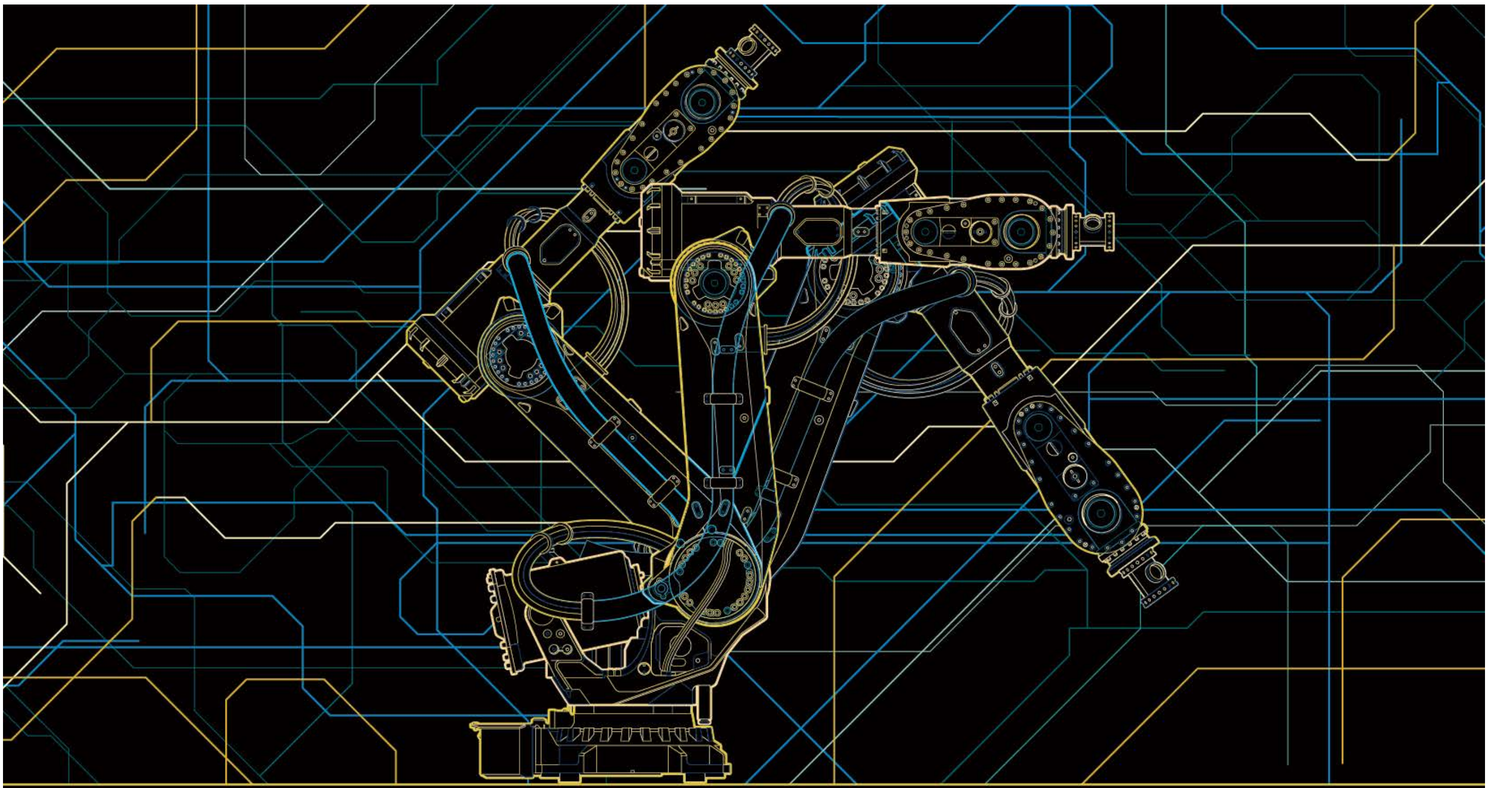


Application manual
RobotWare Cutting

Power and productivity
for a better world™

Trace back information:
Workspace RW 6-0 version a3
Checked in 2014-11-11
Skribenta version 4.1.349

Application manual
RobotWare 6

RobotWare 6.00

Document ID: 3HAC043508-001
Revision: D

ABB Engineering (Shanghai) Ltd.
5 Lane 369, ChuangYe Road
KangQiao Town, PuDong District
SHANGHAI 201319
CHINA

Table of contents

Overview of this manual

About this manual

This manual describes the additional option RW Cutting for IRC5 and provides instructions on how to install and program the system.

Usage

The manual should be used to program a cutting system.

Who should read this manual?

This manual is intended for:

- installation personnel
- robot programmers

Prerequisites

The reader should

- be familiar with industrial robots and their terminology
- be familiar with the RAPID programming language
- be familiar with the system parameters and how to configure them

References

Reference	Document Id
Operating manual - IRC5 with FlexPendant	*3HAC050941-001*

Revision

Revision	Description
-	First edition
A	New functions of laser table and improved function of friction tuning are added.
B	New argument of BevelAngle is added for 2D standard shape cutting instructions. Error numbers are added in error handling part.
C	Two Routines are added. The action to make a changed laser table active is changed.
D	Update all interface screen shots to RobotWare 6.00.

This page is intentionally left blank

1 Safety

Safety of personnel

A robot is heavy and extremely powerful regardless of its speed. A pause or long stop in movement can be followed by a fast hazardous movement. Even if a pattern of movement is predicted, a change in operation can be triggered by an external signal resulting in an unexpected movement.

Therefore, it is important that all safety regulations are followed when entering safeguarded space.

Safety regulations

Before beginning work with the robot, make sure you are familiar with the safety regulations described in the manual *Operating manual - General safety information*.

Safety for laser equipment

Laser presents a danger to the eyes and the human body. Before turning on the laser, ensure all protection mechanisms are working correctly, and also ensure you wear eye protection according to all safety regulations.

This page is intentionally left blank

2 Installation

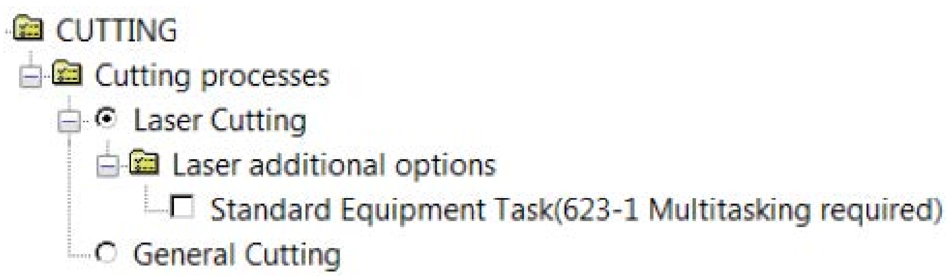
Overview

RobotWare Cutting is a software product primarily developed for laser cutting and other similar cutting methods requiring advanced robot motion performance. The software provides an interactive user interface on the Flexpendant that helps program, configure and operate an ABB Robot in cutting applications. RW Cutting contains motion performance tools, such as automatic friction tuning, iterative learning control, and WristMove. These tools can substantially improve the path accuracy in laser cutting applications.

RW Cutting 6.00 and later version is installed as an additional option with Installation Manager.An virtual use license key is also installed together, which can be used to create a virtual Cutting system in RobotStudio. A list of prerequisites for RW Cutting is listed in Table below.

RW Cutting 6.00 functions	Dependent base options of RW6.00 or later version
Base functions	687-1 Advanced Robot Motion 617-1 FlexPendant Interface 616-1 PC Interface
Laser Cutting- Standard Equipment Task	623-1 Multitasking
Offline programming by RobotStudio Cutting PowerPac	603-1 Absolute Accuracy

RobotWare Cutting is extendable for different cutting processes and methods. RW Cutting 6.00 has three configurations of cutting system. User need to select the appropriate configuration when building the system. See figure below.



Configuration	Features	comments
Laser Cutting - Single task	All laser cutting process data and functions, e.g. speed modulation,Concise laser equipment interfaces. See *Equipment interfaces and classes on page 53* for more information	For simple laser cutting system

Continues on next page

Configuration	Features	comments
Laser Cutting - Standard Equipment Task	Includes all of the above features,plus standard laser equipment interfaces, standard laser equipment operator HMI.Also includes equipment class template files. See *Laser Cutting on page 51* for more information	For advanced laser cutting cell
General Cutting	Cutting process signals and event routines, No equipment class. See *General Cutting Process on page 49* for more information	Extendable base for other cutting processes

Limitations

Note

Note that you cannot install RW Cutting when 633-1 Arc and any Arc sub-options are also installed.

3 Programming

Prerequisites

Before a cutting program is created, the equipment IO and equipment classes (see *Equipment interfaces and classes on page 53*) and additional axes, if any, must be configured.

The TCP and `Wobj` need to be well defined.

Continues on next page

3.1 Programming the cutting program

3.1.1 Program structure

Laser cutting program structure

A laser cutting program consists of a sequence of cutting instructions in order to cut a part. These instructions may include many two-dimensional (2D)holes and several three-dimensional (3D) free form cut paths. The arguments of these cut instructions can be divided to two groups- motion related and process data related. It is best to create a specific routine for each 3D free form cut path, and then make a module file that contains the cut instructions and routines together with their motion related data. Then, save all the process data (the cutdata variables etc.) into a separate module file (ProcessData.mod). Finally, save the calibration data (e.g. `tooldata`, `wobj`) into another module (CalibData.mod). It will make the process data management and cut program reuse easy later.

3.1.2 Cutting instructions

Cutting instruction type

There are two types of cutting instructions, 2D Shape cut instructions and 3D free form cut instructions. See example below of a laser cutting program. It contains one circle hole cut instruction and one 3D free form cut path that includes following positions: `pStart`, `p1`, `p2`, `p3`, `pEnd`.

```
LsCutCircleJ 2, p0, v1000, 10, Arc3_r3_o3, iCW,
     LsTableConf52,\Frcl:=fd2, tLaser\Wobj:=wobjPart;
LsCutLStartpStart,v1000, LsTableConf52, fine, tLaser\Wobj:=wobjPart;
LsCutC p1,p2, v100,LsTableConf52, tLaser\Wobj:=wobjPart;
LsCutL p3, v100, LsTableConf52, tLaser\Wobj:=wobjPart;
LsCutLEndpEnd,v100, LsTableConf52, tLaser\Wobj:=wobjPart;
```

Notice that all the laser cutting instructions begin with `Ls`, and the characters `J`, `L`, and `C` in the instruction name are equal to `MoveJ/MoveL/MoveC`. By the arguments `LsTableConf52`, all information about laser cutting process data can be get.

For shape cut instructions, the values for the arguments ID value and for Friction tune variable data must be unique in program except for global tuning data which can be used in multiple instructions. In above example, `ID = 2`, Friction tune variable data is `fd2`. See *RAPID reference on page 69* for more information.

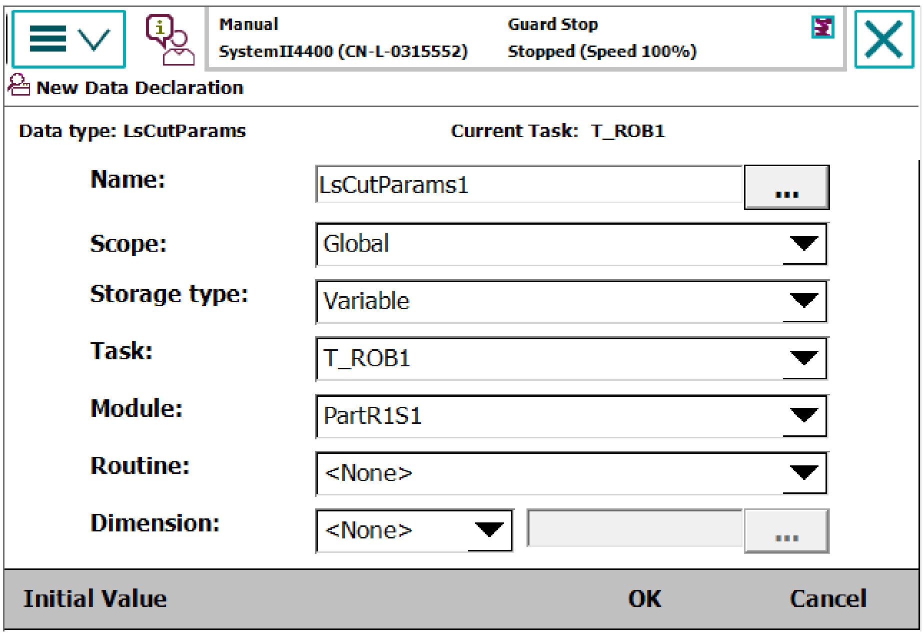
3.1.3 Defining cutting process data

Create cutting parameters

Since 5.15.01, Laser Table is introduced in laser cutting, it is not recommended to create customized cutting process data. However, If users decide to create their own laser cutting parameters, they can use LsCutParams as optional argument of the laser cutting instruction.

If users decide to create general cutting parameters, they can use the cutdata argument for general cutting instructions.

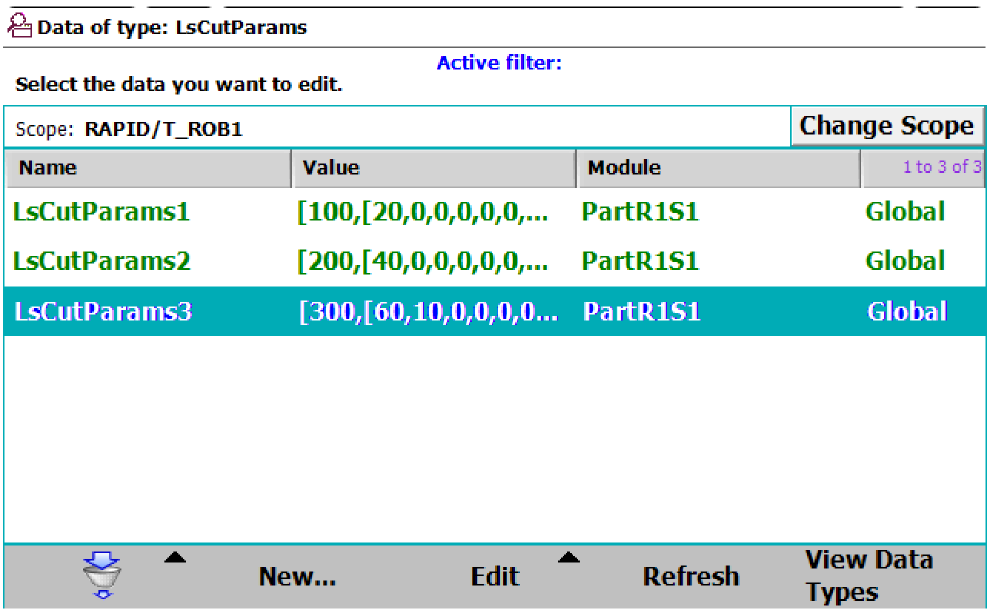
Create cutting parameters

<table>
<tr><th></th><th>Action</th></tr>
<tr><td>1</td><td>From the ABB menu, open the Program Data window.</td></tr>
<tr><td>2</td><td>Select a cutting method type: LsCutParams or cutdata</td></tr>
<tr><td>3</td><td>Tap New.
The data properties are displayed.

</td></tr>
<tr><td>4</td><td>Tap the name button and specify a new name.</td></tr>
<tr><td>5</td><td>If the data needs to be saved in another module, tap the Module drop-down menu and select the desired module. “ProcessData” is recommended.</td></tr>
<tr><td>6</td><td>Give initial Value or edit the value later</td></tr>
<tr><td>7</td><td>Tap OK.</td></tr>
</table>

Continues on next page

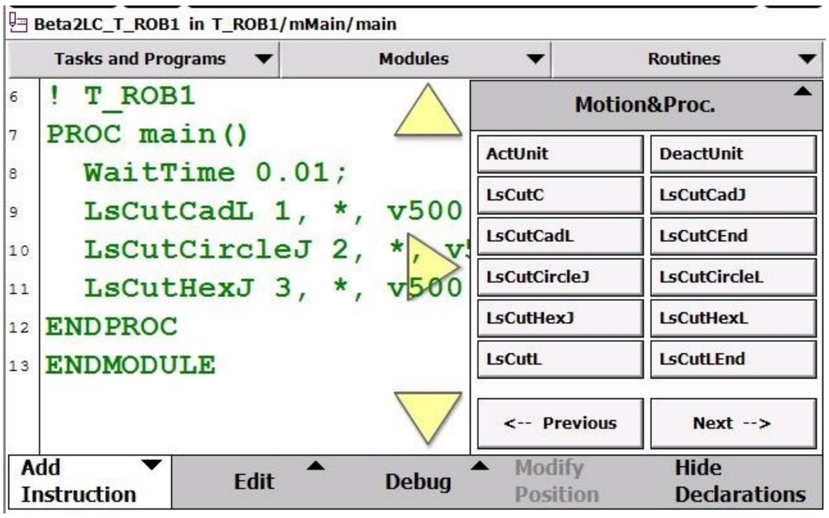
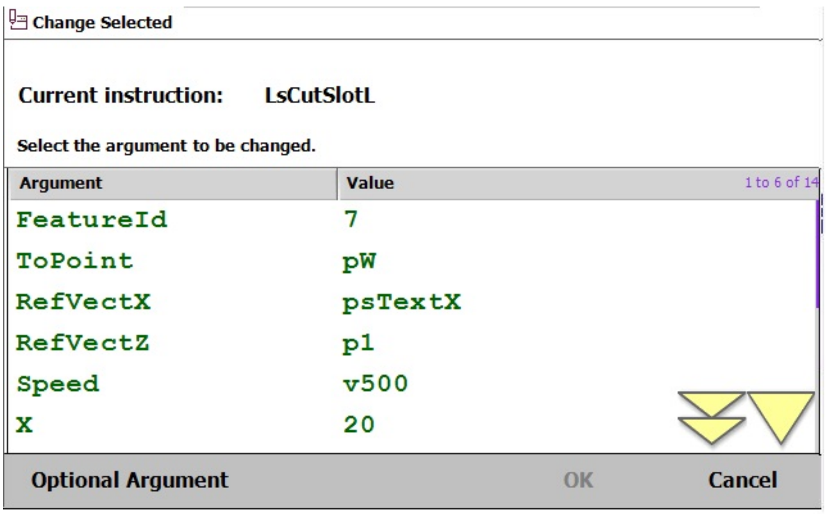
Create cutting process data

It is recommended to create different cutting process data variables for different sizes of the shapes and for every trim cut path. As a result, the process data can be easily tuned for different types of shapes and paths. To see an example of this, see figure below.

