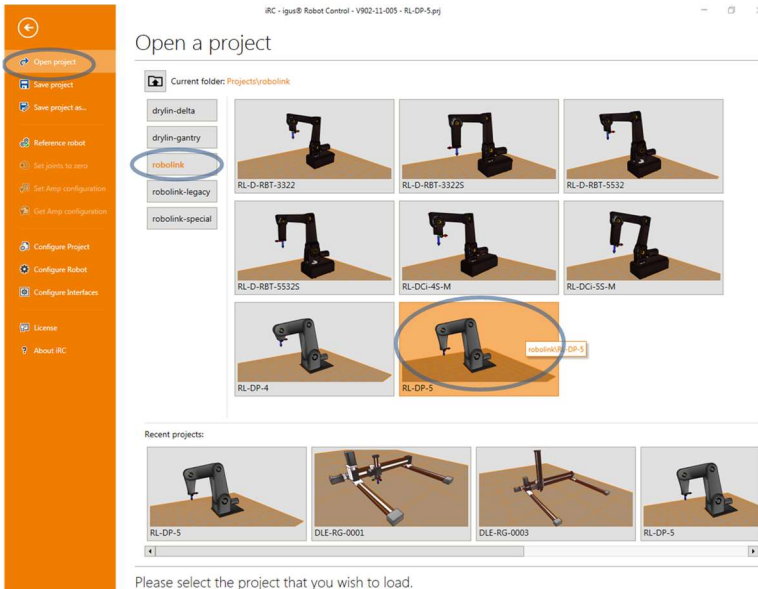


Figure 7.2: Selecting Robot type of the "File" tab in the "Open project".

## 7.1.2. Mouse: Navigation and Moving Robot in the User Interface

To navigate in the igus® Robot Control 3D environment, a 3-button mouse is recommended:

- Left button:
  - Selection of icons and features in the software ribbon.
  - Moving a Robot axis: place cursor over a joint (it will highlight), then click and move the cursor up and down while holding down the left mouse button.
- Middle button/wheel:
  - Navigation in the scene to turn the Robot: move the cursor while holding down the middle mouse button.
  - Mouse wheel: zoom in/out to the current cursor position.
- Right key: pan



The function of the left mouse button can be changed in the "Scene" tab under "Navigation" section, with movement options: "Select", "Pan", "Rotate" and "Zoom". To get back to the initial starting point click "Reset".

## 7.2. Connecting the Robot

### 7.2.1. Connection to Hardware

The real Robot can be controlled like the simulated one, only the hardware has to be connected first by clicking "Connect", "Reset " and "Enable" icons in the "Physical robot" ribbon group in the "Motion" tab (see Figure 7.3). After that the robot needs to be referenced.

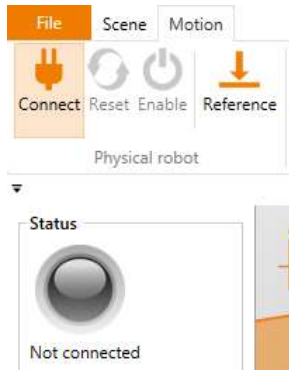


Figure 7.3: Icons for connecting to the hardware, resetting errors and enabling the motors; referencing icon; and "Status" indicator.

1. **"Connect"**: Establish connection to the hardware.
  - This step initializes the USB CAN interface or the Ethernet connection.
  - The "Status" indicator on the left side changes from grey to red.
  - Several error messages are displayed below the "Status" indicator.
2. **"Reset"**: Reset the errors.
  - This key is used to reset the error memories of the electronic modules of the Robot Controller.
  - The axis positions are transferred from the real Robot to the simulation environment. The 3D visualization of the Robot should now correspond to the current position of the real Robot.



**This must be checked with every error reset! If the values do not match, a referencing must be performed, described in Section 7.3.**

- The "Status" indicator remains red. The error messages are deleted, only "Motors not enabled" remains. If other error messages are displayed, try again and follow the instructions in the Robot documentation.
3. **"Enable"**: Activation of the motors.
    - The "Status" indicator is now green.

## 7.2.2. Moving the Robot

It is now possible to move the Robot with: jog buttons, a mouse in the user interface or a gamepad, see Sections 7.1.2 and 7.4.

## 7.3. Referencing the Robot



- After startup the Robot needs to be referenced. Prior to referencing, the axes of the Robot can only perform movements in "Joint" mode. This is to avoid collisions during un-referenced robot operation.
  - Only after referencing, cartesian movements or the start of a program are allowed.
- The reference status is displayed on the left side of iRC.

The Stepper Modules store the motor position in an EEPROM. However, due to gravity or other forces, the joints can move and drop while motor-power is off. In that case, the Stepper Modules will no longer report the correct position to the software, because they were switched off, while the robot has moved. In order to synchronize the position between software, Stepper Motor Module and Robot Axis, a referencing operation must be carried out.

### 7.3.1. Axis 1-4: Referencing via half disc-triggered reference sensors

- The axes of the Robot are equipped with inductive sensors. These are triggered by a metal disc on one half of the axis and free on the other half. During referencing, the Robot moves in the direction of the transition between half-disc/no-half-disc. Once the transition point has been found, it carries out a smaller, slower motion to find the exact midpoint. At that transition point, the offset-corrected encoder reading read and defined as zero position. The offset, i.e. the difference between the position of the reference switch and 0° of the axis, can be changed, see Section 13.4.

### 7.3.2. Axis 5 – Referencing via pin-triggered reference sensors

- The 5<sup>th</sup> axis of the Robot is unusual: there is no half disc, but a "referencing pin".
- This pin should ideally be jogged close to the "referencing position sensor" before starting the referencing (Figure 7.4). This speeds up the referencing process of this particular axis.

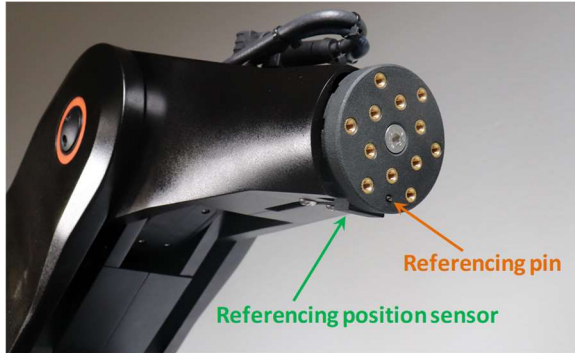


Figure 7.4: Referencing tools of Axis 5.

### 7.3.3. Steps to Referencing

1. Start the Robot Controller and igus® Robot Control.
2. Press the "Connect", "Reset" and "Enable" buttons (Figure 7.3).
3. Move the 5<sup>th</sup> axis of the Robot close to the reference position.
4. Click on "Reference" icon (Figure 7.3) in the "Physical robot" ribbon group in the "Motion" tab (or "Reference robot" via "File" tab) to open the referencing window.
5. Click the buttons to start referencing the axes, see Figure 7.5. Several joints can perform the referencing in parallel.
6. You can also click on "Reference all", then the axes start referencing in a sequence defined in the project file.
7. As soon as all joints have stopped, click "Reset" and "Enable". Now the Robot is fully operational.

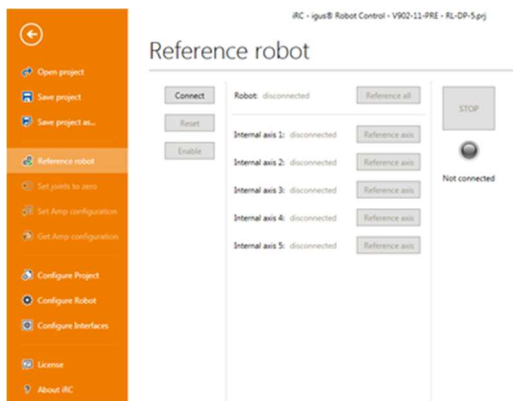


Figure 7.5: Referencing window.

## 7.4. Moving the Robot with Software Buttons or Gamepad

The Robot can be moved (or "jogged") manually while no program is running. The two main controls can be found in "Motion" ribbon:

- "Motion parameters" (to choose motion modes and speed with software buttons),
- "Input" (to connect to a gamepad).

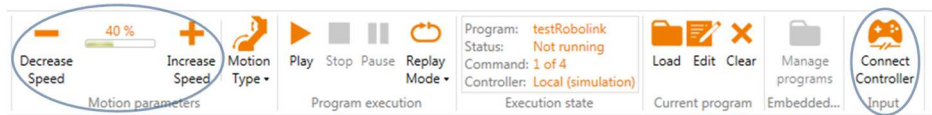


Figure 7.6: Control icons for moving the Robot (marked in blue).

1. **Software buttons** allow to select "Motion Type", which include three modes and in each case the movement speed can be changed between 0 and 100% (Figure 7.6):
  - "Joint": A click on A1 to A6 moves the corresponding robot axis (if available). E1-E3 move the external joints. This could be a **linear or rotational axis** (Figure 7.7).
  - "Base" (cartesian mode), the Robot moves in straight lines along the X, Y and Z axes of the base coordinate system.
  - "Tool" (cartesian mode), the Robot moves in X, Y and Z of the current tool coordinate system.

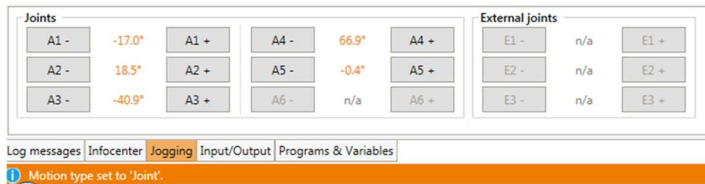


Figure 7.7: The software buttons for "Joint" movements in "Jogging" tab. In both cartesian modes, the buttons change to X, Y, Z, A, B and C.

2. A **gamepad** is potentially the most intuitive way to move the Robot. Figure 7.8 shows the key assignments. By pressing "Connect Controller", igus® Robot Control connects to a gamepad. If the connection was successful, an OK sign is displayed on the controller icon. The device must be of the "Joystick" or "Gamepad" type. You can find further information on establishing a connection in the "LogMessages" tab (on the bottom of the screen).



Legend:  
 1. Change motion type  
 2. Change key assignments:  
 Switching between X, Y, Z and A, B, C in Cartesian motion mode, or A1, 2, 3 and A4, 5, 6 in Joint motion mode.

Figure 7.8: Key assignments of the Gamepad.

