FESTO

CIROS®
Studio 1.0

User's Guide

562194 EN
08/08

Order No.: 562194
Edition: 08/2008
Author: U. Karras
Graphics: U. Karras
Layout: 08/2008, U. Karras, J. Saßenscheidt

Contents

1. Introduction

1.1 The 3D Simulation System CIROS®

Welcome to the new release 1.0 of CIROS® Studio. It replaces the previous product COSIMIR® Professional Release 4.2. The new name CIROS® (**C**omputer **I**ntegrated **R**obot **S**imulation) shall point out that the software kernel of the previous simulation system COSIMIR® was completely renewed. The concept and the main features of COSIMIR® Professional are kept but the user interface is new designed. The data types of COSIMIR® and CIROS® are fully compatible. CIROS® Studio is part of the CIROS® Automation Suite.

CIROS® Studio is the universal 3-D-simulation-system, suitable for various application ranges, adaptable in composition, efficient and convenient for everyday work. Area of product appliance is widely spread. It ranges from the usage of 3-D-simulation in training and education to the realisation of digital factories up to real-time simulation of complex, immersive and virtual environments.

CIROS® Studio runs on PC based operating systems Windows 2000™, Windows XP™ und Windows VISTA™.

CIROS® Studio enables you to create a detailed planning of industrial manufacturing work cells, to test the reachability of critical positions, the development of robot and PLC programs and the optimization of the cell layout. All movements and handling processes can be simulated to check collision problems and to optimize cycle times.

The Modeling Extensions for CIROS® support the composition of robot-based work cells. Efficient modeling is provided by using component libraries containing machinery, robots, tools, conveyor belts, part feeders, etc. Free 3D modeling and import from CAD systems are also possible via the standard data format STEP.

1.2 Text Formats

Different text formats are used for certain text contents as well as for keyboard shortcuts.

Text Format	Used for
bold	Commands, menus, and dialog boxes.
italic	Enter text instead of the italic printed text.
CAPITALS	Acronyms, directory and file names. You can use lower case letters, too.
"quotation marks"	Options, chapter titles, and links.

Text formats for plain text

Text Format	Means
KEY1+KEY2	If you have to press two keys at the same time a plus sign (+) is printed between the two keys.
KEY1-KEY2	If you have to press two keys one after another a minus sign (-) is printed between the two keys.

Text formats for keyboard shortcuts

1.3 System Requirements

Minimum configuration:

Processor:	At least Pentium IV 1 GHz
Memory:	512 MB RAM
Hard disk:	5 GB free disk space
Operating System:	Windows 2000™/XP™/VISTA™
Graphic Adapter:	Graphics card with Open-GL support, 128 MB RAM
CD ROM drive:	
Interface:	One free serial interface for connection to the robot controller (drive unit) or a network interface for the TCP/IP connection. One USB port for the USB license key.

Recommended configuration:

Processor:	Intel Core Duo 2,2 GHz
Memory:	1 GB RAM
Harddisk:	10 GB free disk space
Operating System:	Windows 2000™/XP™/VISTA™
Graphic Adapter:	Graphics card Nvidia 7800GT, 512 MB RAM
Monitor:	19“ with 1280x1024 pixel resolution
DVD ROM drive:	
Interface:	One free serial interface for connection to the robot controller (drive unit) or a network interface for the TCP/IP connection. One USB port for the USB license key.
Internet access	
email client	email-account for online upgrade of the license key

1.4 Installation Instructions

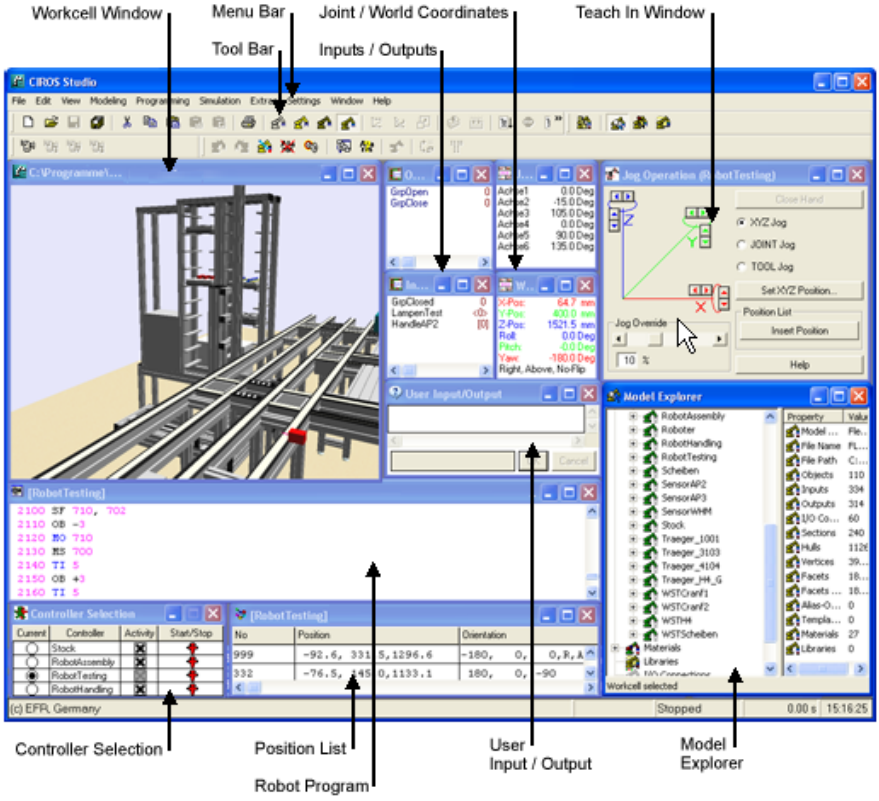
The product package CIROS® Studio consists of a DVD, a manual with comprehensive installation instructions, this user guide as pdf-file on the DVD and a USB license key. You may separately order this user guide as a print out version. The installation does not need a license key. The license key is only required for running the software.

You may find all further details in the installation instruction manual of the CIROS® Automation Suite.

2 Operating

This chapter shows the first steps in using CIROS®.

2.1 The CIROS® User Interface



The user interface was new designed:

- The menu **File** includes all Windows standard functions and following additional ones if you have opened some work cell:
  - Activate Model: This function allows you to release work cells to use them in the educational systems Robotics, Mechatronics and Production.
  - Import and export of files (e.g. the import of CAD files)

- The menu **Edit** includes all standard Windows functions you may expect.

- The menu **View** includes all functions supporting you to use the graphic representation of the 3D-simulation.

- The menu **Modeling** includes all functions you need in order to create or modify models.

- The menu **Programming** includes all functions in order to program robots.

- The menu **Simulation** includes all functions to start and stop the simulation, to configure the setting of the simulation, to activate the collision detection, the sensor- and transport simulation.

- The menu **Extras** provides following special functions:
  - Camera Cruise: This function enables you to create a predefined dynamic change of your viewpoint during simulation or to create videos of your simulation
  - Fault Simulation: This menu provides all functions in order to simulate faults in work cell.
  - Master Frame concept

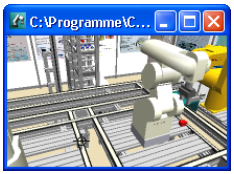
  - Online Management: This menu provides all functions to establish a powerful online communication to a Mitsubishi robot controller.
  - Solution Finder: It provides a powerful script language to create simulation processes.
  - Create Plant: This menu allows you to generate a xml-data model for a superordinate control system.

- The menu **Setting** enables you to configure numerous functions:
  - CAD import
  - Display of the user interface
  - Online Management: Configuration of the online communication interface to the Mitsubishi robot controller.
  - ORL: The ORL name configuration can be used to create I/O names corresponding to Operation Resource Labels.
  - GriP: Configuration of the grip functionality of a robot.
  - Configuration of the IRDATA interpreter
  - Configuration of the Camera Cruise
  - Type of orientation representation
  - Configuration of the programming editor
  - Configuration of the simulation analysis
  - TCP (Tool Centre Point) offset
  - Transport simulation
  - Vertex normals

- The menu Window includes the expected standard functions and the submenu Workspace. This menu supports you in the window configuration of your user interface called workspace. You can save and later restore your own configuration. Additionally, you may use numerous predefined workspace configurations:
  - PLC Operation
  - Manual Operation
  - Fault Operation
  - Teacher mode
  - Robot Programming

2.2 Window Types

The most important window types of the CIROS® user interface are specified in the following list.



Work cell Window

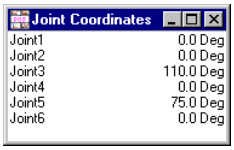
A graphic representation of the currently selected work cell is displayed in the work cell window. Additional views can be opened in the work cell window with the menu function **View →New Window**, allowing you to observe different perspectives simultaneously. The three dimensional representation of the work cell is dependent upon the selected point of view.

- Zoom:
  Mouse wheel or left mouse button and function keys **Ctrl+shift**. The mouse pointer appears in the form of this button, and can then be used to enlarge or reduce the display by moving the mouse.
- Translate:
  Left mouse button and function key **shift**. The mouse pointer appears in the form of this button, and can then be used to move the display by moving the pointer along the coordinate axis.
- Rotation:
  Left or right mouse button and function key **CTRL**. The display can be rotated around the individual coordinate axes.

You can also select various predefined standard views. Use the menu function **View →Standar**d to this end. A dialogue box appears which includes various options:

- Default Setting (O)
- Front view (V)
- Rear view (U)
- Top view (A)
- Left-hand side view (L)
- Right-hand side view (R)

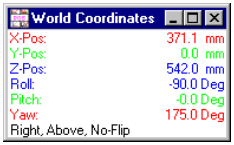
By clicking with the mouse on one of these options, you get immediately the corresponding view if your work cell window is active.



Joint Coordinates

The window joint coordinates shows the positions of the single robot joints. The display unit for rotational joints is degrees, for linear joints it is millimeters. A double click into this window opens the dialog box **Set Joint Coordinates**.

To open the window joint coordinates, press F7 or choose the command **Show Joint Coordinates** from the menu **View → Robot Position**.



World Coordinates

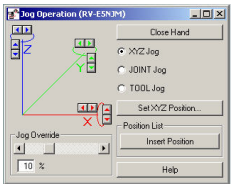
Activate the **Shift+F7** key combination or select the menu function **View → Robot position →Show world coordinates.**

The World coordinates window displays the position and orientation of the TCP (tool centre point) in world coordinates. In addition to position and orientation, the robot's configuration appears in the bottom most line in the window. You may select following different orientation representations by the menu **Settings → Orientation Representation**:

- Roll-Pitch-Yaw angles representation
- Quaternions representation
- Mitsubishi 5-axis coordinates representation

Note

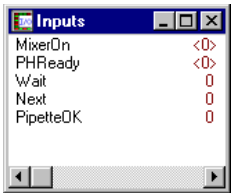
In this case the world coordinate system is always equal to the base coordinate system of the robot.

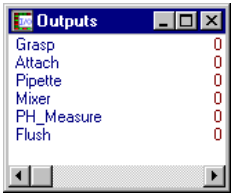


Teach-In

Activate the **F8** key or select the menu function **Programming → Teach-in**. In addition to the designations of the robot's joints, the window that now appears includes two small buttons which can be used to advance the robot's individual joints. The performance of a real robot is simulated when these buttons are activated. The robot is accelerated to the preset speed (override) if one of these buttons is pressed and held. The preset speed is then held constant, and braking to a speed of 0 ensues when the button is released, controlled by means of a acceleration ramp.

By clicking the corresponding option, teach-in can be performed using world coordinates or tool coordinates.



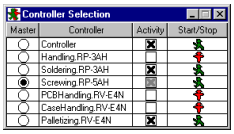


Inputs/Outputs

The window Inputs/Outputs shows the states of the simulated robot-controller's inputs/outputs. The current states of the inputs/outputs are displayed next to their names. 0-signals are displayed in red color, 1-signals are displayed in green color.

The value of an input is displayed in brackets, i. e. [1], if the input is connected to an output. If the signal of the input is forced, the value of the input is displayed in angle brackets, i. e. ‹1›.

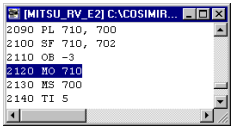
To open the window Inputs/Outputs press **F9/CTRL+F9 or choose the** command Show Inputs/Show Outputs from the menu **View → Inputs/Outputs**.



Controller Selection

The window Controller Selection shows the states of all controllers of the work cell. You are able to choose the master controller and to observe the activity of the different controllers. Display of robot positions, inputs, outputs and teach-in is always done for the emphasized robot (master).

To open the window Controller Selection select the command **Controller Selection** from the **Programming** menu.

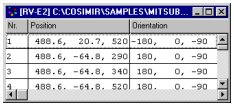


Robot Program

Click the menu function **File →Open** and select the desired file type:

- ***.mb4** (for programming in Melfa Basic IV),
- ***.mrl** (for programming in Movemaster Command)
- ***.IRL** (for programming in IRL = Industrial Robot Language)

Or create a new program with the menu function **File →New** and select the desired data type. The programming languages RAPID for ABB robots, KRL for KUKA robots, V+ for Stäubli or Kawasaki robots are only optional available.



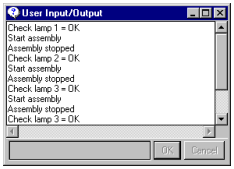
Position List

The screenshot shown on the left contains a position list for a robot. The name of the associated object is specified in the header.

Click the menu function **File -› Open** and select the desired file type, i.e.

- ***.pos** (for Mitsubishi robot)
- ***.psl** (for programming in IRL).

Alternatively, create a new position list with the menu function **File →New** and select the desired data type as above.



User Input/Output

The **User Input/Output** window opens automatically if the robot program contains commands for reading and writing of data via the serial interface to and from the robot control.

Because of the simulation of a robot control, the data is not sent physically via the serial interface, but it is sent to the **User Input/Output** window where the data is displayed.

2.3 Camera Cruise

The Camera cruise can record different views of an active work cell window. During simulation these views are recovered in rotation. A new view between two views is determined by linear interpolation. Thus the viewpoint moves uniformly. At the configuration of the Camera Cruise you can schedule times for holding a certain view and for zooming to another viewpoint. As the Camera Cruise is synchronized to the simulation time the viewpoint movement is always synchronized to the simulation of the work cell. You can also save a Camera Cruise in a video file. At this several compression methods are supported. In the video file (File extension.AVI) all view during the cruise are saved. The video file has the same name and is stored in the same directory as the model file (Extension .MOD) of the actual simulation model.

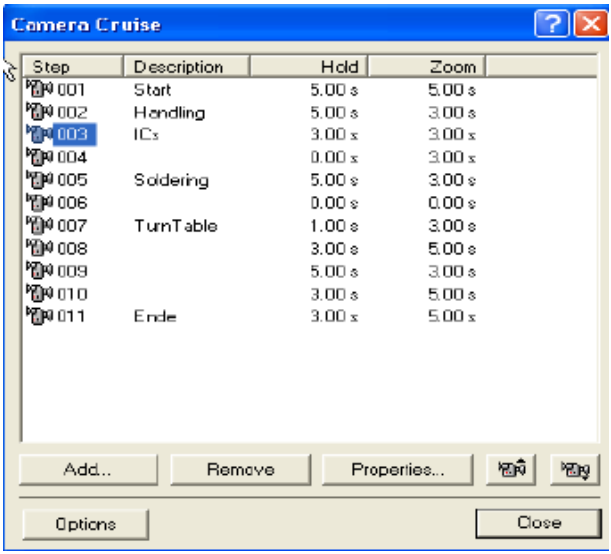
Switching Camera Cruise on
To switch a camera cruise on, use the menu function **Extras → Camera Cruise →Camera Cruise**. If the camera cruise is switched on, the view follows the configured cruise of the camera during simulation.

Recording Camera Cruise
To record the view of a camera cruise, first switch on the camera cruise. Then use the menu function **Extras →Camera Cruise →Camera Cruise Record**. The view will then be recorded to a video-file which will be saved in the model folder under the name ‹model name›.avi.

Playback a Camera Cruise Video
To play back a recorded Camera Cruise in CIROS use the menu function **Extras →Camera Cruise →Camera Cruise Play**. This will open the video file in your operating systems default media player.

Stop recording
The menu function **Extras →Camera Cruise →Camera Cruise Stop** stops the recording of a camera cruise.

Configure Camera Cruise
To setup a Camera Cruise for a simulation model use the menu function **Settings →Camera Cruise**. All setting of the camera cruise are saved to the current work cell's .ini file. To apply changes, write access to this file must be granted. For backup purposes or further use in other work cells, the list of steps can be exported to and imported from a file. To import or export the list use the menu functions **File →Import and File →Export** and select file type CIROS Camera Cruise (file extension .ccc).



Options

View list
This list contains all views of the Camera Cruise. To select a certain view click the number in column step. You can open a context-sensitive menu by clicking the right mouse button.
Double-clicking a step changes the view of the active work cell window to the view of the camera cruise step.

Add

To add the current view to the list click **Add.**

Remove

To remove the selected view from the list click **Remove.**

Properties	To edit the properties of the selected view with dialog box **Camera Cruise - Step X** click **Properties**.
	To move up the selected view click this button.
	To move down the selected view click this button.

3. Modeling

There are several tools (i. e. model libraries and the Model Explorer for CIROS®) providing a comfortable modeling of robot-based work cells. By means of a simple example work cell, a short introduction in work cell modeling is given in this chapter.

3.1 Model Hierarchy

The CIROS® model hierarchy contains the following element types:

Objects
The highest units in the element structure are the objects.
Example: A robot is an object.

Sections
Sections are assigned to objects. One degree-of-freedom can be associated to each section that is moveable relatively to the previous section.
Example: Each joint of a robot is a section.

Hulls
Hulls are assigned to sections and are responsible for the graphical representation.
Example: A face, a box or a polyhedron are hulls.

Gripper Points
An object needs a gripper point to grasp other objects. Gripper points are assigned to sections.
Example: At the flange of a robot a gripper point is modeled.

Grip Points
To be grasped by another object an object needs a grip point.
Grip points are assigned to sections.

Example: A grip point is associated to a work piece that has to be grasped.

3.2 Model Libraries

CIROS® Studio provides a wide range of model libraries for. Use these model libraries to add new objects or model parts to a work cell . Following model libraries are available:

- Robots
  - ABB robots
  - Adept robots
  - Fanuc robots
  - KUKA robots
  - Mitsubishi robots
  - Reis robots
  - Stäubli robots
  - Miscellaneous
- PLC
  - Logic controller
  - Siemens S5/S7
  - Logic controller
  - Miscellaneous controllers
- Miscellaneous
  - Grippers
  - Primitives
  - Materials
  - Mechanisms
  - LEDs
  - Sensors
  - Textures
- Modeling Essentials
  - Predefined working objects , i.e. gravitational surface, Linear 3 axis kinematics, replicator, trashcan, transceiver, transponder etc...
- Extended Mechanisms
  - Gear Box, servo motor, cardan shaft, crank slider, flap door, hydraulic linear axis, etc..
- Festo FMS:
  - This library contains numerous prepared CNC- and robot assembly work cells, conveyor systems and automatic warehouses to build up a automated plant system.

The work cells provided prepared robot and PLC programs to realize manufacturing and assembly processes which have to be supervised by some control system.

- MPS stations: This library contains all actual MPS stations with prepared S7 programs
- MPS 500: This library enables you to build up MPS systems with integrated conveyor.

The dialog box **Model Libraries** can be opened as follows:
Modeling → Model Libraries

Toolbar 

The menu command **File -› New** provides you various options to open a new work cell:

- MPS system
- Production Line
- Project Wizard
- Work cell

If you select MPS system then enter a name for the new work cell and the editor to create a MPS system will be opened:

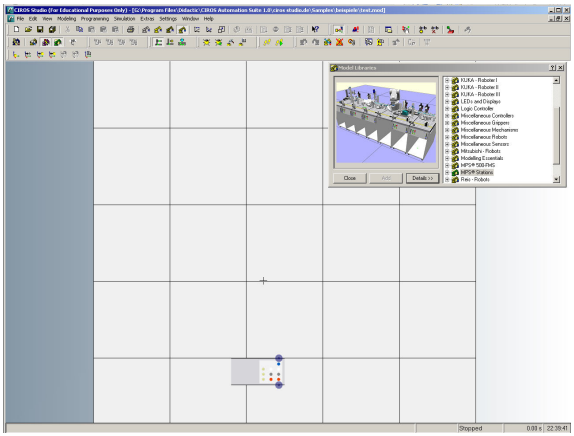


Figure: MPS system

If you select **Production Line** then enter a name for the new work cell and the editor to create a production line will be opened:

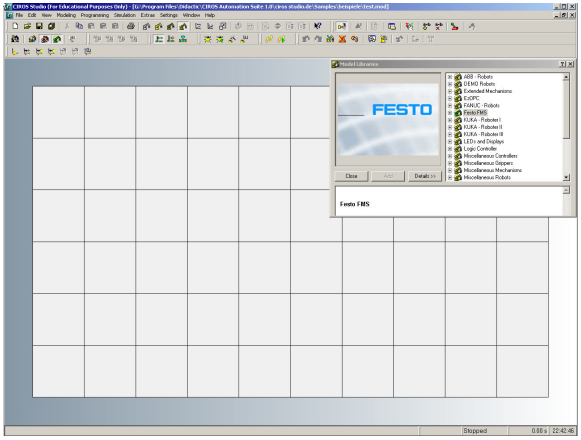


Figure: Production line

If you select Project Wizard then following dialog box will be opened:

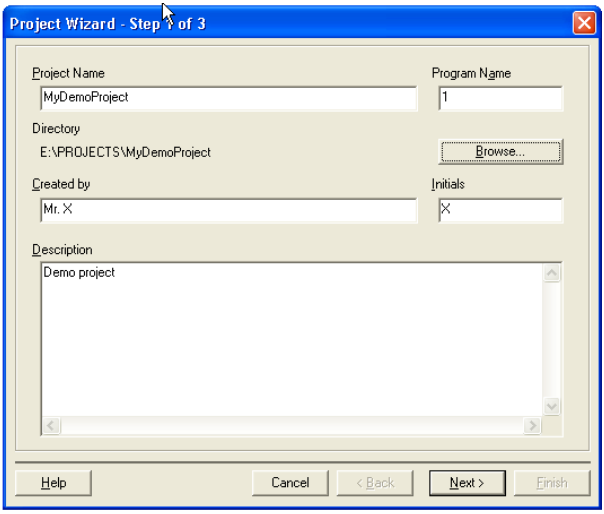


Figure: Project Wizard

Project Name

The project name is used identify the project. It will be the filename after saving, and you must use this name to open the project later. The default suggestion for the project name is "UNTITLED". During installation a directory "Project Name" below the CIROS/CIROS Programming directory is created automatically. According to the selected project name a subdirectory with just that name is created and all files belonging to the project are stored there.
Note: The character ‹'› (apostrophe) within the project name is automatically replaced by the character ‹_› (underscore).