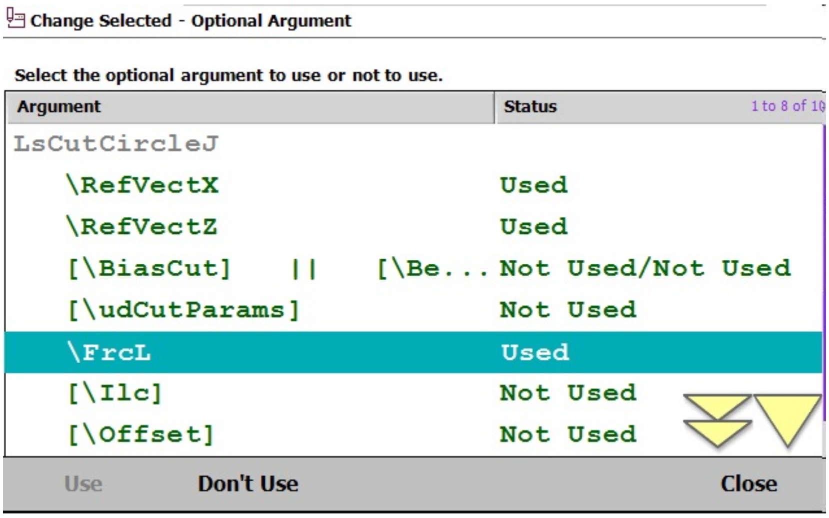


3.1.4 Programming cutting instructions

Procedure

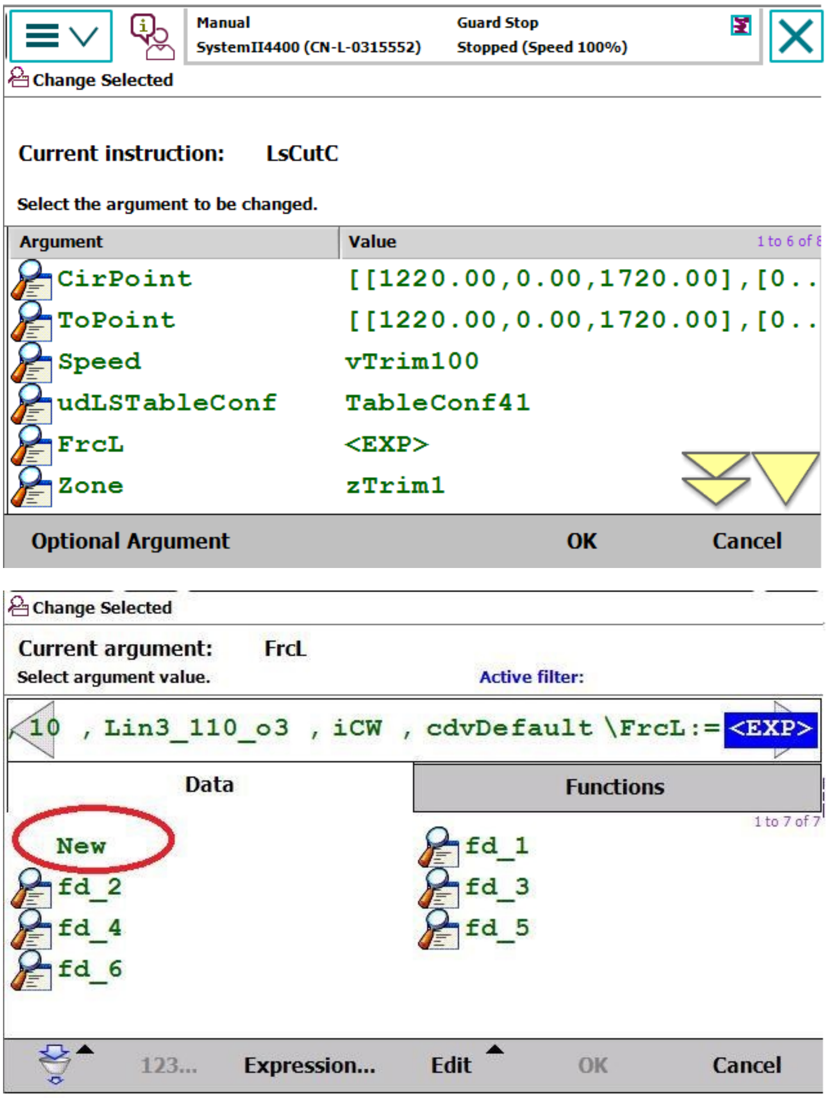
	Action
1	Jog the robot to the desired position. Adjust the orientation of the cutting tool, and ensure it is perpendicular to the cut surface.
2	In the Program Editor, open the instruction pick list. Then select the picklist **Motion & Process**.
3	Select the cutting instructions. The instruction will be added to the program, as illustrated below. The arguments are set according to the last cutting instruction that was programmed.  The instruction is now ready for use.
4	Add any optional argument when it is needed. Tap the instruction twice, a list of argument is displayed 

Continues on next page

	Action
5	To select optional arguments, tap **Optional Argument**, select the argument in the list, then tap **Use**.  Tap **Close**, back to the argument list

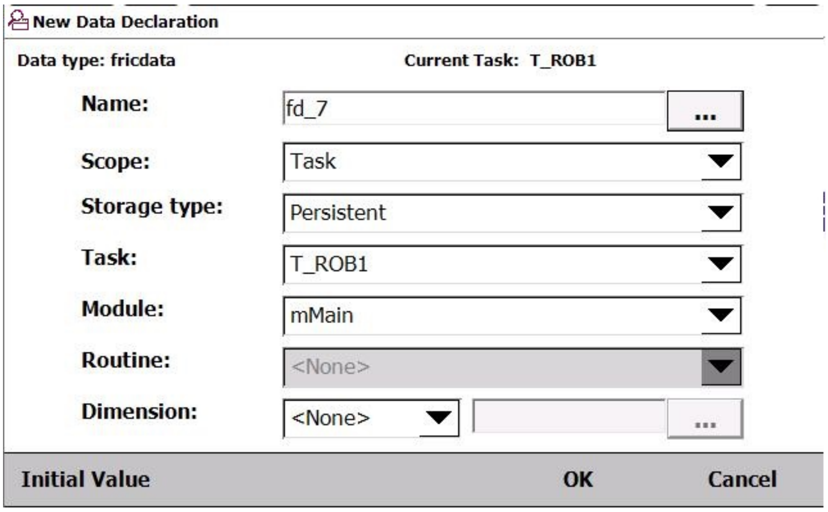
Continues on next page

	Action
6	Tap the new argument, e.g. `FrcL`, and enter a new value for it.



Continues on next page

<table>
<tr><th></th><th>Action</th></tr>
<tr><td>7</td><td>Tap OK, three times in following views, then the new optional argument is added.

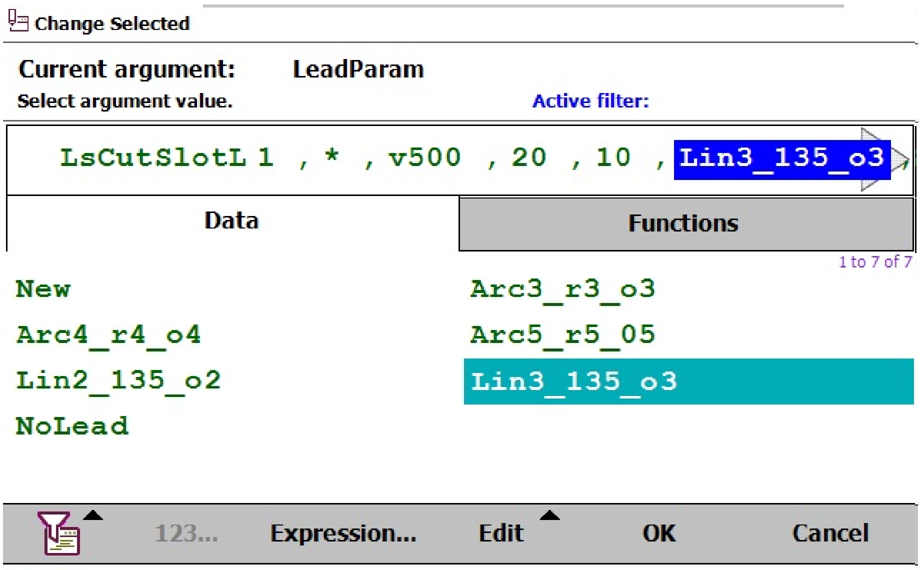

</td></tr>
</table>

Read *RAPID reference on page 69* for more information of cutting arguments and optional arguments.

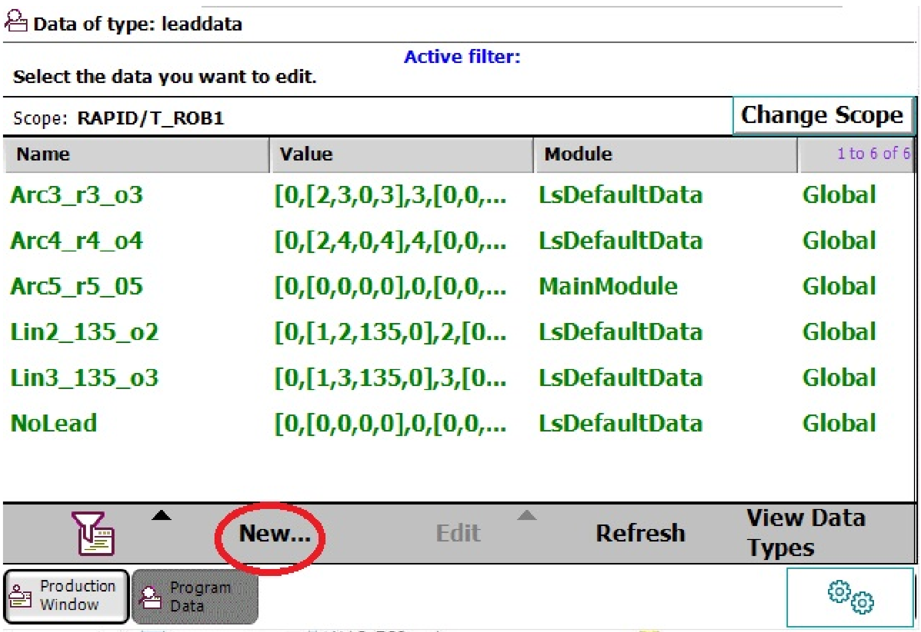
3.1.5 Select and create Lead in / lead out data

Edit leaddata

<table>
<tr><th></th><th>Action</th></tr>
<tr><td>1</td><td>There are default leaddatas for 2D shape cut instruction, user can select the appropriate one from the list.

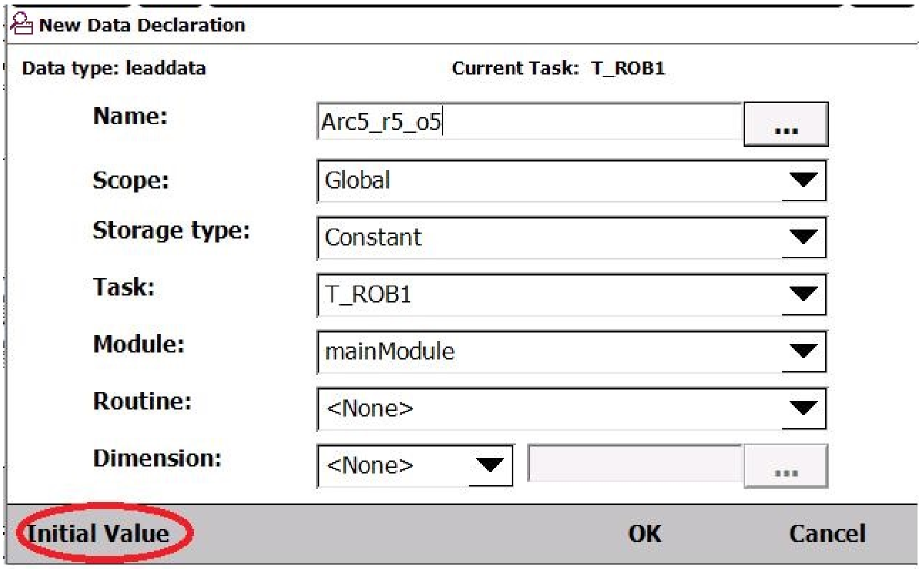


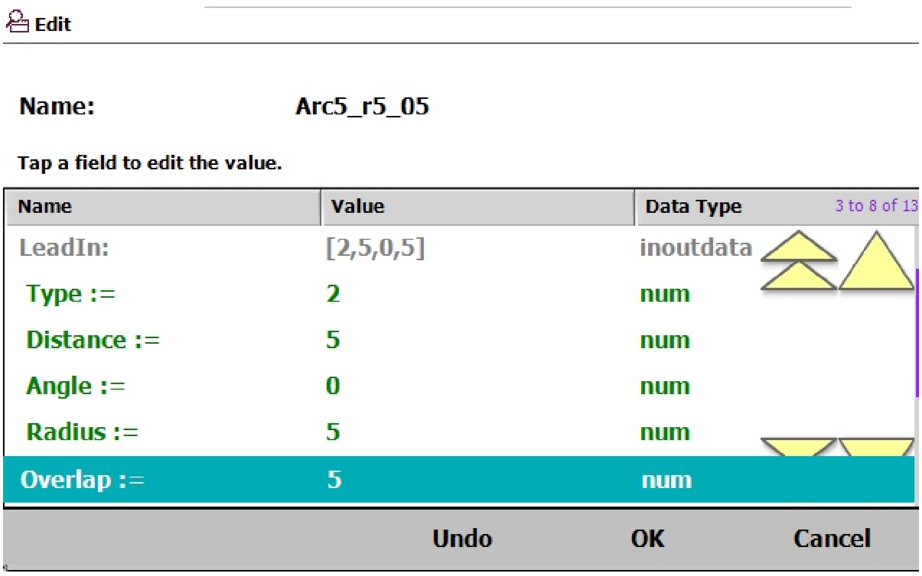
See leaddata on page 148 for more information</td></tr>
<tr><td>2</td><td>If there is no proper <code>leaddata</code>, create new <code>leaddata</code>.


</td></tr>
</table>

Continues on next page

	Action
3	Give the new leaddata correct name to represent its value. e.g. `Arc5_r5_o5` means circular leadin, distance is 5mm, radius is 5mm, overlap is 5mm, no lead out.





Tap **OK**, then the new `leaddata` is now ready for use.

Note

Don't change the value of a lead data.Create new leaddata variables instead.

3.1.6 Select and create LsTableConf data

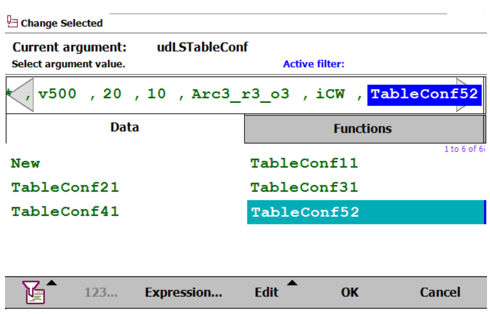
Select LsTableConf data

After create a new laser cutting instruction, a `LsTableConf` argument would be displayed for user's selection.

By using different `LsTableConf` data, users can use different cutting parameter data and piercing data in the laser cutting parameter table according to different process condition and requirements.

There are five pre-defined laser table configurations: `LsTableConf11`, `LsTableConf21`, `LsTableConf31`, `LsTableConf41` and `LsTableConf52`. They are used in different situations.

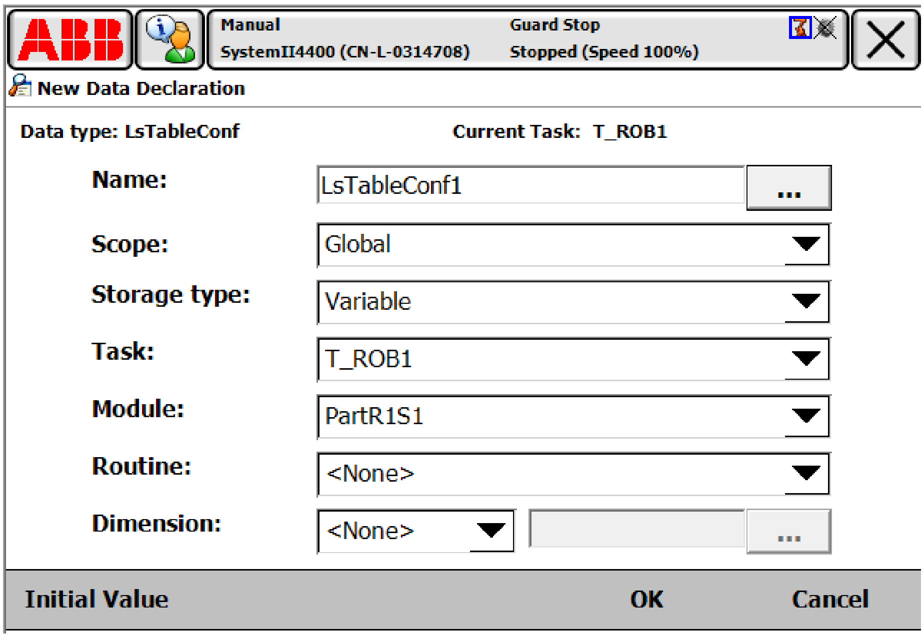
- `LsTableConf11` is supposed to be used in big 2D shape cutting and general piercing.
- `LsTableConf21` is supposed to be used in small 2D shape and general piercing.
- `LsTableConf31` is supposed to be used in fast 3D path along the straight line and general piercing.
- `LsTableConf41` is supposed to be used in slow 3D path around the corner and general piercing.
- `LsTableConf52` is supposed to be used in special cutting and special piercing.

	Procedure	Illustration
1	Users have five default `LsTableConf` to select.	
2	If the existing LsTableConf data met the need of users. Choose the existing LsTableConf data, and click **OK** to confirm.	
3	If existing LsTableCon can not meet users' demand, click **New** to create a new LsTableConf data.	
4	The following procedures are the same with step 3 to step 7 of *Create LsTableConf data on page 25*	

Continues on next page

Create LsTableConf data

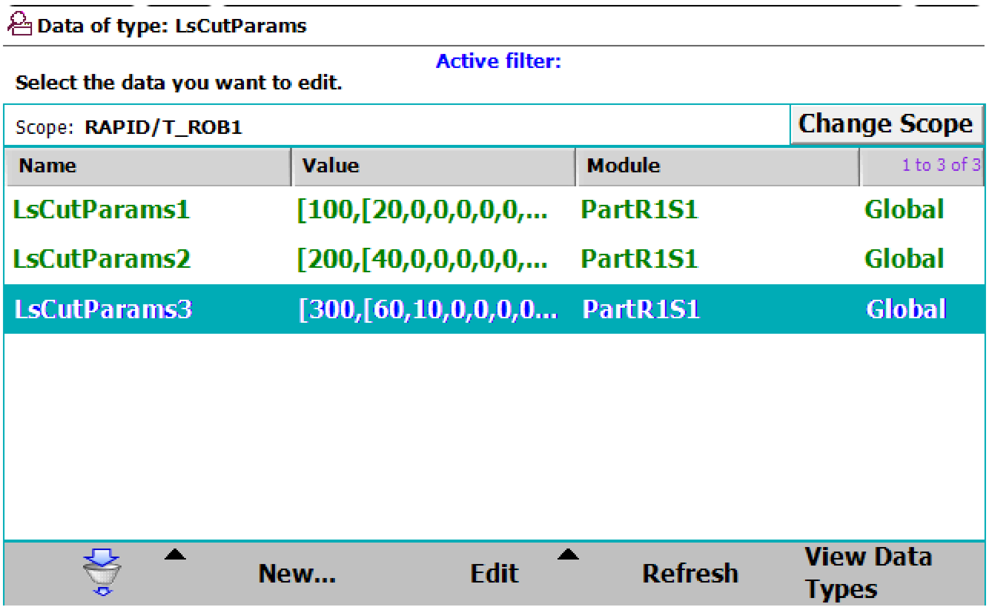
If users want to use a combination of cutting data and piercing parameter data which is not existing pre-defined configuration, users can create a new `LsTableConf` data.