## 7.5. Starting Robot Programs

The robot program must be loaded and started.

1. Load the program: click the "Load" folder icon in the "Current program" ribbon group of the "Motion" tab and select a program, e.g. igus5DOF\_TestMotion.xml

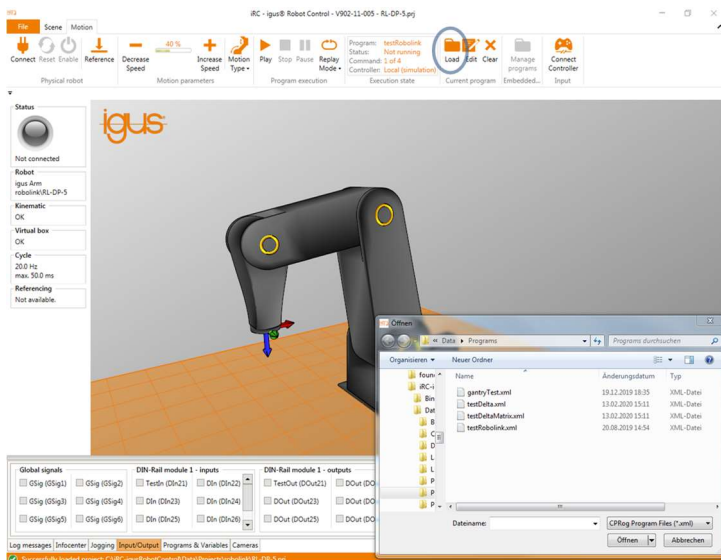


Figure 7.9: Loading a new program (circled in blue).



## 2. Adjust the speed-override:



- Before starting a new program, set the speed to e.g. 20%.
- Be particularly attentive during the first complete program run and keep the Emergency Stop Button ready.

## 3. Start the program:



- Click the "Play" icon in the "Program execution" ribbon group of the "Motion" tab.

## 4. Stop or pause the program:

- After pressing the "Pause" icon, the Robot can continue with the program by clicking the "Play" icon again.
- After pressing the "Stop" icon, the program starts with the first command when the "Play" icon is clicked again.
- The "Replay Mode" can be set to:
  - Single (program stops after a single cycle).
  - Repeat (program does not stop until "stop" or "pause" is pressed).
  - Step (program steps line-by-line. This is useful for debugging a program).

## 7.6. Digital Inputs and Outputs

The easiest way to connect a programmable logic controller (PLC) is via digital inputs and outputs. Each Robot Controller comes with one DIO Module. This provides 7 inputs and 7 outputs (see Section 4.4). If additional inputs and outputs are required, up to two additional DIO Modules can be integrated, see Section 10.1. A total of up to 3 DIO Modules can be controlled.

The state of the inputs and outputs can be monitored in the "Input/Output" section of iRC.

Both inputs and outputs can be set manually:

- Outputs can be set manually, when no program is running.
- Inputs can be set only in simulation when there is no robot connected. Hence one can test the reaction of programs on different inputs in the absence of hardware.

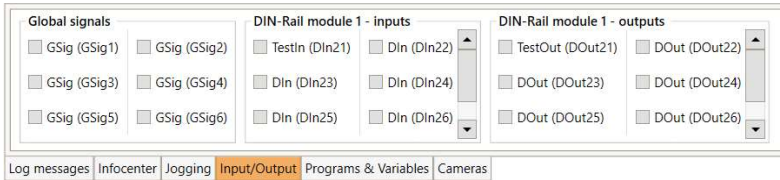


Figure 7.10: "Input/Output" section of the iRC software (tab at the bottom of the window).

See Section 4.5 on how to set up the electrical connections and Section 10.1 on how to extend the number of available Digital Inputs / Outputs. It is also possible rename the different inputs/outputs.

The programming with digital Inputs / Outputs is shown in Sections 0 and 8.6.

## 7.7. Software Interfaces

Various interfaces are available from the Robot Controller:

- PLC interface for controlling the Robot via a PLC, in particular for starting and stopping programs.
- Plug-in interface to integrate e.g. cameras. The plug-in then transmits target positions to the Robot Controller.
- Container Runtime Interface (CRI) to enable further interaction. This interface can be used, for example, to generate workpiece-specific programs from a database.
- ROS interface to integrate the Robot into the Robot Operating System ([www.ros.org](http://www.ros.org)).

See Section 10 for the configuration of these interfaces.

## 7.8. Updating the Software

Updates of the iRC software can be found on the [igus LowCostAutomation](http://www.igus.com) website.

**BackUp:** Please rename your old iRC-igusRobotControl folder to e.g. C:\iRC-igusRobotControl\_BAK before starting the installation. This way, you can switch back to the old version in case of doubt. The following has to be transferred from the previous installation:

- The created robot programs;
- Changes in the project or robot configurations.

# 8. Programming the Robot with iRC

igus® Robot Control allows the creation of robot programs. The type of programming is called “Teach-In-Programming”, which works as follows:

1. Move the robot manually to the position you want to record
2. Record the position and define how to reach this position (linear/joint motion)
3. Repeat these steps and in between add digital output commands or program flow commands.

The integrated Program Editor is provided for setting up and editing these robot programs.

## 8.1. The Program Editor

In the robot program each command consists of one line, e.g. “Joint Motion” or “Wait”. Only commands that direct the program flow are separated into several lines, e.g. “Loop” and “EndLoop”. The Program Editor is opened with the “Edit” button in the “Motion” tab.

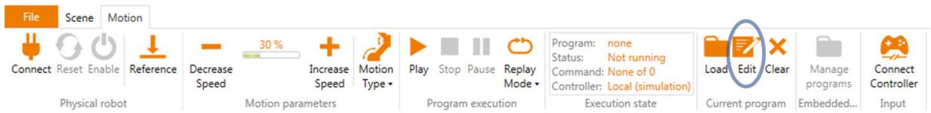


Figure 8.1: Opening the Program Editor with the “Edit” button.

Then the following window opens, here with a short program:

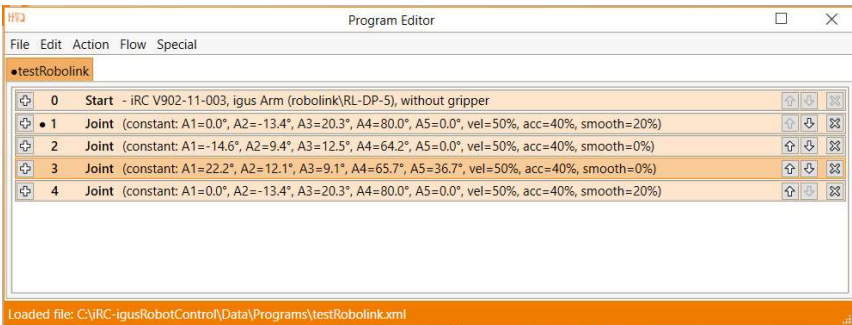
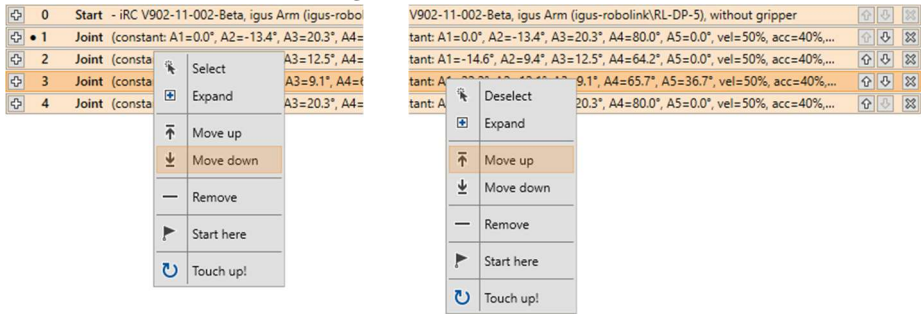


Figure 8.2: Program Editor with a short program.

The following sections show how to write programs with the Program Editor.

### 8.1.1. Changing the Command Sequence

To move the command down, click on “Move down”, on “Move up” to move up. It is also possible to use the arrows on the right side of the commands.

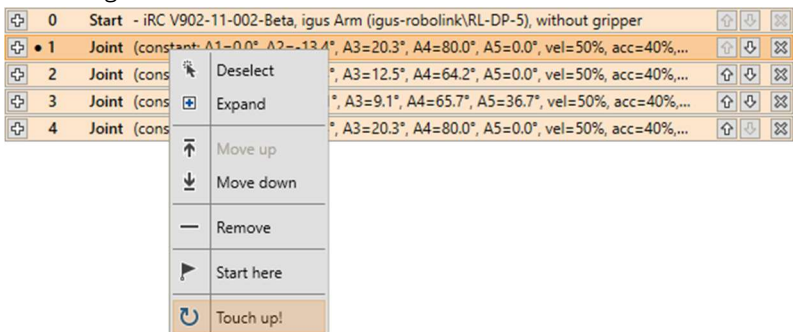


The Program Editor will prevent commands that would render the structure of a program illegal. If moving of a command in a up or down is impossible, the respective buttons and menu items will be greyed out.

## 8.1.2. Touching up Commands

Certain commands require position constants as parameter. Often, it is desirable to use the current position of the robot. Entering it by hand can take quite some time and is error-prone. For such cases the possibility to “Touch up” the command is offered. You can do this by, either:

- selecting the command (see Section 1.3.1) and then pressing Ctrl+T
- clicking “Touch up!” in the context menu that pops up when clicking on the header with the right mouse button:



This functionality is also found in the menu: “Edit”->“Touch up!”

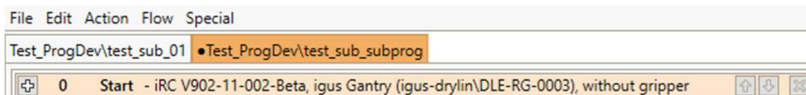
The Program Editor will then replace the position constant in the command with the current position of the Robot.

### 8.1.3. Current Starting Command

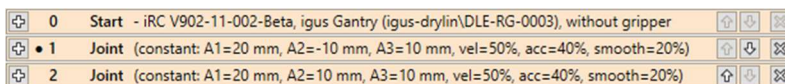
It is possible to execute programs command-wise, or to pick a specific command as starting point of a program for debugging purposes.

The command that will be executed first, next time the program is started, or – if the program is currently running – the currently executed command is indicated in the Program Editor.