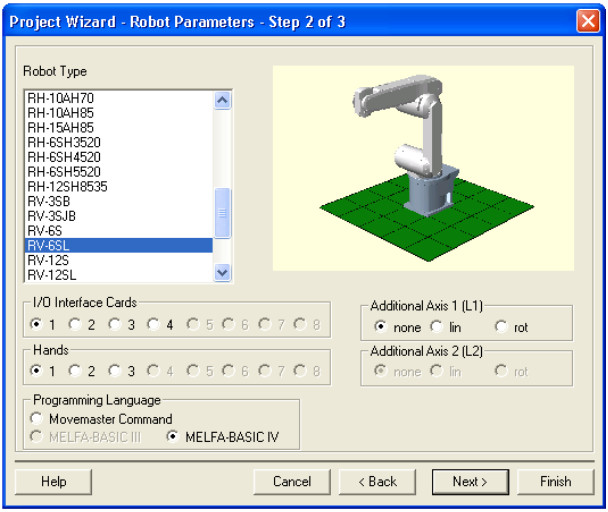
Program Name

Enter the desired program name into this edit field. The program name is used as a suggestion when downloading a program into the drive unit. After downloading the program you may use this name to start the program or for a subprogram call.

Directory / Browse...

	The Directory field shows the currently selected location to save the actual project. It shows the drive and path, not including the filename. You may alter the location by using the browse button. The default suggestion for the directory is the actual directory.
Created by	Enter a name to identify the author of the project, robot program, etc..
Initials	Enter the author,s initials, e.g. for referencing in the project description or the project history.
Description	This field may be used for a description of the project.
General	All data entered within this dialog will be saved, if you change to another step of the project wizard or leave the wizard using Finish.
	All data entered within this dialog and during the actual use of the project wizard will be lost if leaving the wizard using **Cancel.**

Step 2/3



Robot Type

Use this list box to select your robot type. The selected robot is shown in the upper right area of the dialog.

I/O-interface cards

Selects the number of interface cards of your drive unit. The maximum number of cards to select depends on the actual robot type:

- Movemaster RE-xxx: 3 cards
- Movemaster RV-M1/2: 2 cards

Hands

Select the number of hands of the robot here.

Programming Language

The programming language selected here is used for the creation of a program file and selects the der syntax checker. This item is only available if the selected robot type supports more than one programming language. If there is only one possible language for the robot or the controller this language is selected automatically.

Additional Axis 1 (L1)

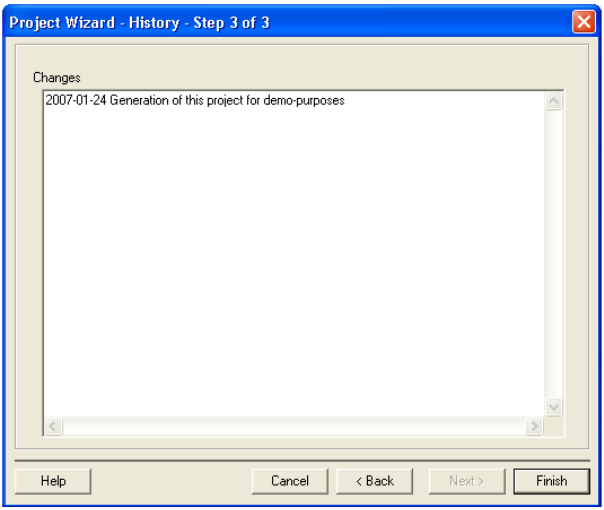
Use this item to determine the first additional axis. You may specify the type of the axis. The selection **lin** describes a travel axis, **rot** a rotating table.

Additional Axis 2 (L2)

Use this item to determine the second additional axis. You may specify the type of the axis. The selection **lin** describes a travel axis, **rot** a rotating table. This item is only available if the first additional axis is selected as **lin** or **rot** and if the robot type supports 2 additional axis.

Step 3/3

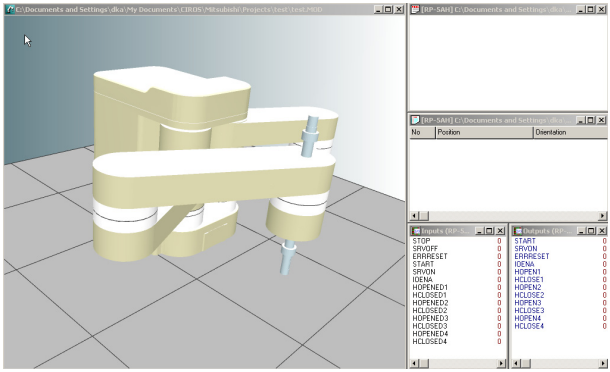
Use this item to determine the second additional axis. You may specify the type of the axis. The selection lin describes a travel axis, rot a rotating table. This item is only available if the first additional axis is selected as lin or rot and if the robot type supports 2 additional axis.



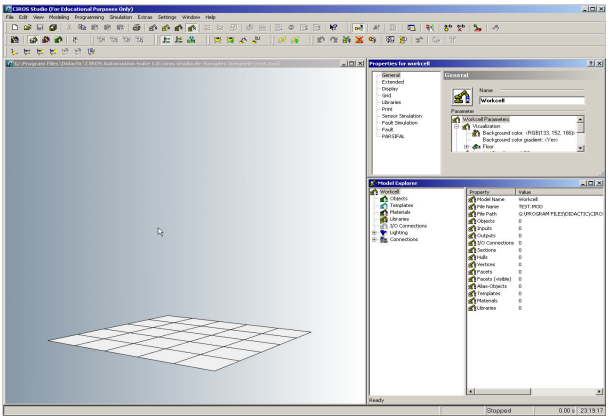
Changes

This field may be used for a description of the changes of the project. You may enter any sentences, words, characters and symbols.

Then click on the button Finish and the work cell with the selected robot, the programming window and position list will be shown. Using the menu function **Window →Workspace →Robot programming →Program, Position List and I/O's** you get an ideal display of your application windows.

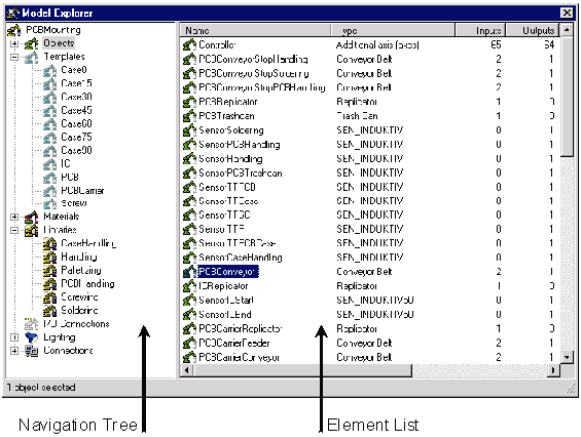


If you select Work cell then following workspace with integrated Model Explorer will be opened:



3.3 Model Explorer

Use the Model Explorer to access all the elements of a work cell. Besides objects and associated elements you are able to maintain materials, libraries, lighting sources, and I/O connections, too.



The Model Explorer's window is divided into two parts.
In the left area a **navigation tree** contains folders with the different elements of a work cell.

If you select a folder in the navigation tree the **element list** in the right area of the Model Explorer is filled with the folder's elements.
To access an element select the element in the navigation tree or in the element list by clicking on the element using the mouse.

By clicking the right mouse button you can open a **context** menu with most important commands depending on the current element selection.

The Model Explorer can be opened as follows:

Menu Modeling → Model Explorer

Keyboard CTRL+T

Toolbar 

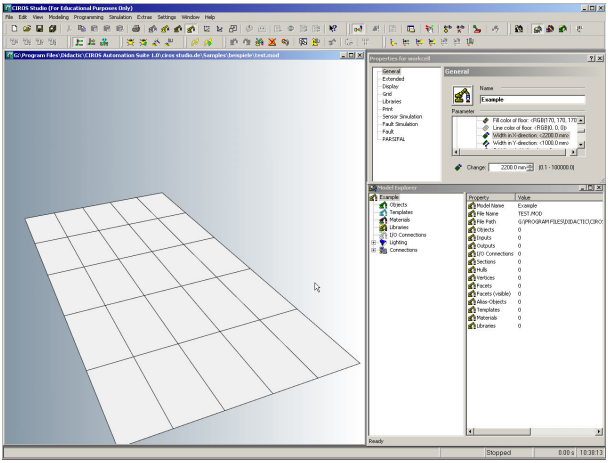
3.4 Example: Work cell Modeling

In this chapter modeling of a simple work cell is described step by step. Programming and simulation of this work cell are described in the next chapters.

Choose command **New Work cell** from the **File** menu to create a new work cell. Specify the filename (e. g. “Example. mod”) for the new work cell.
After creating the new work cell you are able to specify a different work cell name as well as properties for the work cell (e. g. background color, floor color, and floor size).

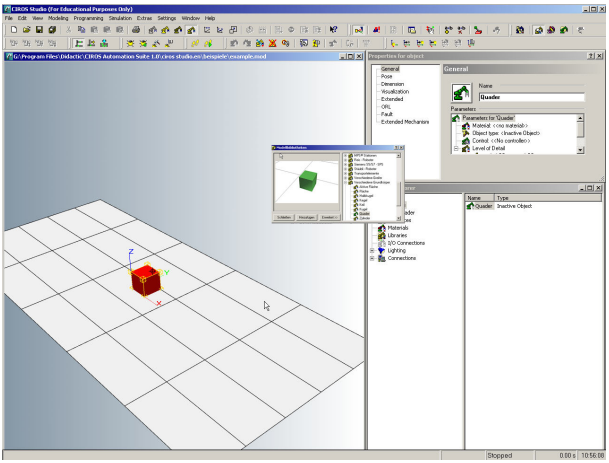
Use the dialog **Properties for work cell** while the work cell name is

selected in the **Model Explorer** to change work cell properties. Change the x-value of the floor seize to x = 2200 mm.



Open dialog box **Model Libraries** by choosing command **Model Libraries** from the **Modeling** menu or clicking the button in the toolbar.

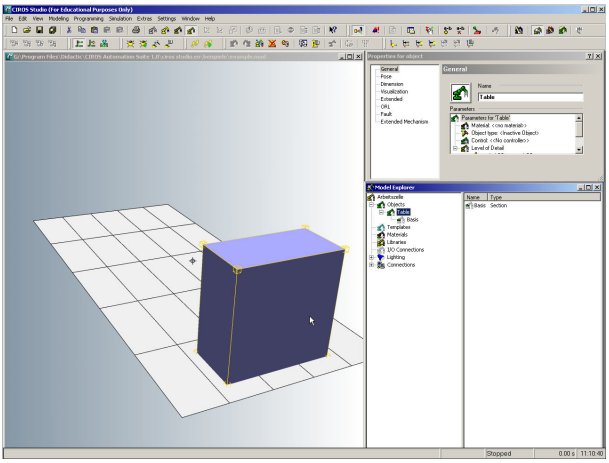
Add a box from the model library **Miscellaneous Primitives** to the work cell.



Edit the element properties in dialog box **Properties for object** as follows:

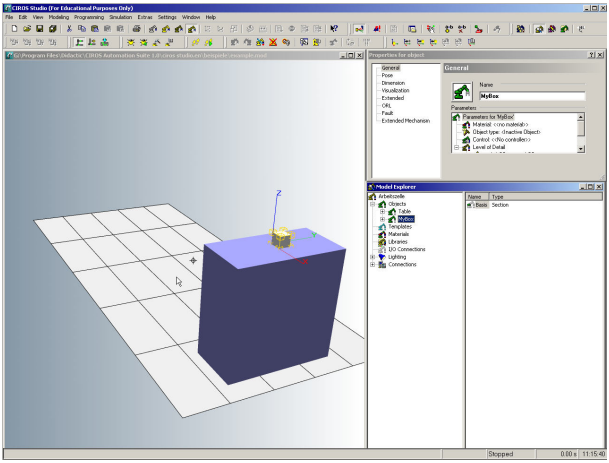
Pose (x,y,z)	750 mm, -250 mm, 0 mm
Dimension (x,y,z)	250 mm, 500 mm, 500 mm
Visualization	Light blue

In the Model Explorer rename the object “Box” to “Table”.



Add a second box to your work cell. Rename it by MyBox and edit the element properties in dialog box **Properties for object** as follows:

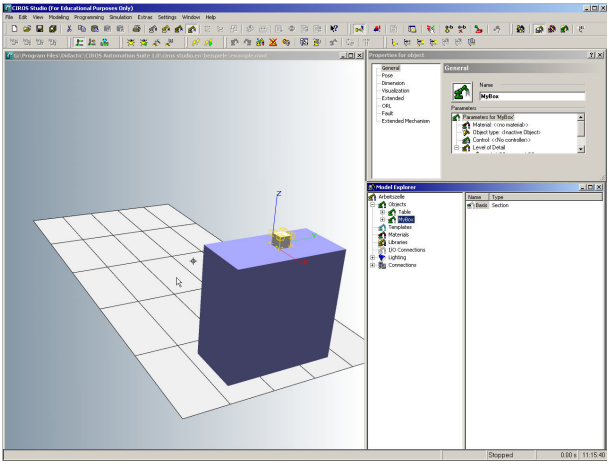
Pose (x,y,z)	850 mm, 0 mm, 500 mm
Dimension (x,y,z)	50 mm, 50 mm, 50 mm
Visualization	Light grey



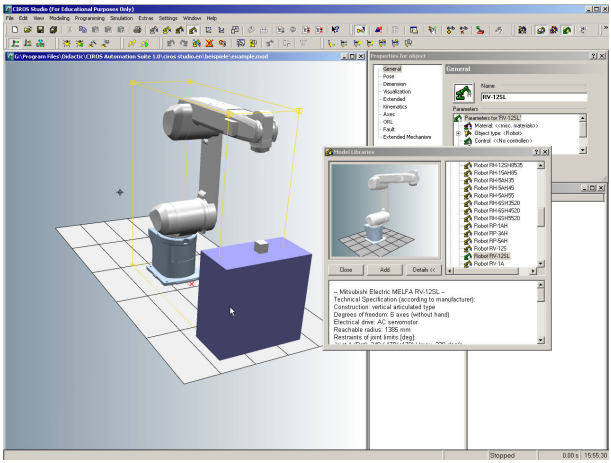
To let a work piece be grasped by a robot you have to assign a grip point to the work piece. Select the group base of the object MyBox and open the context menu. Choose the command **New -› Grip point**.

Rename the grip point to **Workpiece** and move the grip point in the centre of the object by choosing following coordinates relative to the **section** coordinate system:

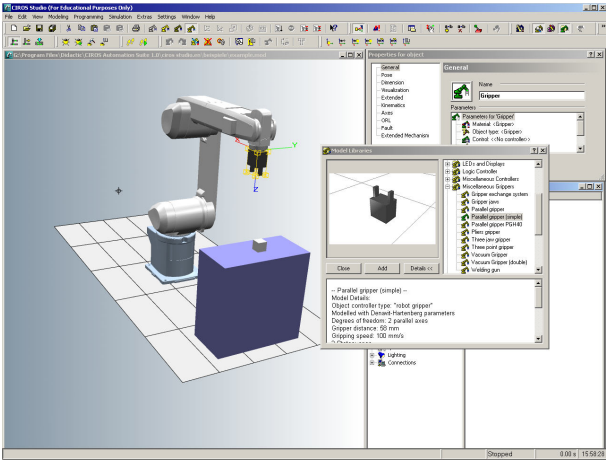
Pose (x,y,z)	25 mm, 25 mm, 25 mm
(R,P,Y)	180 °, 0 °, 180 °



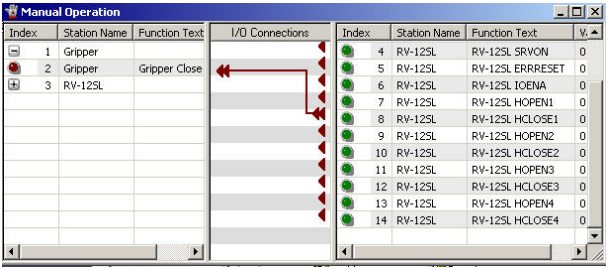
Select the Mitsubishi Robot RV-12SL from the library **Mitsubishi Robots** and click **Add**.



Select the Parallel Gripper (simple) of library “Miscellaneous Grippers” in dialog box **Model Libraries** and click **Add**. The gripper is attached to the robot’s flange automatically.



To simulate the electrical connection between the robot controller and the gripper, select the menu command Modeling -› Manual Operation to open the window Manual Operation:



The window shows the inputs on the left side and the outputs on the right side. In between the I/O connections will be shown. Now click on the arrow next to the output **HCLOSE1** of the robot in the area **I/O Connections** and draw a connection line to the input **Gripper Close** of the gripper .

Close the window Manual Operation and save the work cell by clicking the button in the toolbar.

You can open the modeled work cell from the following installation directory of CIROS® Studio: ‹InstallationDirectory›\ GettingStarted\Mitsubishi\Modeling\ Example.mod.
As an exercise you can do similar examples using ABB or KUKA robots. For example, select the ABB robot IRB 2400 10/16 or the KR6 of the robot libraries. To do the I/O wiring, select the output [inactive 000] of the robot controller. Press F2 to rename the output and call it **Grasp**. Now the output is activated and can be used as above.

You find numerous samples in the folder **Samples** of the installation directory of CIROS® Studio. All these samples are write-protected so that you have these examples always ready for presentation in the

original status. The online help system of CIROS® Studio provides a section **Sample Models**. You can open these sample models in your own folder to work with them.
Further the online help system provides three tutorials for learning the modeling of work cells:

- Tutorial
- Advaneced Tutorial
- Tutorial: PLC controlled package system

The first tutorial helps you to do the first steps in modeling a new work cell. The advanced tutorial shows how to use advanced modeling mechanisms. The third tutorial describes all steps in order to realize a transport system for packages controlled by a PLC.

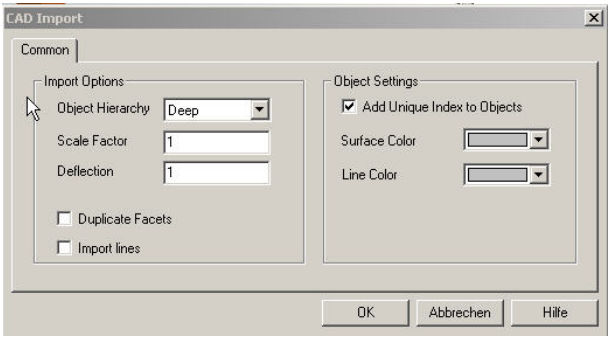
3.5 Modeling via Import

You can add a geometric model to a work cell. Besides the mod – file format, following other formats are available for the import:

- CIROS I/O connections (*.cct)
- CIROS camera cruise (*.ccc)
- CAD format STEP (*stp, *.step)

Use the menu function **File → Import**. To create a work cell you can use at first the included libraries. Further you can also use components of prepared work cells. For that purpose you have to save the components as *.mod files. Then you can import them.

If you want to import a STEP-file you will be asked for the name of the file. Then following dialog box will be opened:



The first options ask you to decide how the CAD data shall be mapped on the object hierarchy of your model:

- A deep hierarchy means that your model consists of many objects containing only one section after the import.
- A flat hierarchy means that your model consists of only few objects with several sections.

Scale Factor

The scale factor enables you to adapt the seize of the objects in your work cell in compare to the dimension of your CAD model. The default value is 1.

Deflection

The value of the deflection determines when a new facet will be generated for the curvature of your model. The default value is 1 (1 mm). If the value is too high then the curvature of your model might look rather angled.

Duplicate Facets

Facets are only visible in direction of their normal vector. If for example a facet is modeled such that its normal vector points to the interior of the object then the facet is only inside visible. In this case you may use the dialog box **Facets Duplicate.**
Then the facet will be duplicated but with the opposite orientation.

Hence it will be also visible from outside. However, this procedure increases the number of the facets of your model.

Import Lines

If your CAD file contains polylines then you can import them.

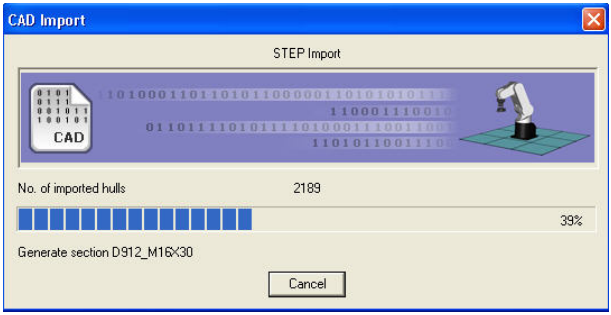
Add Unique Index to Objects

CAD files allow that several objects appear with the same name. This option makes sure that the objects get an index after import.