	Action
1	From the ABB menu, open the **Program Data** window.
2	Double click a cutting method type: LsTableConf. A list of existed LsTableConf data would display. By default, there would be 5 LsTableConf data listed and installed in the LSDefaultData module, which can not be modified by users.
3	Tap **New**. The data properties are displayed. 
4	Tap the name button and specify a name.
5	If the data needs to be saved in another module, tap the Module drop-down menu and select the desired module.
6	Click **Initial Value** to give initial value or edit the value later
7	Tap **OK**.

Continues on next page

Modify LsTableConf data

It is recommended to create different `LsTableConf` data variables for different cutting process, as 2D small, 2D normal, 3D small cutting and so on. As a result, the `LsTableConf` data can be easily edited for different types of shapes and paths. To see an example of this, see figure below.



Click **Edit** to modify the selected `LsTableConf` data. Following modification can be done:

Action	Description
Delete	Delete the current selected `LsTableConf` data.
Change Declaration	Change the name of the selected `LsTableConf` data.
Change Value	Change the value of the selected `LsTableConf` data
Copy	Copy the value of the selected `LsTableConf` data and save as another `LsTableConf` data.

3.2 Functions for Cutting when program execution is stopped

3.2.1 Functions in Manual/Auto mode

Functions in Manual mode

Functions (program execution is stopped) in Manual mode:

- Load equipment operator HMI
- Laser Table editing
- Shape list and detail information
- Quick argument editing
- Frame editing
- Process data tuning
- Friction data tuning
- Offset data editing
- State blocking

Functions in Auto mode

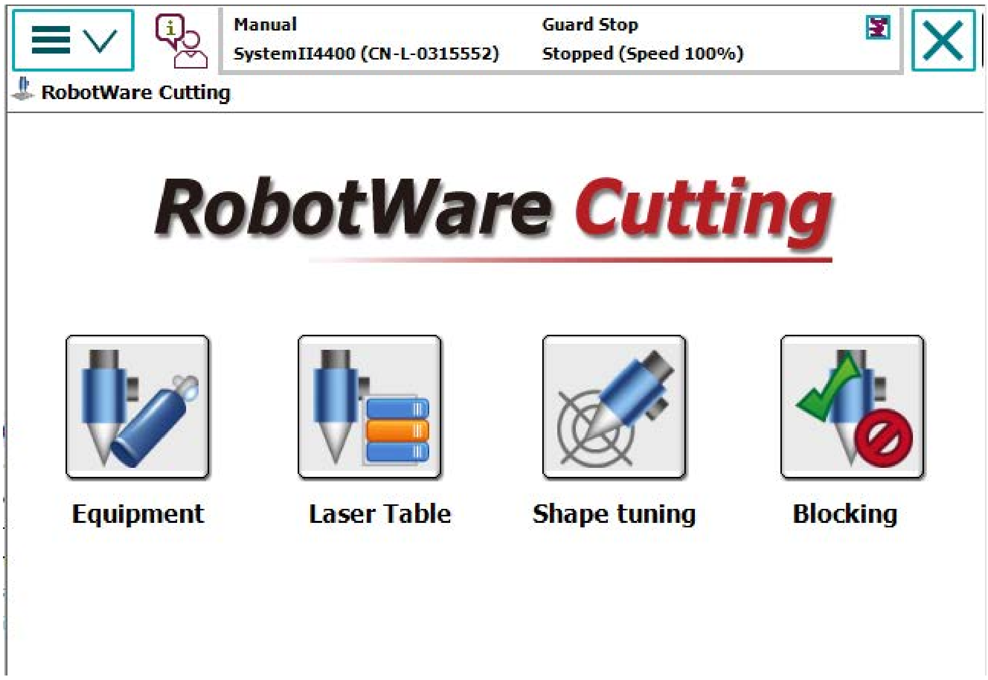
Functions (program execution is stopped) in Auto mode:

- Laser Table editing
- Shape list and detail information
- Process data tuning
- Friction data tuning
- Offset data editing

3.2.2 RobotWare Cutting on Flexpendant

Start RobotWare Cutting

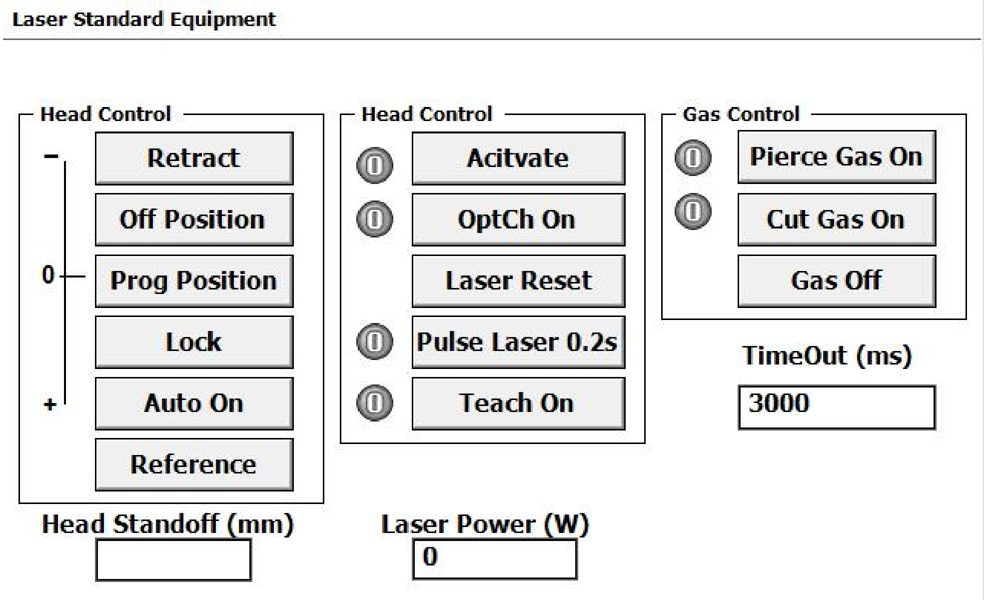
To start **RobotWare Cutting**, tap the **ABB** menu and then tap **RobotWare Cutting**. When **RobotWare Cutting** is loaded, all the cutting functions can be accessed.



3.2.3 Equipment operator HMI

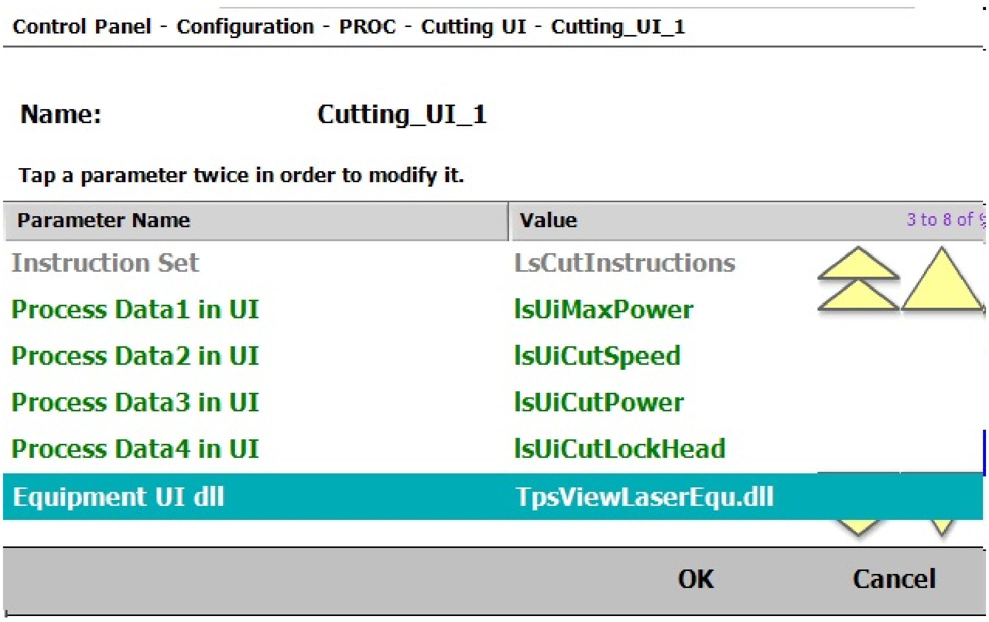
Equipment User Interface

Tap the **Equipment** button, the related equipment HMI that can be accessed is displayed.



Load Cutting UI

To load customized equipment HMI, the **Cutting UI** must be set with **RobotStudio** or **Flexpendant**: **Configuration-Topics-PROC-Cutting UI**, input the **Equipment UI dll** (UIname.dll). See figure below. And the corresponding dll files (UIname.dll and UIname.gtpu.dll) must be deployed under HOME folder of the system. See *System Configuration on page 63* for more information.



Continues on next page

Note

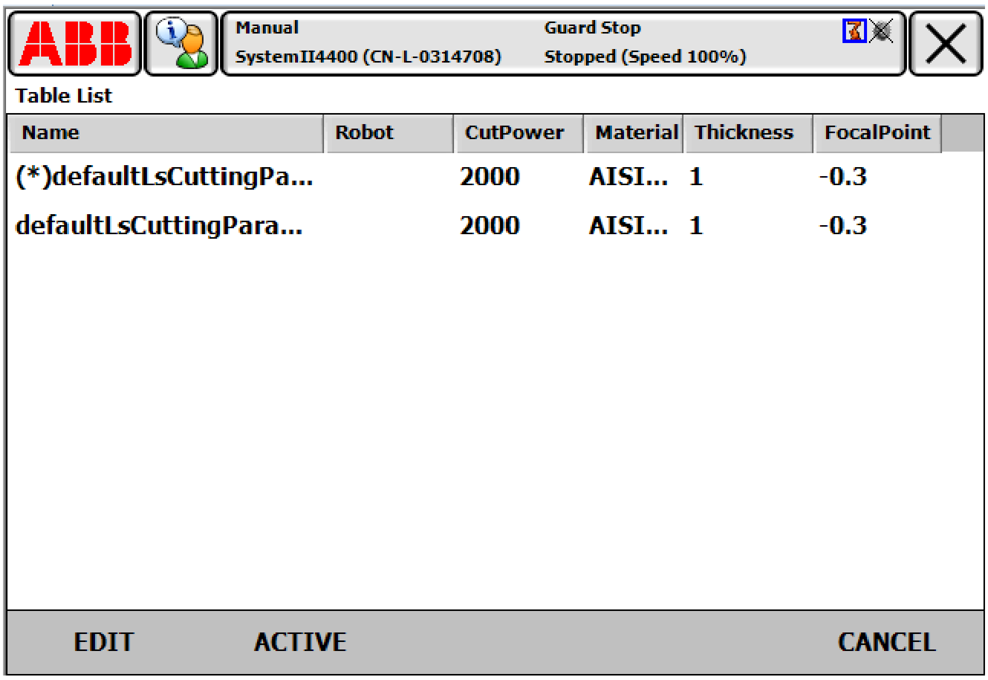
Only the Laser cutting – standard equipment task has default equipment HMI installed, see the figure above. For other configurations such as general cutting and laser cutting single task, user must create their own equipment HMI with RobotStudio ScreenMaker.

3.2.4 Laser Table editing

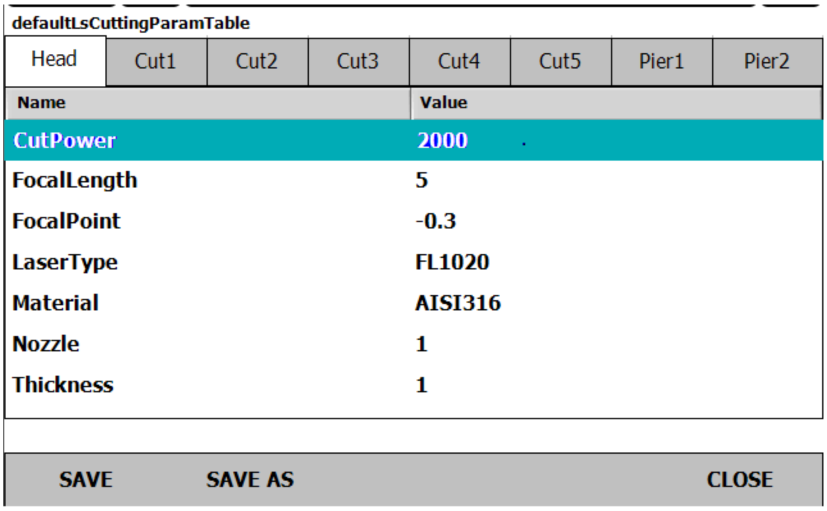
Overview

A laser table consists three parts: one table head, several cutting parameter groups and several piercing parameter groups. See `LsCutHead`, `LsCuttingData`, `LsPiercingData`. It is a Rapid module file stored in HOME/LsCuttingParamTables. It is not a variable in Rapid programs.

In the **RobotWare Cutting** main menu, click the **Laser Table** button, all laser tables files in HOME/LsCuttingParamTables would be listed and displayed. In this view, users can edit laser tables and set one active laser table.



Continues on next page

<table>
<tr><th>Action</th><th>Description/Illustration</th></tr>
<tr><td>Edit</td><td>Click the Edit button, information as for the table head, cutting tables and piercing tables would be displayed.
Due to space limitation, only the first 5 cut data and the first 2 pier data are shown in this view.


Click a field, the field information would be displayed. Users can edit the data directly and click SAVE to the current laser table file. Or save the modified parameter table into a new laser table file by click SAVE AS.
NOTE!
When the robot is running, Laser Table can not be changed and can not be saved.</td></tr>
<tr><td>Active</td><td>Select a cutting parameter table and click ACTIVE. The selected table would be the active parameter table. An asterisk would be displayed before the name of the active parameter table.
NOTE!
Only one cutting parameter table can be active in the same time.</td></tr>
</table>

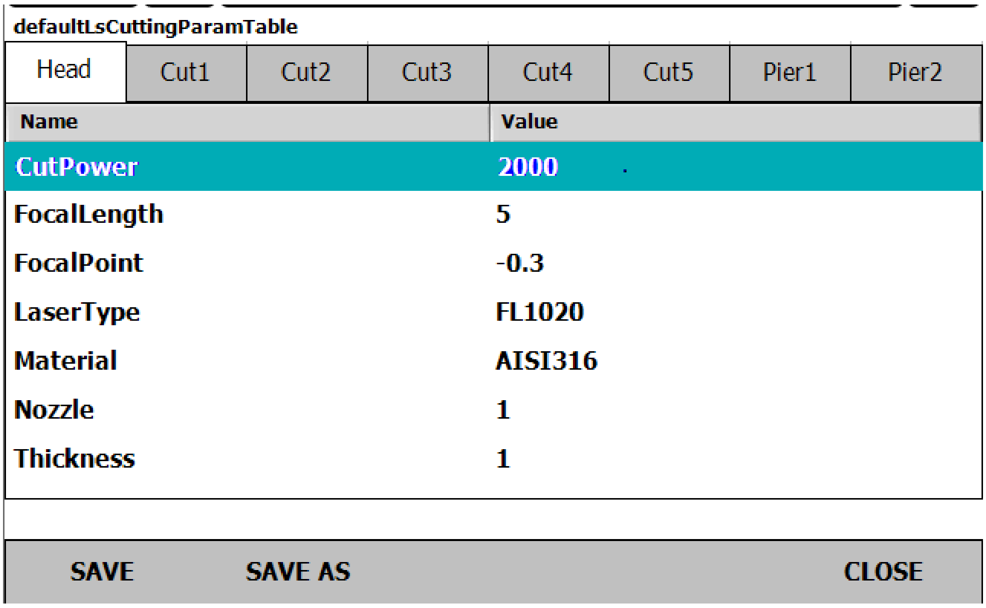
Note

If the active table is changed or modified, it will not take effect automatically until users restart the controller. Another solution is to add "LoadLaserTable" in the head of each routine.

3.2.5 Shape list settings and detail information

Shape list and detail information

Tap the **shape tuning** button, the related shape tuning view can is displayed.



The shape tuning view is divided into two areas: The left area is the shape instruction list of the current selected module, the right area is the detail information of the selected shape instruction.

Note

The shape instruction list is not sorted by the ID value and not list in the execution sequence, but it is just represent the sequence of instruction in the module text.

A list of shape instructions and their descriptions are shown below.

Column	Description	
Shape	6 types of shape with icon: Circle, WristCircle, Rectangle, Slot, Hex, CAD (customized shape)	
FrcL	No Icon	Friction tuning status of related shape instruction: FrcL argument is not used.
	(grey circle)	FrcL argument is used but not tuned (default value 100)
	(green circle)	The friction data has been tuned
	(red circle)	The friction data has conflicting name with other shape instruction. If the friction data is a global friction data, more than one instructions use the same friction data is allowed.
ILC	No Icon	ILC fine tuning status of related shape instruction: ILC argument is not used.
	(grey circle)	ILC argument is used but the result is not tuned.
	(green circle)	ILC argument is used and the related tune result file is in the folder HOME/Ilc_T_ROB1.

Continues on next page

Column	Description	
ID	ID in red color	The feature ID of related shape instruction. ID value or variable name is conflicting with other shapes.

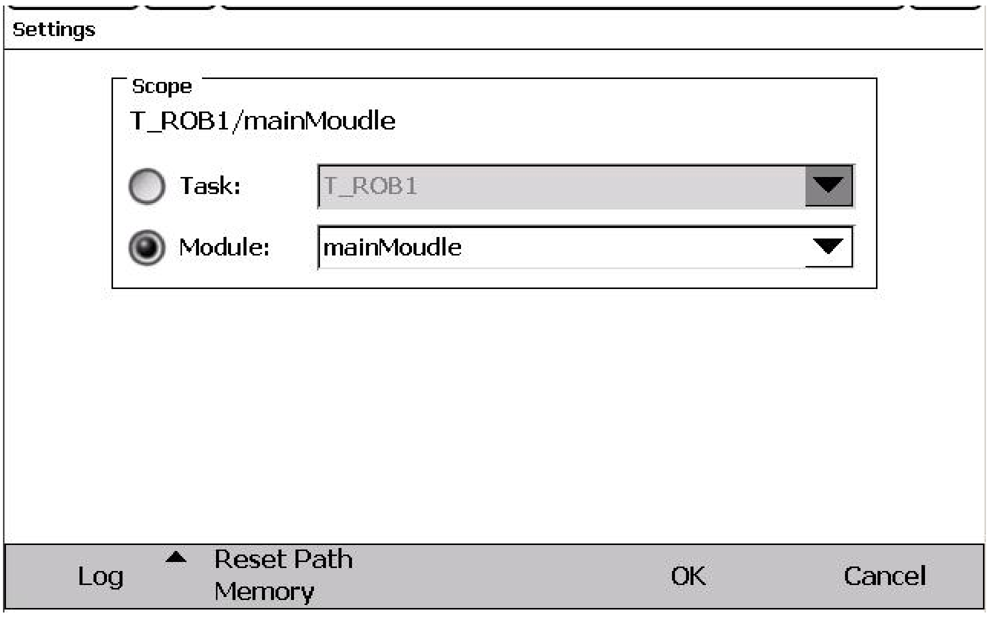
Arguments for shape instructions can be divided into 5 groups. The detail information shows an entire shape instruction in one screen. Descriptions of the arguments for shape instructions are shown below.