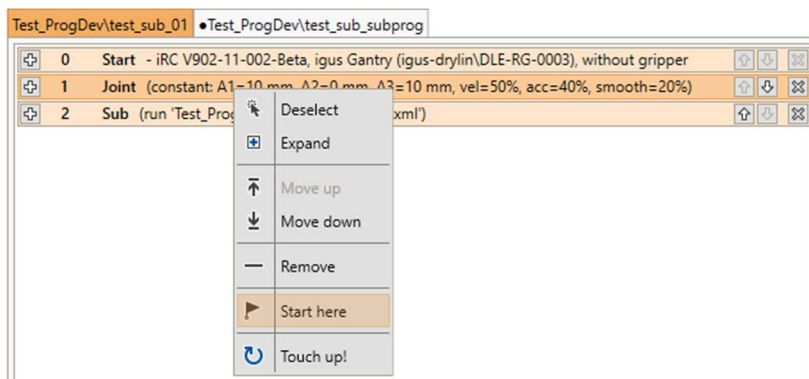
The subprogram that contains this command is designated by a filled black circle in front of the Program Name:



The command itself is designated by a filled black circle in the header bar of the command:



To choose a specific command as starting point for execution, click on “Start here” in the context menu that pops up when clicking on the header with the right mouse button:

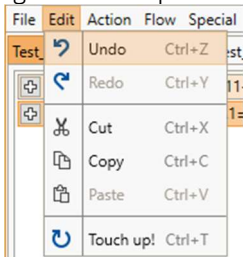


## 8.1.4. Undo and Redo

All changes made in the Program Editor can be undone / redone.

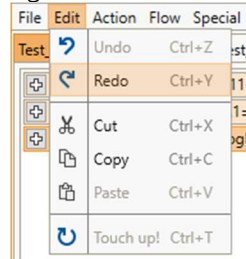
To **undo** you can:

- Press Ctrl+Z.
- Click on the menu item “Edit”->“Undo” (it will be greyed out if no changes have been performed so far).



To **redo** a previously undone change you can:

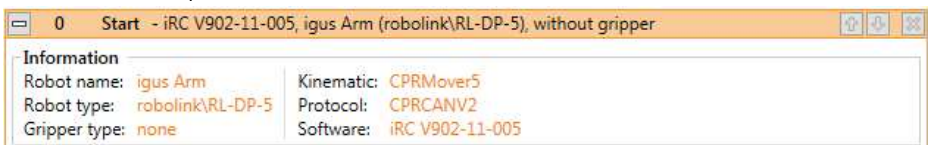
- Press Ctrl+Y.
- Click on the menu item “Edit”->“Redo” (it will be greyed out if no changes have been undone before).



## 8.2. Comments and Information in Programs

### 8.2.1. Program Information

The Program Editor will introduce the **Start** pseudo-command at the beginning of each program. It does not represent a real command in the sense that it will not be executed in any way. Its purpose is to display information about the current hardware, software and kinematic. It is not possible to move or remove it.



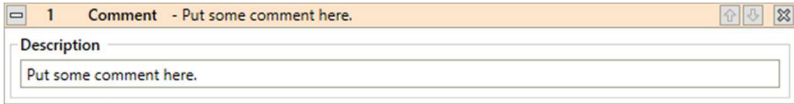
### 8.2.2. Descriptions

Any command of a program contains a description. It can be filled with any text and should be used to describe what the intention behind the respective command is to other programmers.

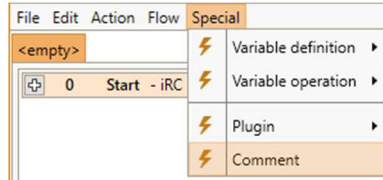


## 8.2.3. Comments

The **Comment** command can be used to include pure descriptions into programs. It has no effect on the robot during execution, its sole purpose is to provide a Description.



It is accessible in iRC's Program Editor through the menu entry "Special"->"Comment":



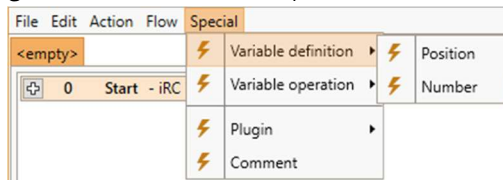
## 8.3. Variables and Variable Access

Two types of variables are supported in programs for iRC and TinyCtrl:

- Number variables: these can be used to store integer or floating point numbers
- Position variables: these can be used to store Cartesian positions and joint positions. Whether such a variable will be interpreted as cartesian position or joint position depends on context.

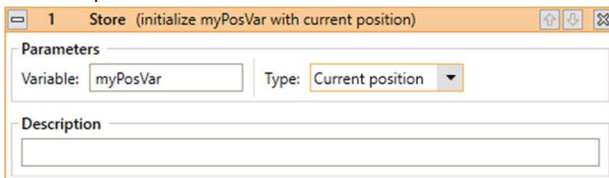
### 8.3.1. User Variables

It is possible to define user-variables with the **Store** command, which is accessible in iRC's Program Editor through the menu entries under "Special"->"Variable definition".



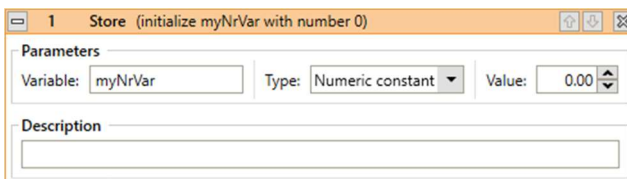
It is possible to choose three types of Store operations:

- "Current position":



A position variable will be initialized with the cartesian and joint position that the robot has when the command is being executed.

- "Number constant":



A number variable will be initialized with the constant given under "Value".



- “Position constant”:

A position variable will be initialized with the constants given under “Cartesian position”, “Joint position” and “External axes”. Depending on the kinematic model of the current robot certain axes may be unavailable.

The name of the variable can be set under “Variable”. If a variable with the same name has already been defined its value and type will be overwritten. All variables are global, i.e. they will also be accessible from subprograms.

### 8.3.2. System Variables

The following predefined variables are available without the need to define them:

- **#position**  
The current position of the robot. The units of the Cartesian position are mm for the components X,Y,Z and degrees for the Euler angles A,B,C. Joint values are measured in mm or degrees, depending on the type of the joint.

Note that the system controls the values of predefined variables – they cannot be changed by the program. The names of predefined variables always start with a “#”.

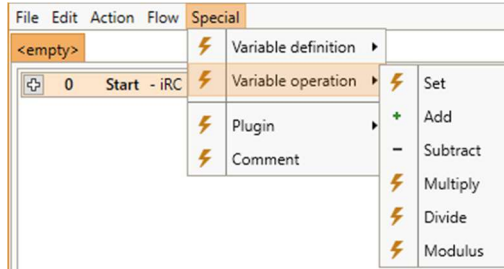
### 8.3.3. Element Access

The following elements of the variables can be accessed with the “.” Operator:

- **Position:** X, Y, Z                                   E.g.: myvariable.x
- **Rotation:** A, B, C                                   E.g.: myvariable.a
- **Joints:** A1, A2, A3, A4, A5, A6, E1, E2, E3   E.g.: myvariable.a5

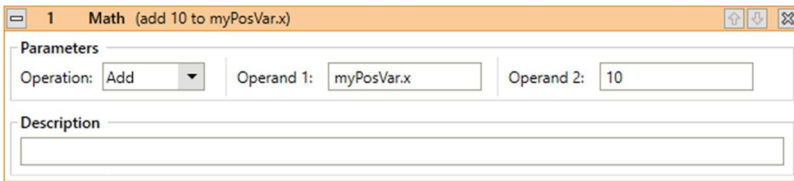
## 8.3.4. Operations on Variables

Operations on variables can be performed by using the **Math** command, which is accessible in iRC's Program Editor through the menu entries under "Special"->"Variable operation".



"Operand 1" defines the first operand of the operation that is to be performed. It will also be used to store the result of the operation. Only names of user-defined number variables or components of user-defined position variables are allowed.

"Operand 2" defines the second operand of the operation. It can contain numeric constants, names of number variables or components of position variables.



The following operations are supported and can be chosen under "Operation":

- **Set:**  
This will set the first operand to the value of the second operand.
- **Add:**  
This will increment the first operand by the value of the second operand.
- **Subtract:**  
This will decrement the first operand by the value of the second operand.
- **Multiply:**  
This will multiply the first operand by the value of the second operand.
- **Divide:**  
This will divide the first operand by the value of the second operand.
- **Modulus:**  
This will store the remainder of the division of the first operand by the second operand into the first operand.

### 8.3.5. Monitoring Variables

You can monitor the current values of all defined variables under iRC in the "Programs & Variables" tab:

Position variables					
#position					
X = 326.7 mm	Y = 0.0 mm	Z = 262.3 mm	A = 180.00°	B = 0.00°	C = 180.00°
A1 = 0.00°	A2 = 10.00°	A3 = 0.00°	A4 = 80.00°	A5 = 0.00°	

Number variables	
#cycles	23
#partsfail	0
#partsgood	0

Log messages | Infocenter | Jogging | Input/Output | **Programs & Variables**

## 8.4. Execution Flow

### 8.4.1. Conditional Expressions

Conditional expressions can be used in if-then-else commands, conditional loops and as abort conditions in motion commands. The conditional statements can be combinations of digital inputs, global signals, Boolean operations and comparisons. Examples:

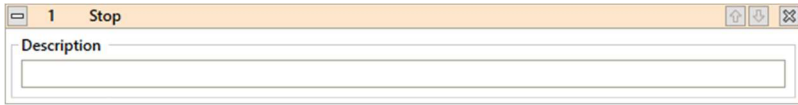
- `DIn23` True, if digital input 23 is high
- `DIn23 AND !DIn27` True, if digital input 23 is high and digital input 27 is not high
- `modelclass = 31` True, if variable modelclass is 31
- `mempos.x > 350.0` True, if the x component of the position variable mempos is higher then 350

The syntax of conditional statements is given by the following EBNF definition:

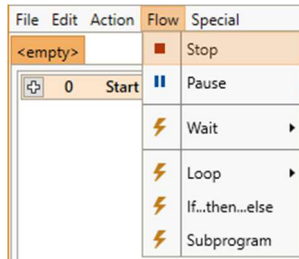
Expression	:= [ "]" <Boolean> { <BooleanOperator> <Boolean> } ...
Boolean	:= <BooleanConstant>   <Expression>   "(" <Expression> ")"   CompExpression   "(" <CompExpression> ")"   <DigitalInputs>   "(" <DigitalInputs> ")"
BooleanOperator	:= "And"   "Or"
BooleanConstant	:= "True"   "False"
DigitalInputs	:= <ChannelType> <ChannelId>
ChannelType	:= "Din"   "GSig"
ChannelId	:= an integer value
CompExpression	:= <CompValue> <Comparator> <CompValue>
CompValue	:= <Variable>   <Number>
Variable	:= <NumberVariable>   <PositionComponent>
NumberVariable	:= Name of a number variable
PositionComponent	:= <PositionVariable> "." <Component>
PositionVariable	:= Name of a position variable
Component	:= "x"   "y"   "z"   "A"   "B"   "C"   "A1"   "A2"   "A3"   "A4"   "A5"   "A6"   "E1"   "E2"   "E3"
Number	:= Integer or Float number
Comparator	:= "="   ">"   "<"   ">="   "<="

## 8.4.2. Stop

The **Stop** command stops the program execution.