Surface Colour/Line Colour

If the CAD file does not provide a colour for a surface or a line, then this option makes sure that after import the corresponding surface or line has the selected colour.

Conform your settings by OK then the import will be started:



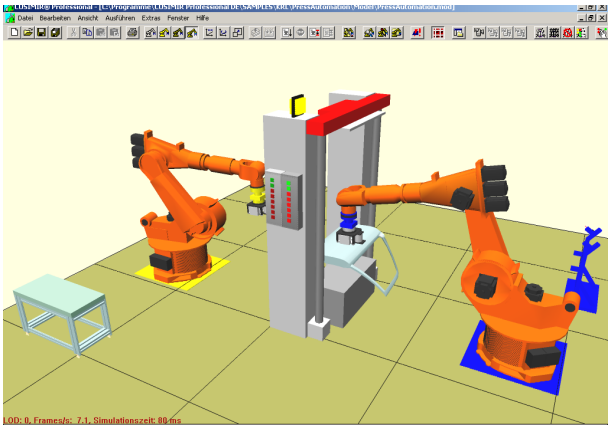
After import of non mod-files you must in general adapt the geometric model corresponding the model hierarchy of CIROS®.

3.6 Integration of a PLC into a work cell

During modeling of complex work cells the system loads standard components e.g. robots, grippers, assembly – lines and PLCs from libraries. Only cell or user specific sections must be modeled with the help of elementary hulls. After inserting all cell objects you have to link the appropriate inputs and outputs of the controllable objects with the PLC and create the control programs. Robot programs in different

languages can be created in CIROS® and linked with position lists. The programming of the PLC in Step5/Step7 has to be done by means of a PLC environment, how they are offered by PLC manufactures or by software companies.
The PLC programs which are also part of the project can be loaded in CIROS® and interpreted during the work cell simulation.

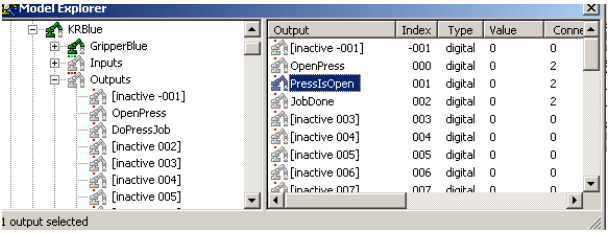
Following multi robot work cell is available as demo in your CIROS® Studio version (<InstallationDirectory>\Samples\KRL \PressAutomation\Model\PressAutomation.mod).



At first, you have to model the press, the table and the door using geometric primitives of the library. In a next step, you have to integrate the appropriate mechanisms, see chapter 6. Load the two KUKA robots KR 125 from the library and place them using the Model – Explorer. The robot on the yellow surface is denoted by KRYellow und the robot on the blue surface is called KRBlue.
Insert a Siemens PLC S7 with 8 digital inputs (DI) and 8 digital outputs (DO). Again use the Model – Explorer to place the PLC at the correct position.

As an example we want to outline the realization of a communication between the PLC and the robot KRBlue. The PLC shall send the message at which time the press is open such that the robot can place the door into the press.

1. Open the list of outputs of the object PLC in the Model – Explorer. Rename the second output with **PressIsOpen**. You select the appropriate output and you open by right mouse click the context menu. Choose the submenu **Rename** and rename it by above name.
2. As before, rename an input of the robot KRBlue by **PressIsOpened**.
3. Connect the output **PressIsOpen** of the PLC with the input **PressIsOpened** of the robot. This can be easily as follows: Select the output range of the PLC. Then the list of all outputs will be shown in the right side of the Model – Explorer window.

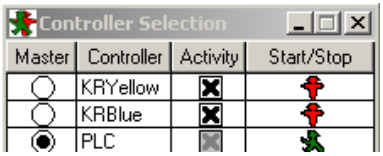


4. Select the output **PressIsOpen** with the mouse and drag it to the input **PressIsOpened** of the robot. The connection is ready.

Load of a S7 program

The S7 simulator interprets executable S7 programs. Each work cell can contain several PLCs. In each PLC a S7 program is running. While loading the PLC module the appropriate S7 program is loaded like-wise. Of course, you can exchange the standard S7 PLC-program by another S7 program. Before the first usage of a work cell the PLC does not contain any program. So the appropriate S7 program must at first be assigned to the PLC. Proceed as follows:

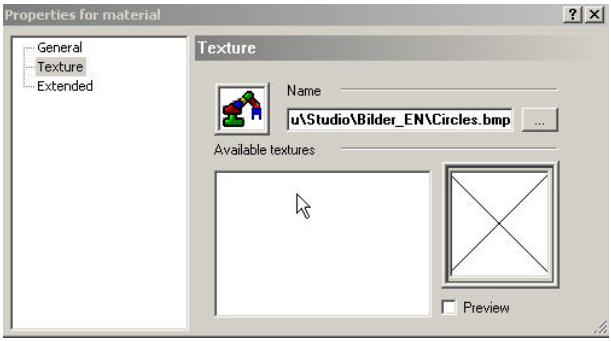
- Make sure that the simulation is stopped, see chapter 5.
- Choose the command **Programming → Controller selection**. Following window is opened:



- Select in the column **Master** the desired PLC by clicking the appropriate field, here **PLC** .
- Choose the command **File → Open**. This opens the **File open** dialog. Select in the combination field **type of file** the entry **S7 project**. This will display all files in the current directory according this data format.
- Select the desired S7 project by a double click on the file name or by selecting the file and pressing the **Open** button afterwards.

3.7 Texture Mapping

Use **texture mapping** to let pictures display on object's surfaces in a work cell. The texture mapping can be easily realized as a material property. Provide your object's surface with a material. Select the material and open the dialog box **Properties for Material**. Now you can change the texture of this material by choosing some *.bmp file.



3.8 Masterframe Concept

The masterframe is a user defined frame (x,y,z,R,P,Y), which can be used for the positioning of elements or the tcp of the current robot during graphical modeling. The masterframe is controlled by the following toolbar buttons:

Showing / hiding the masterframe:
Use this button to show/hide a coordinates system which represents the masterframe

Positioning the masterframe:
The masterframe can be positioned either by directly entering it's coordinates using the **MasterFrame Location** dialog or by moving it to the position of an element automatically:

Sets the masterframe to the position of the 3D-Cursor.

Sets the masterframe to the position the currently selected element(s). If more than one element is selected, the geometrical center of the selection will be calculated and used as current position of the selection.

Sets the masterframe to the position of the current robot's TCP.

Opens a dialogue, in which the position of the masterframe can be entered directly.

Using the masterframe to position elements:

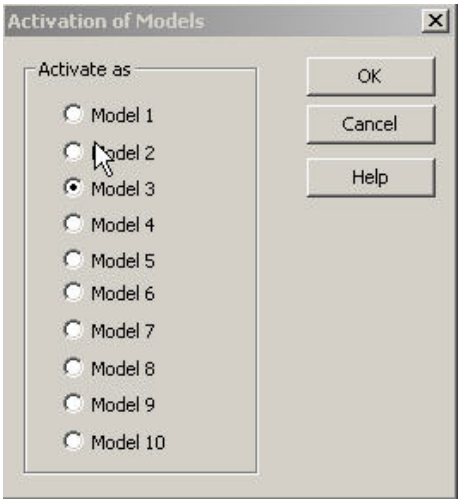
Moves the current selection to the position of the masterframe. If more than one Element is selected, each of the selected elements will be moved to the position of the masterframe.

Sets the TCP of the current robot to the position of the masterframe.

3.9 Activate Model

The menu command **File -› Activate Model** is only available in CIROS® Studio. This command generates a check sum for the actual model which will be saved in the license key.
Using these check sums it is possible to make up to ten own work cells usable in the educational CIROS® packages CIROS® Mechatronics/ CIROS® Advanced Mechatronics/ CIROS® Robotics/ CIROS® Production. Note that the activated models must be compatible to the corresponding versions of CIROS® (e.g. CIROS® Mechatronics can only run one PLC controller).

Open the model you wish to activate in CIROS® Studio. Select the command **File -› Activate Model**. Following dialog box will be opened:



The license key has 10 memory cells (model 1-10). Select one of them and click on OK to save the check sum.
The corresponding model can now be used in all educational packages using the same license key.
Note that it can happen you overwrite a already existing check sum. Then the corresponding work cell is no longer usable in one of the educational packages of CIROS®. Thus it is advisable to maintain a list of all activated models.

4 Programming

Programming of the Mitsubishi robots, KUKA robots, ABB robots, Adept and Stäubli robots can be done in CIROS® Studio in their native programming languages:

- Movemaster Command or Melfa Basic IV
- KRL
- RAPID
- Vplus

Note that the programming languages KRL, RAPID and Vplus are not part of the standard delivery of CIROS® Studio.

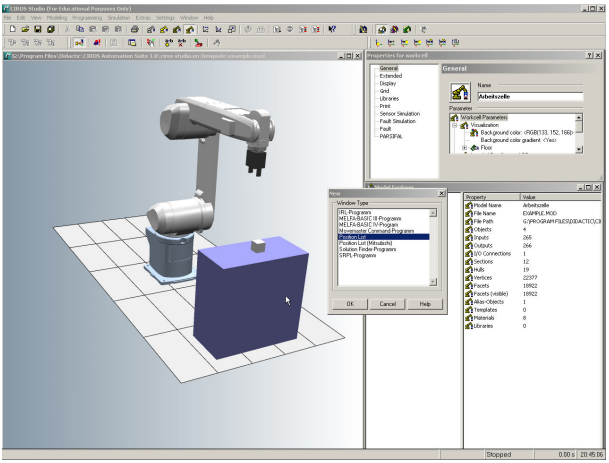
You can also program the robots using the universal robot programming language IRL (**I**ndustrial **R**obot **L**anguage). You will find details about these different programming languages in the chapter Programming of the help system.

4.1 Example: Work cell Programming

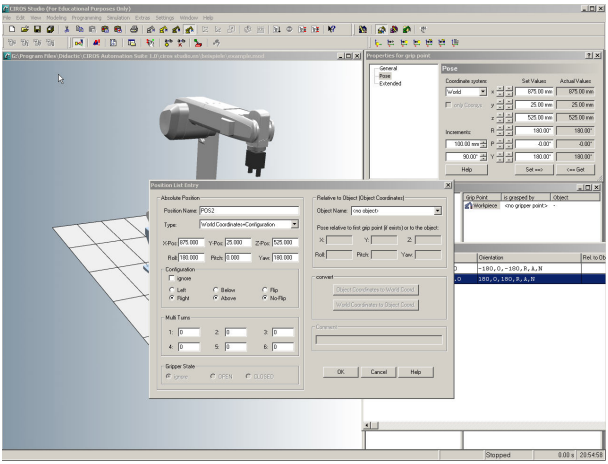
This example shows the programming of the Mitsubishi robot RV-12SLin IRL. The work cell modeled in the previous chapter is used.

Choose command **Open** from the **File** menu to open the example work cell of the previous chapter.
Create a new position list by choosing command **New** from the **File** menu and selecting item **Position List**.

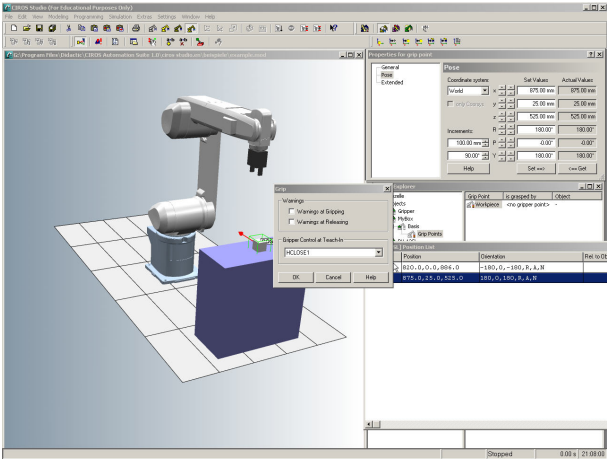
You are able to open dialog box new by clicking button in the toolbar or by using the shortcut **CTRL+N**.



Accept the initial position of the robot as first position (POS1) of the position list. Use the shortcut CTRL+F2 to accept current robot positions in position lists.
Select the last free entry in the position list and use the shortcut CTRL+F2 a second time. This position (POS2) has to be changed.
Select the grip point of the object **MyBox** in the Model Explorer and open its property **Pose**. Choose the world coordinate system as reference coordinate system.
Select the position by clicking on the entry with the left mouse button and open the dialog box **Position List Entry** via the context menu.
Transfer the position data of the grip point to position POS2. Click on OK and close the dialog box:



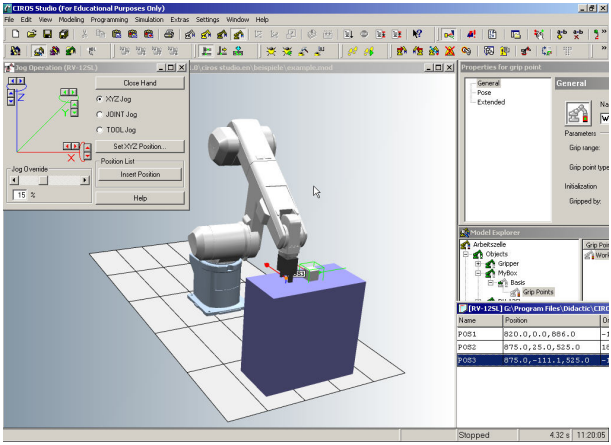
Use the menu Settings -› Grip to configure the grip control via the Teach-In (F8) window. Select the output HCLOSE1 and press OK.



Press Close Hand respectively Open Hand in the Teach-In window in

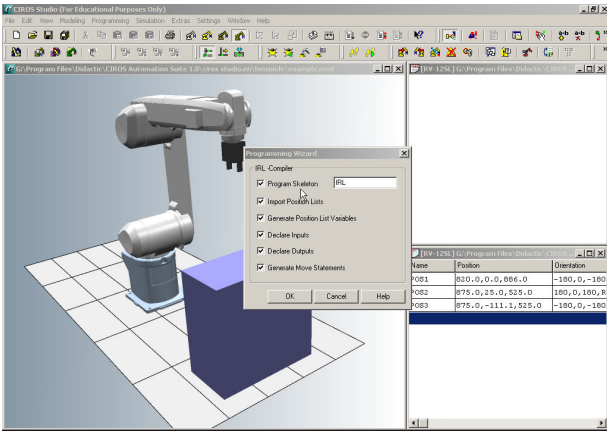
order to open or close the gripper of the robot. Confirm the warnings that no object near the gripper or no object can be gripped. You may also switch off these warnings.

Move the robot to position POS2 by double click on the position entry in the position list. Select the modus **XYZ-Jog** (world coordinates) in the Teach-In window and press the button to move the robot in negative y-direction. Insert the new position POS3 in the position list by the short cut Ctrl+F2.

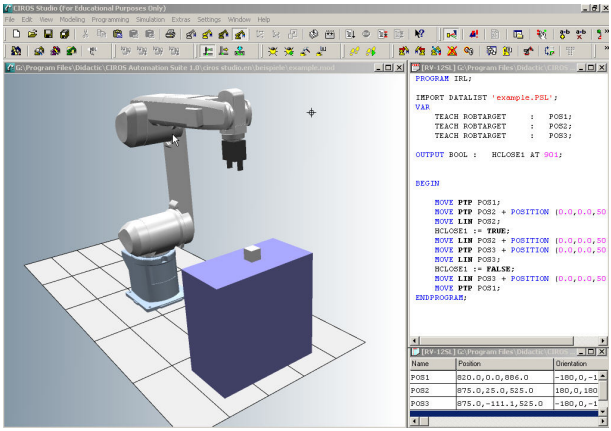


Save the position list as Example.psl. Select the command **Reset Workcell** from the menu **Simulation**.

Use the menu command **File -› New -› File** to open a new IRL program. Save the program as **Example.IRL**. Close the Model-Explorer and the property dialog box.
Use the program wizard to generate a simple body of the program. Activate the programming window and select the menu **Programming -› Programming Wizard**. Select all options and press OK.



Change the program to a simple pick and place task where the safety positions should be 50 mm above picking and placing position. Delete all input and output variables except the output variable HCLOSE 1.



Sample Program:

```
PROGRAM IRL;
IMPORT DATALIST 'Example.PSL';
VAR
   TEACH ROBTARGET : POS1;
   TEACH ROBTARGET : POS2;
   TEACH ROBTARGET : POS3;
   OUTPUT BOOL : Grasp AT 0;

BEGIN
   MOVE PTP POS1;
   MOVE PTP POS2 + POSITION (0.0,0.0,50.0);
   MOVE LIN POS2;
   GRASP := TRUE;
   MOVE LIN POS2 + POSITION (0.0,0.0,50.0);
   MOVE PTP POS3 + POSITION (0.0,0.0,50.0);
   MOVE LIN POS3;
   GRASP := FALSE;
   MOVE LIN POS3 + POSITION (0.0,0.0,50.0);
   MOVE PTP POS1;
ENDPROGRAM;
```

The editor provides a highlighting of the syntax for your support. You may adjust the highlighting using the menu **Settings -› Program Editor**. Save the program and select the command **Compile+Link** from the menu **Programming**. The syntax will be checked, the IRDATA code will be generated and downloaded in the virtual robot controller.
In window **Messages** all used system modules, program modules, and position lists as well as errors and warnings are displayed. If no errors and warnings are displayed,, then close the message window and your program can be started.

IRDATA is a standardized low level code needed to download robot programs created in different programming languages.
CIROS® processes IRDATA code created by compilers for different high level programming languages.

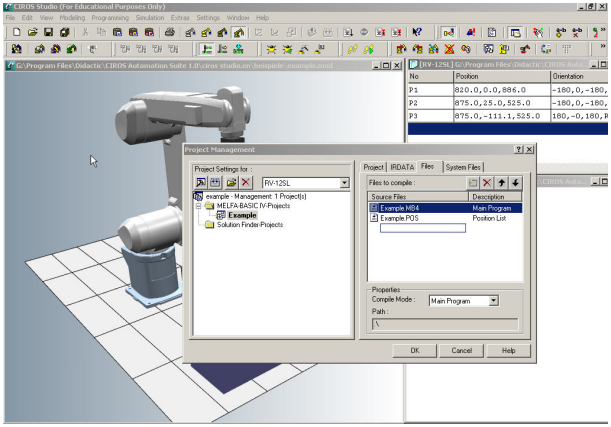
IRDATA files created by CIROS® may be only used inside CIROS®. Do not use IRDATA files created by CIROS® with robot controllers.

Melfa Basic IV Project

As above you may also create a Melfa Basic IV program. For that purpose you have to open a MRL-position list and a Melfa Basic IV program. The new files will be named as **Example.pos** and **Example.mb4**. For the purpose of simulation the program as well as the position list must be integrated into a MELFA Basic IV project.

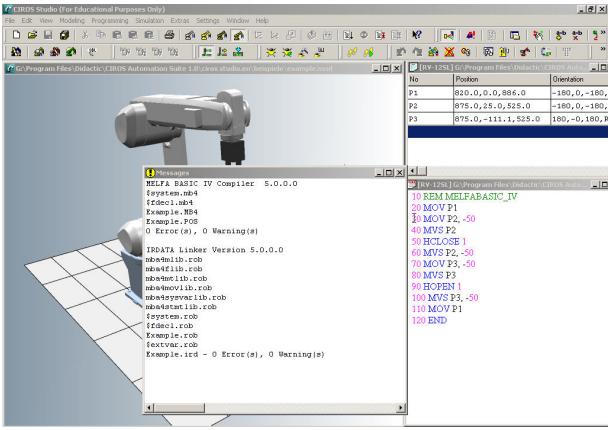
Create a new MELFA Basic IV project in dialog box **Project Management** (command Project Management from **Execute** menu). Choose command **Add Project** or click button to add a new project named “Mitsubishi.prj”.

In page **Files** of the dialog box add the program and the position list by clicking button and selecting the files. Declare the program “Mitsubishi.mb4” as **Main Program**.



The program is integrated into the current project.

Close dialog box Project Management and activate the program window. Choose command **Compile + Link** from the **Programming** menu (CTRL+F9 or button) to check the syntax of the program and to translate it into IRDATA code.



You have developed an executable robot program for the work cell simulation.

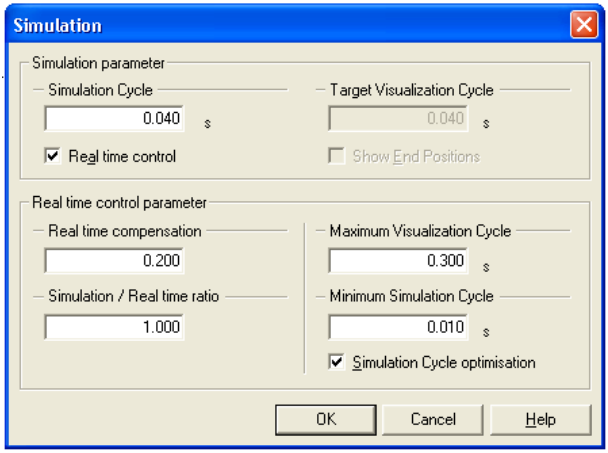
The next chapter contains an example of simulating the modeled and programmed work cell.

5. Simulation

This chapter contains the simulation of programs developed offline in CIROS®.