Argument group	Description
Shape	The ID value, shape, size, leadParam and cut direction.
Frame	ToPoint, tooldata, wobj, and the reference vectors of x and z axes of the cut frame.
Process	Cut process data variable and four selected components. See *System Configuration on page 63*, cutting UI setting for more information
Tuning	Friction data variable and value, and the ILC tuning result
Offset	Offset variable for minor adjustment based on the measurement result.

Shape list scope setting

To change the shape list scope, Tap the menu item **Settings**.

Select the motion task and the module, and then Tap **OK** to confirm or tap **Cancel** to discard the changes.When the scope is changed or the selected module is modified, the shape list is updated automatically.

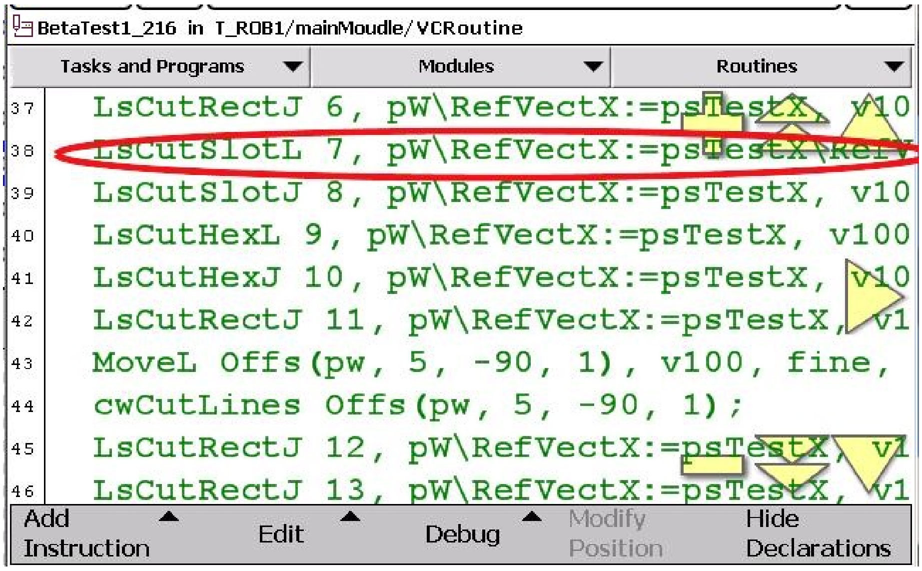
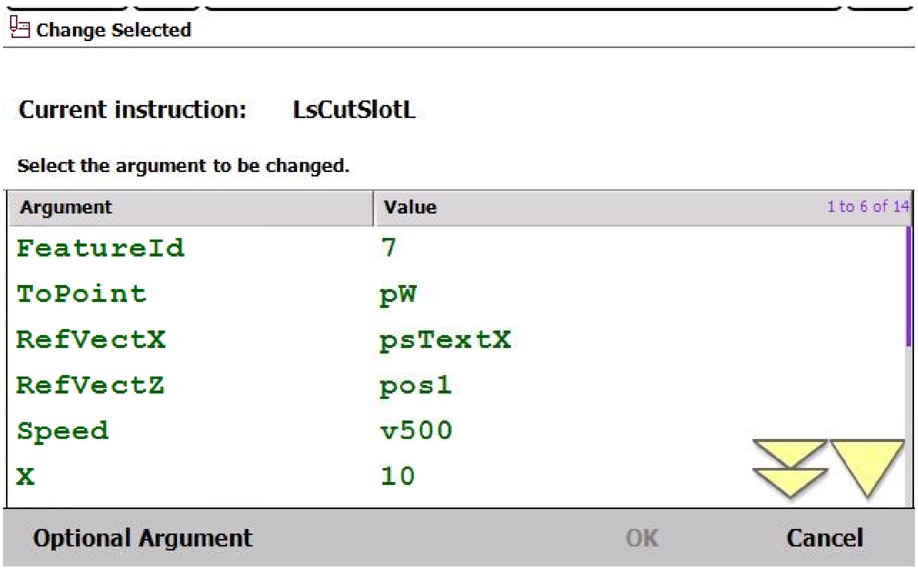


3.2.6 Quick argument editing

Argument editing

Arguments above can also be quickly edited by tap the **...** icon in the **Shape** column.

Arguments edit

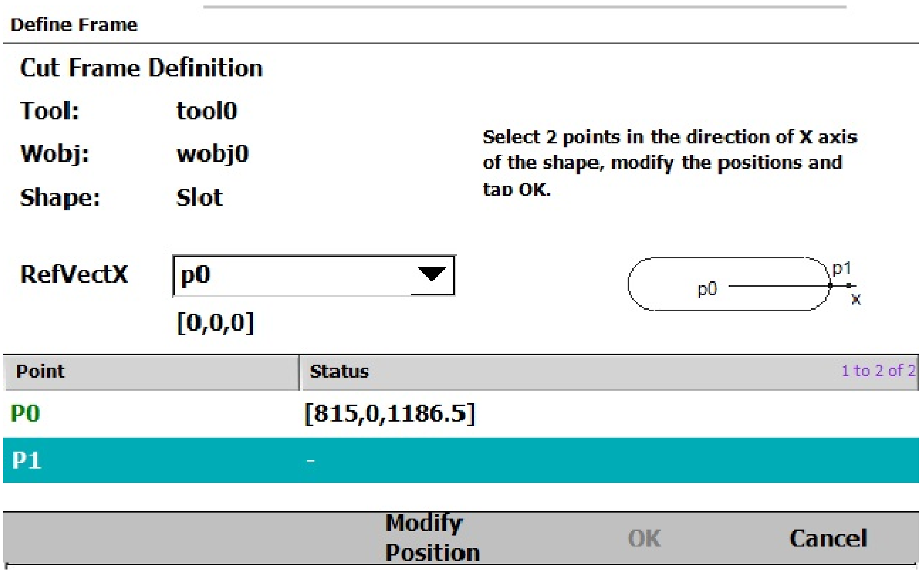
<table>
<tr><th></th><th>Action</th></tr>
<tr><td>1</td><td>Tap “...” button of the shape group.The selected instruction is shown in the program editor.

</td></tr>
<tr><td>2</td><td>To add, edit or delete instructions, select the instruction you want to edit; To add, remove or edit optional arguments of instructions, tap the selected instruction again, the argument list is shown, see figure below. The shape tuning view is updated automatically after your changes are confirmed.

</td></tr>
</table>

3.2.7 Frame editing

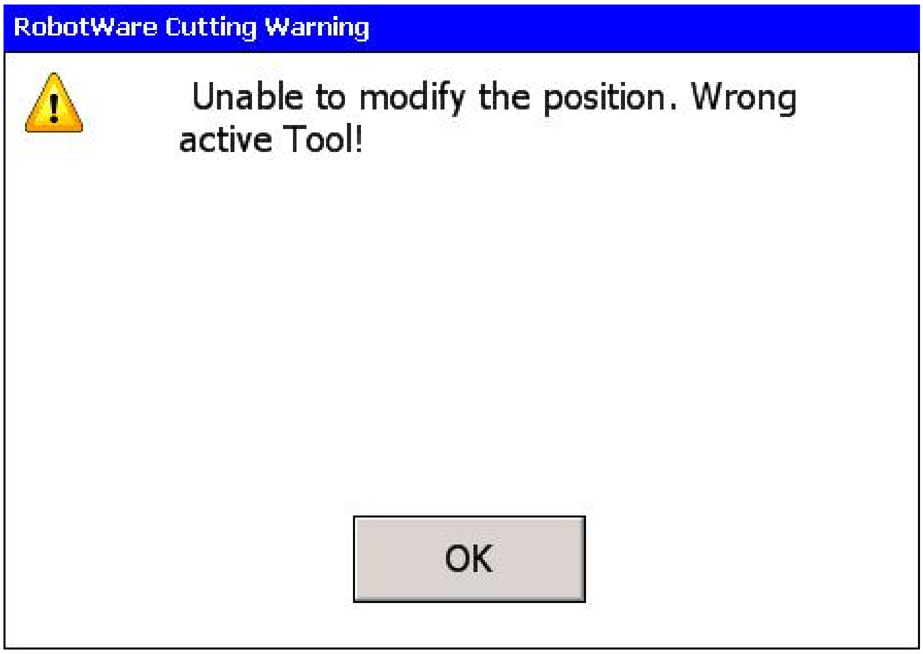
Frame editing

Click the **...** icon in the **Frame** column, frames can be edited in the interface displayed.

Frame editing can be accessed only when the optional arguments `\RefVectX` and `\RefVectY` are used.

	Action
1	Tap “…” button in Frame group. The frame editor is displayed. 

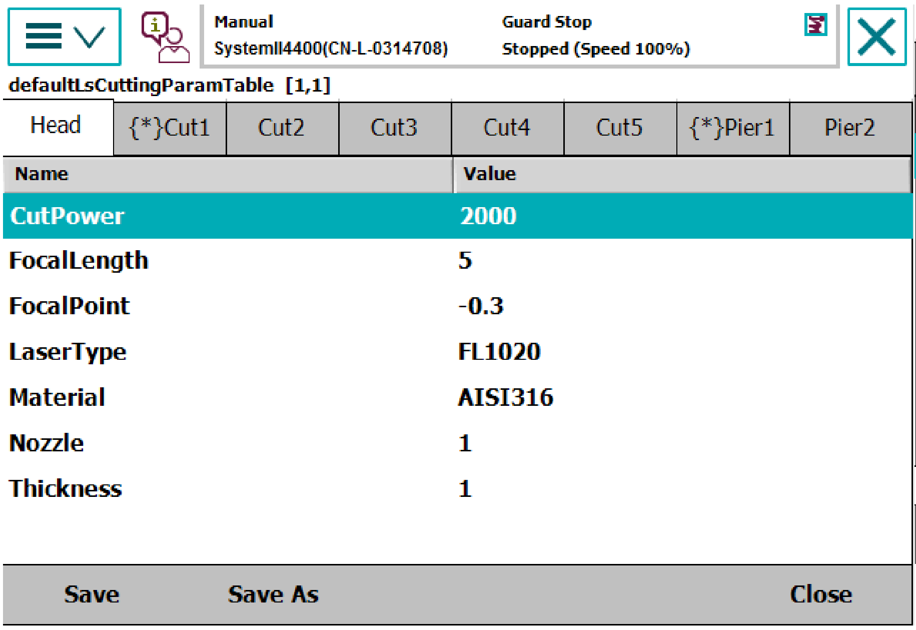
Continues on next page

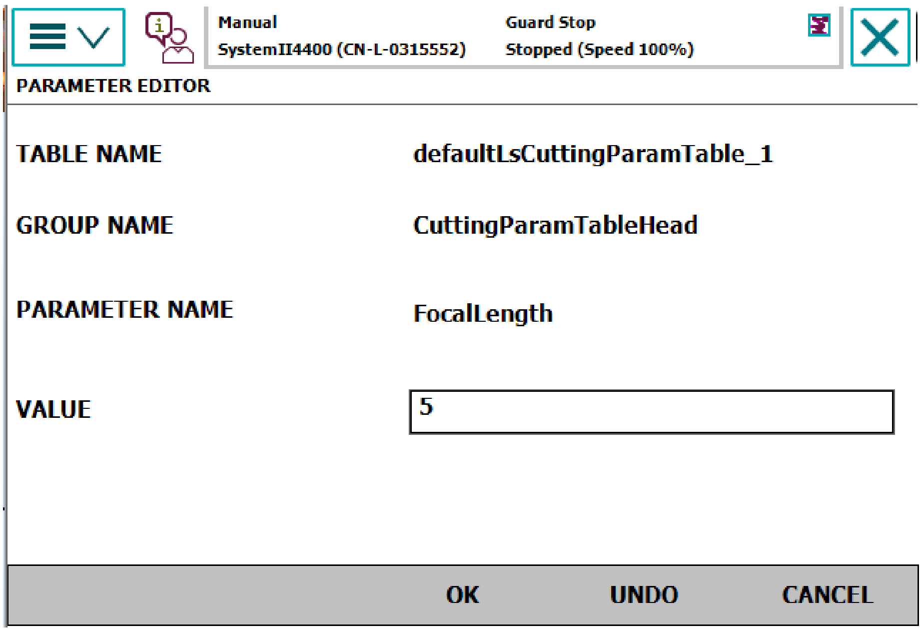
	Action
2	The frame could be defined by two points. By default, the P0 point will be the ToPoint argument that is used in the instruction. Tap **Modify Position** to modify the P0 and P1 points. Note: If the active tool and work object are not the same as the TCP and Wobj that are used in the instruction, the following warning message appears: 
3	Tap **OK** to confirm, or **Cancel** to discard the changes.

3.2.8 Process tuning

Table configuration editing

When a laser cutting parameter table are used in the current cutting instruction, following actions can be done:

<table>
<tr><th></th><th>Action</th></tr>
<tr><td>1</td><td>Tap the ... icon in Process group.The detailed information of the currently active parameter table would be displayed.


Components marked with an asterisk are the currently used</td></tr>
<tr><td>2</td><td>Selected the desired field to edit the value.

</td></tr>
<tr><td>3</td><td>Tap Undo to discard changes.</td></tr>
</table>

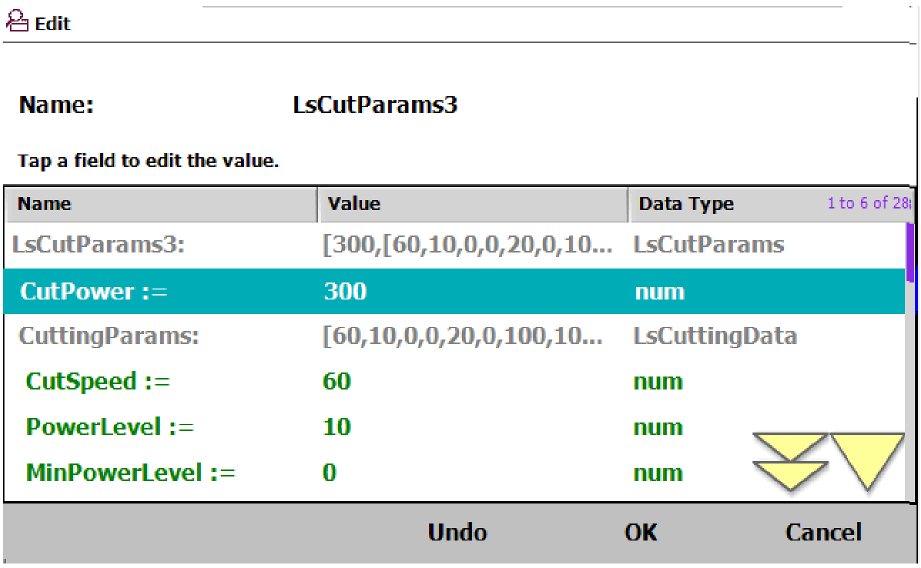
Continues on next page

	Action
4	Tap **OK** to confirm, or tap **Cancel** to discard the changes and return back to the former view.
5	Click **SAVE** to save the modification.

Process data editing

When `LsCutParams` data are used in the current cutting instruction, following actions can be done:

	Action
1	Tap the **...** icon in **Process** group.The LsCutParams is displayed.
2	Selected the desired field to edit the value.



	Action
3	Tap **Undo** to discard changes.
4	Tap **OK** to confirm, or tap **Cancel** to discard the changes. The interface would return back to the laser cutting parameter table edit view.

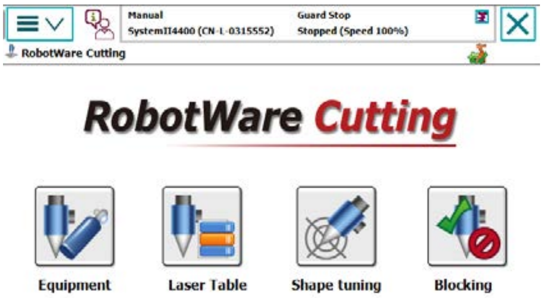
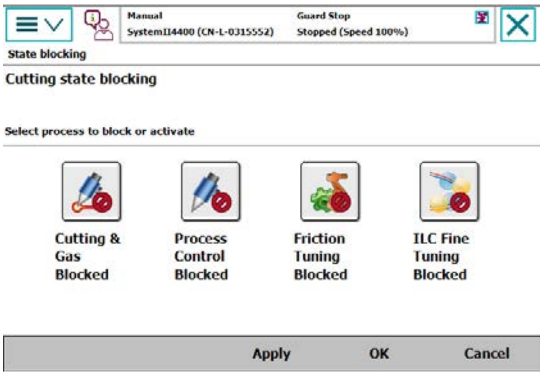
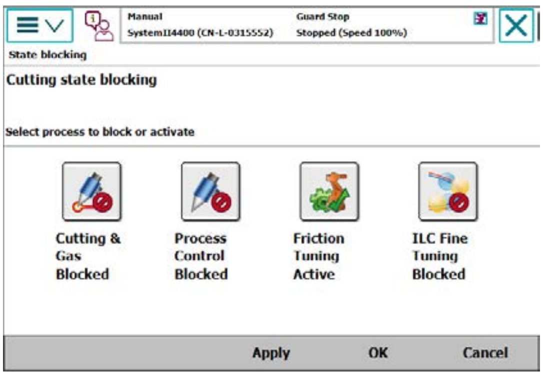
Data masking

If `LsCutParams` is used in the cutting instruction, there are a maximum of four components of cutting parameters that can be shown in the Process group.This depends on how the Cutting_UI is configured. See *System Configuration on page 63* for more information.