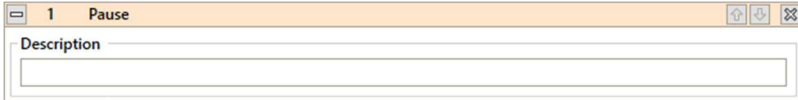


It is available through the menu entry "Flow"->"Stop".

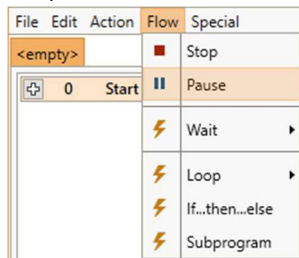


## 8.4.3. Pause

The **Pause** command pauses execution of the program. Execution can later be resumed by the user.

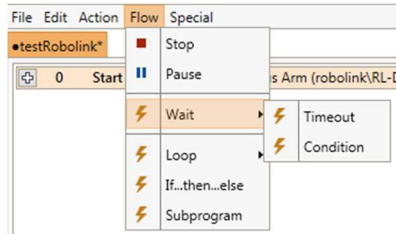


It is available through the menu entry "Flow"->"Pause".



## 8.4.4. Wait

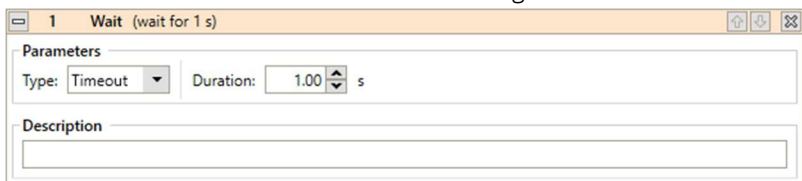
The **Wait** command allows the robot to wait for a fixed amount of time or until a conditional statement evaluates to true. It is accessible through the menu entries under “Flow”->“Wait” in iRC’s Program Editor.



The different modes can be chosen under “Type”:

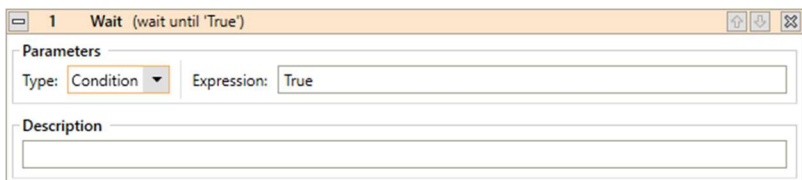
- “Timeout”:

This will cause the robot to wait for the time that is given under “Duration”.



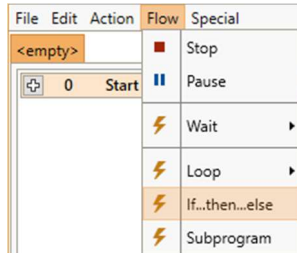
- “Condition”:

This will cause the robot to wait until the conditional statement given under “Condition” evaluates to “True”.

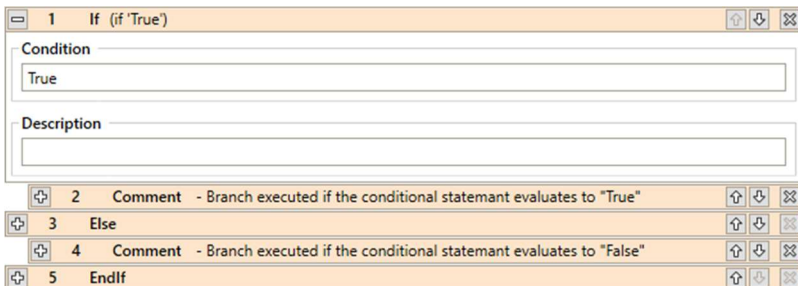


## 8.4.5. If-then-else

The If command branches execution of the program depending on the value of a conditional expression. It is accessible through the menu entry "Flow"-*"If...then...else"* in iRC's Program Editor.



The conditional expression given under "Condition" must match the syntax described in Section **Fehler! Verweisquelle konnte nicht gefunden werden.** . The branch between "If" and "Else" will be executed if it evaluates to true, the branch between "Else" and "EndIf" will be executed otherwise.



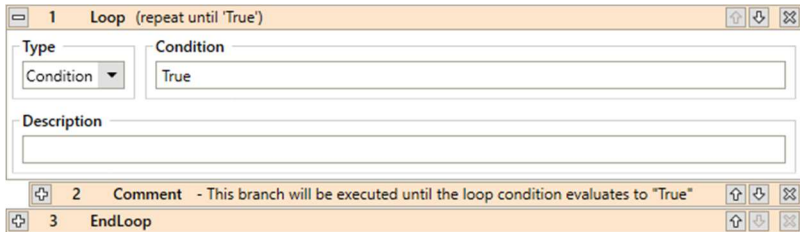
## 8.4.6. Loops

The **Loop** command allows to define execution loops. It is possible to choose between the following types of loops under "Type":

- "Condition"

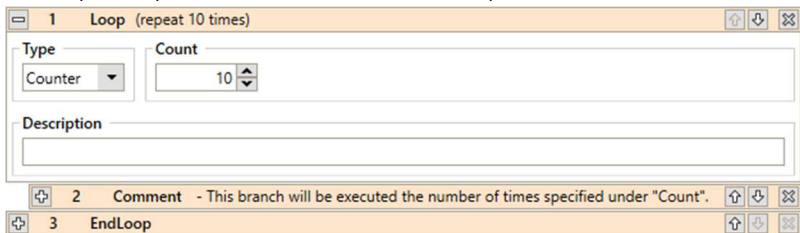
The loop will repeat until the conditional expression given under "Condition" evaluates to "True". It must follow the syntax described in Section **Fehler!**

**Verweisquelle konnte nicht gefunden werden.** .

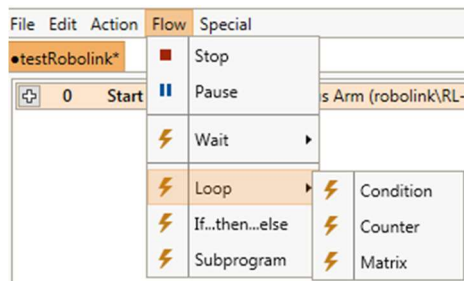


- "Counter"

The loop will repeat the number of times that is specified under "Count".



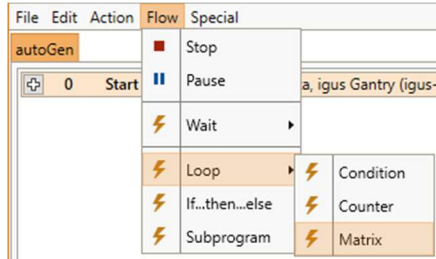
The loop command is accessible via the menu items "Flow"->"Loop"->"Condition" and "Flow"->"Loop"->"Counter".



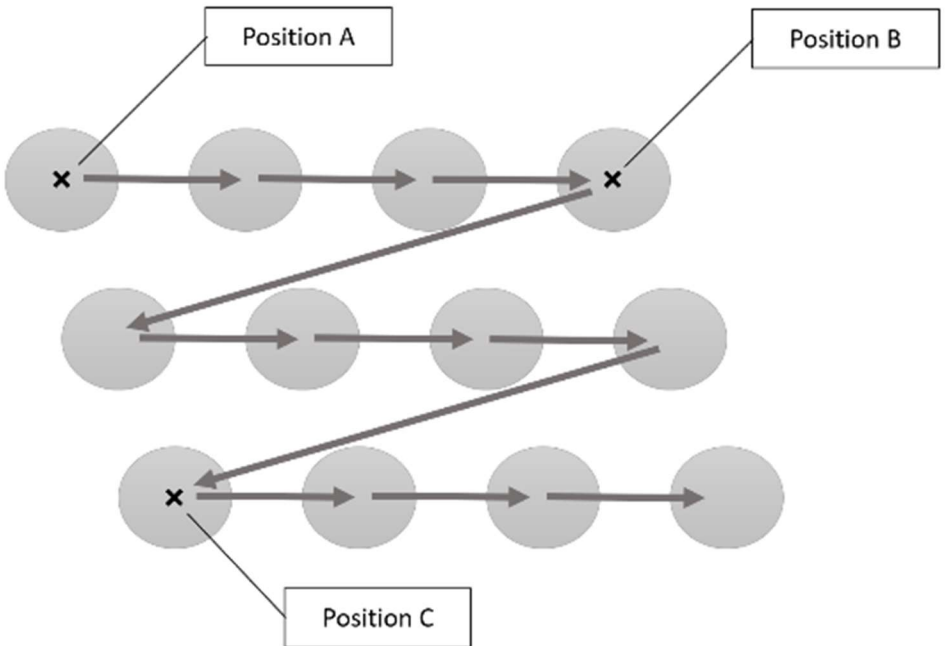


## 8.4.7. Matrix Loops

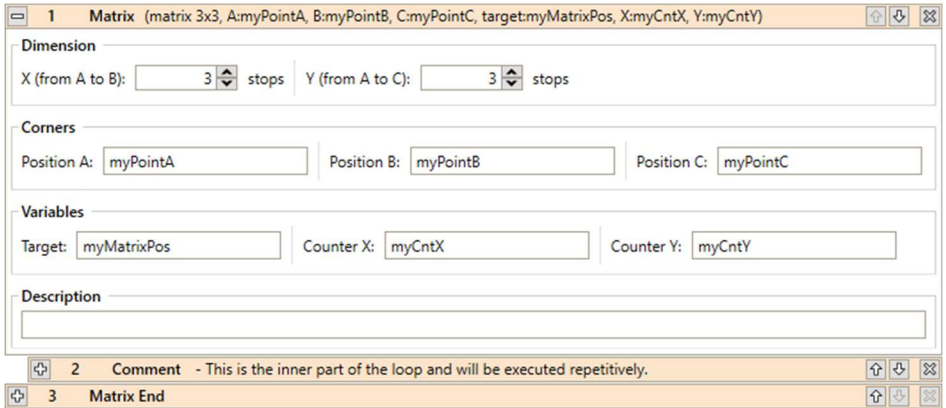
The **Matrix** command is designed to execute loops that allow the robot to perform raster movements, for example for palletizing tasks. It can be accessed through the menu item "Flow"-">"Loop"->"Matrix".