5.1 Settings

Choose the command **Simulation Settings** from the **Simulation** menu to configure the simulation.



Simulation cycle

The simulation cycle specifies the intervals, in which the simulation controller interpolates the states of robots. Additionally it specifies the cycle time for available PLCs and the recalculation time for all extension modules (e.g. sensor simulation, transport simulation,...)
A high value results in a fast simulation, but with only very few interpolation steps. Too high values may result in important steps not being calculated. A low value calculates more interpolation steps, but therefore decreases the simulation speed due to the need of more machine time.

Example

Suppose a robot needs exactly one second for a certain motion command. Depending on the simulation cycle, the number of interpolations would be as following:

Simulation cycle:	0.040	0.100	0.200	0.500	1.000
Number of interpolations:	25	10	5	2	1

Target Visualization Cycle

The visualization cycle specifies the intervals, in which the model in the work cell window shall be refreshed. The value can be interpreted as "refresh work cell window each visualization cycle seconds". A very low value means that the window will be refreshed very often, which may due to a higher need of machine power, result in slowing down the simulation.
Since the simulation is recalculated each simulation cycle, the value of the visualization cycle must always be equal to or greater than the simulation cycle.

Show End Positions

This option ensures that the state at the end position of a robot motion visualized, even if it lies between two visualization cycles.

Real Time Control

Select this option to enable the real time visualization. The visualization cycle will then be adjusted dynamically to provide real time views of a running simulation.

Real Time Compensation

This parameter determines a constant (amplification P) to control the Visualization Cycle. Values range from 0.1 to 0.6. A small value means a slower compensation, higher values may force fluctuations or even oscillations.

Maximum Visualization Cycle

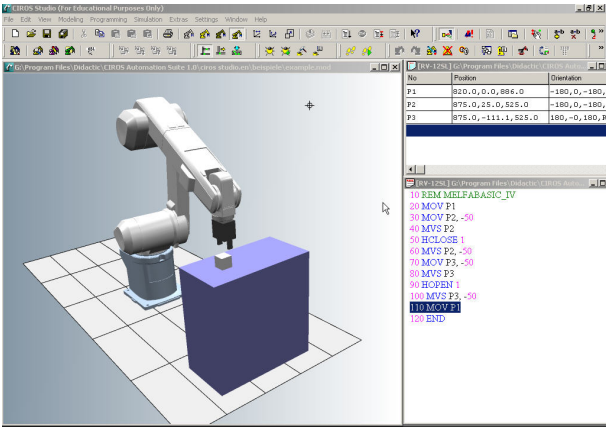
Selecting **Realtime** makes the system set the Visualization Cycle automatically to reach a synchronization between the simulation time and the real clock.

In case that a model is very complex, it can happen that real time control is not possible due to too high machine power requirements. This would result in permanently increasing the visualization cycle. To avoid this effect, the maximum visualization cycle can be limited to a certain value. The range of the visualization cycle is always between the simulation cycle and the simulation cycle.

Simulation / Real Time Ratio	The parameter entered determines the relation between simulation time and real time. The default value 1.0 controls the simulation time according to realtime, a value greater 1.0 makes the simulation time run faster then realtime. Selecting the value 5.0 makes the simulation time run five times faster than realtime. A simulation period of 50 seconds will take 10 seconds in realtime.
Simulation Cycle Optimization	Select this option to use spare computing power of your machine in order to improve the simulation cycle. The simulation cycle will then be optimized dynamically, depending on the unused computing power. The lower limit for the simulation cycle can be defined in the field **Minimum simulation cycle**.
Minimal simulation cycle	This field defines a lower limit for the option **Simulation Cycle Optimization.** The selection Model Update switches the update of model calculations like belts or process simulation from the very small cycle Controller Cycle to Simulation Cycle. The setting Controller Cycle may lead to decreasing performance for some models, on the other hand Simulation Cycle may evaluate to some inaccuracies.

5.2 Example: Work cell Simulation

Open the example work cell of the previous chapters.

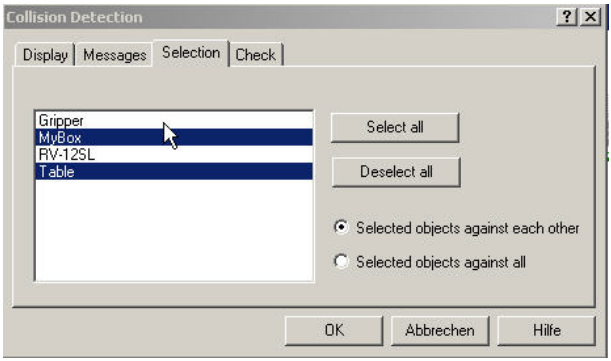


To start simulation, choose command **Start** from the **Simulation** menu. The program is simulated step by step. The simulation time is displayed in the status bar. Because of the source code sequence trace the currently executed command is highlighted in the program window. Before you start simulation a second time choose command **Reset Work cell** from the **Simulation** menu. This command resets all objects as well as the robot

5.3 Collision Detection

By the collision detection you can detect collisions in your applications. You are able to select objects for checking.

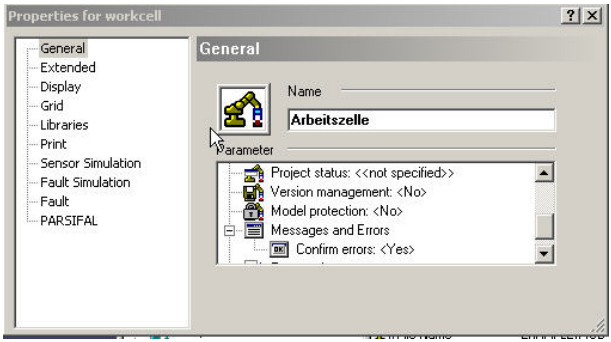
Use the menu function **Settings → Collision detection** to this end. Click the **Selection** index card. The index card displays a list of all of the objects included in the work cell. Select the Selected objects against each other option, in order to determine whether or not the selected objects collide with each other.



You may adjust the distance between two objects, which leads to a collision message. Click the **Check** indexing card. Here you are also able to detect collisions while gripping. This is useful, if you want to detect the inexact positioning of a gripper.
Objects, which are controlled by the collision detection, can be coloured corresponding their status. The color of the edges or areas can be chosen by preferences. Secondly, collisions can be put out as confirmed or unconfirmed messages. You are able to adjust this by clicking on **Messsages** index card.

Warning

If you select the option of confirmed or unconfirmed messages, then you have to open the property window General of the work cell. Make sure that the parameter **Confirm Errors** of the class **Messages and Errors** is set to **Yes**.



Now you can activate the collision detection by the menu function **Simulation → Collision detection**.

Example

Replace line 220 in above sample program Mitsubishi. mb4 as follows:
80 MVS P3, +10
A collision between work piece **MyBox** and the table will be generated.

5.4 Sensor Simulation

The sensor simulation extends the capability of CIROS® to simulate complete work cells. Many sensors used in production automation can be parameterized and simulated realistically. Moreover the visualization of measuring ranges helps to prevent errors in the planning stage. This cannot be done in reality.

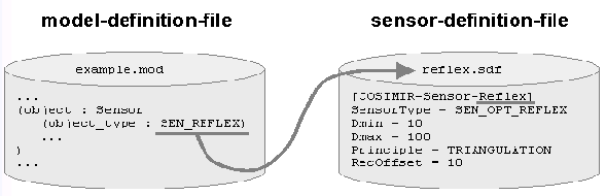
Switching on and off

The sensor simulation is switched on by default. In some cases it might be useful to switch off the sensor simulation to save computing time. To switch the sensor simulation on resp. off select the command **Simulation → Sensor Simulation**.

Settings of the sensor simulation

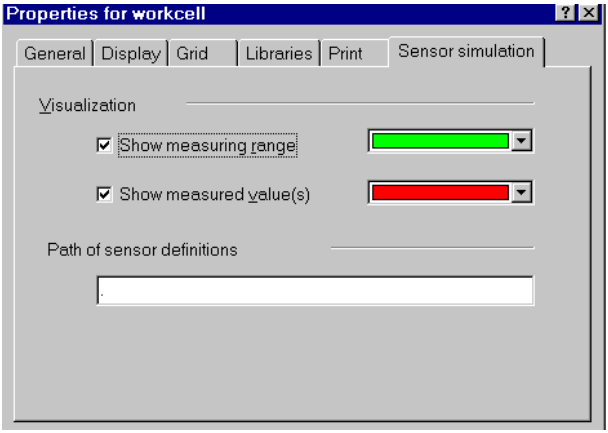
By adjusting the settings of the sensor simulation you can choose how the measuring range and the measured value of a sensor shall be displayed within a work cell window. Moreover you can specify a path in which CIROS® searches for **sensor-definition-files.**

In a sensor-definition-file (*.sdf) all parameters which characterize a sensor type are stored. All sensors are marked by an object type (in MOD-file **object_type**) which begins with **SEN_**. The following letters specify the filename of the SDF which contains the description of the sensor (see figure). Single parameters are identified by keywords.



The extension of the SDF is always ".sdf". The file has the same format as a Windows-INI-file. It starts with a section name which is given in squared brackets. This name always starts with **CIROS-Sensor**. After that the name of the sensor definition file without extension follows. Upper and lower case are not distinguished.

To modify the settings of the sensor simulation you must at first open the dialog box "Sensor Simulation Settings":
Open the model-explorer. Use the right mousebutton to click the work cell. A context menu pops open. Open the submenu **Properties** and select the tab **Sensor Simulation** .



The following settings can be modified:

Display mode

Show measuring range

If this option is activated the measuring range of each sensor is displayed by a line respectively a fan of lines. You can choose the color of these lines by clicking the button "Color...". Measuring rays normally start at the origin of the measuring coordinate system.

Show measured value

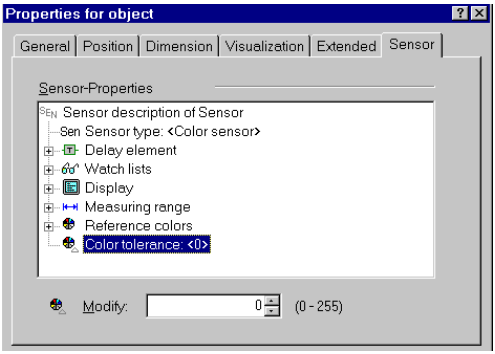
If this option is activated the measured value of each measuring ray of each sensor is displayed by a line (normally starting at the origin of the measuring coordinate system) to the detected object. You can choose the line color by clicking the button "Color...". The measured value of one single ray is not displayed if no object is detected by this ray.

Sensor-definition-file

You can determine the path (the directory) in which CIROS® searches for sensor-definition-files (Files with the extension "sdf"). If the path is not specified CIROS® searches the directory from which the current mod-file was loaded.

Setting sensor parameters

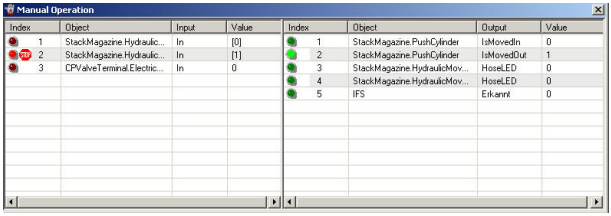
Parameters of every sensor which is used within a work cell can be modified using the property page **Sensor-Properties**. This page can be opened as follows:
Open the model explorer. Select the sensor whose parameters you want to modify and press the right mouse button to open the context menu. Select the menu item **Properties**. The page **Sensor-Properties** is displayed now. If for example a color sensor is selected the page looks as shown in the figure:



In the upper part of the page single parameters can be selected which then may be modified in the lower part of the page.
Some parameters may exist multiply like for example "reference color". If you want to add another element of such a parameter please click the right mouse button on any already existing entry of this parameter to open the context menu. Choose **New** to generate a new entry with default values.
If you want to delete an element of such a parameter please open the context menu on the entry you want to delete and choose menu item **Delete**. Notice that only the last element of a coherent chain of elements can be deleted.

5.5 Manual Operation

To switch the manual operation on resp. off please select the command **Modeling → Manual Operation**.
The dialog **Manual Operation** shows the inputs on the left side and the outputs on the right side. You can change the input values.



By a double click a red LED you can change the corresponding input value. Is the input connected, the connection is released temporarily and the input is forced to 0 or 1. The connection is established again:

- Executing the menu command **Simulation → Reset Work cell**.
- Saving or closing the work cell.
- Executing the menu command **Restore I/O connections** in the context menu of the dialog **manual operation**.

If you select a connected input or output, the corresponding output or input is highlighted. By a double click in the column **Value** of the inputs, you can enter values directly, especially for analog inputs.

Context Menu Inputs

Set/Reset
Executing this menu command the input is forced temporarily to 0 or 1.

Disconnect Controllers
The connection of all connected inputs for all controllers are released and forced to their current value.

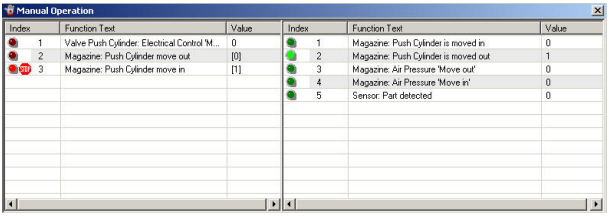
Restore I/O Connections
All I/O connections that were changed via the dialog **manual operation** are established again. All inputs, that were forced via the dialog **manual operation** are set to the condition before calling the dialog **manual operation**.

Show Value Changes
Activation of this menu entry results in a green highlighting of all inputs that have changed at last.

Stop at Value Change
Activation of this menu entry results in assigning or deleting of stops to inputs. An icon shows the user if a stop is assigned. If a stop is assigned and the input value changes, the simulation run is stopped.

Delete all Stops
Deletes all assigned stops.

Function Text
Via the context menu you can change this dialog to the function text view. The dialogs then looks like:



Context Menu Outputs

There are similar functions available for the output signals.

5.6 Hose- and Cable Track Simulation

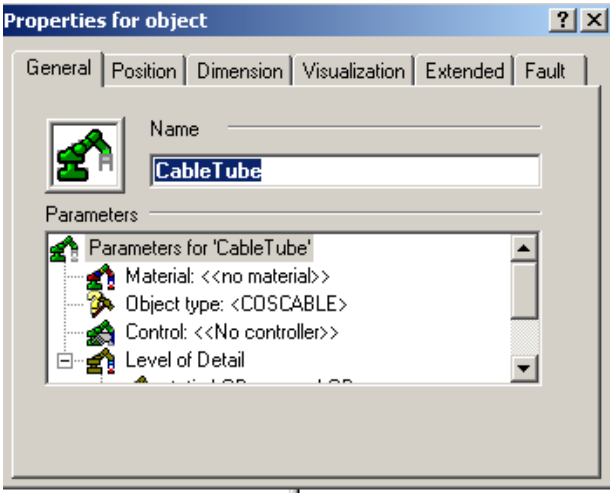
The COSCABLE module offers the simulation of flexible hoses and cable tracks for CIROS®. Shape and position of it is calculated in simulation frequency. Because of the simple way of modeling it easy to equip work cells with powerlines, robots and other mechanisms with properly sized hoses for electric and pneumatic power in the virtual world.

Even lengths of hoses can easily be adapted and interactively set properly on static and moving parts.
This new object type has full support for collision detection.

Modeling

The object type "COSCABLE" supports three different types of hoses and tracks:

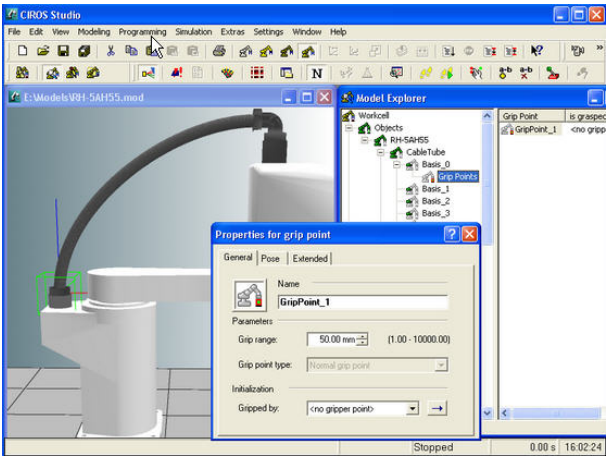
- Type 1
  simulation of hoses consisting of lined up segments which poses are calculated by the object controller.
- Type 2
  simulation of hoses consisting of a single polyhedron by directly modifying the positions of its polyhedron points (vertices) by the COSCABLE controller.
- Type 3
  simulation of cable track systems, consisting of lined up segments running on lines and half-circles, which poses are calculated by the object controller.
  Specification is done by the object property dialog, setting the object type to COSCABLE.



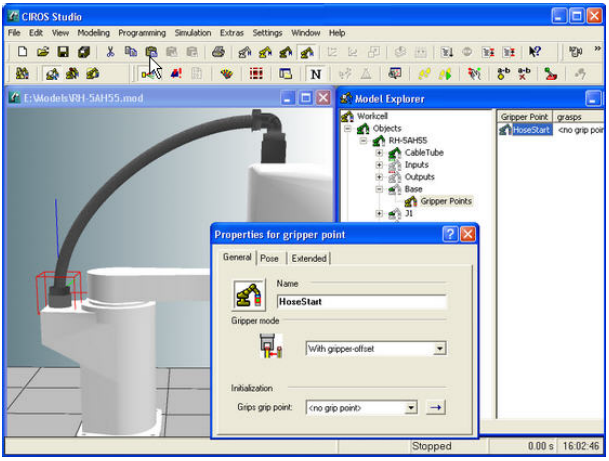
Modeling of Start and End

The first and last segment of this object type needs to contain grip points for defining the start and end of the object. These grip points own a property with the name of a corresponding gripper-point and by this way they are attached to each other.

The starting grip point has got the name **StartGripper** and the end **EndGripper**. The following screenshots are viewing these correlations:



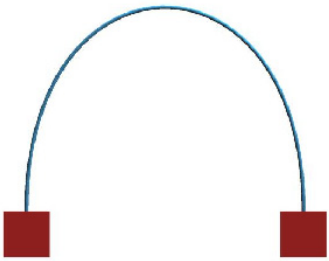
Properties and value of the grip point of the first segment of the tube.



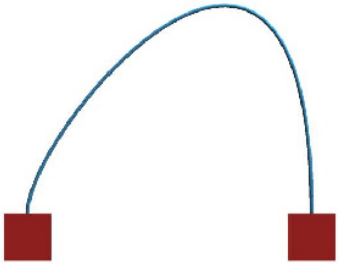
If the Object is a flexible conduit, the z-axis of the grip points are directing tangentially into the hoses fixed position.
If the object is a cable track the z-axis of the start grip point is the moving direction of the track.

Type 1

The hose is modeled by single overlapping segments. The positions of these segments are calculated by the object controller.
The first segment is the topmost listed element in the explorer view respectively the last. Enumerations have no relevance.
The length of the object can be adapted by the property **Length** and its unit is mm.
Assigning the property **RatioSE**, the balance of the hose can be adapted, as shown in picture 1 and 2. The default value is "1.0":



RatioSE = 1

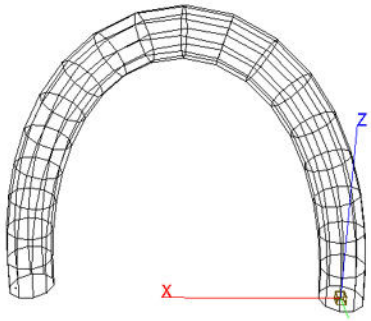


RatioSE = 5.0

If the length is set too large the segments do not overlap and become singular parts, as shown in following picture

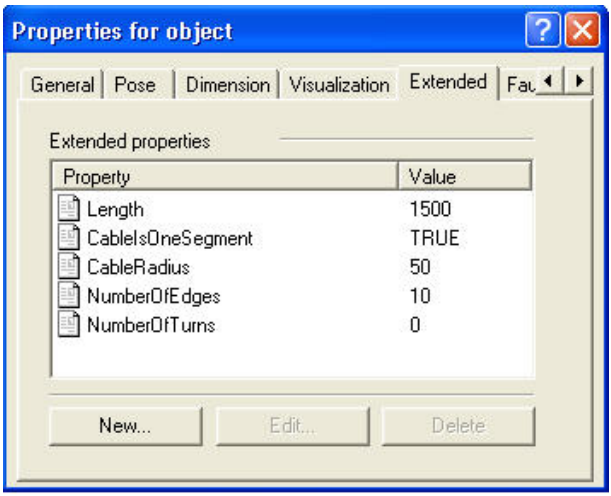
Type 2

This type of hose is not built by N sections but by one cylindrical polyhedron which is bent by modification of its polyhedron points (vertices) containing N+1 layers. The object controller itself can not generate such polyhedron. It has to be modeled with CIROS® or a text editor. The controller calculates the vertices only.



Ployhedron of a bent hose

The wireframe representation shows the structure of such a single-object hose. This object type needs some more properties:



Type 3

If a cable track is needed, another property must be set. Additionally to the properties of type 1 the property **CableDuct** with the value **TRUE** must be present.

The property **RatioSE** does not have an effect in this mode. The movement direction of the track is the z-axis of the start gripper point. The height of the cable track results automatically form the position of the end gripper point, because the start and end line are parallel and are leading through the start and end gripper.