3.2.9 Friction data tuning

friction data tuning

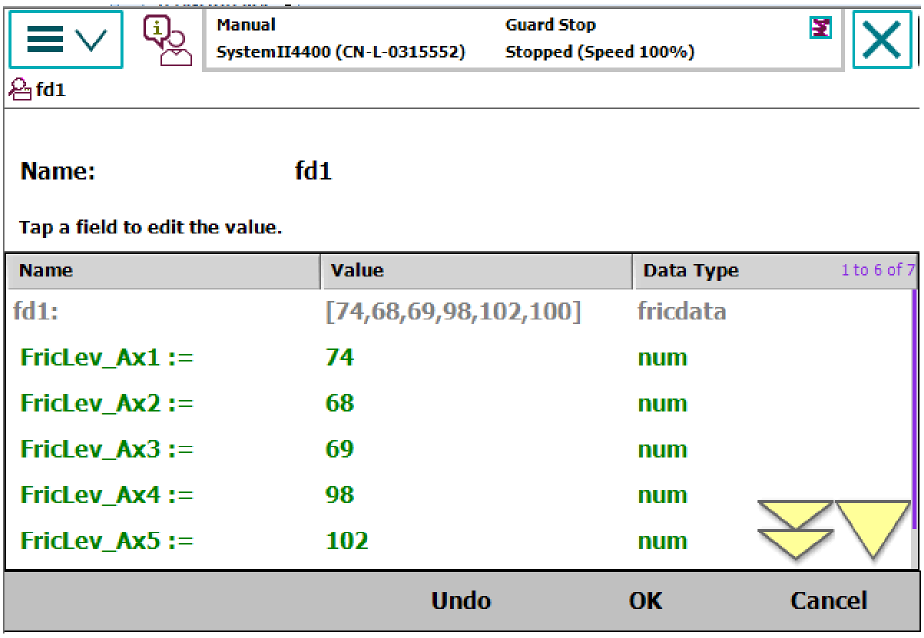
To tune a friction data:

	Description	Illustration
1	In the **RobotWare Cutting** main interface, click **Blocking.**	
2	In the State blocking interface, click the **Friction Tuning Blocked** button to activate friction tuning function. The **Friction Tuning Active** icon would display.	
3	Click **OK** to confirm.	
	In the ABB main menu, click Program Editor to enter the program interface. Click **Debug** and select **PP to Main** in the appeared menu.	
	Click the **Run** button. The friction data would be tuned and updated.	

Continues on next page

General friction data editing

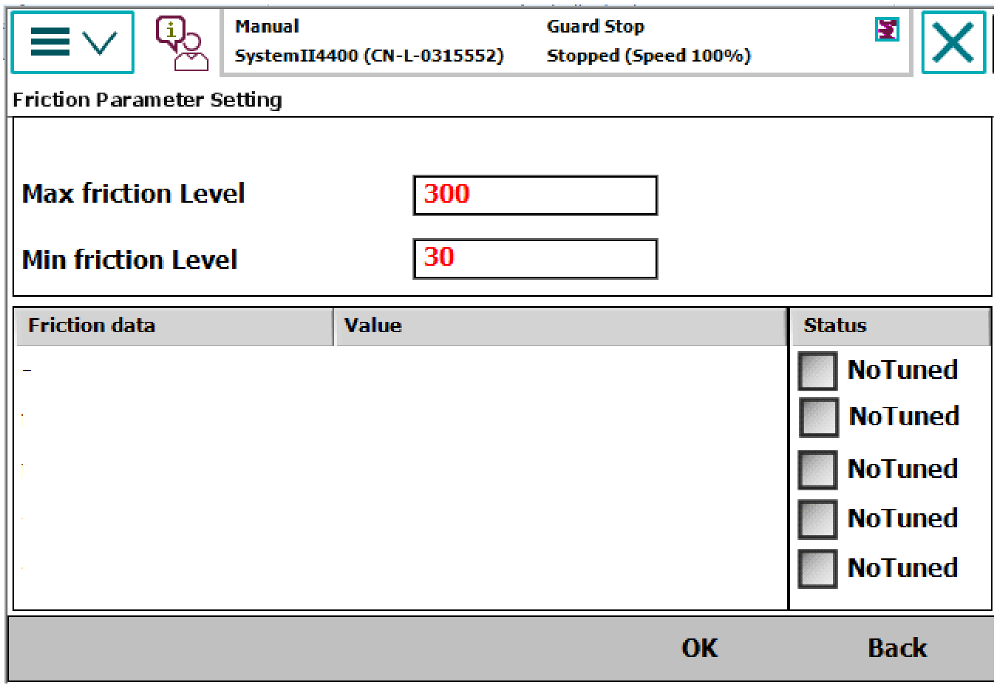
Friction data tuning can be accessed only when the optional argument `\FrcL` is used. Besides auto friction tuning, users can also modify the friction data manually. Please only do it when necessary.

	Action
1	In **Tuning** group, tap the **...** icon.The FrclData can be accessed. 
2	Selected the desired field to edit and change the value.
3	Tap **Undo** to discard changes and keep in current dialog.
4	Tap **OK** to confirm and **Cancel** to discard the changes and back to shape tuning view.

Continues on next page

Advanced friction data editing

Click the **Advanced** button in the main shape tuning view interface, the **Friction Parameter Setting** interface would show up. Users can set the maximum and minimum friction level for all friction data tuning, view and modify the tuning status of global friction data in this interface.

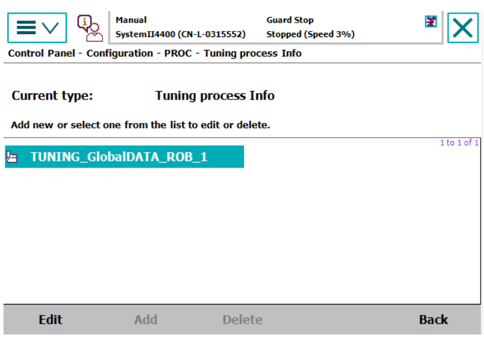


Tip

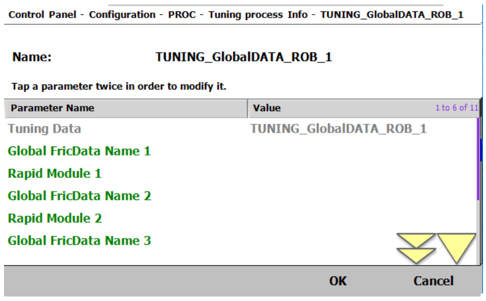
Correctly setting the Max. friction level and Min. friction level can reduce about 20% of the tuning time.

Global friction data configuration in TPU

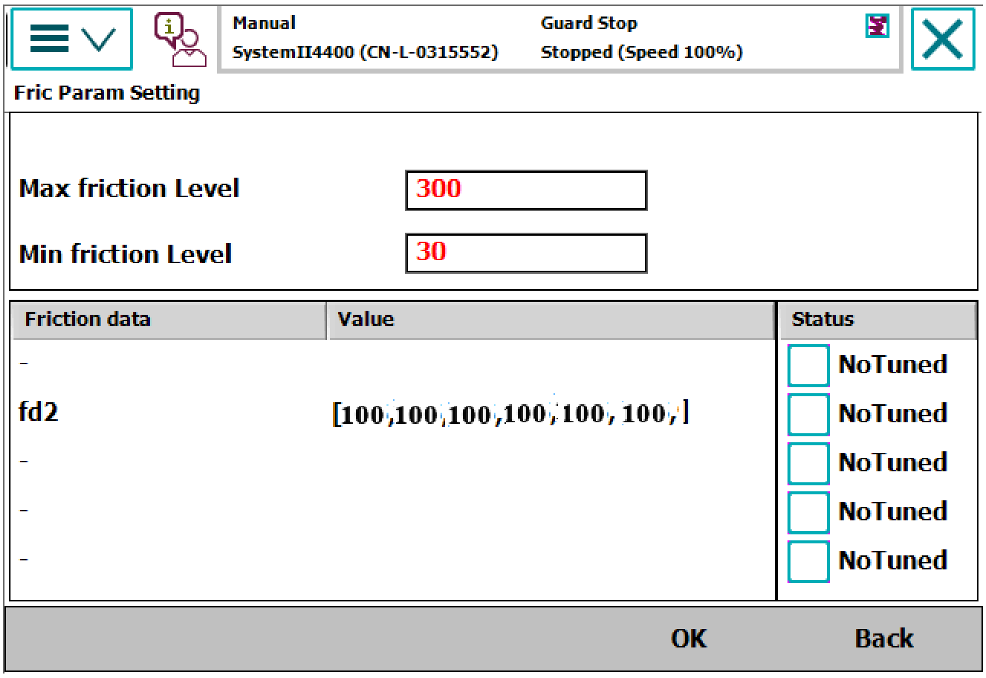
Global friction data can be configured in TPU.

	Description	Illustration
1	Enter **Control Panel** -> **Configuration** -> **PROC**, then enter **Tuning Process Info**.	

Continues on next page

	Description	Illustration
2	Click **Edit** and enter the global friction data edit interface.	
3	Input the global friction data name and the Rapid Module where the data is saved.	
4	Click **OK**.	

```
After setting in the TPU, the value and status of the global
friction data can be shown as below:
```



Users can change the tuning status by ticking or clearing the **NotTuned**/**Tuned** check box which would change its status after ticking or clearing.

When the friction data is **NotTuned**, it would be tuned when run the tuning process. And after tuning, the updated status would be displayed in this interface.

Continues on next page

When the status of the friction data is **Tuned**, then it would not be tuned when run the tuning process.

Note

When a global friction data is used in several 2D shape instructions, it will be only tuned in the first executed instruction. So it is suggested that only use this global friction data in nearby shapes' cutting to ensure the cutting accuracy.

3.2.10 Offset data editing

Offest data editing

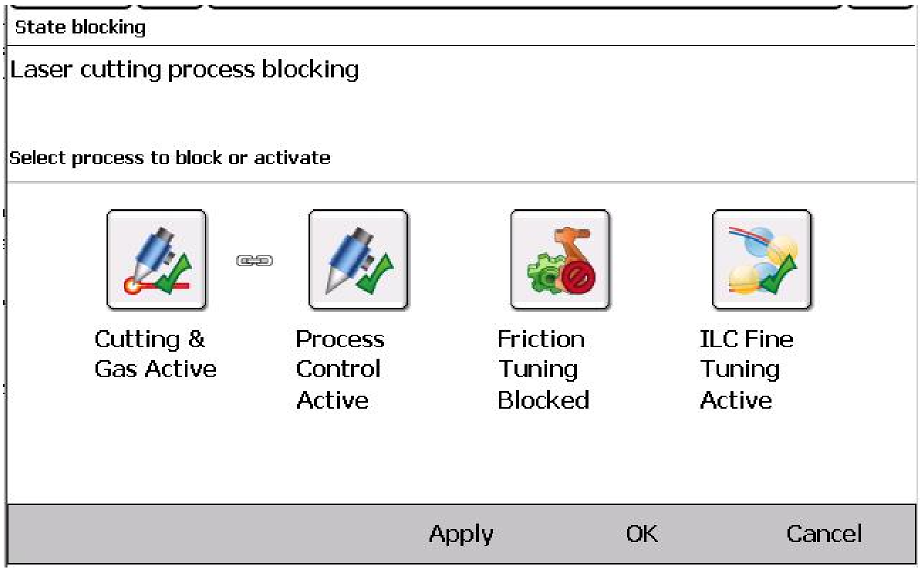
Offset data editing can be accessed only when the optional argument `\offset` is used. Tap the **...** icon in the **Offset** group. The offset data could be accessed.

	Action
1	Tap the ... icon in **Offset** group, the offset data be accessed.
2	Selected the desired field to edit and change the value. See *ShapeOffsetData on page 153* for more information
3	If modifying the offset data in Auto mode, clear the result of shape generation and enable recalculation based on the offset value by tapping “Shape Tuning - Setting - Reset path memory” before put it back to production.

3.2.11 State blocking

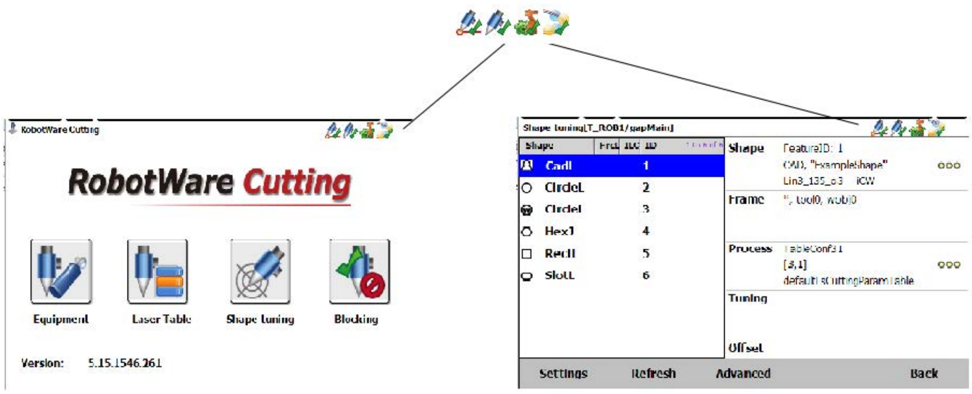
State blocking activate

State blocking lets the possibility activate/deactivate the state of Cutting&Gas, process control, Friction tuning and ILC fine tuning. This may be useful during programming or testing phases.

	Action
1	Tap **Blocking**. The following dialog box is displayed: 
2	Tap a process icon to block or activate it.
3	Tap **Apply** to confirm the changes and stay in the current dialog box.
4	Tap **OK** to confirm or **Cancel** to discard changes and back to main view.

Activiated process display

The parts of the process that have been activated are shown on the top border in the main window and the shape tuning window.



Continues on next page

State	Description
Cutting & Gas 	To activate the all the process control, including cutting power source, gas and tool. Executing cut instructions will cut the workpiece. It is used in production. It is locked with Process Control when it is active.
Process Control 	To only activate part of the process control but not to cut the workpiece. E.g. enable the cut head control, but not activate the laser power source and gas. It is used to verify the path and IO signals during dry run the program.
Friction Tuning 	To activate the advance shape tuning. Then shape cut instructions will run the tuning motion if the /FrcL argument is given.
ILC Fine Tuning Active/Continue 	To activate the iterative learning control as fine tuning of the shape. Then shape cut instructions will run the tuning motion if the /Ilc argument is used. ILC Active: start the calculation from zero. ILC Continue: continue the calculation based on the result of last time.

See *SetCutWareState on page 128* for more information.

3.3 Functions during program execution

Motion pointer tracking

Functions (program execution) in Manual & Auto mode:

- Motion pointer tracking

If the shape tuning view is open before the program is running, the shape list will highlight the shape instruction that the robot is executing.

4 User guide

4.1 General Cutting Process

Cutting process

The general cutting process can be divided into four phases: preparing cutting, cutting start (piercing), cutting (moving along the tool path), cutting end. The entire process should be synchronized with the motion of the robot to ensure cutting the work piece correctly.

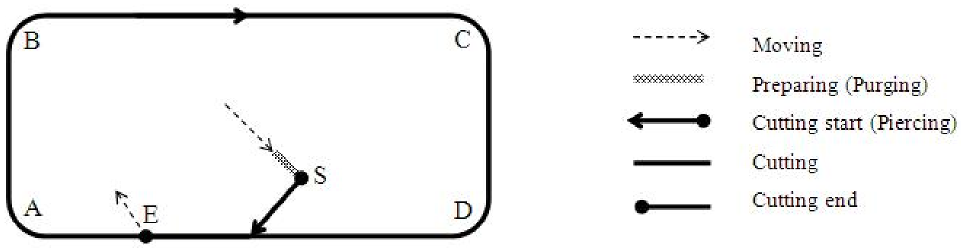
Process signals

RW Cutting defines internal signals to synchronize cutting process and robot motion. Those internal signals are listed in below.

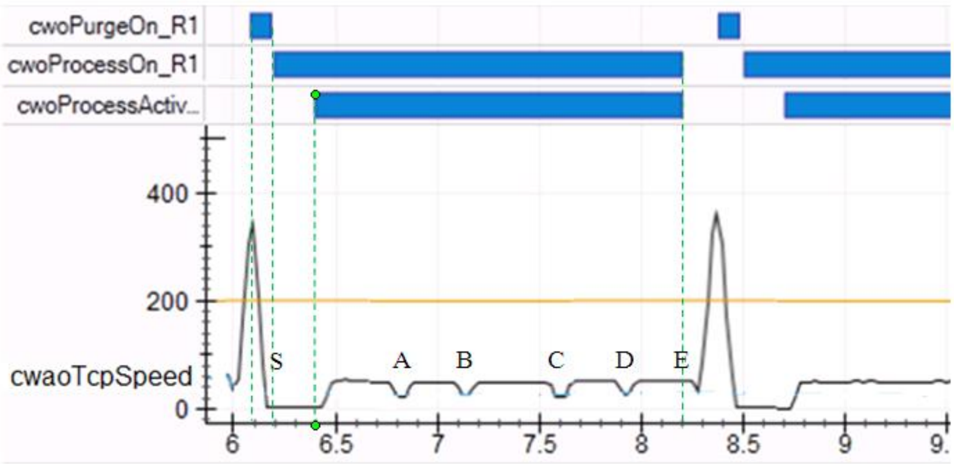
Robot motion	Signal name	Trigger	Phase of cutting process
Approaching the start point	cwoPurgeOn		Preparing cutting
Start_Fly or Stop at start point	cwoProcessOn	rising edge	cutting start (piercing)
Move along the tool path	cwoProcessActive		cutting
TCP speed change (>2mm/s), e.g. slow down at the corner	cwaoTcpSpeed		cutting
at specific position on the path (to change the process data)	cwoDataChange		cutting
End_Fly or Stop at end point	cwoProcessOn	falling edge	cutting end

Note: the name of internal signals for T_ROB1 have "_R1" is appended to the end of the internal signal name.

For example, in following figure the system is cutting a rectangle hole by using a laser. The robot approaches the point S, stops at the point S and then starts cutting (piercing). Then, it goes through corner A, B, C, D, and then passes the point E. Finally, it stops the laser on the fly, leave. See below graph on how related signals are synchronized with the robot motion.



Continues on next page



Process event routines

RW Cutting provides event routines to customize the behavior for different cutting processes.

Signal name	Trigger	Other conditions	Event routine name
cwoPurgeOn	rising edge	PurgeTime > 0	`cwPurgeProcess`
cwoProcessOn	rising edge		`cwProcessOn`
cwaoTcpSpeed	change>2mm/s	Enable Speed Modulation	`cwSpeedChange`
cwoDataChange	rising edge	Only for 3D free form cut	`cwDataChange`
cwoProcessOn	falling edge	OR Program Stop	`cwProcessOff`

Above event routines are called when related signals are triggered and certain conditions are met. These routines are defined in a system module called CwProcUser.sys for general cutting, and LsProcUser.sys / LsProcCont.sys for laser cutting. Users can modify these routines to adapt the specific cutting process to their needs.

See *System Configuration on page 63* for more information about the configuration of cutting processes.