The illustration below shows the movement pattern that can be achieved by using the Matrix command.



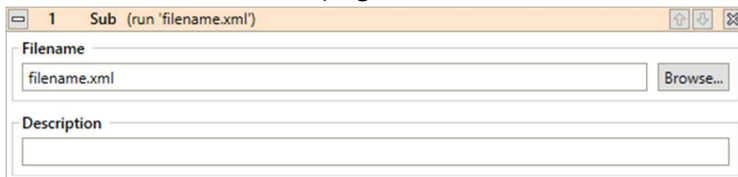
The position variables given as "Position A", "Position B" and "Position C" define the corners of the area that will be covered by the matrix loop (see illustration above). The number of steps that are to be made are given by "X (from A to B)" and "Y (from A to C)". For example, in the illustration above, X would be 4 and Y would be 3.



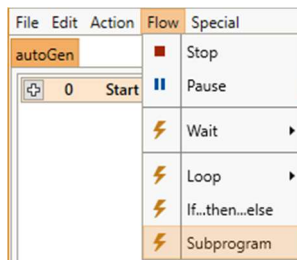
The block between “Matrix” and “Matrix End” will be executed for each step. The position variable given to “Target” will contain the position of the current target point for the current step. Row and column of the current step will be stored in the number variables given to “Counter X” and “Counter Y” respectively.

## 8.4.8. Subprograms

The **Sub** command can be used to call subprograms.



The path to the file of the subprogram is given under “Filename”. It is relative to the Programs subfolder of iRC’s Data folder. The command can be accessed via the menu item “Flow”->“Subprogram”.



## 8.5. Motion

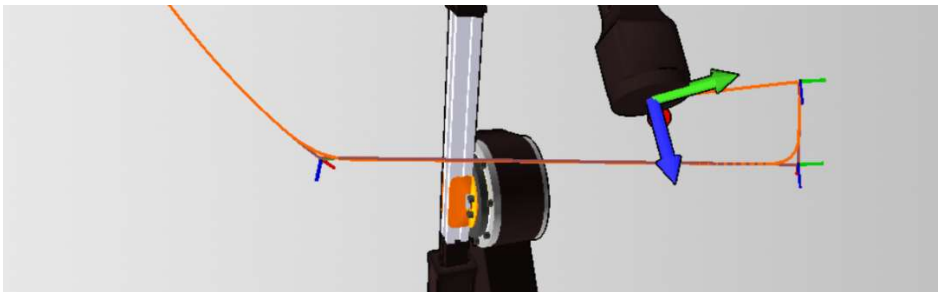
### 8.5.1. Abort Conditions

An abort condition can be given to any motion command. It is a conditional expression, following the syntax described in Section 8.4.1. During the execution of the motion command the statement is continuously evaluated, and the moment it evaluates to “True” the robot will stop moving. It can be specified under “Abort condition” for each motion command respectively:

Abort condition
False

### 8.5.2. Acceleration and Smoothing

In Motion commands joint accelerations can be defined, to prevent abrupt movement of the robot. This can be done for a single motion command, but also for a combination of up to a maximum of 10 commands. It is also possible to smooth out the corners within the path defined by blocks of consecutive motion commands. This allows the robot to go through the corners without coming to a full stop. If corner smoothing is set to 0% the robot stops at every corner for a short moment. This leads to a more abrupt motion.



Any non-motion command will divide such a motion set and interrupt smoothing.

#### Parameters:

Acceleration:  % Smoothing:  %

- “Acceleration”:  
Percentage of the maximally possible acceleration.
- “Smoothing”:  
Scope of the smoothing in percent.

### 8.5.3. Joint Motion

The **Joint** command moves the robot to a (absolute) target position given in joint coordinates. The resulting motion of the TCP will typically be a curve and not a straight line. The target position can be given in the following ways (choose corresponding “Source”):

- “Constant”:

The target position is a constant value for each axis.

The screenshot shows the configuration window for a 'Joint' command with the source set to 'Constant'. The title bar reads '1 Joint (constant: A1=0.0°, A2=10.0°, A3=0.0°, A4=80.0°, A5=0.0°, vel=50%, acc=40%,...'. The 'Source' dropdown is set to 'Constant'. The 'Parameters' section includes 'Velocity: 50 %', 'Acceleration: 40 %', and 'Smoothing: 20 %'. The 'Constant' section has input fields for 'Axis 1: 0.00 °', 'Axis 2: 10.00 °', 'Axis 3: 0.00 °', 'Axis 4: 80.00 °', 'Axis 5: 0.00 °', and 'Axis 6: n/a'. The 'Abort condition' is set to 'False' and the 'Description' field is empty.

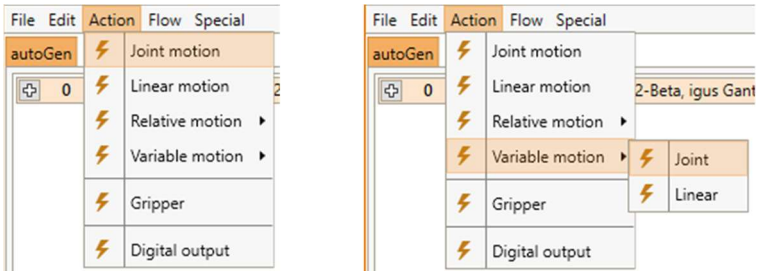
- “Variable”:

The target position is taken from the position variable given to “Variable”.

The screenshot shows the configuration window for a 'Joint' command with the source set to 'Variable'. The title bar reads '1 Joint (variable: var=myPosVar, vel=50%, acc=40%, smooth=20%)'. The 'Source' dropdown is set to 'Variable'. The 'Parameters' section includes 'Velocity: 50 %', 'Acceleration: 40 %', and 'Smoothing: 20 %'. The 'Variable' section has a text input field containing 'myPosVar'. The 'Abort condition' is set to 'False' and the 'Description' field is empty.

The movement speed is given by “Velocity”. It is measured in percent of the maximally allowed movement speed for the respective robot axes.

The Joint command can be accessed in iRC's Program Editor under the menu entries "Action"->"Joint motion" and "Action"->"Variable motion"->"Joint".

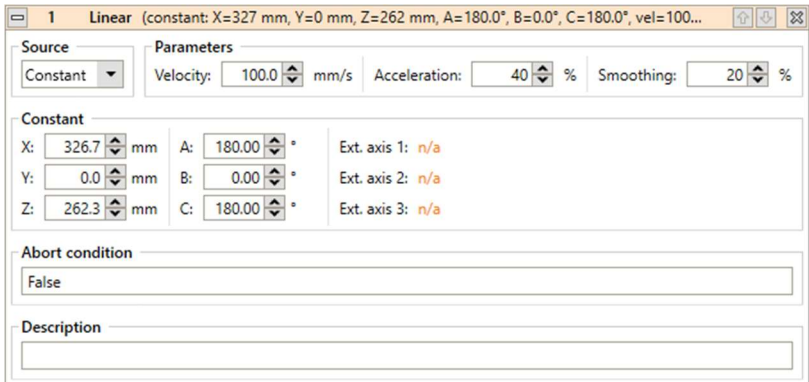


## 8.5.4. Linear Motion

The **Linear** command moves the robot to an (absolute) target position given in Cartesian coordinates. The resulting motion of the tool center point (TCP) will follow a straight line. The target position can be given in the following ways (choose corresponding "Source"):

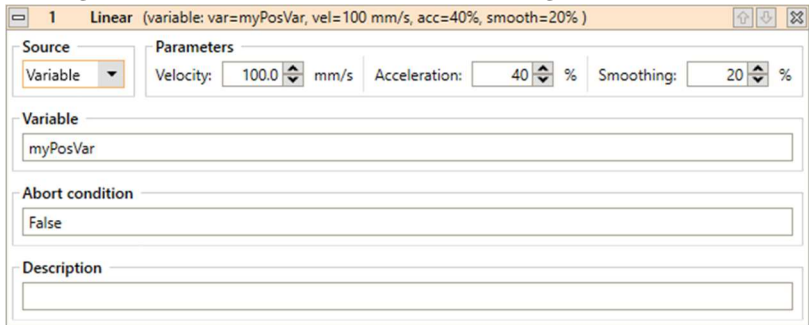
- "Constant":

The target position is a constant given by Cartesian coordinates X,Y,Z and Euler Angles A,B,C and positions of external axes if supported by the current robot kinematic.



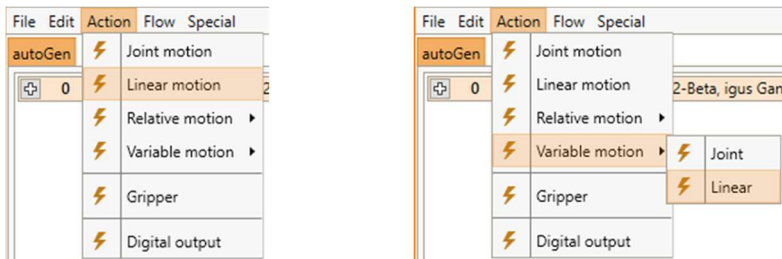
- “Variable”:

The target position is taken from the position variable given to “Variable”.