Cable track with its start gripper point. Its Z axis is the movement direction

Object Properties for the COSCABLE Object

Property name	Value range	Mode of operation	Relevant for type
Length	greater than 0.0	Sets the length of the object in mm. Within this length the segments are aequidistantly arranged.	1,2,3
RatioSE	greater than 0.0	Balance of the hose of shifting it more to start or end. 1.0 is default.	1,2
CableDuct	TRUE or FALSE	Flag for cable track. If present and true the object is a cable track.	2
CablesOneSegment	TRUE or FALSE	If present and true the object is a single segment polyhedron.	2
CableRadius	greater than 0.0	If CableIsOneSegment =TRUE this value is the radius of the cable.	2
NumberOfEdges	corresponding to the model of the hose	If CableIsOneSegment =TRUE this is the number of edges, the polyhedron has. The number of layers is calculated automatically.	2
NumberOfTurns	integer	Complete torsion twists of the hose. If the hose should be twisted less than a complete turn, the start or endpoint can be rotated around its z-axis.	1,2

Hints:

Static conduits:
The design of the workcell looks more nice, if static hoses or conduits are visualized too. During modelling these parts can be laid more easily by using this flexible object type. After proper installation the object type can be reset to "inactive object". All calculated vertices, boxes, segments will remain in this position, if the workcell is saved.

Type 1 or type 2?
Type 1 is much more easily to built, because one segment can be replicated by simple copy and paste, but needs more vertices than type 2. Type 2 requires a completely modelled single segmented polyhedron with N Layers, which vertices are calculated by the object controller.

Textures
Textures on hoses or conduits make them much more attractive.

Estimation of length
The object controller is ideal for estimation of length of hoses or conduits or cable tracks, because the parameter "Length" is adjustable very easy and the user has instant visual feedback of his manipulations. With this method the lengths for real, wry, bent or twisted cables and tracks can be found very easily.

CAD data for cable tracks
Many manufacturers of cable tracks are offering 3D-Data download on their home pages, which can be imported by CIROS. This makes modelling much easier and faster.

Quality of Simulation
The hose, calculated by the object controller, mathematically is a bezier curve. Its run can be very near to reality but can not be compared to it, because dynamical effects like mass, weight, stiffness or inner life are not taken into account.

Examples	You can find a demo model for a hose on a robot in ‹CIROS directory›\samples\CableSimulation\RobotWithHose\RH-5AH55.mod You can find a demo model for a cable track at a storage system in ‹CIROS directory›\samples\CableSimulation\StorageWithCableTrack\X-Y-Storage.mod

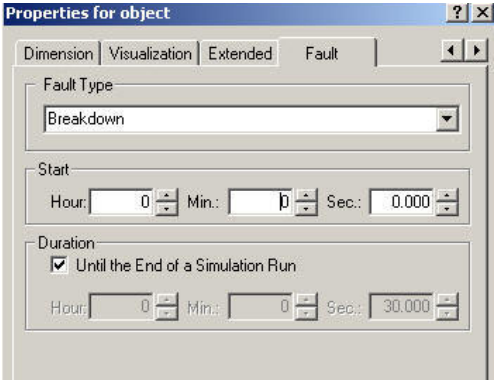
5.7 Fault Simulation

The module **Fault Simulation** enables the user in CIROS® to simulate faults that occur during the production process. The fault simulation is switched off by default. To switch the fault simulation on resp. off select the command **Extras → Fault Simulation**.

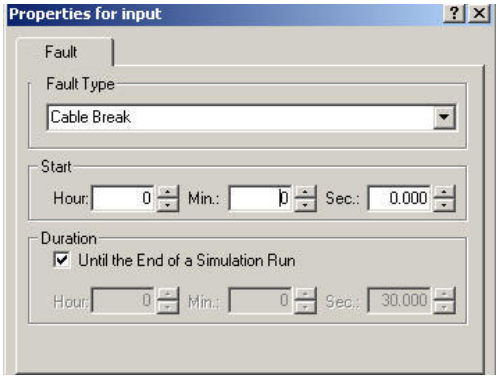
Modeling

The fault simulation discriminates the following faults:

- **Object faults**
  Select the dialog "Properties for object" of the object that should have a fault. Here you can enter the start and the duration of the fault.

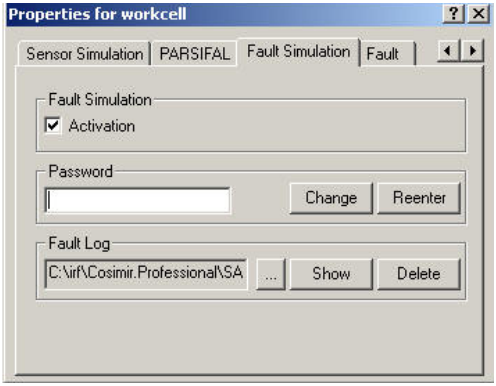


- **I/O Connection Faults**
  Select the dialog "Properties for input" of the input that should have a fault. Here you can enter the start and the duration of the fault.



Settings

In the dialog **Properties for work cell** you can change the following settings:



- **Fault Simulation Activation**
  Here you can switch the fault simulation on and off.

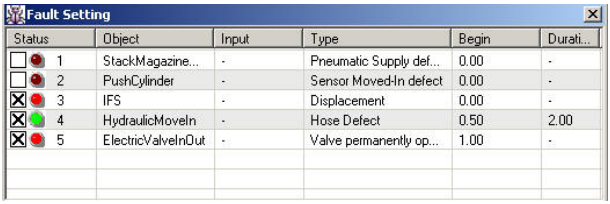
- **Password**

Here you can change the password to activate the teacher mode. Select **Reenter** to leave the teacher mode.

- **Fault Log**
  Here you can specify a new file for the fault log. The fault log records all actions of the fault localization.
- **Show/Delete**
  Here you can show the fault log or you can delete it.

Fault Setting

You can activate or deactivate the fault setting either executing the menu command **Extras -› Fault Simulation → Fault Setting** .In the dialog **fault setting** you get an overview of the modeled faults.



Enabling the check box you can specify the faults to be considered. The LED shows the fault status of the object.

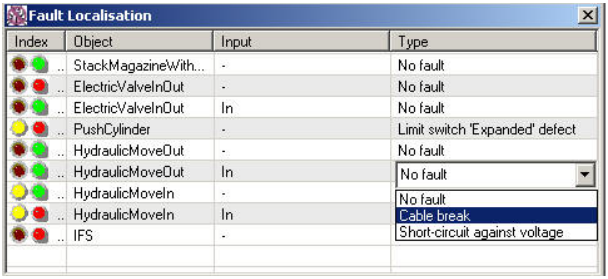
LED is of	The fault is not yet active or is ignored
Red LED	The fault is active.
Green LED	An active fault is over.
Yellow Bar	This fault is localized and it is ignored.

By a double click on the column **type** of a single fault , a list opens that contains possible types for this fault . Thereby you can change the fault type of an object or an I/O connection.

By a double click on the column **begin** of a single fault , you can enter a new beginning of a fault .
By a double click on the column **duration** of a single fault , you can enter a new duration of a fault.

Fault Localization

You can activate or deactivate the fault localization executing the menu command **Extras -› Fault Simulation → Fault Localization**.

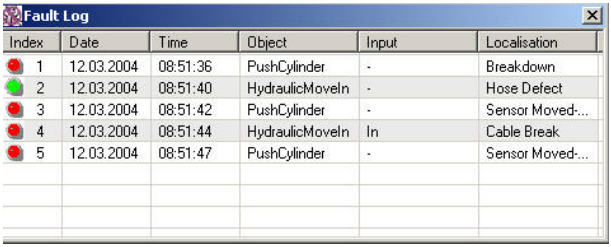


In the teacher mode (password) both LED columns will be shown. If the teacher mode is not activated you will only see the first LED column.

Yellow LED	A localization is set.
Green LED	The localization complies with the fault in the fault setting.
Red LED	The localization does not comply with the fault in the fault setting.

By a double click on the column **Type** of a single row, a list opens that contains possible types to localize. These names correspond to the names in column **Type** of the fault setting.

Fault Log



You can activate or deactivate the fault executing the menu command

Extras -› Fault Simulation → Fault Setting .The fault log records all actions of fault localization.

Red LED	The fault was not correctly localized.
Green LED	The fault was correctly localized.

6. Mechanisms

In CIROS® the simulation of so called base mechanisms is a powerful feature for simulation of work cells. A mechanism is assigned to an object using the object's type. Depending on the mechanism the object structure (concerning number of I/Os, number and configuration of sections and joints) is given. The mechanism can only be simulated correctly if the given object structure exists.

To model a mechanism, use the model libraries. Add an object with a mechanism to the work cell to guarantee that the object structure is correct. Afterwards, change the shape as well as the dynamics and I/O names of the object to model your own mechanism.

Please note that to control any of the mechanisms the input values have to change from low to high to start the mechanism. Moreover the output values containing the state of the mechanisms are only updated if the outputs are connected to an input.

6.1 Gripper

Use the gripper mechanism to simulate grasping of workpieces. If system input 0 of the gripper object is set to high, the gripper grasps an object that has a free grip point in the grip range of the gripper's gripper point. All sections of the gripper object that have a degree of freedom are moved to their upper limits. Thus the movement of gripper chucks is simulated.

Mechanism: Gripper			**Object Type:** Gripper		
System Input/Output	**Input**	**Index**	**Type**	**Value**	**Description**
		000	digital	1	Closes the gripper.
				0	Opens the gripper.
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	Gripper is not closed.
				1	Gripper is closed.

Examples	
Object	**Model Library**
Parallel Gripper	Miscellaneous Grippers
Three Jaw Gripper	Miscellaneous Grippers

6.2 Conveyor Belt

If there is another object with a free grip point above a conveyer belt object, the object is moved along the active surface of the conveyor belt if the grip point lies inside the grip range of the active surface. This only works if system input 0 of the conveyor belt is set to high. If the object is moved up to the end of the active surface system output 0 is set to high.

Mechanism: Conveyor Belt **Object Type:** Conveyor Belt

System Input/Output	Input	Index	Type	Value	Description
		000	digital	1	Switches the conveyor belt on.
				0	Switches the conveyor belt off.
		001	digital	1	Conveyor belt transports backwards.
				0	Conveyor belt transports forward.
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	There is no object at conveyor's end.
				1	There is an object at conveyor's end.

Examples	
Object	**Model Library**
Conveyor Belt	Miscellaneous Mechanisms

6.3 Push Cylinder

The push cylinder is extended if system input 0 is set to high. If there is an object with a free grip point in the grip range of the push cylinder's gripper point the object is moved by the push cylinder. The push cylinder is retracted if system input 0 is set to low.

Mechanism: Push Cylinder **Object Type:** Push Cylinder

System Input/Output	Input	Index	Type	Value	Description
		000	digital	1	Extends the push cylinder.
				0	Retracts the push cylinder.
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	Push cylinder is not extended.
				1	Push cylinder is extended.

Examples

Object	Model Library
Push Cylinder	Miscellaneous Mechanisms

6.4 Rotary Drive

The mechanism rotary drive is based on the push cylinder mechanism.

Mechanism: Rotary Drive		**Object Type:** Rotary Drive			
System Input/Output	**Input**	**Index**	**Type**	**Value**	**Description**
		000	digital	1	Rotary Drive moves to upper limit.
				0	Rotary Drive moves to lower limit.
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	Rotary Drive is not at upper limit.
				1	Rotary Drive is at upper limit.

Examples	
Object	**Model Library**
Rotary Drive	Miscellaneous Mechanisms

6.5 Turntable

All axes of turntable object are moved to the upper limits if system input 0 is set to high. If there is an object with a free grip point inside the grip range of the turntable's active surface the object is moved with the turntable.

Mechanism: Turntable **Object Type:** Turntable

System Input/Output	Input	Index	Type	Value	Description
		000	digital	1	Turns the turntable.
				0	–
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	Turntable is moving.
				1	Turntable stand still.

Examples

Object	Model Library
Turntable	Miscellaneous Mechanisms

6.6 Two Way Push Cylinder

The function of the two way push cylinder is similar to the function of the push cylinder. For the purpose of control there are two system inputs. According to this there are two system outputs for the cylinder's state.

Mechanism: Two Way Push Cylinder **Object Type:** Two Way Push Cylinder

System Input/Output	Input	Index	Type	Value	Description
		000	digital	1	Extends the push cylinder.
				0	–
		001	digital	1	Retracts the push cylinder.
				0	–
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	Push cylinder is not extended.
				1	Push cylinder is extended.
		001	digital	0	Push cylinder is not retracted.
				1	Push cylinder is retracted.

Examples

Object	Model Library
Two Way Push Cylinder	Miscellaneous Mechanisms

6.7 Turning Mover

The turning mover consists of two sections and a vacuum gripper that can be controlled by setting system inputs 2 and 3. Use system inputs 0 and 1 to control the position of the turning mover.

Mechanism: Turning Mover			**Object Type:** Turning Mover		
System Input/Output	**Input**	**Index**	**Type**	**Value**	**Description**
		000	digital	1	Moves to position A.
				0	–
		001	digital	1	Moves to position B.
				0	–
		002	digital	1	Grasp
				0	–
		003	digital	1	Release
				0	–
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	Turning mover is not at position A.
				1	Turning mover is at position A.
		001	digital	0	Turning mover is not at position B.
				1	Turning mover is at position B.

Examples	
Object	**Model Library**
Turning Mover	Miscellaneous Mechanisms

6.8 Parts Feeder

Use the mechanism parts feeder to model depots for workpieces etc. The associated object contains a whole string of gripper points. The sequence of these gripper points is important for the function of the parts feeder that is filled by moving objects with free grip points in the grip range of the feeder's gripper points. Note that the first of the feeder's gripper points must not be covered. In case of setting system input 0 of the parts feeder to high the object at the second gripper point is moved to the first gripper point, the object at the third gripper point is moved to the second gripper point etc. If there is an object at the position of the first gripper point system output 0 is set to high.

Mechanism: Parts Feeder **Object Type:** Parts Feeder, optional "with Gravity"

System Input/Output	Input	Index	Type	Value	Description
		000	digital	1	Requests a new part.
				0	–
	Output	**Index**	**Type**	**Value**	**Description**
		000	digital	0	No part is available.
				1	There is a part available.

Examples

Object	Model Library
Parts Feeder 1	Miscellaneous Mechanisms
Parts Feeder 2	Miscellaneous Mechanisms

6.9 Proximity Sensor

This simple proximity sensor checks if there is an object with a free grip point in the grip range of the sensor's gripper point.

Mechanism: Proximity Sensor **Object Type:** Proximity Sensor

System Input/Output	Output	Index	Type	Value	Description
		000	digital	0	No grip point detected.
				1	Grip point detected.

Examples

Object	Model Library
Proximity Sensor	Miscellaneous Mechanisms

6.10 Replicator

Use the replicator mechanism for creation of new objects based on templates. The extended properties of the replicator object contain the assignment of system inputs and templates (example; template 0 = “Workpiece”). If a system input is set to high, a new object based on the associated template is created at the gripper point of the replicator object.

Mechanism: Replicator			**Object Type:** Replicator		
System Input/Output	**Input**	**Index**	**Type**	**Value**	**Description**
		000	digital	1	Create first configured object.
				0	–
		001	digital	1	Create second configured object.
				0	–
		...	...	...	...
		00n	digital	1	Create n-th configured object.
				0	–

Examples	
Object	**Model Library**
Replicator	Miscellaneous Mechanisms

6.11 Trash Can

The trash can is the counterpart of the replicator. Use the mechanism trash can to remove objects at runtime. Each system input of the trash can object is associated to a gripper point. If system input 1 is set to high and there is an object with a free grip point inside the grip range of grip point 1 of the trash can object, the object is removed.
Please note that all objects that have been removed by this mechanism are not recovered by choosing command Reset Work cell from the Edit menu.

Mechanism: Trash Can **Object Type:** Trash Can

System Input/Output	Input	Index	Type	Value	Description
		000	digital	1	Remove object at gripper point 1.
				0	–
		001	digital	1	Remove object at gripper point 2.
				0	–
		...	...	...	...
		00n	digital	1	Remove object at gripper point n.
				0	–

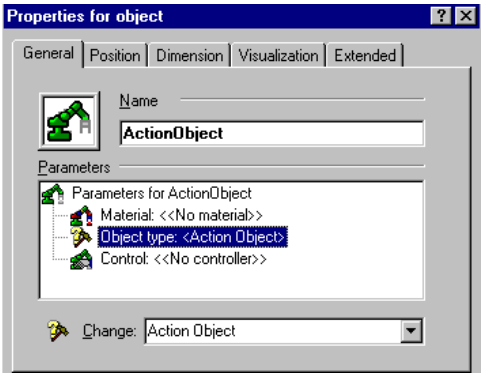
Examples

Object	Model Library
Trash Can	Miscellaneous Mechanisms

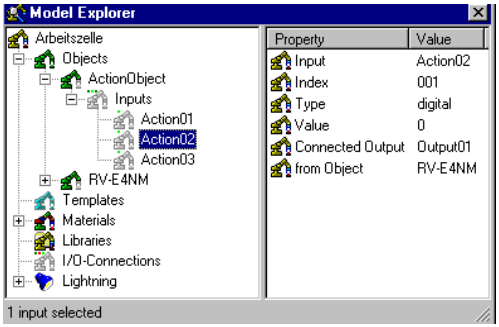
6.12 Action Objects

Use Action Objects in CIROS® to execute different actions because of output values of any object.

The Action Object must have inputs only und have to be configured of type **Action Object**.

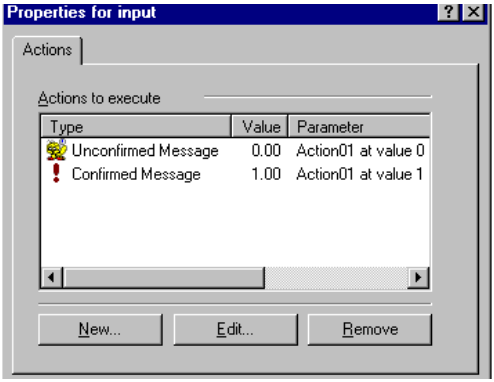


The inputs of the Action Object are connected to outputs of simulated objects (e. g. robots).



Each input of the Action Object is able to be configured. For different

input values different actions can be defined.



Now, certain actions have been associated to certain output values of the simulated object.

Properties for Input

Configure the properties of an Action Object's input with the corresponding dialog in the tab control. Select the input to be configured in the Model Explorer and open **Actions** in the tab control **Properties for input**.
Click **New** to associate a new action with the selected input.
Click **Edit** or **Remove** to edit or to remove the action selected in the list.

Actions The following actions are available:

Action	Description	Parameter
Unconfirmed message	Writes an unconfirmed message in window Messages of CIROS®	Message text
Confirmed message	Opens a message box that has to be confirmed.	Message text
Show picture	Shows a picture.	Filename of picture (.bmp-Format)
Play sound	Plays an audio file. Use the switch cyclic to let the sound be played as long as the input value of the Action Object is the configured value.	Filename of sound (.wav-format)
Play video	Plays a video file.	Filename of video (.avi-Format)
Show www-page	Opens a WWW page in the standard browser.	URL of HTML-page.
Open CIROS® work cell	Opens a work cell.	Filename of work cell.
Restart simulation.	Executes the menu command **Edit → Reset Work cell** and starts the simulation.	
Track object	Starts the object tracking.	Name of object to be tracked
Make object invisible	Makes an object invisible.	Name of object to be invisible
Make object visible	Makes an object visible.	Name of object to be visible
Switch on light source	Switch on a certain light source.	Index of light source to be switched on

Switch off light source	Switch off a certain light source.	Index of light source to be switched off

Action	**Description**	**Parameter**
Open CIROS® help	Opens a HTML page of CIROS® help.	HTML page to be opened in CIROS® help..
Open popup-window	Opens a popup-window containing text.	Text for popup-window.
Set material transmission	Sets the transmission factor of one material to the value of the analoge input.	Material name
Move to camera cruise lookat	Sets the lookat in the work cell window that is associated to the camera cruise.	Index of camera cruise stepl.
Set **Level of Detail**	Sets the **Level of Detail (LOD)**	**Level of Detail**
Import CIROS® work cell.	Imports one CIROS® work cell into the current model.	Filename incl. full or relative path.

7 Communication Interfaces

CIROS® Studio provides several communication interfaces:

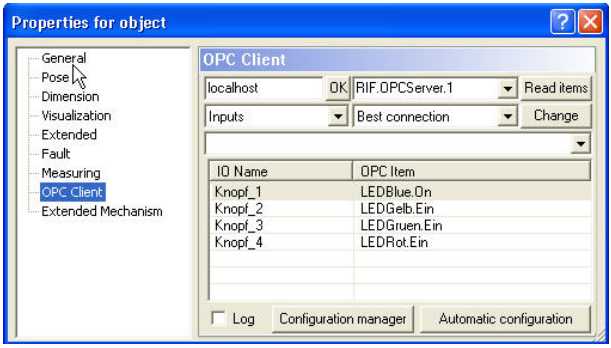
- OPC Client
- OPC coupling
- PARSIFAL
- Robot controller interface to Mitsubishi robot controller

7.1 OPC Client

The OPC-Client can be used to swap out the controller of an object to an OPC-Server. All or some of the configured IOs and robot joint values can be exchanged between the client and the server. The required configuration of COM interfaces in the Windows operating system is described in chapter OPC controller connection.

Connecting

Select an object and change its controller type to OPC-Client. The dialog box Properties for object will then be extended by a tab named OPC Client.



Options:

Host name

Use this field to enter the host to which you want to connect. The default value is localhost. If you entered a name, press OK. CIROS then starts to read all OPC-servers of the selected host.