4.2 Laser Cutting

4.2.1 Overview

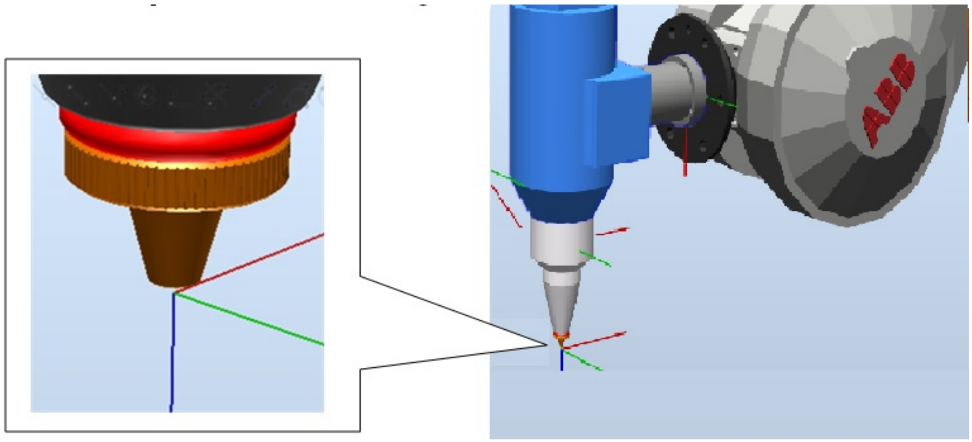
Cutting source and material

Laser cutting cuts material with a fiber laser or a CO_2 laser. RW Cutting supports cutting thin metal with fiber laser. A robotic laser cutting system includes the laser source and the cutting head.

4.2.2 Calibration

Definition

Defining the TCP and `Wobj` precisely is important in order to achieve high quality cutting results. The TCP of the cutting head is the focus position of the laser beam, and should be located under the center of the nozzle at a standoff distance (normally 1.0 mm). The cutting head should be located in the center of the travel path -in other word zero position when define the TCP. The Z axis should be located parallel to the head movement direction, and it should be positioned pointing down. To see this position, see below figure.



There are 2 ways to define TCP: manually and automatically. Bullseye is recommended to be used to define the TCP automatically.

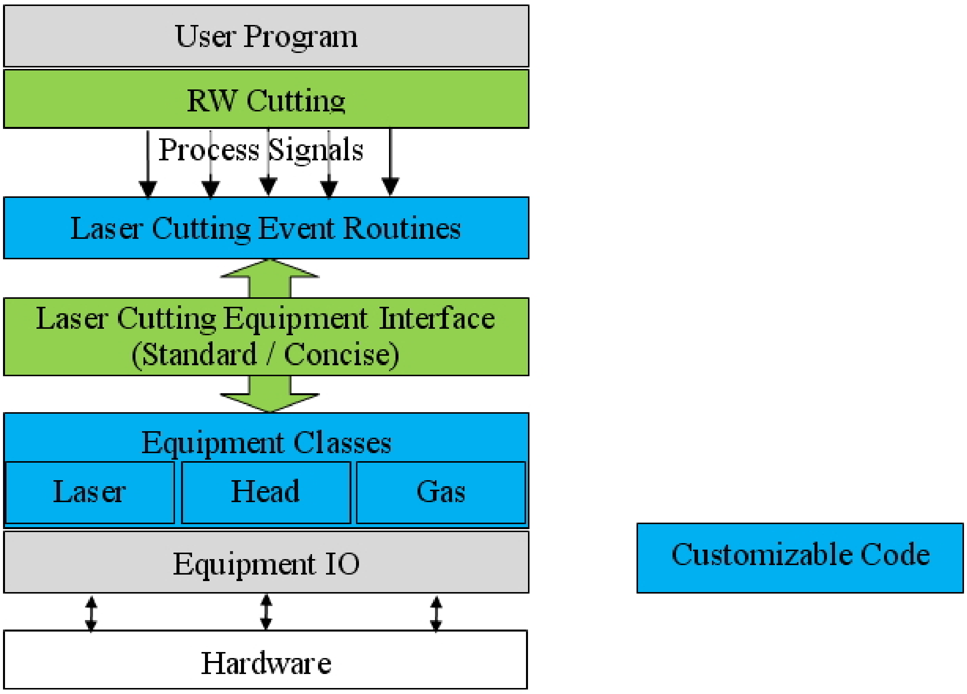
Defining the work object is necessary for using offline programming and for repeating consistent production process. Users can create calibrations first in RobotStudio Cutting PowerPac first, then calibrate the points created in offline program in Robotware Cutting. To use this method, a good fixture and a master work piece are needed. To define work object in Robotware Cutting, perform the following procedure:

> **Note**
>
> Users must create calibrations in RobotStudio Cutting PowerPac first, then calibration can be done in RobotWare Cutting.
>
> Before the calibration in RobotWare Cutting, please execute **PP to Main**. For example, go to the **Production Window** in the main menu, and click **PP to Main**. Then calibration can be done.
>
> After the calibration, if the max error/mean error displayed is too big. Redo the calibration is suggested for users.

4.2.3 Equipment interfaces and classes

Equipment interface and class overview

RW Cutting provides pre-defined event routines for laser cutting process. (See *Process event routines on page 50* for more information). These event routines communicate with equipment classes through laser cutting equipment interfaces. User programs should support the equipment interfaces in the user equipment classes.



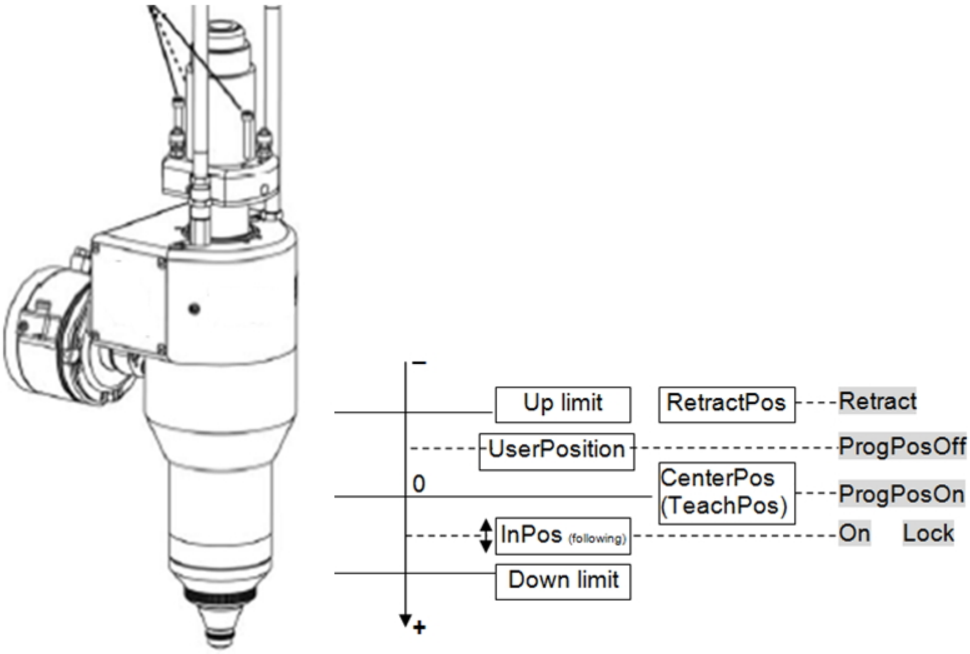
There are two sets of laser equipment interfaces: concise and standard interfaces. Standard laser equipment interfaces define all the common functions for mainstream laser equipment, and they are available for the laser cutting system with standard equipment task. Concise interfaces are a subset of the standard interfaces, which includes basic functions. They are provided as default equipment interfaces for laser cutting systems without standard equipment task.

Head Control Interface

Standard interface	Concise interface	Description
OnInPosCheck	x	Enables auto head control, then waits until it reaches the standoff above the surface. Used during piercing at the beginning of a cut
On	x	Enables auto head control without check. Used after piercing is completed and then starts cutting with a different cut height
Lock	x	Locks the head control during piercing or cutting
Retract	x	Retracts the cutting head after a cut. No check for completion

Continues on next page

Standard interface	Concise interface	Description
ProgPosOn	x	Used for alignment operations. Moves the head to the center position
ProgPosOff	x	Used for alignment operation. Moves the head back up
SetUserPosition		Used to set the distance that the head retracts between cuts
DoRefMove		Used to reference the head after a crash occurs. Also used during initialization of the head
AutoCal		Manual operation to create a new characteristic curve (Precitec)
Height	x	Sets the Cut height. Used with OnInPosCheck and On
SetCharCurve		Selects the characteristic curve (Precitec)



Laser Control Interface

Standard interface	Concise interface	Description
Activate	x	Ramps up the laser. Indicates that the laser is ready to turn on
	ActivateNoWait	Ramps up the laser without confirmation
ActivateOptCh		Selects optical channel that is used for dual laser channel
ResetOptCh		Sets optical channel back to 0 (IPG). To turn the laser off. Be used when requiring faster cycle time.
Reset	x	Reset error
Deactivate	x	Deactivates the laser

Continues on next page

Standard interface	Concise interface	Description
On	x	Turns on laser beam
Off	x	Turns off laser beam
CutProgram		Cuts Program (IPG), used with On
CutPower	x	Laser Power, used with On
PulseLaser		Creates a short pulse that is used for setting or checking the focus of the laser beam
TeachLaserOn		Activates the teach light
TeachLaserOff		Deactivates the teach light

Gas Control Interface

The Gas Control Interface is defined in the standard laser equipment interfaces.

Standard interface	Concise interface	Description
PierceOn		Turns on the gas for piercing
CutOn		Turns on the gas for cutting
Off		Turns off the gas after cutting completion
GasType		Sets the gas type. Used with PierceOn and CutOn
GasFlow		Sets the flow volume. Used with PierceOn and CutOn

Note: Implement the equipment interfaces according to your needs. Not necessary to implement all of them.

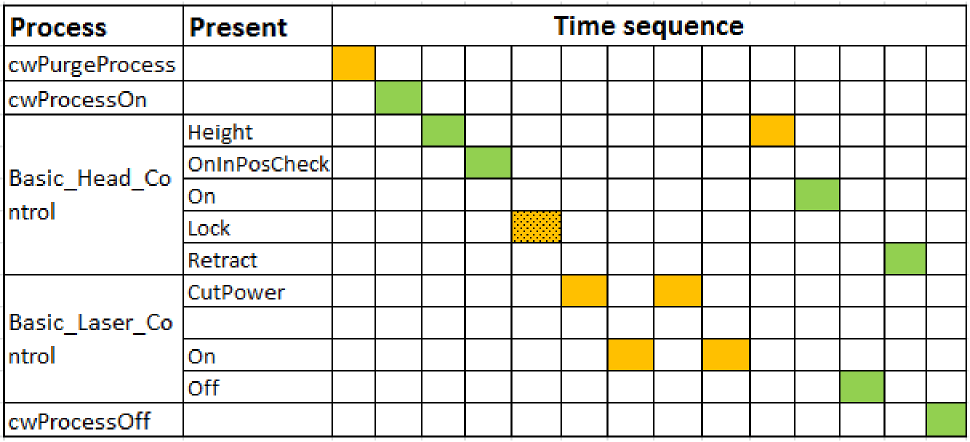
Internal signals

There are three internal signals for laser cutting. The signals should be correctly set in user equipment classes.Description of internal signals are as follows:

Signal Name	Description	Usage
cwoLsReady_R1	Indicates that laser is ready	Ramping up laser take long time. Only activate laser when laser is not ready.
cwoShowLaserOn_R1	Indicates that laser beam is turned	For HMI and simulation
cwaoLaserPower_R1	Indicates the laser power value for monitoring	For HMI and simulation

Continues on next page

Following is the sequence diagram of process signals:



Note

If **Enable Cutting&Gas** and **Enable Process Control** are enabled simultaneously, then the cutting command would execute the procedures in green and orange in sequence.

If only **Enable Process Control** is enabled, then the cutting command would only execute procedures in green.

User equipment classes

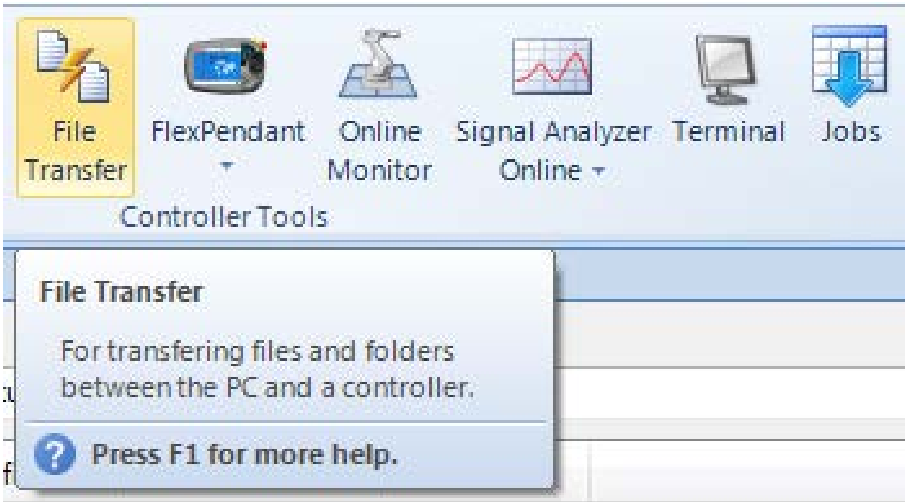
Users must configure the equipment IO and write the equipment classes before cutting the work piece with the laser equipment. RW Cutting provides template files of equipment interfaces and also provides example files of equipment classes. Users can customize them for the specific laser equipment.

The template file LsEquUser.sys supports the standard equipment interface and it is loaded in the background equipment task tProcContR1. The template file LsProcUser.sys supports the concise equipment interface, and it i included in the laser cutting system without equipment task.

Steps to modify the equipment class files in the background task:

1	In order to modify LsEquUser.sys running in the background equipment task, RobotStudio Online must be used.
2	Before editing LsEquUser.sys online, the background task must be stopped by reset the DO signal cwoProcContTaskOK_R1.
3	User can also create their own equipment class files and load them into the background task. In this case, the value of string variables stProc_Gas_Control, stProc_Head_Control, stProc_Laser_Control in LsProcCont.mod” must be changed according to user equipment class routines name.

Continues on next page

4	Save the modified files LsEquUser.sys or LsProcCont.mod back into the controller system folder HOME/Cutting with FileTransfer of RobotStudio Online. 
5	The last step is to warm start the system, and then these files under the folder HOME/Cutting will be loaded automatically into background task and run again.

4.2.4 Command interface and equipment operator HMI

Command interface signals

Command Interface is the communication interface between the foreground motion task and the background equipment task. It is also used in communication between the HMI and the background equipment task.

Note: Laser cutting single task does not require the command interface, because the motion task directly access the equipment IO.

The command interface includes six internal signals. To see descriptions, see below table.

Command Signals	Description
cwgoHdCommand_R1	Head Command ID, 0~15(4bit)
cwoHdExecCommand_R1	Rising edge triggers the command execution; Falling edge indicates the command done.
cwoHdCommandFailed_R1	Set if the command failed.
cwgoLsCommand_R1	Laser & Gas Command ID, 0~15(4bit)
cwoLsExecCommand_R1	Rising edge triggers the command execution; Falling edge indicates the command done.
cwoLsCommandFailed_R1	Set if the command failed.

The sequence of communication: motion task and HMI set the commands ID and triggers the execution signal. Then, the background task executes the command and then sends a reply message that contains the results.

The command ID is defined in the equipment task (LsProcCont.sys). See below table.

Head Command ID	Functions (see the equipment interface for detail)
1	On
2	Lock
3	Retract
4	ProgPosOn
5	ProgPosOff
6	SetUserPosition
7	DoRefMove
8	AutoCal
9	SetCharCurve
Laser Command ID	
1	Activate
2	ActivateOptCh
3	ResetOptCh
4	Reset
5	Deactivate
6	PulseLaser

Continues on next page

7	TeachLaserOn
8	TeachLaserOff
Gas Command ID	
13	Gas Off
14	Pierce Gas On
15	Cut Gas On

Laser equipment operator HMI

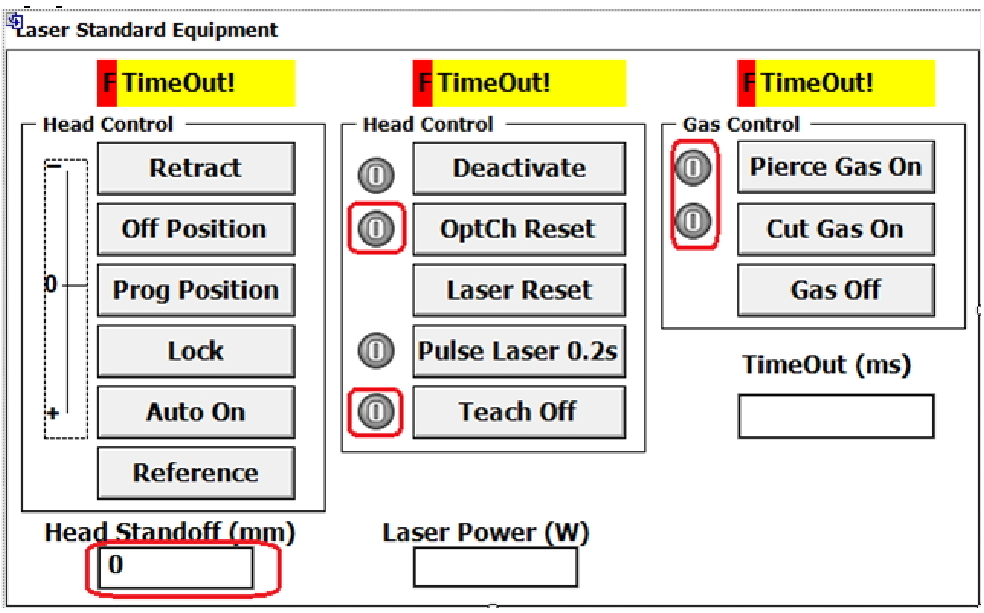
A standard laser equipment operator HMI is provided with standard equipment task. It implements above command interface. It is made by RobotStudio ScreenMaker. The ScreenMaker project file can be found in the folder "Mediapool\CUTTING\ScreenMaker\LaserStdEqu".

Note: Laser cutting single task does not contain the operator HMI. However, users can make an HMI with RobotStudio ScreenMaker and then import it into RW Cutting UI. The default project name is "LaserEqu". See the help files in RobotStudio ScreenMaker for more information.

Users must customize the operator HMI for the specific equipment with ScreenMaker. There are three things that can be customized:

1. Bind the value or property of UI controls with system IO or Rapid data.
2. Disable unused UI controls or add new UI controls and screens
3. Use local language.

Below UI controls have no default binding data. Users create the binding according to the equipment IOs.