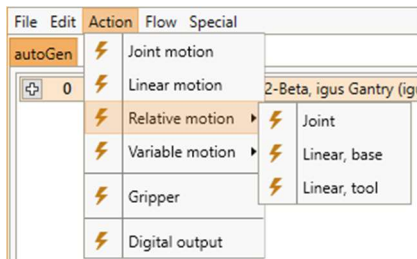
The movement speed is given by “Velocity”. It is measured in mm/s, if it exceeds the maximally allowed movement speed of the robot it will cause a kinematic error during execution.

The Linear command can be accessed in iRC’s Program Editor under the menu entries “Action”->“Linear motion” and “Action”->“Variable motion”->“Linear”.



## 8.5.5. Relative Motion

The **Relative** command allows to move the robot relative to its current position. It can be accessed via the menu items under “Action”->“Relative motion”.



The following modes of relative motion can be chosen under “Type”:

- “Joint”:

The relative offset is specified in joint coordinates. The movement speed is given by “Velocity”. It is measured in percent of the maximally allowed movement speed for the respective robot axes.

The screenshot shows a configuration window for a 'Relative' motion mode. The title bar indicates '1 Relative (joint: A1=0.0°, A2=0.0°, A3=0.0°, A4=0.0°, A5=0.0°, vel=20%, acc=40%, smooth=20%)'. The 'Type' dropdown is set to 'Joint'. The 'Parameters' section includes 'Velocity: 20 %', 'Acceleration: 40 %', and 'Smoothing: 20 %'. The 'Offset' section shows values for six axes: Axis 1: 0.00°, Axis 2: 0.00°, Axis 3: 0.00°, Axis 4: 0.00°, Axis 5: 0.00°, and Axis 6: n/a. The 'Abort condition' is set to 'False' and the 'Description' field is empty.

- “Linear – Base”:

A linear movement with an offset specified in Cartesian coordinates will be performed. The coordinate system used for the offset is the robot coordinate system. The speed is given by “Velocity”. It is measured in mm/s, if it exceeds the maximally allowed movement speed of the robot, it will cause a kinematic error during execution.

The screenshot shows a configuration window for a 'Relative' motion mode. The title bar indicates '1 Relative (base: X=0 mm, Y=0 mm, Z=0 mm, vel=50 mm/s, acc=40%, smooth=20%)'. The 'Type' dropdown is set to 'Linear - Base'. The 'Parameters' section includes 'Velocity: 50.0 mm/s', 'Acceleration: 40 %', and 'Smoothing: 20 %'. The 'Offset' section shows values for X, Y, and Z axes: X: 0.0 mm, Y: 0.0 mm, Z: 0.0 mm. The 'Abort condition' is set to 'False' and the 'Description' field is empty.

- “Linear – Tool”:

A linear movement with an offset specified in Cartesian coordinates will be performed. The coordinate system used for the offset are tool coordinates. The movement speed is given by “Velocity”. It is measured in mm/s, if it exceeds the maximally allowed movement speed of the robot it will cause a kinematic error during execution.

1 Relative (tool: X'=0 mm, Y'=0 mm, Z'=0 mm, vel=50 mm/s, acc=40%, smooth=20%)

Type: Linear - Tool

Parameters: Velocity: 50.0 mm/s Acceleration: 40 % Smoothing: 20 %

Offset

X': 0.0 mm	A': n/a	Ext. axis 1: n/a
Y': 0.0 mm	B': n/a	Ext. axis 2: n/a
Z': 0.0 mm	C': n/a	Ext. axis 3: n/a

Abort condition: False

Description:



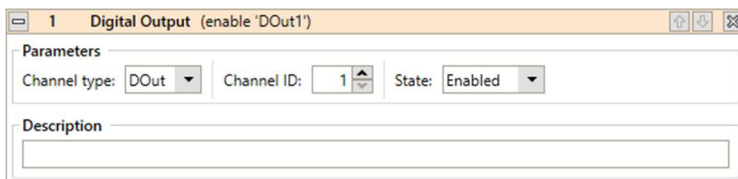
## 8.6. Gripper and Digital IO

### 8.6.1. Digital Inputs

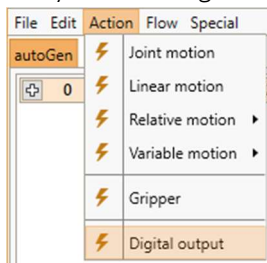
Digital inputs and global signals are accessible as reserved words in conditional statements as expressions in Section 4.1. For global signals these are GSig1, GSig2, ... , for digital inputs these are Din1, Din2, ... . Depending on the state they will either evaluate to "True" or to "False".

### 8.6.2. Digital Outputs

The **Digital Output** command can be used to set the states of digital outputs and global signals.

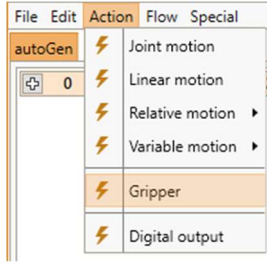


Under "Channel type" it is possible to specify if a digital output or a global signal are to be set. "Channel ID" specifies the channel of the digital output or global signal. Under "State" the desired state after execution of the command is specified. The command is accessible in iRC's Program Editor by the menu entry "Action"->"Digital output".



### 8.6.3. Opening/Closing the Gripper

The **Gripper** command allows to control the gripper of the robot. It is accessible in iRC's Program Editor by the menu entry "Action"->"Gripper".



You can set the desired aperture, measured in percent, under "Aperture". A value of 0% represents a fully closed gripper, 100% represents a fully opened gripper. For grippers that can only be either fully opened or fully close the threshold between these states is at 50% aperture.



## 8.7. Camera

We are currently working on this feature. The command specification will follow soon.

## 9. Stand-alone Operation with Embedded Computer and the Operating Panel

To run the robot without a Windows PC connected, the Embedded Computer is necessary. It runs the TinyCtrl software as Robot Control Software. The Embedded Computer allows to jog the Robot Arm (using the Operating Panel), and to replay robot programs. To set up new programs it is connected via Ethernet to a Windows Computer running iRC.

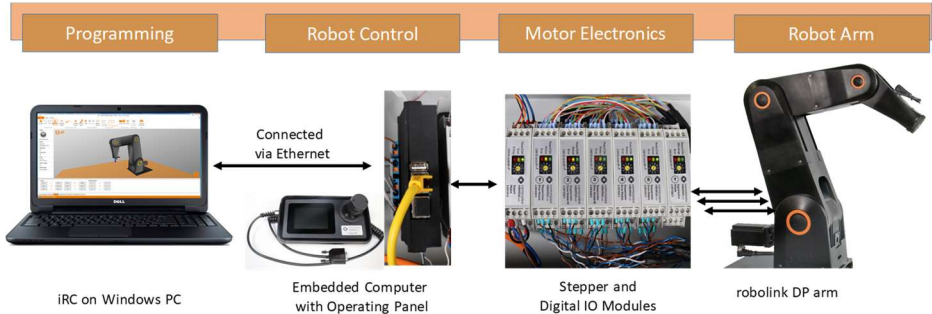


Figure 9.1: robotlink DP with Embedded Computer as Robot Control.

This section shows how to operate the robot with Embedded Computer and Operation Panel.

After all electrical connections to the Robot have been established, the Robot has been switched on and the Emergency Stop Button has been released, the errors first need to be "reset" and then the robot needs to be "enabled".

When moving the Robot, always keep a hand on the Emergency Stop Button to prevent it from hitting an object unexpectedly, for example if it is about to collide with the table.

## 9.1. Reset Errors/Enable Robot



On the touchscreen display, press the "Enable" button in the upper menu (top right) to get to the Enable page.



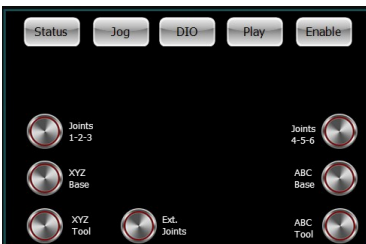
Now press "Reset":  
The status changes to "MNE" (Motor Not Enabled)



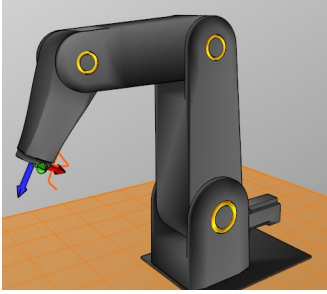
Press "Enable":  
The status changes to "No Error".  
"Not ref'd!" (in red) means not referenced.

## 9.2. Moving the Robot with the 3-Axis Joystick

Once the Robot is enabled, the axes of the Robot can be moved.



1. To do this, press the "Jog" button at the top of the display
2. Press A1.  
Then move and turn the joystick.  
You can now move the axes 1, 2 and 3.
3. Press A4.  
Now, move axes 4 and 5 by moving and rotating the joystick.



4. Repeat steps 2 and 3 until:
  - the Robot is roughly in a "gallows position" and
  - the "reference pin" (Figure 7.4) on the flange of 5<sup>th</sup> axis points towards the Robot base.

### 9.3. Referencing

To enable an automatic program sequence, the Robot electronics must be referenced. The type of referencing movement depends on the Robot or encoder type.



1. To do this, press the square "Enable" button at the top edge of the display.
2. The Robot must be in "NoError" status. If it is in an error state, enable it by pressing the round "Reset" button followed by the "Enable" button.
3. Now press the "Ref All" button to reference all axes. The Robot now carries out search movements for each axis.
4. Referencing is complete when there is a 1 next to Ref A1-A5 respectively. For a 4-axis Robot, Ref A5 will remain at 0. Ref E1 is intended for additional external axes.
5. After referencing, the axes are in the error state. This is necessary because the actual position has changed in the referencing process.



6. Now press the round "Reset" button, followed by the round "Enable" button. The Robot is now referenced.

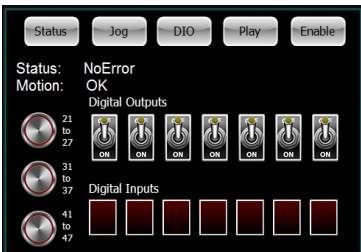
### 9.4. Starting and Stopping a Program



1. To do this, press the "Play" button at the top edge of the display and use the "prev" and "next" buttons to select the *Igus5DOF\_TestMotion* program.
2. Load the program using the "Load" button
3. Let the program play once using "Single Play".
4. "Cont. Play" plays the loaded program continuously.
5. "Stop" stops the movement.
6. The slider "Override" can be moved to the right to increase the speed of the movement or to the left to decrease it.