Server name

	Next to the host entry you will find a drop down box with all available OPC servers. Use this field to select the desired OPC server. After clicking on read items, the list below will be filled with available OPC items. Note: The OPC server must be running.
Inputs/Outputs/Joint Values	This pull down box can be used to select, which part of your CIROS object you would like to configure. Outputs are read from the server to CIROS, inputs are written to the server by CIROS. Joint values can only be read. Depending on the choice made in this field, the number of available items and the display of already configured elements of your object may vary.
Connection	Select the desired connection. The default value Best connection uses the best available connection. This is automatically detected using the following sequence: ASYNC V2.00, ASYNC V1.00, SYNC. Alternatively you can select an entry to use only a specific type of connection.
Change configuration	Use the table to select the entry which you want to configure. From the list of items select the item, which shall be connected to the selected element. Then press Change. The table entry will be extended by the new OPC item. If a desired item is not available for configuration, please check the read/write permission settings of your OPC server.
Log	Check this option to log extended output to the message window. Automatic configuration:
Configuration manager	The Configuration manager button opens the Configuration Manager, which allows you to manage different configurations.

7.2 OPC Controller Connection

The OPC coupling of CIROS enables you to connect external controllers to the simulation system. The external controllers can be of different kind: controllers that perform basic logical and arithmetic calculations or controllers that control one or several kinematics. Programmable logic controllers (PLC) belong to the first group and robot controllers to the second.

The communication interface used for the coupling of the controller and the simulations system is OPC (OLE for Process Control). OPC is a standardized software interface definition that facilitates and standardizes the data access of PC-based applications within the automation engineering field. For a controller that is to be coupled to CIROS an OPC server must be disposable that provides controller internal information to communication partners. The OPC coupling of CIROS is realized as an OPC client which retrieves the necessary information from the server and passes them to the CIROS kernel.
The coupling is realized as an extension module and is fully integrated into the user interface of CIROS. Since the coupling also has an adequate controller interface to CIROS an external controller shows exactly the same behaviour as a controller that is integrated in CIROS and that emulates the program processing and the kinematic functions.

Setup and Installation

Your simulation computer (PC) must meet the following requirements for the operation of the CIROS OPC coupling:

- 32 bit Windows operating system (Windows 2000, Windows XP,)
- Ethernet network card
- TCP/IP protocol

Please consider the following notes for the communication between CIROS and the OPC server.

- Microsoft network:
  Integrate the CIROS-PC and the OPC server into a common Windows network and assign unequivocal computer names.

- User management:

Install two identical users with the same user name and the same password on the CIROS-PC and on the OPC server. Grant administrator rights to both users.

- Registration of the interface classes of the OPC server:
  Register the class descriptions of your OPC server in the registry of the CIROS-PC. Note: In most cases it is required to install the OPC Server also on the client PC in order to register the DCOM-classes properly.

DCom Configuration:

OPC is based upon the DCOM (Distributed Component Object Model) technology developed by Microsoft for distributed applications. Hence, the features of DCOM are used, when CIROS and the OPC server of the external controller are running on different computer systems.
DCOM uses several security mechanisms that control the access to local applications over network. Numerous system settings that are dependent on the network environment are necessary for the establishment of a network connection.

Firewall Settings

The firewall of both machines should be configured to allow connections to TCP Port 135 and OPCEnum.exe (usually found at ‹Windows Directory›\System32) .

Note

You can find examples for DCOM settings in the section **Control-› OPC coupling -› Setup and Installation** of the online help system.

7.3 PARSIFAL

Use the Extension PARSIFAL to connect the simulation system to other simulation and controller systems.
To configure PARSIFAL, use the corresponding tab of the Properties for workcell dialog.

Choose from the list of parameters the parameter to configure. Depending on the type of the chosen parameter some control elements are displayed at the bottom of the dialog box.
You are able to configure the following parameters for PARSIFAL:

- Save client connections with the workcell
- Logging in message window
- Logging in log file (.\bin\PARSIFAL.log)
- Cycle time of text-based notification for model update
- Automatic model partitioning

7.4 Robot Controller Interface

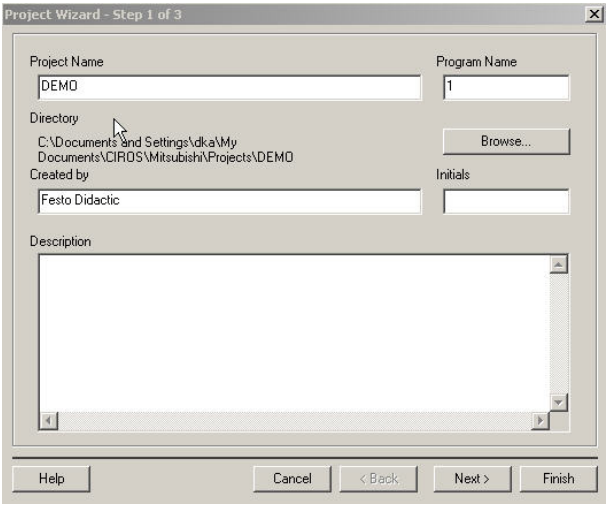
The RCI Explorer (Robot Controller Interface) is the new information-processing center of CIROS® Studio. With the RCI Explorer you can up- and download programs, simply by drag-and-drop. Start programs in any slot you want and keep track of the actual state of your robot parameters, error messages, system variables, etc.
It has never been so easy to test your programs in deep.

The new Debugger supports a breakpoint-oriented online debugging of

your robot programs. Set breakpoints wherever you want. Start your program in any line; use single stepping forward and backward. The online debugger is the ultimate test tool for detailed online testing of your programs.

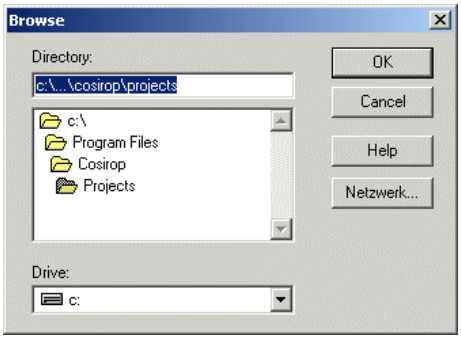
How to Setup a new Project with CIROS® Studio

Before you can write robot programs you have to open your work cell or you start the project wizard using the menu function **File → Project Wizard**.

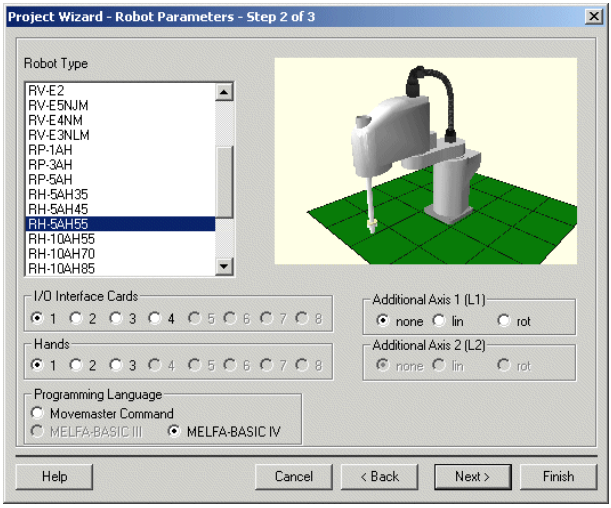


Insert your desired **Project Name** into the appropriate field of the dialog box. The dialog box will come up with the project name "UNTITLED". In the example the project name is **DEMO**. You can enter any valid file name (without file name extension) into this field. As **Program Name** insert e. g. 3.
For each new project you create, CIROS® Studio will create a new directory with the name of the project. CIROS® Studio uses this directory to store all the programs that belong to your project.
The program name is the name CIROS® Studio uses to download the robot program into the robot controller.

The directory used for this project is displayed under Directory. If you want to change the directory or if you want to create a new directory press the button Browse which will open the browse dialog.



Insert your name into the text box **Created by**, your initials into **Initials**, and a short description of the project task into **Description**.
Proceed to the second step of the project wizard by pressing the **Next** button. This will display the second step of the Project Wizard.



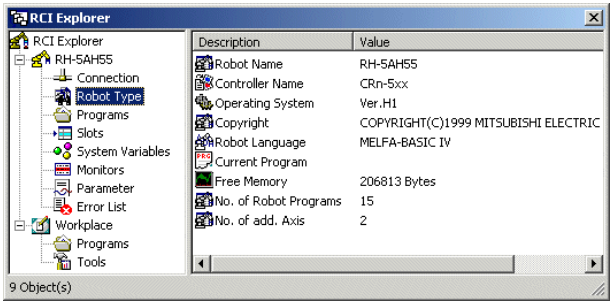
Select the robot type you are working with, from the list **Robot Type**.

Then, select the number of **I/O Interface Cards** (only required for the I/O Monitor), the number of **Hands** (only required for opening and closing hands in the Jog Operation), the number and type of **Additional Axes** (required for Jog Operation) and the **Programming Language**, which is essential for opening of the correct program window. You can only choose between different programming languages for robots of the RV-EN and the A series (RV-A, RH-AH, RP-AH). Robots of the RV-M and RV-E series can only be programmed in **Movemaster** Command (MRL).

Press the button **Finish**, to create the project. A work cell window, the RCI Explorer, an empty robot program window, the associated position list window, and the message window will be opened and arranged on your monitor. Several files for the project itself, the program and the position list are created in the directory **Project Name**.

How to Work with the RCI Explorer

The **RCI Explorer** is an information and data exchange center. It gives you an overview of the current state of the robot and provides an intuitive way to upload, download, start, debug, and monitor robot programs.



The RCI Explorer contains two folders. The first folder is the robot folder. The name of the folder matches the robot name, you have configured in the second step of the project wizard. This example makes use of the robot **RH-5AH55**.
The robot folder provides access to the data on the robot controller. The

second folder is the **Workplace**. It contains the data on your PC in the project directory.

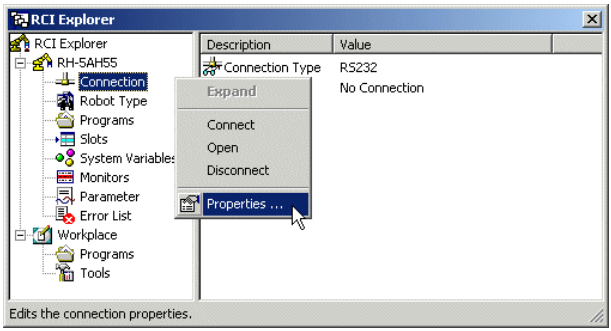
In the robot folder you find information concerning
- the connection to the robot controller,
- the connected robot type,
- the programs, currently available on the robot controller,
- the contents of the program slots,
- the system variables,
- the state of the robot using a wide range of monitors,
- the robot parameters,
- and the most recent errors.

In the workplace folder you find
- the programs and position lists currently available in the CIROS® Studio project directory on your PC,
- and the CIROS® Studio tools.

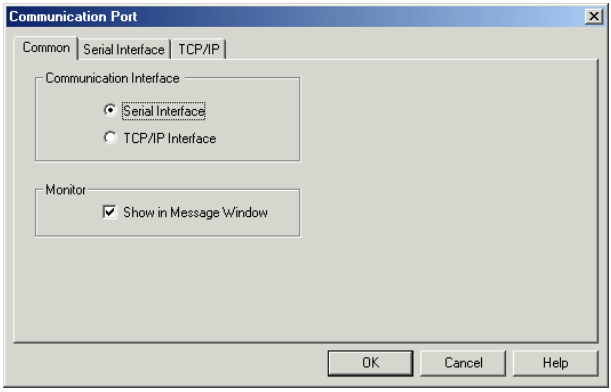
Before you can exchange programs and position lists between the CIROS® Studio PC and the robot drive unit the following you have to do the following steps.

1. Connect a serial interface of the CIROS® Studio PC to the RS-232 interface of the robot controller. Be sure to use a serial cable that connects the hardware handshake signals (DTR, RTS, CTS) of the serial interface, too. See the robot manual for details.
2. Use the **RCI Explorer** to check the connection properties. Open the context menu of the **Connection** by clicking with the right mouse button on the folder **Connection**. Select **Properties** from the context menu.

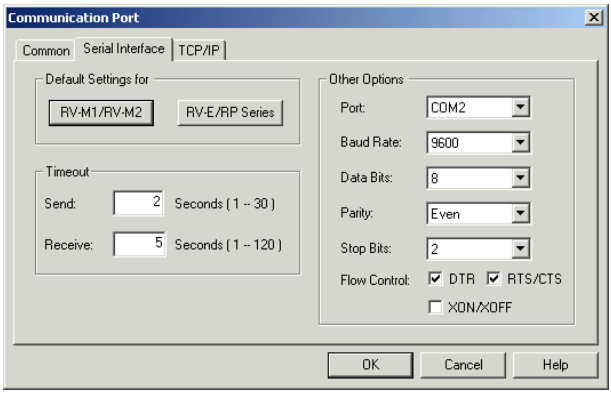
How to Establish a Connection to the Robot Drive Unit



3rd Check if your Communication Interface is the Serial Interface.
Alternatively, you can open the dialog Communication Port Common also with the command **Extras → Settings → Communication Port**.
The command is only available if a work cell is loaded.

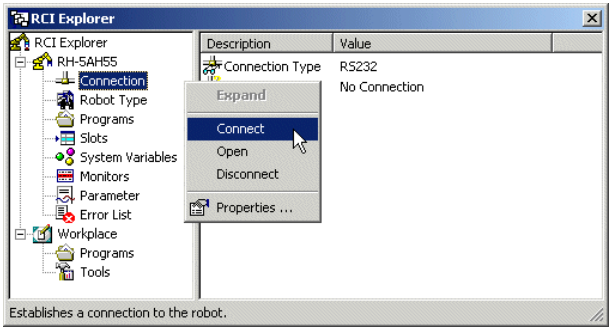


4. Switch to the property page Communication Port Serial Interface to select the port and to set the communication parameter. Select the correct port and set the communication parameters of the serial interface appropriately. The default port is the serial port that has been selected during the installation.



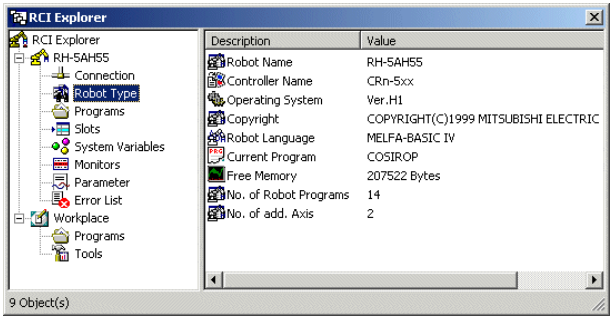
5. Disable the teaching box.
6. If you have a robot of the A series (RV-A, RH-AH, RP-AH), set the key switch on the robot controller to **Auto(Ext.)**.
7. Establish the logical connection between CIROS® Studio and the robot drive unit by executing the command **Execute/Init Connection**, which will communicate with the robot and determine the type of the robot and some parameters of the robot and display them in the dialog box Robot Type. If this dialog box will be displayed the communication to the robot drive unit has been established and programs and position lists can be down- and uploaded.

8. Establish a connection between CIROS® Studio and the robot controller. Use again the context menu of the **Connection**. Select **Connect** to establish the connection.
   Alternatively, you can use the command **Execute → Init Connection** to establish a connection. If the connection initialization is successful, the dialog Robot Type displays information regarding the robot type, the robot programming language, and the amount of free memory.



How to Check the Robot Type

Check, if the configured robot type matches the actual type of the connected robot. In the RCI Explorer select the folder **Robot Type**:

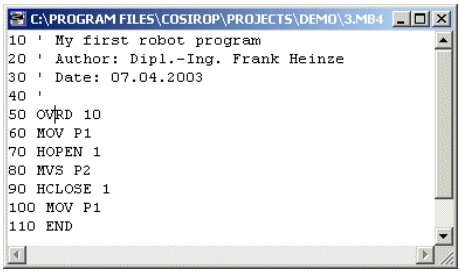


You get information concerning the type of the connected robot, the version of the operating system, the actual robot language, and the number of available robot programs on the robot controller.

How to Write a Robot Program

You are now ready to write your first robot program. Activate the window with the robot program by clicking into the window or by selecting the window with the command Window/1,2,3
Now you can freely edit your program using the keyboard and the mouse.

The online help contains a detailed syntax description for every command. The **F1** key opens the help page of the command where the cursor is located. See the **Reference Manual** or the **Instruction Manual** to get further information.
A simple robot program in MELFA-BASIC IV might look like this:



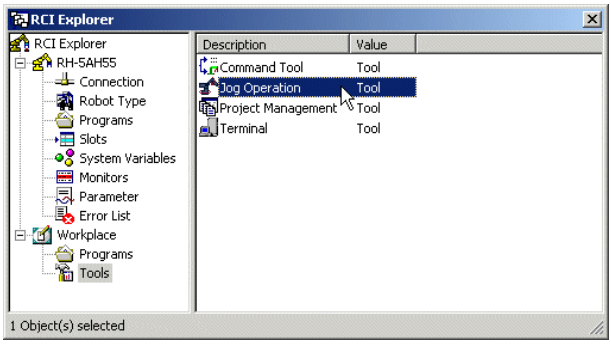
Use the context sensitive help to get help on the command **OVRD**. To open the help page place the cursor inside the OVRD command and press the **F1** key.
To renumber a program sequentially use the command **Edit → Renumber**. This command can be used to renumber the whole program (in this case no part of the program must be selected) or only the selected range of the program.

If you have added new program lines that should be inserted between other lines in the program according to their line numbers, use the command **Edit**/**Sort** to sort the whole program in ascending order according to the line numbers.
Save the program with the command **File → Save**.

How to Create a Position List Using the Robot

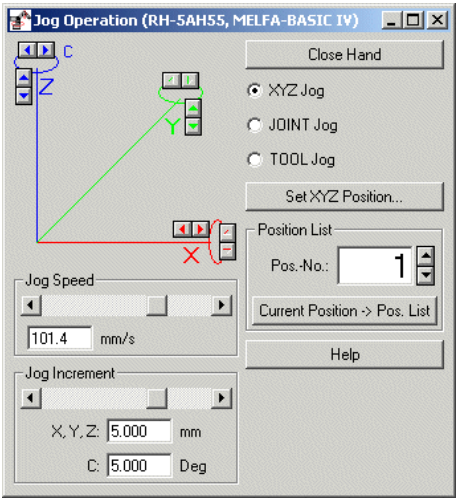
Read and observe the safety instructions of the SAFETY MANUAL carefully before operating or programming the robot with CIROS® Studio! Nobody **must** be in the safeguarded area, when using the JOG operation.

Usually, you use the T/B (teach pendant) to create position data. Have a look at your Instruction Manual for a detailed description of how to create position data. Once you have created position data with the T/B it is easy to upload the position data into a position list on your PC. Simply use the program upload. You will get your position data as a position list and additionally any program lines you might have created with the T/B.
Alternatively, you can create and modify a position list interactively using CIROS® Studio. Open the window Jog Operation by double clicking on the tool **Jog Operation**.

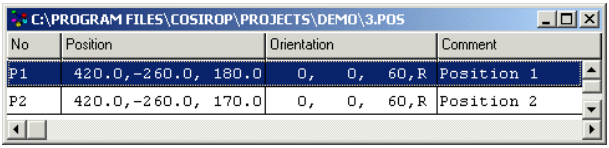


Optionally, you can also open the window Jog Operation by the command **Execute → Jog Operation**. Use this window to jog the robot in **JOINT, XYZ** and **TOOL** coordinates.

Use the button **Current Position → Pos. List** to insert the current robot position into the position list as position number **Pos.-No.** To override an existing position just enter the position number into the text box **Pos.-No.** or use the spin buttons next to the text box. Set the **Jog Speed** and the **Jog Increment** to appropriate values.



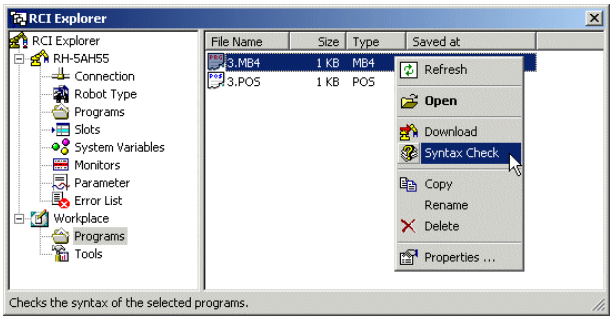
Create two positions P1 and P2 in your current position list and save the position list with the command File/Save. Your position list might look like this:



Discover next, how to check the syntax of your program.

How to Check the Syntax of your Program

Your first robot program and the associated position list are now located on the PC in the project directory. In the RCI Explorer they are accessible in the folder Programs in your **Workplace**. Click with the right mouse button on the name of the robot program and open the context menu of the program. Select **Syntax Check** to check the syntax of your program.



You can check the syntax of your program also by using the command **Execute/check Syntax**. The syntax checker finds syntactical errors in your program and displays the errors in the message window.

Load the syntactically correct program and position list into the robot controller.