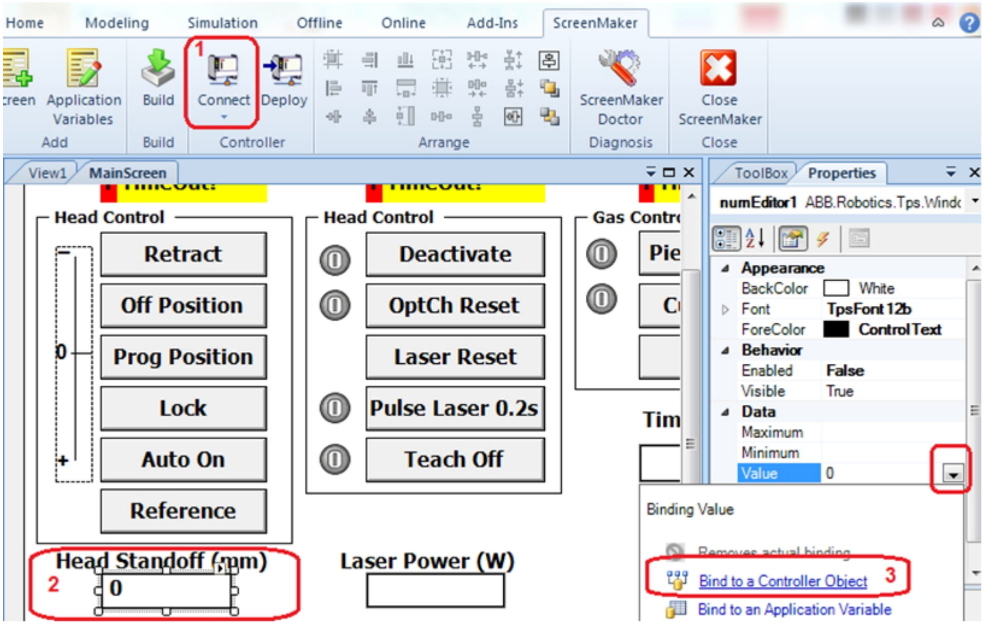


To bind the head standoff analog input signal with the UI control, perform the following:

1. Connect the IRC5 controller. Then add the binding value for the selected UI control numEditor.

Continues on next page

2 Choose the AI signal of head standoff from the signal list.



3. After all the modification is done, press the **Deploy** button to download the HMI .dll files to the HOME folder of the system.

4.2.5 Service routines

General

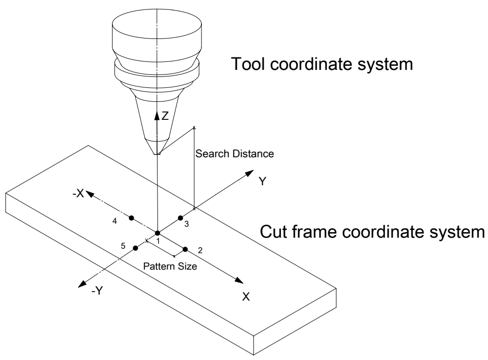
The laser cutting head has a distance sensor to measure the distance between the nozzle and the metal surface of the work piece. RW Cutting provides service routines based on the distance sensor in order to automatically move the cutting head to the surface perpendicularly and in precise distance.

`AlignToSurface` routine configuration

The `AlignToSurface` routine finds the surface (cut plane) through a five points method, and the it aligns the cutting head perpendicular to the surface. It is used to find the perpendicular direction of cut plane precisely, which can be used to define the Z direction of the cut frame.To configure this routine, perform the following:

1. Let cutting head move to center position (teach position)
2. Jog cutting head close to the cut surface (roughly perpendicular to the surface)
3. Run the service routine `AlignToSurface`
4. The robot will move up a little, and then search down until it finds the surface. Then the robot moves to next point and performs the search movement again. After the robot completes 5 points, it moves back to the original position in a perpendicular direction to the surface. (See below figure)

Note: Don't execute `AlignToSurface` above a hole.



There are two ways to find the surface. Using the first method, the tip of cutting head doesn't touch the surface; the system reads the distance from the analog input signal of the sensor. Using the second method, the tip of cutting head touches

Continues on next page

the surface; the system monitors the digital input signal of TipTouch. These two signals should be configured correctly to make the AlignToSurface service work. See *Laser Cutting on page 51* for more detail information.

There are 4 parameters for adjusting the alignment as following:

Align setting	Description	Rapid Variables in LsAdvFunc.sys
Pattern Size	Indicates the distance between 5 points. Default value is 6mm.	`num nAlignmentPatternSize`
Search Distance	Indicates the draw back distance before search start. Default value is 10mm.	`num nSetZSearchDist`
Align Standoff	Indicates the final distance between the tip and the surface after alignment completes. Default value is 1mm.	`num nAlignmentStandoff`
TipTouch	Indicates whether TipTouch search is used. Default value is false.	`bool bCwUseTipTouchSearch`

The default value is defined in the laser equipment setting. See *Laser Cutting on page 51* for more information.

`Set_Z_Distance`

This routine moves the cutting head close to the surface until the distance is Align Standoff. However, it doesn't align the cutting head. To use it, users should follow the same steps as using `AlignToSurface`.

4.3 System Configuration

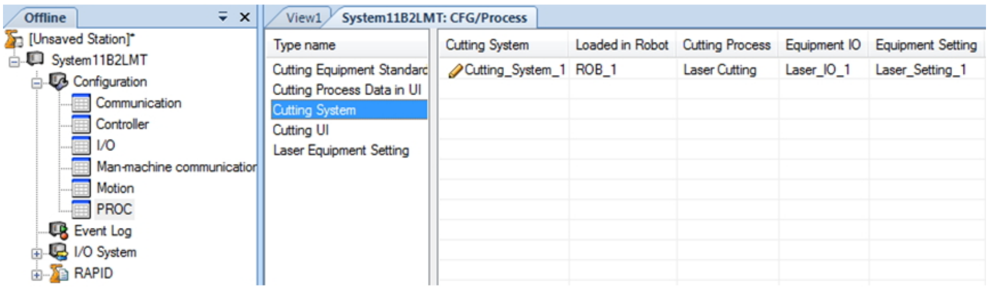
Overview

The system configuration of RW Cutting is divided into 4 groups.

- Cutting System
- Cutting UI
- Laser equipment setting
- Tuning process Info

Cutting System

The top level of Configuration parameters for RW Cutting is Cutting System.See Figure.



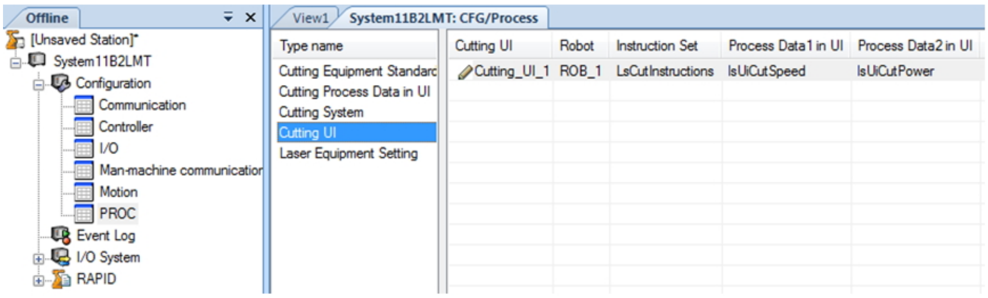
Below are the customizable parameters of Cutting system.

Item	Description	Comments
Stop Event Routine	Indicates whether the routine will be called when program is stopped	Default routine will run `cwProcessOff` internally
Restart Event Routine	Indicates whether the program is stopped during the cutting phases. This routine will be called when restarting the program.	Default routine will pop up a message box asking to restart the cutting process before moving.
TimeOut of Purge	Indicates the maximum time of preparing cutting phase, e.g. purging gas	Default 2s, Max 60s
TimeOut of ProcessOn	Indicates the maximum time of cutting start phase, e.g. piercing	Default 5s, Max 120s
TimeOut of ProcessOff	Indicates the maximum time of cutting end phase.	Default 2s, Max 60s
Disable Confirm of First Cut	In the manual mode, during the first cut movement, if FLASE, the system will show a message dialog and let users comfirm if the laser is actived; otherwise, system will not show this dialog.	Default true
Auto-ILC	If TRUE, system will raise a warning and automatically redo ILC when the ILC data has been changed. Otherwise,system will raise an error and let user manually handles it.	Default false

Continues on next page

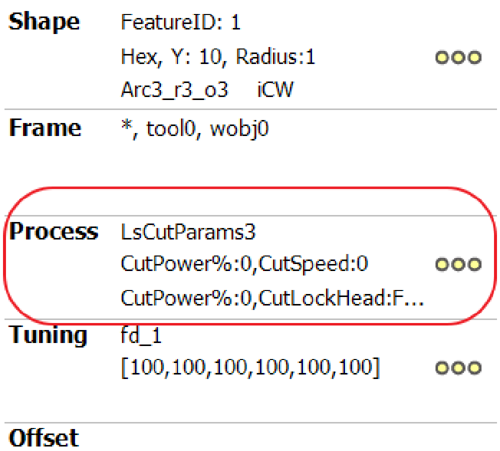
Cutting UI

RW Cutting UI can also be customized by changing the UI setting.



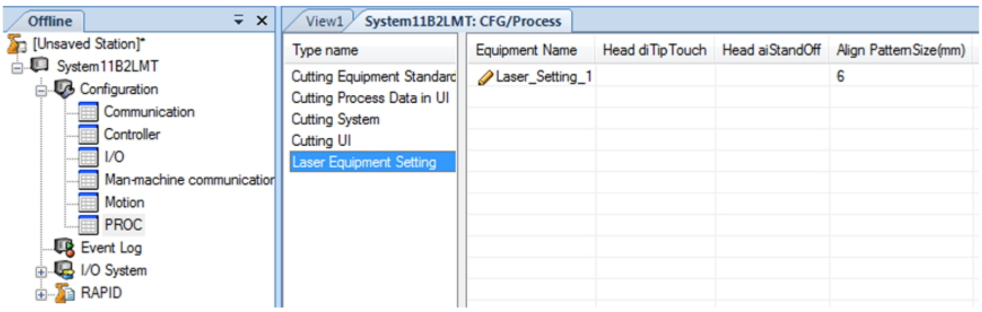
If `LsCutParams` is used in the cutting instruction, the parameters to customize the Cutting UI are in the below table.

Item	Description
Process Data1,2,3,4 on UI	Selects the process data components of `LsCutParams` shown on the shape tuning view of Cutting UI. See below Figure. Three datatypes are supported: `num`, `boolean`, and `string`.
Equipment UI dll	Set the file name of equipment UI dll, which can be launched by Cutting UI



Laser equipment setting

The parameters of laser equipment setting should be correctly set before using advance functions of the laser equipment (e.g. `AlignToSurface`).



Item	Description

Continues on next page

Head diTipTouch	
Head aiStandOff	
Align Pattern Size	Indicates default distance between 5 points. (6mm)
Align Search Distance	Indicates default draw back distance before search start. (10mm)
Align Standoff	Indicates default final distance between the tip and the surface after alignment completes. (1mm)
TipTouch Search	Indicates whether TipTouch search is used. Default value is false
Active Laser Cutting Table	Indicate the name of the currently active laser cutting table.

Tuning process Info

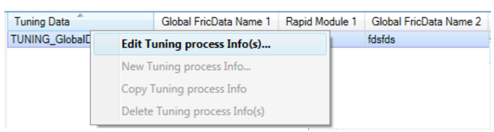
Tuning process Info is used to create and view global tuning data. In RobotStudio Cutting PowerPac, up to 5 global tuning data can be created. Only when global tuning data are created here, the **Advanced** function in the **Shape Tuning** function of RobotWare Cutting can work.

Item	Description
Tuning Data	Default specific type name designated to global type friction data. Don't change these default text.
Global FricData Name 1 Global FricData Name 2 Global FricData Name 3 Global FricData Name 4 Global FricData Name 5	User can define the name of the global friction tuning data used in cutting instructions in this field. If the user does not use the global data, this column would keep the default value of **TRUE**.
Rapid Module 1 Rapid Module 2 Rapid Module 3 Rapid Module 4 Rapid Module 5	The module where the tuning data is stored.If the user does not use the global data, this column would keep the default value of **TRUE**.

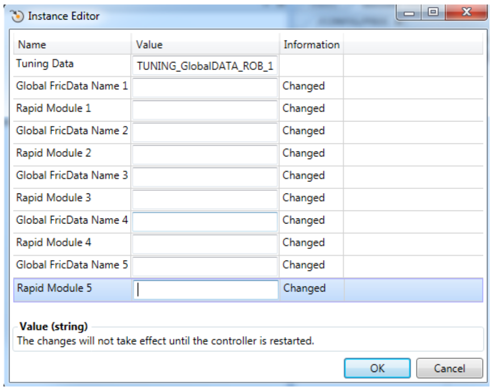
NOTE!

If users want to delete the global friction data has been used, please delete the text in the Global FricData Name and Rapid Module fields. Then warm restart the virtual controller, then the modification would take effect, and the value in the fields would return to **True**.

Edit the global tuning data

	Description	Illustration
1	Right click the column of the global data to be used.	

Continues on next page

	Description	Illustration
2	Double click **Edit tuning process info(s)**, the **Instance Editor** to the data would display.	
3	Name the global friction data and appoint the Rapid Module in which the data would be saved.	
4	Click **OK** to confirm.	

When users need to use the global tuning data, users need to edit the data to give it a name to be used in instructions and assign a module to store the data.

4.4 Tuning

Tuning factors and method

To reach optimal accuracy, the entire laser cutting system needs to be well tuned. There are many factors that will influence the overall accuracy of the work piece. If the final cut result doesn't fulfill the requirement, it is recommend to perform the following check in order:

- External factors: Robot installation e.g. the robot base, the rigidity of the tool, the fixture. Influence: Vibration during the TCP is accelerating or decelerating and causes wave on the cut path.
- Internal factor: Robot itself. Different types of robot have different up limit of accuracy. Generally, small robots has better accuracy than big robots.
- Robot System setting: Calibration data, the TCP, Wobj definition precision, Tool Load and payload value. Influence: Bad path accuracy.
- Robot program of cut path: correct movement arguments, e.g. position, speed, zone of the free form cut path. Influence: deviation from the design path and speed.
- Software tuning: Friction tuning and ILC fine tune.

For external factors, the best solution is to redesign the cell and remove weaknesses of the mechanical components. However if it can't be changed, software adjustments can compensate it to a certain extent. For example, try decreasing the acceleration of the robot by RAPID instruction: `AccSet 30,30`; or Tune the DF value with TuneMaster.

For the cut path program, the best solution is to optimize the cut path with RobotStudio Cutting PowerPac. Cut speeds should be stable, especially at the corners. For example, use customized speeds and zones for free form cut paths to decrease the orientation speed and zone radius.

```
speeddata vTrim1:=[100,90,200,15];
zonedata zTrim1:=[FALSE,0.5,5,40,10,35,5];
```

For software tuning, it is better to do it in the last step after the entire program have been completed and adjusted. Tuning data depends on the specific path, and it takes long time to run the tuning. If any argument of the cut shape or the path is changed, the previous tuning data is lost. The following guidelines should be observed when tuning:

- Use the tuning functions only when necessary. Sometime, the result is also acceptable without tuning.
- ILC is only used as fine tune, please use it together with Friction tune, and don't use it independently.
- Activate the tuning state in the last step after the program is completed and tested, and then run it in auto mode. After the friction tuning has been done, a friction data automatically calculated based on the tuning would replace the default value of the tuning data. Then **Block** the Friction tuning function to run the program with the friction data tuned.

Continues on next page

- If the program of a part continues to run in production for a few weeks, or an old program is put back into production again, redo the tuning to update the tuning data.

5 RAPID reference

5.1 Instructions

5.1.1 LoadLaserTable

Usage

`LoadLaserTable` is used to load another laser table during program running, when a different cutting parameters are needed due to different cutting processes or material.

Basic examples

The following example illustrates the instruction `LoadLaserTable`.

Example 1

```
LoadLaserTable\stActiveTableName:="newTable"\force:=FALSE;
```

Load laser table newTable, if the table is not been loaded before.

Arguments

```
LoadLaserTable[\stActiveTableName][\force]
```

`stActiveTableName`

Data type:`string`

Name of the laser table to be set active.

`force`

Data type:`bool`

When this argument is true, load the table even the table has been loaded.

When this argument is false, check if `stActiveTableName` is the same as the currently active laser table. If not, load the table and update the controller configuration and active the `stLsTableName`.

Error handling

The following recoverable errors are generated and can be handled in an error handler. The system variable `ERRNO` will be set to:

ER-RNO	Argument	Error/Warning	Description/Solution
5111	`stActiveTableName`	Failed to load LS cutting parameters from table file.	Save the LS table file in the folder HOME/LsCuttingParamTables.
5110	`stActiveTableName`	Laser cutting parameter table file is not found.	LS cutting parameter table is not found. Users can find laser tables in the folder HOME/LsCuttingParamTables

Syntax

```
LoadLaserTable
  [stActiveTableName ':='] < expression (IN) of string> ','
  [force ':='] < expression (IN) of bool >
```

5.1.2 CutCadL/CutCadJ/LsCutCadL/LsCutCadJ

Usage

`CutCadL/CutCadJ/LsCutCadL/LsCutCadJ` is used to cut customized shapes. The shape is described by a point array and referenced on the cut frame, and it provides tuning switches to improve the shape accuracy, and the offset argument to adjust the deviation. The instruction has the following functions:

- Start the cutting process at start (piercing) point
- Cut the shape with the specific lead-in and lead-out path
- Stop the cutting process at specific position on the lead-out path
- Auto tune the friction level of robot and reference points to improve the shape accuracy

The difference between `CutCadL` and `CutCadJ` is they use different movement types (`MoveL/MoveJ`) to approach the start point. `CutCadL/CutCadJ` and `LsCutCadL/LsCutCadJ` use different data types of cutting parameters, which are defined for different cutting processes. `CutCadL/CutCadJ` and `Cutdata` are for general cutting, and `LsCutCadL/LsCutCadJ` and `LsCutParams` are for laser cutting process. Although `LsCutCadJ` is used in the examples, `CutCadL/CutCadJ` or `LsCutCadL` could also be used.

Basic examples

The following example illustrates the instruction `LsCutCadJ`.

Example 1

```
LsCutCadJ 1, p1, v1000, “ExampleShape”, Lin3_135_o3, iCW,
     LsTableConf52, tLaser\Wobj:=wobjPart;
```

The robot moves to the start (piercing) point with a speed v1000. A customized shape is cut that is referenced to the default cut frame with the origin p1. “ExampleShape” is the name of the file that defines the customized shape. `Lin3_135_o3` defines the leadin/leadout path as a linear leadin path with distance 3mm and in angle 135, and overlap of 3 mm. The cut direction is clockwise (`iCW`). Cutting process data, for example, cut speed, cut power and pierce time, etc. can be obtained from the currently active laser tables by `LsTableConf52`. The `tooldata` used is `tLaser` and work object is `wobjPart`.

Arguments

```
LsCutCadJ FeatureId, ToPoint, [\RefVectX], [\RefVectZ], [\BiasCut],
     Speed, CadShapeName, LeadParam, CutDirection, LSTableConf,
     [\LsCutParams], [\FrcL], [\Ilc], [\Offset], Tool, [\WObj],
     [\FeatureName]
```

`FeatureId`

Data type:`shpno`

`FeatureID` is a unique integer value that is used as an array pointer for feature specific data stored in the system. The data range is from 1 to 600.