### 9.5. Manually Setting the Digital Inputs/Outputs

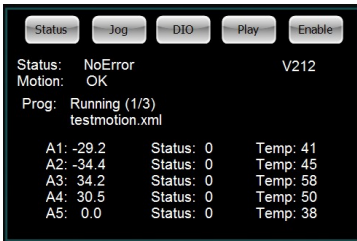
The digital outputs can be activated and deactivated by pressing the illustrated switches.



- If e.g. a gripper is connected to the DIO Module, this can be activated or deactivated by switching the illustrated switches.
- If a signal is present at the digital input, it will be displayed in the digital inputs field at the bottom edge of the display.
- The three buttons on the left side allow you to switch between several DIO Modules.

## 9.6. Display of Status Information

Status information can be displayed by pressing the "Status" button at the top left corner of the display.



The name of a program is displayed, e.g. while a program is running.

The axis positions of axes 1-5 (or 1-4 for 4-axis Robots) are displayed (A1-A5).

The temperature of the Stepper Modules is also displayed. Due to the varying loads on the individual axes different holding currents are applied, which lead to changing temperatures of the Stepper Modules.

## 9.7. Organize Programs on the Embedded Control

This iRC function is only available in combination with the Embedded Computer. When writing programs on the Windows iRC software, these programs are automatically also downloaded to the Embedded Computer.

With time there might be the necessity to re-organize these programs, e.g. to delete some of them. This is possible with the "Manage Programs" icon in the "Motion" tab.

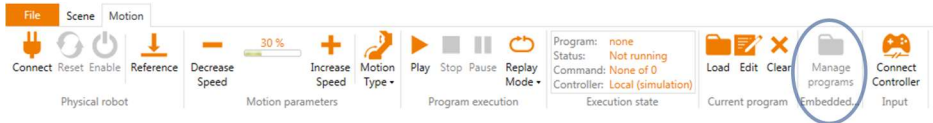


Figure 9.2: igus® Robot Control ribbon with "Manage programs" icon (circled in blue).



# 10. Project Configuration

## 10.1. Program

With “Configure Project / Program” settings, the starting set up can be defined:

- File: which program to load during startup
- File logic: which logic program to load during startup. The logic file runs parallel to the main program allowing e.g. to set lights.
- Override: the override (speed) that is set after startup.
- Replay mode: the replay mode that is set after startup

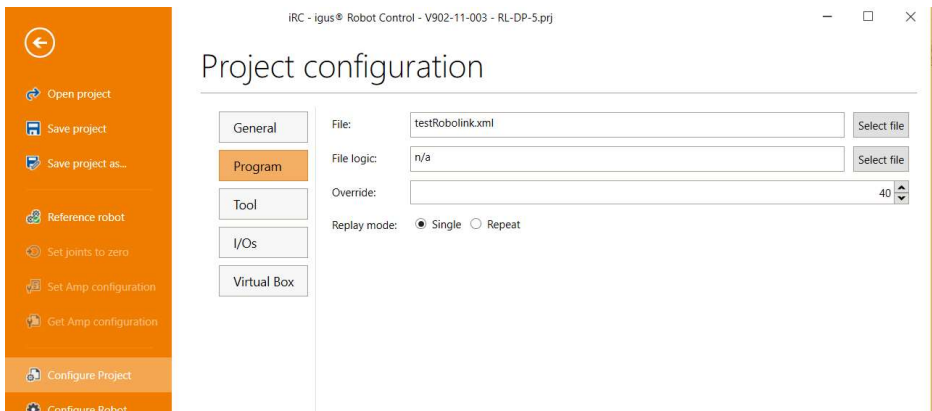


Figure 10.1: Configuration of the starting program in the backstage area.

## 10.2. Tool

With “Configure Project / Tool” settings, the tool attached to the robot is chosen. This setting changes the robot kinematic. Programs written using a different kinematic setting cannot be replayed because this might lead to a collision.

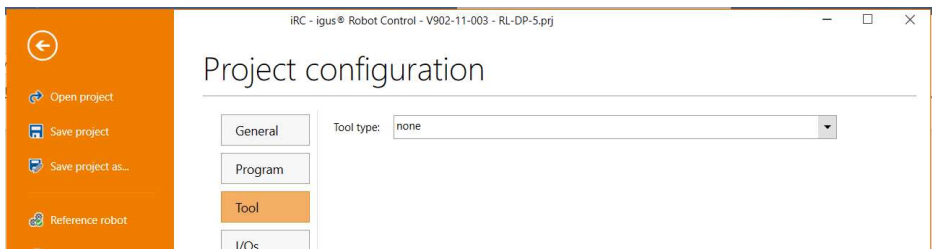


Figure 10.2: Configuration of the tool in the backstage area.

## 10.3. Inputs /Outputs

With “Configure Project / I/Os” settings, the number of available inputs and outputs can be configured and renamed. The corresponding names will show up in the Input/Output tab.

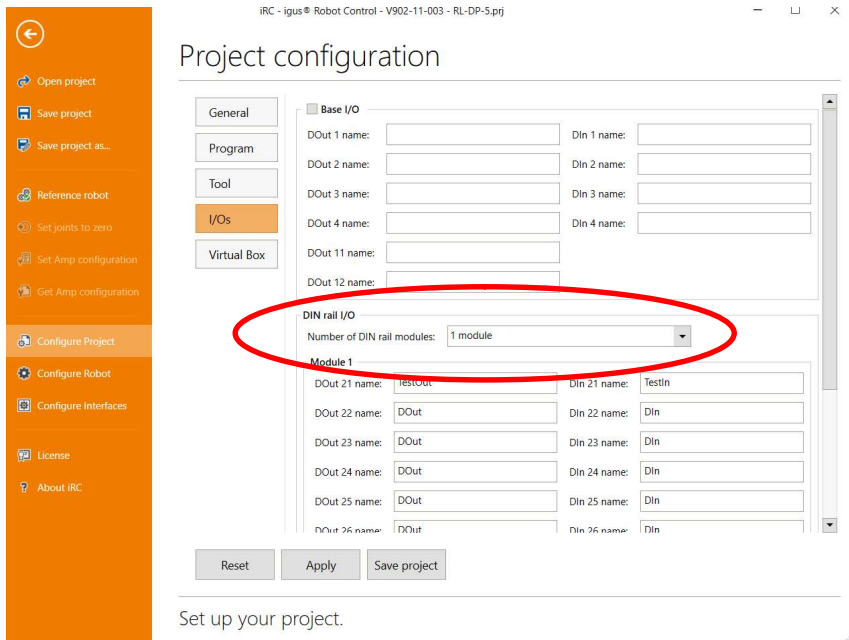


Figure 10.3: Configuration of Inputs /Outputs in the backstage area.

For the robolink robot control, the “Base I/O” area is not used. Relevant is the “DIN rail I/O” section.

The standard Robot Control comes with one DIO Module with 7 inputs and 7 outputs. You can define names for the I/O here.

If another DIO Module is added, then the setting has to be changed to “2 modules”, or “3 modules”.

The settings are automatically synced with the Embedded Computer.

## 10.4. Virtual Box

The “Configure Project / Virtual Box” settings allow to restrict the motion range of the robot arm. These restrictions are monitored both, in linear and in joint motion.



This functionality assists to avoid mechanical damages. It is not a functionality rated for personal safety!

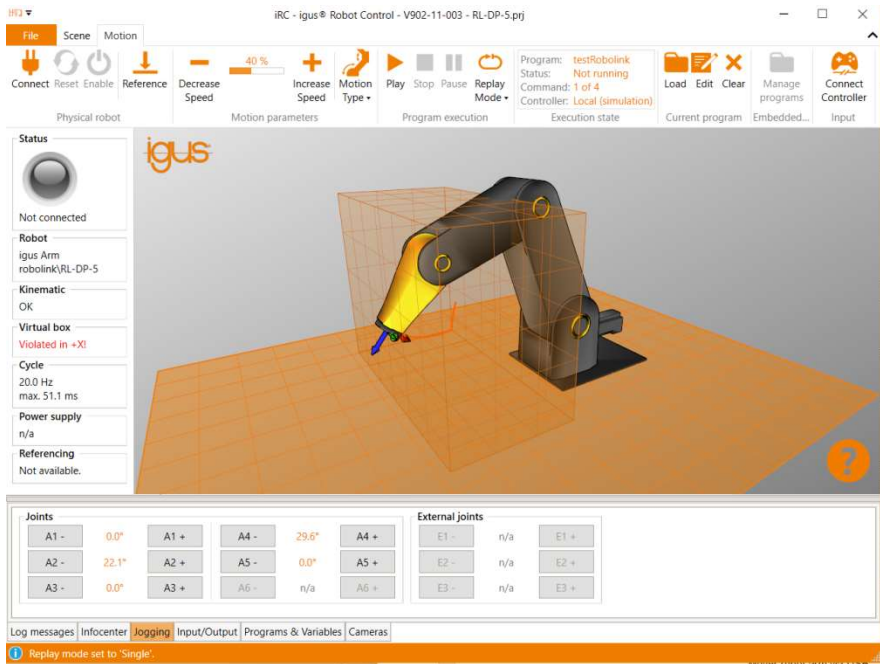


Figure 10.4: Robot Arm violating the virtual box.

In manual jogging mode an error message appears on the left side of the iRC window.

When running a program that violates the virtual box the program is stopped with an error message.

## 11. Advanced Robot Configuration

The robotlink DP robots come with a standard set of parameters. If your application requires changing parameters such as, joint length, etc. please get in contact.

Although many of these parameters can be changed in the robot configuration files, a graphical user interface to help with these changes is not available at present.

## 12. Interfaces Configuration

### 12.1. Cameras

The iRC software has built-in support for receiving target positions from cameras. Currently the IFM O2D is supported.