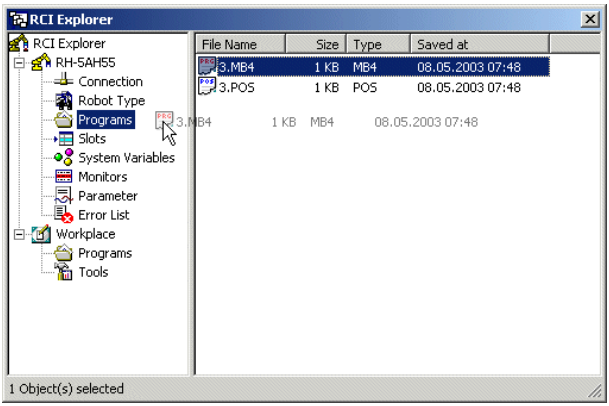
How to Exchange Programs and Position Lists with the Robot

Before you can exchange programs and position lists between the CIROS® Studio PC and the robot controller the connection must have been established. If the connection is established, you can

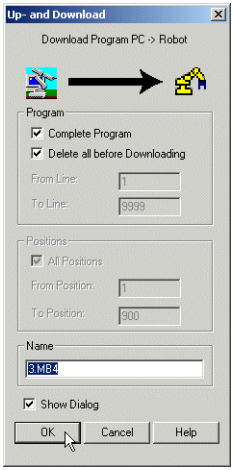
- Download a program to the robot
- Download a position list to the robot
- Upload a program from the robot to the PC

Downloading a program

The RCI Explorer offers a very easy and intuitive way to download a program from the PC to the robot controller. Select the program you want to download to the controller, in the folder **Programs** in your **Workplace**. In the example the program **3. MB4** is selected. Keep the left mouse button pressed and simply "drag and drop" the program into the folder **Programs** of the robot:



If you release the mouse button, the download starts. The download can also be started with the command **Download** from the context menu of the program. Click on the program **3. MB4** with the right mouse button to open the context menu.



Alternatively, you can download a program without the RCI Explorer, too. Activate the window with the robot program by clicking into the window or by selecting the window with the command Window/1,2,3.... Download the program by executing the command **Extras → Online Management -> Download PC → Robot**. The dialog box Up- and Download is displayed.

Usually, it is best to use the default values in the dialog “Up- and Download" and download the complete program. To download the complete program, select the option **Complete Program**. If you want to download only a certain number of lines, switch off the option Complete Program and insert line number for **From Line** to **To Line**. For MELFA-BASIC III and MELFA-BASIC IV the line numbers have to be in the range between 1 and 32767, for Movemaster Command between 1 and 9999. Before the download, the specified line range is cleared in the robot controller. Furthermore, insert the program name, that will be used for your program in the controller. This field is always initialized with the file name of the program on the PC.

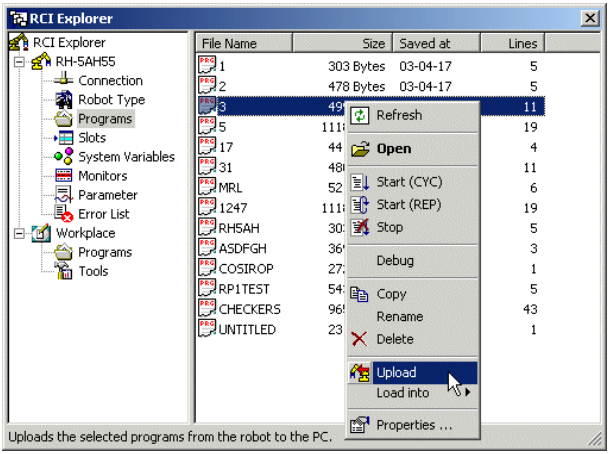
If you are programming in MELFA BASIC and you are using line numbers larger than 9999, be sure to check **Delete all before Downloading**. If this option is chosen, a program in the Drive Unit will be deleted even if it contains line numbers larger than 9999. If this option is not selected, line numbers larger than 9999 will not be erased, independent of the line number you specify in the **To Line** section. Position lists will not be deleted.

Press the **OK** button to start the download. All commands that are

transmitted to the drive unit are displayed on the screen. After each command the alarm (error) status of the drive unit is checked. If an error should occurs the download will be aborted and the erroneous command is displayed.

If you are trying to download an empty program or an empty position list, a warning is issued and you are asked, if you really intend to download an empty program and delete the program on the robot controller. It is saved to cancel the download at this point. The warning keeps you from destroying valuable programs on the controller, in all the cases, when you actually want to upload a file from the robot controller and you select the download by mistake.

If an error is reported during deleting of the old program (command "DL. . . " for RV-E or RV-M robots), delete the program on the robot controller manually. Use the RCI Explorer to delete a program. Open the "Programs" folder on the robot. Open the context menu of the program that is going to be deleted, by clicking with the right mouse button on the program name. Select **Delete**.

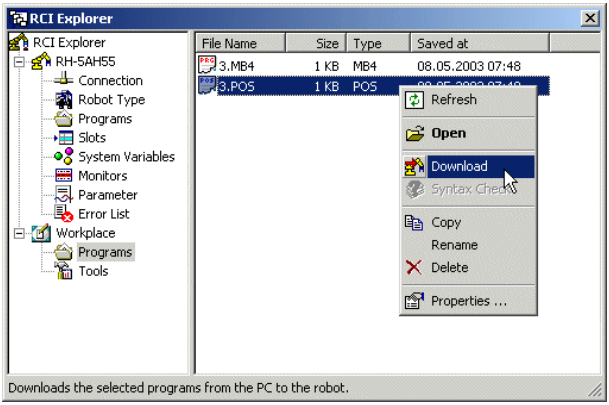


After the successful download of the robot program, discover how to download the position list to the robot controller.

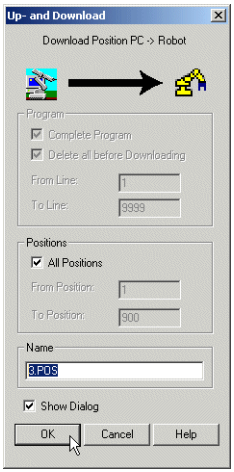
Downloading a position list

The steps required to download a position list are almost the same as for a program. The easiest way, to download a position list from the PC to the robot controller, is to use the RCI Explorer. Select the position list you are going to download to the robot controller, in the folder **Programs** on your **Workplace**. Keep the left mouse button pushed and drag and drop it into the Programs folder on the robot.

The example demonstrates another way to download a position list. Select the position list **3. POS**. Open the context menu with the right mouse button. In the context menu select **Download**.



Alternatively, you can download a position list without the RCI Explorer. Activate the position list window and execute the command **Extras → Online Management -› Download PC → Robot**.



The dialog box Up- und Download is displayed. In most cases, it is best to can use the standard settings. Be sure that the option **All Positions** is selected. If you want to download only a special range of position numbers, switch off the option **All Positions** and insert values for **From Position** and **To Position**. There are no limits for position numbers in MELFA-BASIC III and MELFA-BASIC IV. In Movemaster Command the position numbers are limited to 1 to 999, and for the RV-M1 the range is 1 to 629.

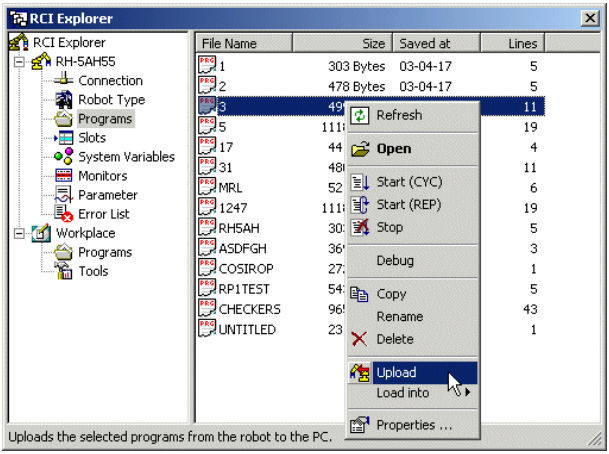
Before the download the old positions in the selected range are cleared. **Note**, that the old positions are not cleared if you use MELFA-BASIC IV.

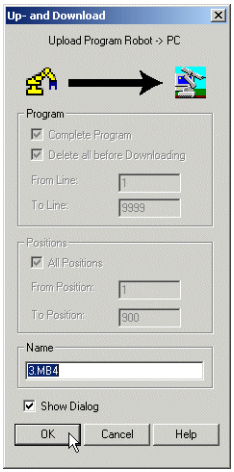
Additionally, you can add a name. The default name is the name of the file position list on the PC.
If you use a different name, be sure to use the same name that you have used for the program beforehand.

It is also very easy, to upload all the programs on the robot controller back to the PC.

Uploading a program

The procedure for uploading is very similar to the downloading procedure. Again, it is best to use the RCI Explorer. Just drag and drop the program from the **Programs** folder of the **robot** to the **Programs** folder of your **Workplace**. Optionally, you can use the context menu of the robot program. Right click with the mouse on the program **3**. Select **Upload** in the context menu to start the upload of a program and its associated position list. Note, that the program is uploaded into your Workplace folder, but it is not opened on your monitor. If you want to upload and open the program, select **Open** from the context menu or simply double click on the program name.





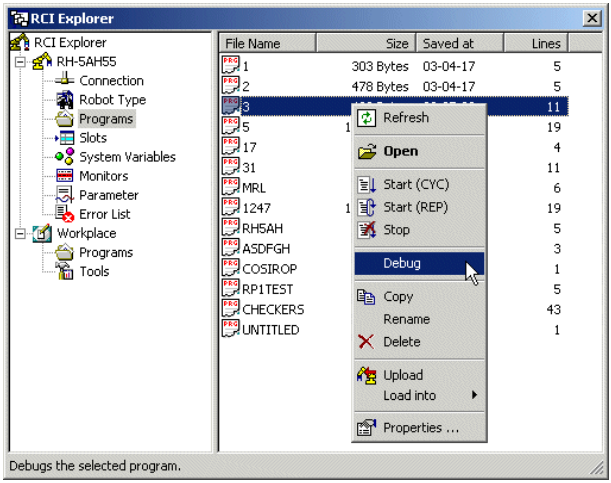
Alternatively, you can also upload programs and position lists without the RCI Explorer. Activate the program window that is to be used for the uploaded program. You can open a new program window with the command **File → New**, too. Then, execute the command **Execute → Upload Robot → PC**. This will again open the dialog box Up- and Download. After pressing the **OK** button the program will be uploaded line by line from the robot controller and finally displayed in the program window.

During this upload the original program file on the disk will be overwritten and there is no way to restore it.
You are now ready to debug your program directly on the robot controller.

How to Debug a Robot Program

Read and observe the safety instructions of the SAFETY MANUAL carefully before operating or programming the robot with CIROS! Nobody must be in the safeguarded area, when debugging your program.

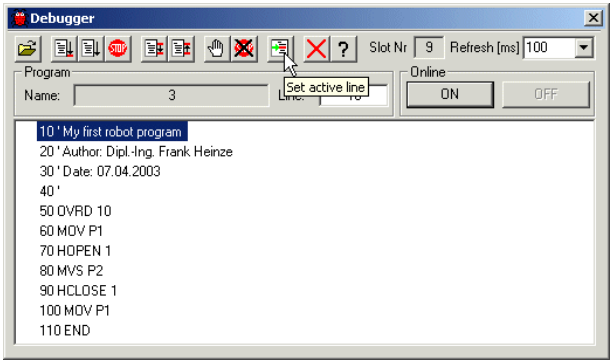
Now, as you have downloaded your robot program successfully to the robot controller, it is time to test your program online on the robot controller. Locate last errors with the online Debugger. Start the Debugger by using the context menu of your program. Right click with the mouse on the name of the program in the folder Programs inside the robot folder. Select **Debug**. In the example the debugger for the program **3** is opened:



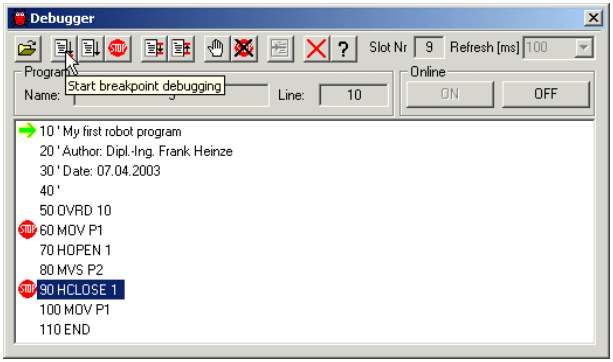
Note!

In the debug mode, the robot actually moves according to the movement commands.

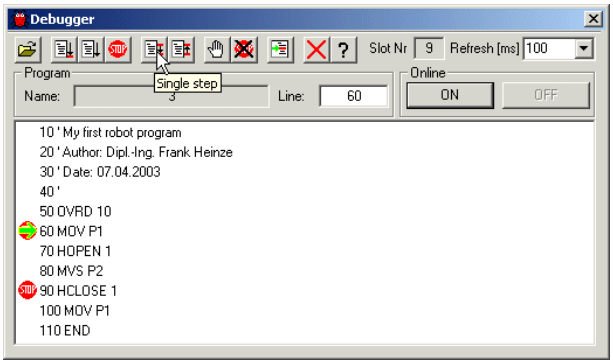
The Debugger opens with the selected program. Set the current line to the first program line. Select the first program line with the mouse and set it as currently active line by pressing the icon shown in the following picture:



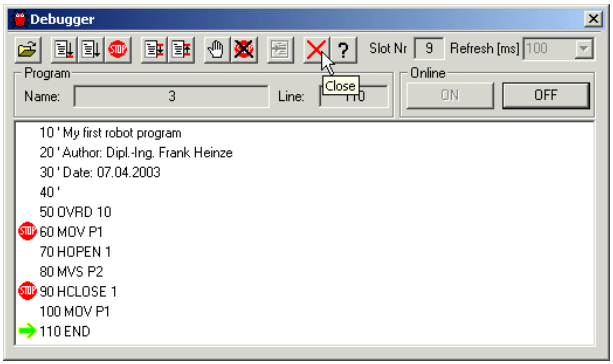
A small green arrow marks the currently active line. This is the line that will be executed next by the robot program. Now, set breakpoints by double clicking on the program lines, on which the program is supposed to stop during debugging. Press on the icon shown in the following picture and start the breakpoint debugging:



Program execution stops at the first breakpoint. Now, continue using the single step mode. In the single step mode, only the command in the currently active program line is executed. Afterwards the currently active line moves on to the next program line. Execute a single step by pressing the icon shown in the following picture:



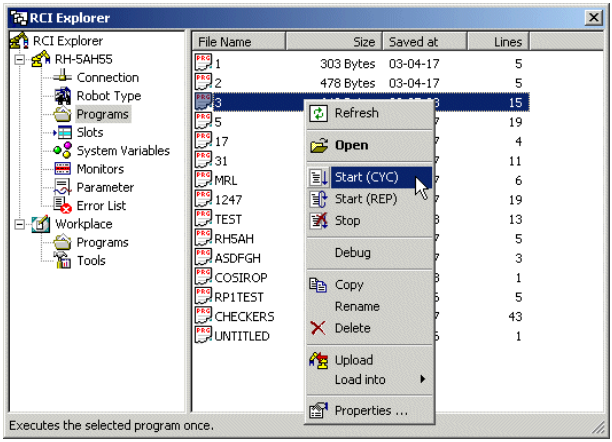
Stop debugging b simply by closing the debugger. Close the debugger by pressing on the icon shown in the following picture:



After testing your program in detail with the online debugger, you can now continue to finally start the program.

How to Start and Stop a Robot Program

It is really easy to start programs using the RCI Explorer. Open the folder **Programs** in the robot folder, click with the right mouse button on the program you want to start and select **Start (CYC)** or **Start (REP)** in the context menu.



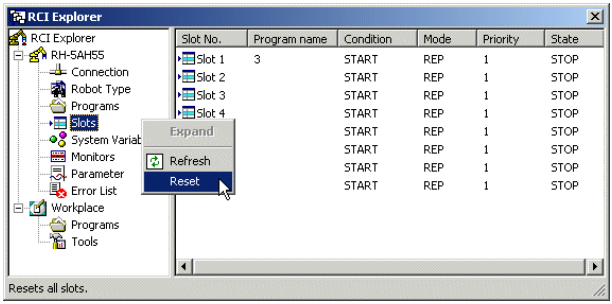
There are two commands to start a robot program. Use the command **Start (CYC)** (1 cycle) to start the program, once. Use the command **Start (REP)** (repeated) to start the program in a continuous loop. With the command **Start (REP)** the program continues until the program is stopped by a user command.

Stop the program with the command Stop from the context menu. It is also possible to stop the program without the RCI Explorer. Use the command **Execute → Program Stop** to stop the program.

If you start a program, the robot controller loads the program into the so-called **slot 1** before it is actually executed. If you want to start the program in a slot different from slot 1, then you have to load the program into this slot, first.

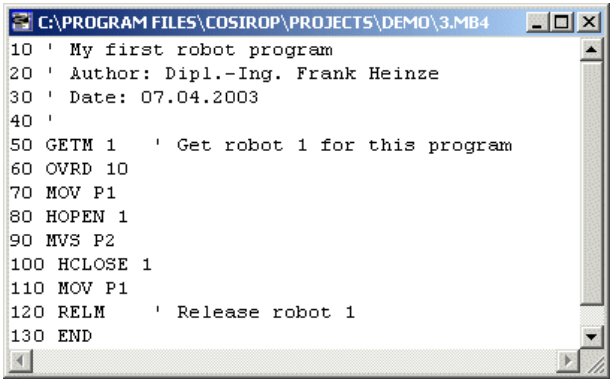
How to Load a Robot Program into a Slot

Robots of the A series (RV-A, RH-AH, RP-AH) have so-called slots. In each of these slots a single robot program can run. Thus several robot programs can actually be executed in parallel. This type of program execution is called multitasking. All other robot types have only one slot for program executions. Thus, they cannot be used for multitasking. A-series types of robots can execute the program in any slot, but slot 1 is the default slot. In order to start a program in a slot different from slot 1, you have to explicitly load the program into the desired slot.
Open the folder **Slots**. If your program **3** is already loaded into a slot (e. g. Slot 1), then reset the slots first. Open the context menu of the folder **Slots** and select **Reset**.

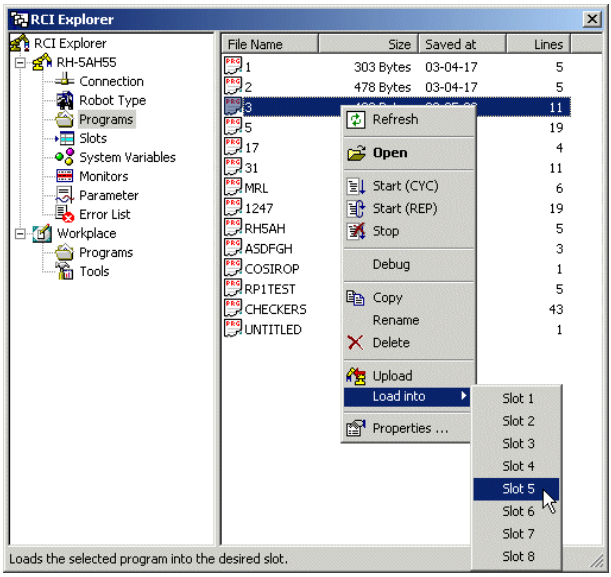


If your parameter settings do not assign any programs to any slot (parameter SLT1 to SLT32), then all the slots should now be empty.

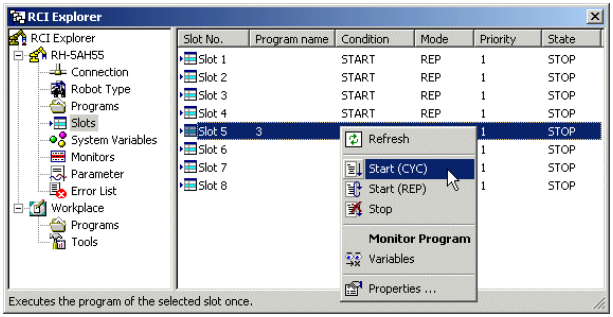
At the moment our program example is not designed to run in a slot other than slot 1. Thus, we have to modify our program first. Add the MELFA-BASIC IC commands **GETM 1** and **RELM**. GETM 1 reserves the robot for the current program. This is necessary, whenever a robot is to be moved by a program that is not running in slot 1. Only one single program can move the robot at any given time. The robot controller assumes that only the program running in slot 1 is moving the robot. Thus, the command GETM 1 is not necessary, if the program is only running in slot 1. But all other programs have to claim the robot first, before they can use it.



Check the syntax of the modified program and download it to the robot. If necessary, check the functionality of the program with the debugger. Load the modified and checked program into slot 5. Click with the right mouse button on the program name in the folder **Programs** in the robot folder. Select **Load to → Slot 5**. The program is loaded into slot 5.



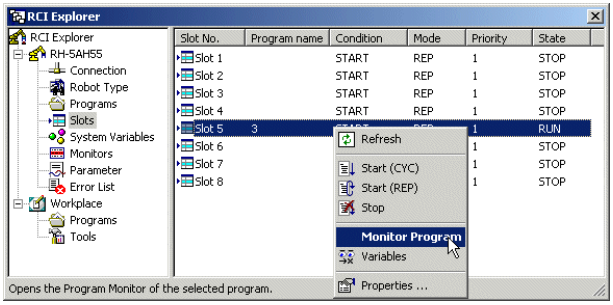
Now, open the folder Slots. Start your program in slot 5 with the command **Start (CYC)** from the context menu.



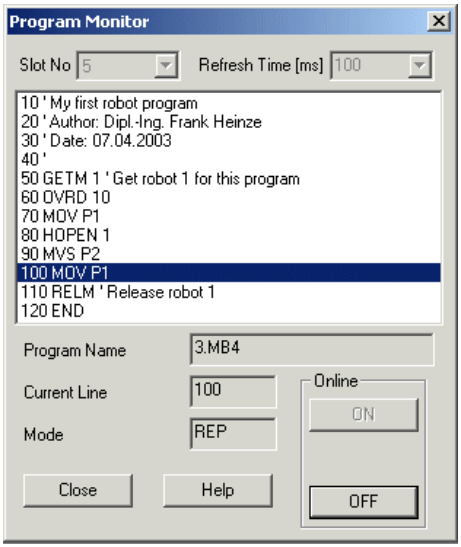
Stop the program execution with the command **Stop** from the context menu. Discover next, how to monitor your running program.