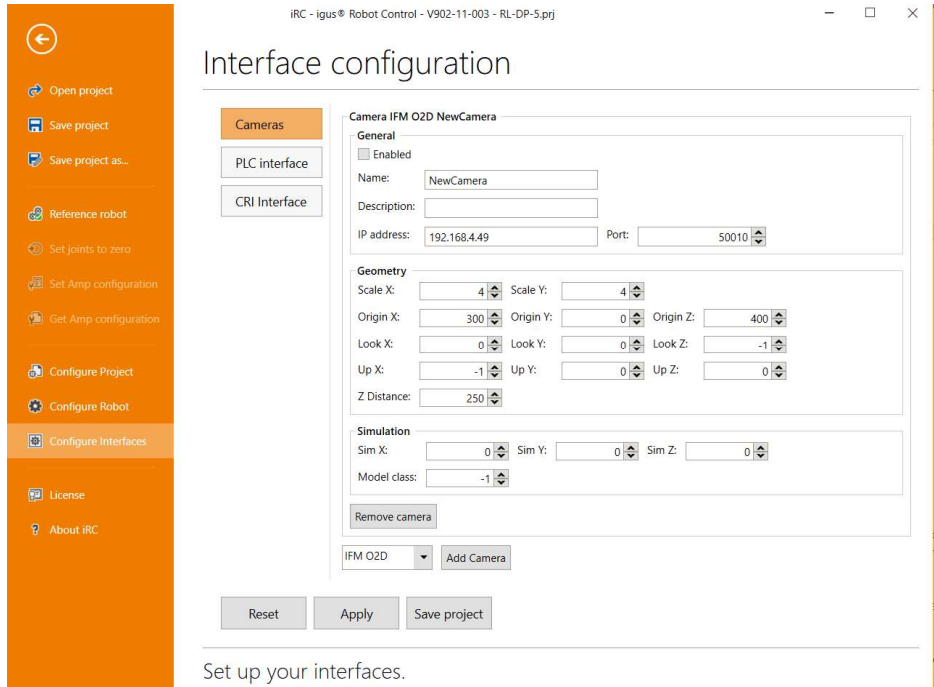


Figure 12.1: Camera configuration dialog.

The work piece position reported by the camera can be accessed with the “Camera” command in the robot program, see Section 8.7.

## 12.2. PLC Interface

The PLC interface (PLC = Programmable Logic Controller) allows the integration of the Robot into a production system controlled by a PLC. In this way, the Robot can work without manual interaction. Also, it is possible to connect third party control panel with push buttons to simplify and restrict operation of the robot in a production environment.

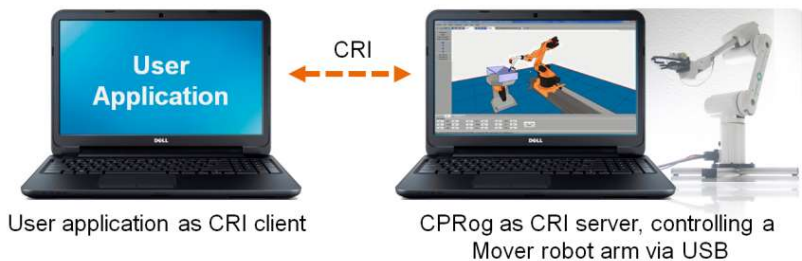
Specifically, this interface assigns certain digital IO channels to special functions, such as:

- inputs: "reset/enable motors", "start/stop program", "reference";
- outputs: "noFault", "enabled" and "program running".

Further details are available on the Wiki (section "PLC interface", <http://wiki.cpr-robots.com/>).

## 12.3. CRI Ethernet Interface

The Commonplace Robotics Interface (CRI) allows the Robot to be controlled via Ethernet. The Robot can be moved, programs can be uploaded and executed. It allows the development of a custom robot control software. The control software (TinyCtrl) on the Embedded Computer and igus® Robot Control on the Windows PC communicate via the CRI interface.



This interface makes it possible to combine the igus® Robot Control functions with application-specific algorithms, such as a teleoperation system or a database. The CRI documentation and a C# sample project for a client can be downloaded on request.

## 13. Troubleshooting

### 13.1. Support Contacts

In case of problems we would be happy to help!

- igus support landing page: <https://www.igus.de/info/robotics-low-cost-robotics>
- Phone: +49 (0) 2203-96498-255
- E-mail: [ww-robot-control@igus.net](mailto:ww-robot-control@igus.net)

Please describe the problem briefly and send the file "logMessages.log" from the folder C:\iRC-igusRobotControl\.

### 13.2. Online Tool - Fault Identification and Recovery

Via the igus support landing page you have access to web pages that allow to:

- ➔ Identify a problem based on user observations
- ➔ Propose solutions to the problem

### 13.3. Configuration of the Stepper Modules

The operating parameters of the Stepper Modules, particularly the motor currents, can be adapted to the application. The Stepper Modules are delivered with a standard parameterization for the specific Robot. Normally no change of the parameters is necessary. If the Robot is to be operated at high loads or speeds, the motor currents can be increased.

This configuration can be done via the software tool "CPR ModuleCtrl". It can be downloaded from the Wiki (<http://wiki.cpr-robots.com/>).

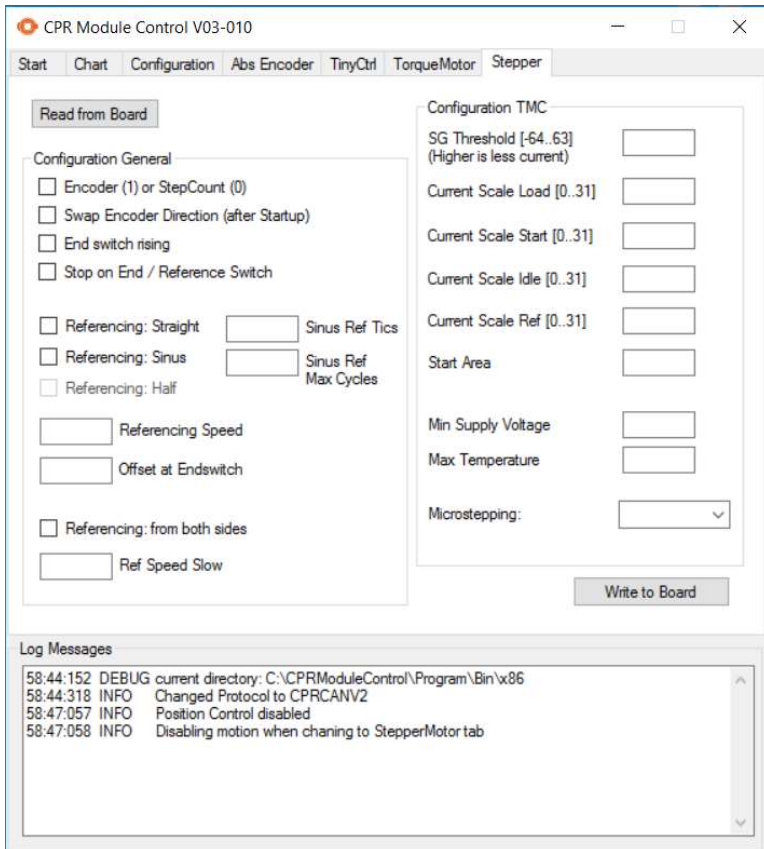


Figure 13.1: CPR ModuleCtrl Software, Stepper Motor Configuration Tab.

Figure 13.1 shows the configuration of the stepper motors. The behavior during referencing is defined on the left side. Motor currents and microstepping are defined on the right side.

- The Current Scale parameters define the max. current during normal operation (Load), during the start-up phase (Start), at standstill (Idle) and during referencing (Ref).
  - The Stepper Module uses Trinamic CoolStep technology to adjust the motor current between these values and a lower limit value, which is a fraction of the Current Scale value.
  - At lower Current Scale values, total power consumption and heat generation are lower. However, a so-called "motor stall" can occur. In this case the current scale values must be increased.
  - At higher values, there is no motor stall, but the system becomes hotter.
- The current scale during the reference movement can be set separately to avoid damage, e.g. by collision.
- The higher the microstepping, the smoother the movement. In operation without encoder a microstepping of max. 1:64 should be used.
- It is possible to reverse the direction of rotation of the encoder. This change is applied after the next cold start of the controller. If the encoder direction does not correspond to the motor direction, the axis does not move correctly but accelerates until a position lag error occurs. In this case, the encoder direction must be changed, and the system restarted. The axis should then function properly.

## 13.4. Calibration of the Robot

The robolink® Robot Arms are referenced using reference switches. For design reasons, however, these are not exactly on the zero positions of the axes. The offset between reference switch and zero position is stored in the EEPROM of the Stepper Modules.

If the Robot was supplied with the Robot Controller, these offsets are already set.

To perform the calibration yourself, you will find the necessary steps on the Wiki (section "Define the Zero Position Offsets", <http://wiki.cpr-robots.com/>).



## 13.5. Error Codes

The Robot Controller provides several status information:

- Status LEDs on the Electronic Modules.
- igus® Robot Control status information, received via CAN status bytes.

### 13.5.1. Status LED of the Electronic Modules



#### Support Module:

- Green LED on: Logic power supply on
- Green LED flashing: CAN communication with the Support Module
- Orange LED on: Error
- Red LED on: Emergency Stop Button pressed



#### Stepper Module:

- Green LED on: Logic power supply on
- Green LED flashing: CAN communication with the Stepper Module
- Orange LED on: Reference switch is active
- Red LED on: Stepper Module is in error state or motor is not enabled



#### Digital In/Out Module:

- Green LED on: Logic power supply on
- Green LED flashing: CAN communication with the DIO module
- Orange LED on: The state of an input or output changes.
- Red LED on: Error

### 13.5.2. CAN-Bus and iRC Status Information

Error	Bit in error byte	Meaning	Measures
Bus dead		The CAN-Bus is not available. Reasons are missing power supply or missing plug connections.	Check the plug connections of the power supply and the CAN line. Restart the control computer.
Temp	Bit 1	The temperature of the Motor Modules is too high.	Check if the ventilation is installed and working. The motor current may have to be reduced.
E-Stop/ Supply	Bit 2	Emergency stop or voltage too low.	Check if the Emergency Stop Button is released.
MNE Motor not enabled	Bit 3	No error. The motors are not released yet.	Press the "Enable motors" button.
COM Comm Watch Dog	Bit 4	The time period without CAN command from the controller was too long.	The position commands via the CAN-Bus must be sent in short intervals. Turn off other programs or update / virus scan functions.
LAG Position Lag	Bit 5	Position lag error. The Robot cannot maintain the target position.	Decrease the speed of movement.
ENC Encoder Error	Bit 6	Error in motor encoder or absolute encoder	Check the encoder cables
OC Over Current	Bit 7	Overcurrent in the motors	Reduce the motor current
DRV	Bit 8	Error in motor driver or motor algorithm	Drive specific

After an "Error Reset" the normal status of the axes is 0x04 (motor not enabled). After releasing the motors the status is 0x00, now the axes are ready for operation.