How to Monitor running Robot Programs

Monitor the execution of your programs. Open the context menu of a slot with the right mouse button and select **Monitor Program** . . . .



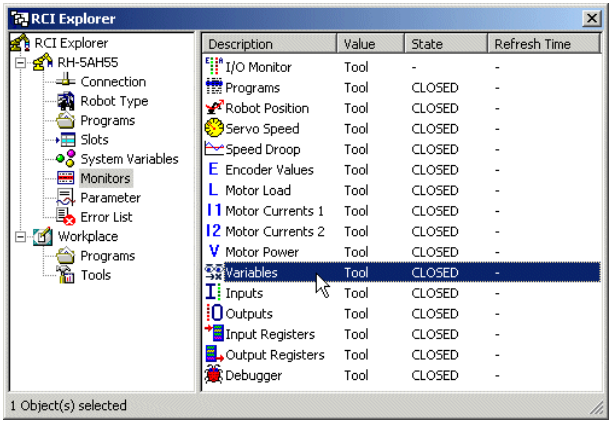
Optionally, double click on a slot to open the Program Monitor of the selected slot:



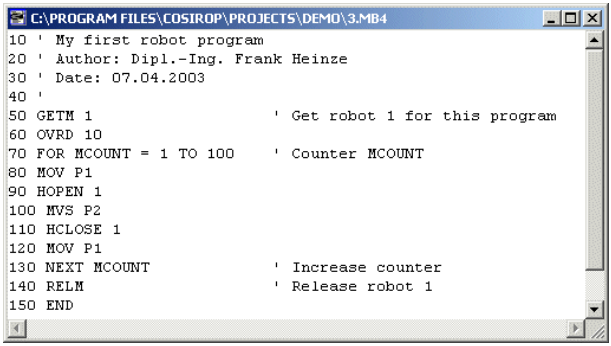
You can actually monitor a lot of different types of robot data. Discover how to monitor variable values.

How to Monitor Variable Values

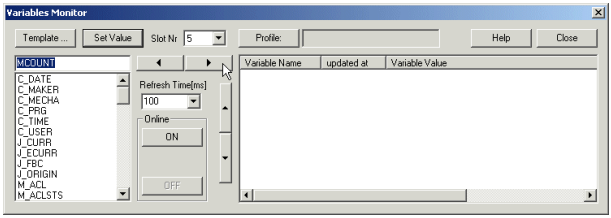
CIROS® Studio offers a wide range of different types of monitors. Use the Variables Monitor to monitor the current values of variables. Open the Variables Monitor by double clicking on **Variables** in the folder **Monitors**.



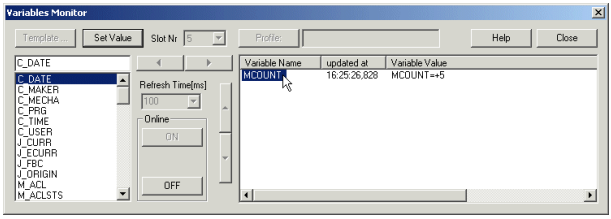
In order to monitor the value of a local variable, add the counter variable MCOUNT to your robot program.



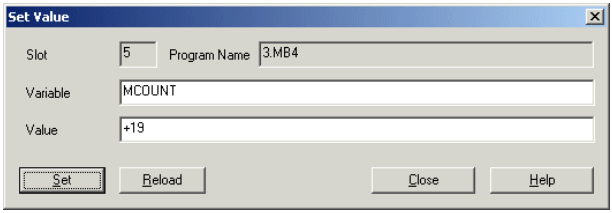
Download the modified program to the robot, load it into slot 5, and start the program. Select slot 5 in the Variables Monitor and add the variable MCOUNT to the list of the monitored variables with the arrow buttons.



Start the monitor by pressing **ON** and observe the changing value of MCOUNT.



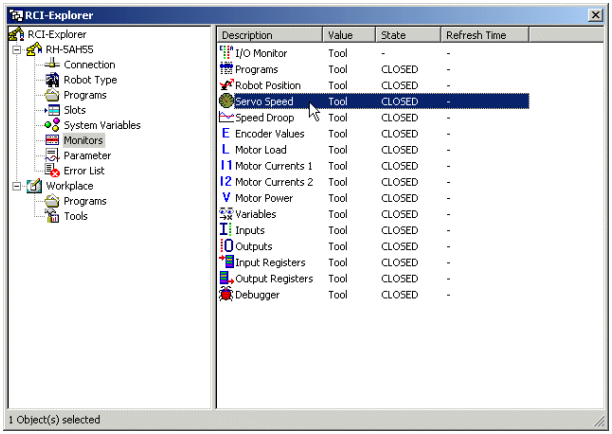
Double click on the variable name to open the dialog **Set Value**. Use it to change the value of the variable MCOUNT. Observe the change in the Variables Monitor.



There are actually many more monitors available. Check the robot state with some of the other robot monitors.

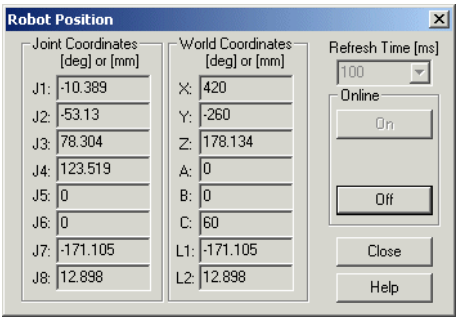
How to Monitor Robot States

Start your program example and again and open some of the monitors. To open a monitor, double click on the monitor in the folder **Monitors**.

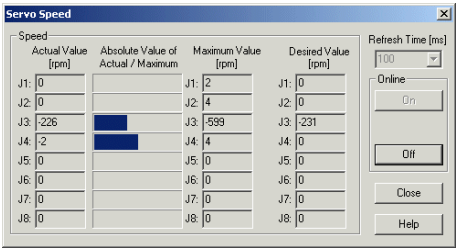


Open the monitor **Robot Position**. Watch the changing joint and world

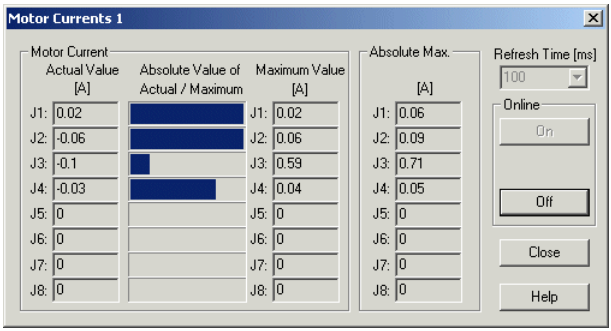
coordinates of the current position as the robot moves.



Open the monitor **Servo Speed**. Observe the changing joint velocities of the robot.



Open the monitor **Motor Current 1**. Observe the changing electric currents of each servomotor.

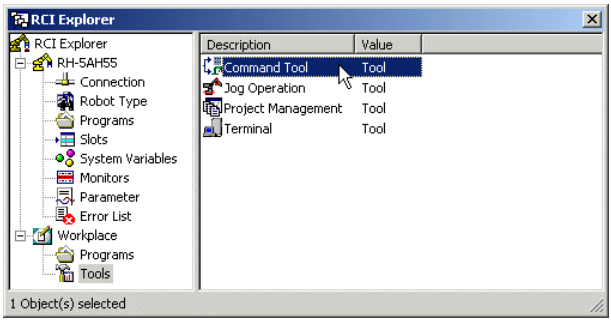


Stop the program. Discover how to send commands directly to the robot.

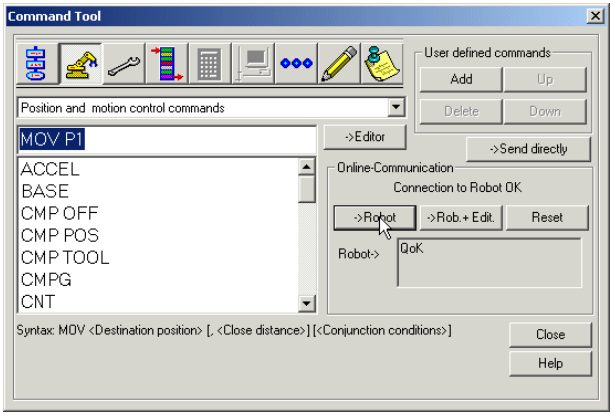
How to Send Commands to the Robot

Read and observe the safety instructions of the SAFETY MANUAL carefully before sending commands to the robot.

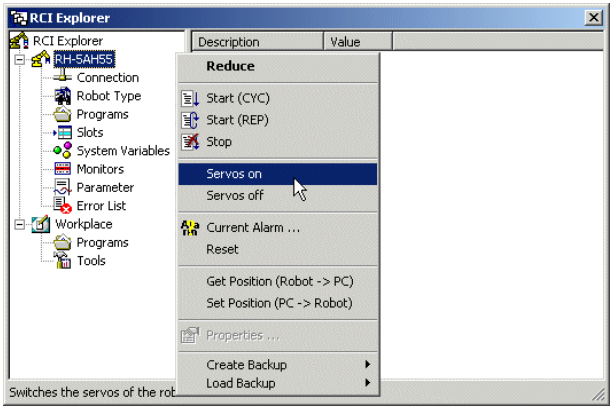
You can interactively send commands to the robot and display the reply of the robot controller with the Command Tool in the **Tools** folder of the RCI Explorer.



Alternatively, you can also open the **Command Tool** with the command **Edit → Command Tool**. Use the **Command Tool** to send commands to the robot as well as to insert robot commands into a robot program.



In order to test the functionality of the Command Tool, reset the slot contents, first. Then, load your example program **3** into slot 1. Switch on the servo motors of the robot with the command **Servos on** from the context menu of the robot.



Send the motion commands **MOV P1** and **MOV P2** to the robot. The robot will move to the desired positions.

The Command Tool offers a structured list of robot commands and displays a short syntax description for each command. Use the Command Tool to send commands to the robot using the button **Robot**. The current command can either be selected from the command list or entered using the keyboard. The Command Tool remembers the last 20 commands, which have been sent to the robot. Choose the class Last commands to display the least recently used commands.

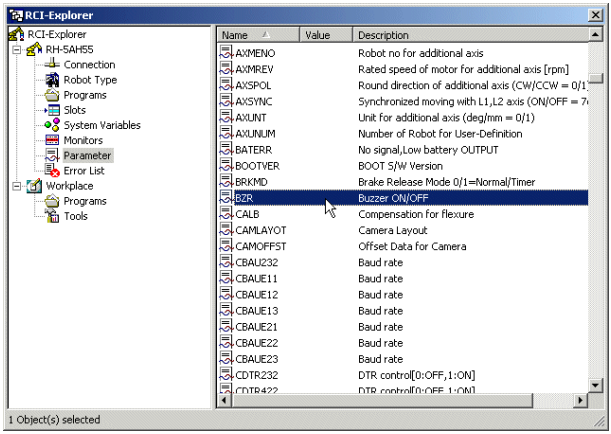
Use the button **Send directly** only, to communicate with a robot program or if you have detailed knowledge about the communication protocol. This command does not add necessary modifications to the command. Instead, the command is sent as it is.
To directly communicate with a running robot program, you can also use

the Terminal.
Furthermore, you can build up your own list of favorite commands by using the button **Add** in the group **User defined** commands. This list can be displayed by choosing the class **User defined** commands. These commands will be stored together with the project.

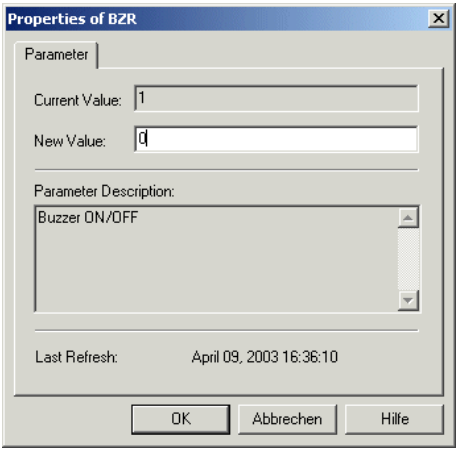
Sometimes, it might be necessary to change internal robot properties. Discover next, how to change robot parameter values.

How to Change Parameter Values

Select the folder **Parameter**. If you are using an A series robot (RV-A, RH-AH, RP-AH), then the list of all available parameters is requested from the robot, first. COSIROP requests this parameter list only once. Afterwards it will be used in any further projects with your robot. To change a parameter, open the Properties dialog by double clicking on the parameter. Double click on the parameter **BZR**.



Enter a new value for the parameter in the Properties dialog. If you are working with an A series robot (RV-A, RH-AH, RP-AH), be sure to switch the key switch on the robot controller to **Teach**. Set the value of "BZR" to 0 and press the **OK** button.



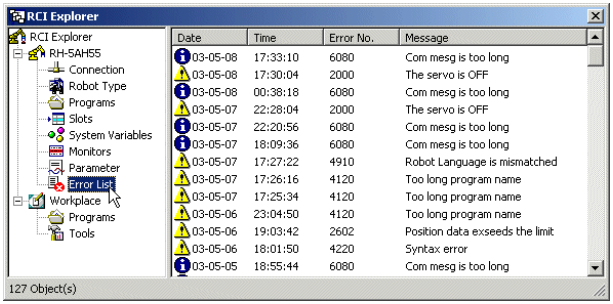
The setting of **BZR** to 0 switches off the acoustic error signal.

Note, that you have to switch the robot controller off and back on again to confirm the changes.

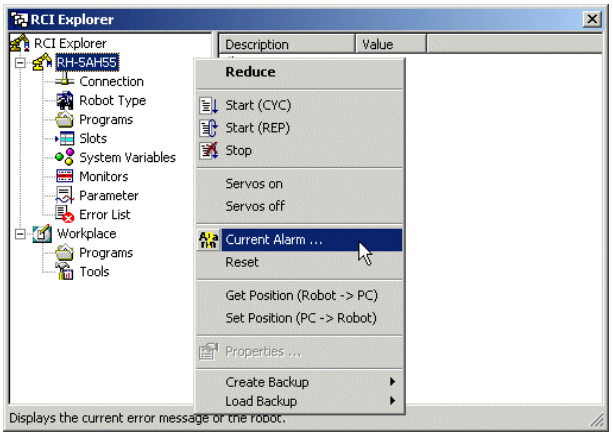
Change the parameter **BZR** back to 1, switch the controller off and on, and switch the key switch back to **Auto(Ext.)**.

How to Check the latest Error Messages

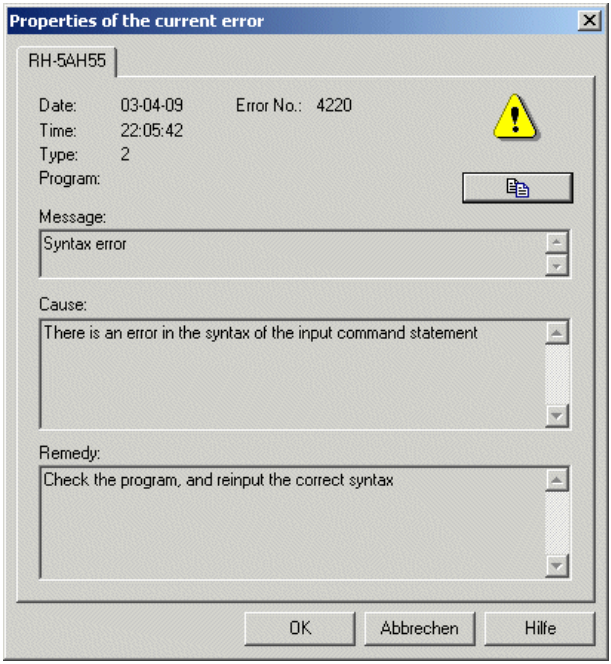
Check the last error messages by selecting the folder **Error List**. COSIROP requests the error list from the robot.



Determine the cause of any current error. Select **Current Alarm** . . from the context menu of the robot.



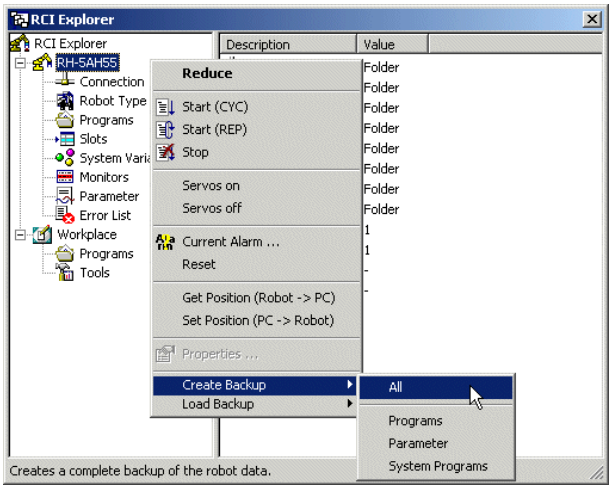
Alternatively, you can determine the cause of any current error with the command **Execute → Current Alarm**.



Closing the dialog box **Properties of the current error** with **OK** resets the current error.

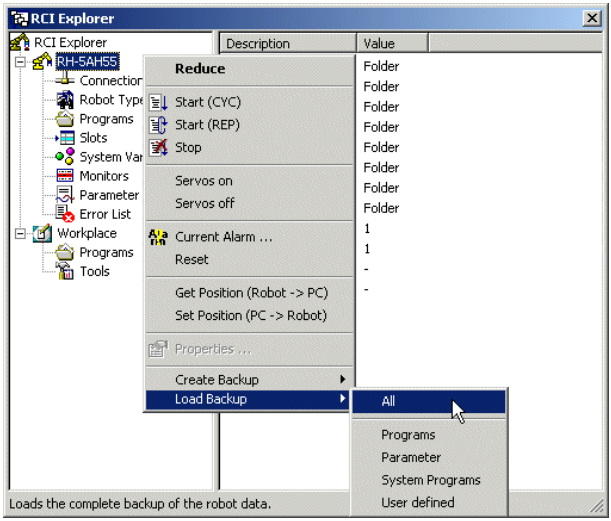
How to Backup your Robot Controller

After you have created a robot project, written programs, and adapted the parameters according to your project, you want to backup your data for future use. Backup the complete data on the robot controller with any robot of the A series (RV-A, RH-AH, RP-AH). Select **Create Backup → All** in the context menu of the robot.



A dialog box pops up. Select a folder for the backup. Be sure to select **an empty folder** for the backup. This backup folder will then contain only files necessary for the backup and none of your PC data will be overwritten during the backup. The backup can take several minutes, especially if there are a lot of programs on the robot controller and you are connected via a slow serial link with the robot.

Store the backup data in a safe place and use it, whenever you need to recover the state of the robot at the backup time. To load a backup, select **Load Backup → All** in the context menu of the robot.



Select the folder that contains your backup data. After your data is successfully recovered, switch the robot controller off and back on again to confirm the changes.

Use the backup folder only for data backups of your robot and for data recovery. Do not try to download single backup files with the program download of CIROS® Studio, if the backup files have the same names as regular CIROS® Studio programs! Under no circumstances try to transfer CIROS® Studio program files with "Load Backup" to the robot. The file formats are different and the robot programs will be destroyed and cannot be recovered.

8. Appendix

8.1 Keyboard Usage

Key	Shortcut
SHIFT+F5	Cascade windows.
SHIFT+F4	Tile windows.
ALT+F4	Quit the program.
F7	Displays the joint values of the robot.
SHIFT+F7	Displays the tool coordinates in world coordinates.
F8	Displays the window Teach-In
F9	Displays the input signals.
SHIFT+F9	Displays the output signals.
CTRL+N	Command File New
CTRL+O	Command File Open
SHIFT+F12	Command File Save
F12	Command File Save as
CTRL+P	Command File Print
CTRL+A	Command Edit Select all
ALT+EINGABE	Command Edit Properties
CTRL+X	Command Edit Cut: Cuts the selected text out of the window and puts it into the clipboard.
CTRL+C	Command Edit Copy: Copies the active window or selected text into the clipboard.
CTRL+V	Command Edit Paste: Pastes the contents of the clipboard into the active window.
CTRL+K	Opens the dialog box for configuration of coordinate systems. Select here which coordinate systems shall be displayed.
CTRL+E	Toggles between Edit Mode and Simulation Mode
CTRL+T	Opens or closes the Model Explorer.

The following shortcuts depend on the type of the activated window. These shortcuts are available in case of an activated work cell window:

Key	Shortcut
CTRL+L	Opens the dialog box for setting the point of view to the work cell.
“+”-KEY	Activates the command zoom-in. It magnifies the view of the work cell.
“-“-KEY	Activates the command zoom-out. It reduces the view of the work cell.
O	Activates the command default settings.
V	Activates the command front view.
U	Activates the command rear view.
A	Activates the command top view.
L	Activates the command left side view.
R	Activates the command right side view.
F	Activates the command full format.
F11	Switches to wireframe representation.
SHIFT+F11	Switches to filled surfaces representation.
CTRL+F11	Switches to flat shaded representation.
SHIFT+CTRL+F11	Switches to smooth shaded representation.
CTRL+D	Opens the rendering dialog box to set the quality and speed of the work cell representation.

These shortcuts are available in case of an activated program window:

Key	Shortcut
CTRL+PAGE UP	Resets the program to the beginning.
CTRL+Q	Continues or starts the current robot program.
CTRL+Y	Continues or starts the current robot program in cyclic mode.
CTRL+S	Stops a running program.

8.2 Abbreviations

Abbreviation	Description
CIROS®	Computer Integrated Robot Simulation
NLP	Native Language Programming
TCP	Tool Center Point