Continues on next page

`ToPoint`

Data type:`robtarget`

`ToPoint` defines the reference point of the shape and the default cut frame. The x and z axes of the default cut frame is in the reversed direction of the x and z axes of TCP. The Y axes is in the same direction of Y axis of the TCP. For more information, refer to Topoint argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\RefVectX]`

Data type:`pos`

`[\RefVectX]` is a unit vector that represents the X axis direction of the cut frame which is related to the wobj coordination system. The x direction of the default cut frame is neglected. It is useful to adjust the TCP without changing the shape direction. If it is not used, the x direction will be derived from the negative tool x-axis. For more information, refer to [\RefVectX] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\RefVectZ]`

Data type:`pos`

`[\RefVectZ]` is a unit vector that represents the z axis direction of the cut frame which is related to the wobj coordination system. The z direction of the default cut frame is neglected. If it is not used, the z-direction will be derived from the reverse direction of tool z-axis. For more information, refer to [\RefVectZ] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\BiasCut]`

Data type:`switch`

`BiasCut` is only used when `RefVectZ` is defined. If `BiasCut` is used, the z direction of the TCP is no parallel with the z direction of the cut frame (defined by `RefVectZ`). With `BiasCut`, it is allowed to cut not perpendicularly to the surface. The maximum angle between the z axis of TCP and the Cut Frame is 20 degree. For more information, refer to `[\BiasCut]` argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`Speed`

Data type:`speeddata`

`Speed` data applies to approaching movements. It defines the velocity of the TCP during the approach to the ToPoint or start (piercing) point. Additionally, it defines the speed of the tool's reorientation and the speed of any uncoordinated additional axes.

`CadShapeName`

Data type:`string`

`CadShapeName` is the file name of the customized shape. It contains a point array that defines the 2D shape. The file should be stored in the folder "HOME/CwCadShapes" and with the extension name ".cad". The length of the file

Continues on next page

name should <=28 characters. RobotStudio Cutting PowerPac can generate the file from the cad model.

`LeadParam`

Data type:`leaddata`

`LeadParam` defines the lead-in/lead-out and overlap parameters.

`CutDirection`

Data type:`cwdirection`

`CutDirection` defines the cut direction: Inside/outside cut clockwise / counter-clockwise(`iCW`/`iCCW`/`oCW`/`oCCW`).

`LSTableConf`

Data type:`LsTableConf`

`LSTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutCadL`/`CutCadJ`. Argument `LsCutParams` defines the laser cutting specific process data for `LsCutCadL`/`LsCutCadJ`, including cut speed and parameters for controlling head, laser and cutting gas during the piercing and cutting phases.

`[\FrcL]`

Data type:`fricdata`

`[\FrcL]` is the friction level of 6 joints of the robot which is used during the processes of cutting a shape. Its value is automatically calculated to improve the path accuracy of cutting a shape. It should be a unique, persistent variable for each cut shape instruction. The value is larger than 0 and less than 300 and varies at different locations in robot work range. It is very important for cutting a good circle.

`[\Ilc]`

Data type:`switch`

`[\Ilc]` is used for switching iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3. iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. The default iteration times of ILC is 3.

`[\Offset]`

Data type:`shapeoffsetdata`

`[\Offset]` is an optional parameter that defines the shape offset specific parameters. It is used for small deviation compensation in location according to

Continues on next page

the measurement in production. The dimension of the customized shape can not to be adjusted.

`Tool`

Data type: `tooldata`

`Tool` is the tool used in the movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool, and it should be defined precisely to achieve good cut accuracy.

`[\WObj]`

Data type: `wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (`wobj0`).

`[\FeatureName]`

Data type: `string`

`[\FeatureName]` is an ptional parameter that defines the feature name.

Program execution

Shape calculation

System calculates the shape path according to the shape parameters. Here is to load the point array in the file. This procedure is performed during running program for the first cycle in Auto mode and recalculated for every cycle in Manual mode. The result is stored in path memory. It will be performed again after the tuning or reset path memory by moving PP to Main or from HMI.

Controlling process equipment

The entire cutting process and each of its phases are coordinated with the robot movements. The robot moves to the start position of the lead-in path. The cutting process is initiated (surface search, laser ramping, piercing, etc.). The robot cuts to the end position (including lead-out). The cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Tuning execution

If the arguments of `Frcl` and `ILC` are used, the robot will move along the shape path in the cut direction and in reverse direction repeatedly, until the system finds the optimal result for the shape. This process takes much longer time (25x) than cutting a shape in normal way. Only running tuning after all the shape and cut data are correctly set. Additionally, it can be run in auto mode.

The turning is activated and deactivated by the cutting state blocking.

Execution in manual mode

The robot moves to the center of the shape (`ToPoint`), stops, calculates the shape, and then moves to the start (piercing) point, stops, starts the cutting process and begins further movement.

Continues on next page

Execution in auto mode

For the first cycle, the program is run with shape calculation which is the same as in manual mode. Since the 2nd cycle, the robot moves to the start (pierce) point directly, and then starts the cutting process and begins further movements.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and the shape is skipped.

Backward

The robot moves to the ToPoint position and the shape is skipped.

Error handling

The following recoverable errors are generated and can be handled in an error handler. The system variable `ERRNO` will be set to:

ERRNO	Argument	Error/Warning	Description
5100	`FeatureId`	Feature ID is out of range	• FeatureId: -1 out of range. • FeatureId: 601 out of range.
5101		Feature ID is conflicting	FeatureId: 10 Has been used
5105	`CadShapeName`	Cad file error	Cad file is not found.
5102	`RefVectZ`	Align angle too large	The angle difference between Vectz and negtive z axis of ToPoint >= 20

Syntax

```
LsCutCadJ
  [FeatureId ’:=’] < expression (IN) of shpno> ‘,’
  [ ToPoint ’:=’] < expression (IN) of robtarget >
  [‘\’RefVectX ‘:=’ <expression (IN) of pos>]
  [‘\’RefVectZ ‘:=’ <expression (IN) of pos>]
  [‘\’BiasCut] ‘,’
  [Speed ‘:=’] < expression (IN) of speeddata > ‘,’
  [CadShapeName ‘:=’] <expression (IN) of string > ‘,’
  [LeadParam ’:=’] < expression (IN) of leaddata> ‘,’
  [CutDirection ’:=’] < expression (IN) of cwdirection> ‘,’
  [LSTableConf’:=’] < expression (IN) of LSTableConf> ‘,’
  [‘\’LsCutParams‘:=’] < expression (IN) of LsCutParams >
  [‘\’FrcL‘:=’ <persistent (PERS) of fricdata>]
  [‘\’Ilc]
  [‘\’Offset ‘:=’< expression (IN) of ShapeOffsetData>]
  [Tool ‘:=’] < persistent (PERS) of tooldata >
  [‘\’WObj ‘:=’ <persistent (PERS) of wobjdata>]
  [‘\’FeatureName ‘:=’ <expression (IN) of string>]
```

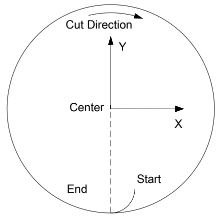
5.1.3 CutCircleL/CutCircleJ/ LsCutCircleJ/ LsCutCircleL

Usage

`CutCircleL`/`CutCircleJ`/`LsCutCircleL`/`LsCutCircleJ` is used to cut a circle hole in the work piece. It provides tuning switches to improve the shape accuracy and it also provides an offset argument to adjust the deviation. The instruction has the following functions:

- Start the cutting process at start (piercing) point
- Cut the shape with the specific lead-in and lead-out path
- Stop the cutting process at specific position on the lead-out path
- Auto tune the friction level of robot and the reference points to improve the shape accuracy

The difference between `CutCircleL` and `CutCircleJ` is they use different movement types (`MoveL`/`MoveJ`) to approach the start point. `CutCircleX` and `LsCutCircleX` use different datatypes of cutting parameters, which are defined for different cutting processes. `CutCircleL`/`CutCircleJ` and cutdata are for general cutting, and `LsCutCircleL`/`LsCutCircleJ` and `LsCutParams` are for laser cutting process. Although `LsCutCircleJ` is used in the examples, `CutCircleL`/`CutCircleJ` or `LsCutCircleL` could also be used.



Basic examples

The following example illustrates the instruction `LsCutCircleJ`.

Example 1

```
LsCutCircleJ 2, p2, v1000, 10, Arc3_r3_o3, iCW, TableConf21,
     tLaser\Wobj:=wobjPart;
```

Continues on next page

The robot moves to the start (piercing) point with a speed of `v1000`. A circle is cut that is referenced to the default cut frame with the center p2, diameter of 10mm. `Arc3_r3_o3` defines the leadin/leadout path as an arc leadin path with radius of 3 mm and overlap of 3 mm. The cut direction is clockwise (`iCW`). Cutting process data can be obtained from parameter tables by `TableConf21`, including, for example, cut speed, cut power and pierce time, etc. The `tooldata` used is tLaser and the work object is `wobjPart`.

Arguments

```
LsCutCircleJ FeatureId, ToPoint, [\RefVectX], [\RefVectZ],
     [\BiasCut]|[\BevelAngle], Speed, Diameter, LeadParam,
     CutDirection, LsTableConf, [\LsCutParams], [\FrcL], [\Ilc],
     [\Offset], Tool, [\WObj], [\FeatureName]
```

`FeatureId`

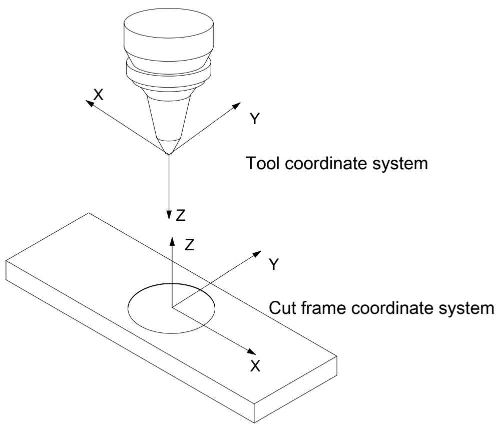
Data type: `shpno`

`FeatureID` is a unique integer value that is used as an array pointer for feature specific data stored in the system. Its data range is from 1 to 600.

`ToPoint`

Data type:`robtarget`

`ToPoint` defines the reference point of the shape and the default cut frame. The x and z axes of the default cut frame is in the reversed direction of the x and z axes of TCP. And Y axis is in the same direction of Y axis of TCP.



`[\RefVectX]`

Data type:`pos`

`[\RefVectX]` represents the X axis direction of the cut frame that is related to the wobj coordination system. The x direction of the default cut frame is neglected. It

Continues on next page

is useful to adjust the TCP without changing the shape direction. If it is not used, the x-direction will be derived from the negative tool x-axis. For more information, refer to the `[\RefVectX]` argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\RefVectZ]`

Data type:`pos`

`[\RefVectZ]` is a unit vector that represents the z axis direction of the cut frame which is related to the wobj coordination system. The z direction of the default cut frame is overwrote. If it is not used, the z-direction will be derived from the reverse direction of tool z-axis. For more information, refer to the `[\RefVectZ]` argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\BiasCut]`

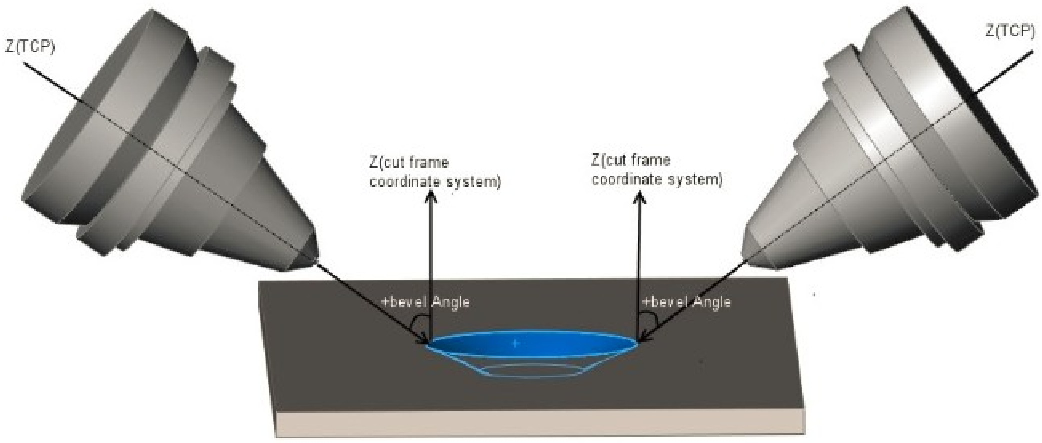
Data type:`switch`

`BiasCut` is only used when `RefVectZ` is defined. If `BiasCut` is used, the z direction of the TCP is not parallel with the z direction of the cut frame (defined by `RefVectZ`). With `BiasCut`, it is allowed to cut not perpendicularly to the surface. The maximum angle between the z axis of TCP and the cut frame is 20 degree. For more information, refer to the `[\BiasCut]` argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*

`[\BevelAngle]`

PICPICKSHAPEData type:`num`

`[\BevelAngle]` produces a given number of bevel angle in the cutting face. Then user can get a better correction result from friction tuning and ILC.



`Speed`

Data type:`speeddata`

`Speed` applies to approaching movements. It defines the velocity of the TCP during the approach to the ToPoint or the start (piercing) point. Additionally, it defines the speed of the tool's reorientation and the speed of any uncoordinated additional axes.

Continues on next page

`Diameter`

Data type:`num`

`Diameter` defines the radius of the circle in mm. The data range is between 3~300mm.

`LeadParam`

Data type:`leaddata`

`LeadParam` defines the lead-in/lead-out and overlap parameters.

`CutDirection`

Data type: `cwdirection`

`CutDirection` defines the cut direction: Inside/outside cut clockwise / counter-clockwise. (`iCW`/`iCCW`/`oCW`/`oCCW`)

`LsTableConf`

Data type:`LsTableConf`

`LsTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutCirlceL`/`CutCircleJ`. Argument `LsCutParams` defines the laser cutting specific process data for `LsCutCirlceL`/`LsCutCircleJ`, including cut speed and parameters for controlling head, laser, cutting gas during piercing and cutting phases. When both `LsTableConf` and `LsCutParams` are used in the cutting instructions, the data of `LsCutParams` would have the priority to be used.

`[\FrcL]`

Data type:`fricdata`

`[\FrcL]` is the friction level of 6 joints of the robot which is used during cutting this shape. Its value is auto calculated to improve the path accuracy of cutting the shape.Up to 5 global tuning data can be used in different cutting instructions.Normal cutting data generated by users should be a unique and persistent variable for each cut shape instruction. The value is larger than 0 and less than 300 and varies at different locations in robot work range. It is very important for cutting a good circle.

`[\Ilc]`

Data type:`switch`

`[\Ilc]` is used for switching iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3. iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into

Continues on next page

reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3.

`[\Offset]`

Data type:`shapeoffsetdata`

`[\Offset]` is an optional parameter that defines the shape offset specific parameters. It is used for small deviation compensation in location according to the measurement in production. The dimension of the customized shape can not to adjusted.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool, and it should be defined precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`

`[\Wobj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (`wobj0`).

`[\FeatureName]`

Data type:`string`

`[\FeatureName]` is an optional parameter that is used to enter the string of the feature name.

Program execution

Shape calculation

System calculates the shape path according to the shape parameters. It takes some time to complete the calculation. This procedure is performed during running program for the first cycle in Auto mode and recalculated for every cycle in Manual mode. The result is stored in path memory. It will be performed again after the tuning or reset path memory by moving PP to Main or from HMI.

Controlling process equipment

The entire cutting process and each of its phases are coordinated with the robot movements. The robot moves to the start position of the lead-in path. The cutting process is initiated (surface search, laser ramping, piercing, etc.). The robot cuts to the end position (including lead-out), and then the cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Continues on next page

Tuning execution

If the arguments of `Frcl` and `ILC` are used, the robot will move along the shape path in cut direction and in the reverse direction repeatedly, until the system finds the optimal result for the shape. It takes much longer time (25x) than cutting a shape in normal way. Only running tuning after all the shape and cut data are correctly set. And it can be run in auto mode.

The turning is activated and deactivated by the cutting state blocking.

Execution in manual mode

The robot moves to the center of the shape (`ToPoint`), stops, calculates the shape, and then moves to the start (piercing) point, stops, starts the cutting process and further movements.

Execution in auto mode

For the first cycle, the program is run with shape calculation, which is the same as in manual mode. Since the 2nd cycle, the robot moves to the start (piercing) point directly, and then starts the cutting process and further movements.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and the shape is skipped.

Backward

The robot moves to the ToPoint position and the shape is skipped.

Error handling

The following recoverable errors are generated and can be handled in an error handler. The system variable `ERRNO` will be set to:

ERRNO	Argument	Error/Warning	Description
5100	`FeatureId`	Feature ID is out of range.	• FeatureId: -1 out of range. • FeatureId: 601 out of range.
5101		Feature ID is conflicting	FeatureId: 10 Has been used
5050	`Diameter`	Shape calculation error	Diameter< 1.4 mm
5102	`RefVectZ`	Align angle too large	The angle difference between Vectz and negtive z axis of ToPoint >= 20

Syntax

```
LsCutCircleJ
  [FeatureId ':='] < expression (IN) of shpno> ‘,’
  [ ToPoint ':='] < expression (IN) of robtarget >
  [‘\’RefVectX ‘:=’ <expression (IN) of pos>]
  [‘\’RefVectZ ‘:=’ <expression (IN) of pos>]
  [‘\’BiasCut]
  |['\'BevelAngle]‘,’
  [Speed ‘:=’] < expression (IN) of speeddata > ‘,’
  [Diameter ‘:=’] < expression (IN) of num > ‘,’
  [LeadParam ':='] < expression (IN) of leaddata> ‘,’
  [CutDirection ':='] < expression (IN) of cwdirection> ‘,’
```

Continues on next page

```
[LsTableConf’:=’] < expression (IN) of LsTableConf> ‘,’
[‘\’LsCutParams‘:=’] < expression (IN) of LsCutParams >
[‘\’FrcL‘:=’ <persistent (PERS) of fricdata>]
[‘\’Ilc]
[‘\’Offset ‘:=’< expression (IN) of ShapeOffsetData>]
[Tool ‘:=’] < persistent (PERS) of tooldata >
[‘\’WObj ‘:=’ <persistent (PERS) of wobjdata>]
[‘\’FeatureName ‘:=’ <expression (IN) of string>]
```

5.1.4 WristCutCircleL/WristCutCircleJ/LsWristCutCircleL/LsWristCutCircleJ

Usage

`WristCutCircleL`/`WristCutCircleJ`/`LsWristCutCircleL`/`LsWristCutCircleJ` is used to cut a small circle hole (3-6 mm) with only two wrist axes. Only moving axis 4 and 5(5and 6, or 4 and 6) of the robot achieves better accuracy of cutting the circle. And robot should move by very low speed, for example: `cutparam.cutspeed`< 10 mm/s. The other behavior is the same as *CutCircleL/CutCircleJ/ LsCutCircleJ/ LsCutCircleL on page 75*.

Continues on next page

Basic examples

The following example illustrates the instruction `LsWristCutCircleJ`.

Example 1

```
LsWristCutCircleJ 3, p3, v1000, 6, Lin2_135_o2, iCW,
     TableConf21\Wrist56, tLaser\Wobj:=wobjPart;
```

The robot moves to the start (piercing) point with a speed v1000. A circle is cut that is referenced to the default cut frame with the center p3,and a diameter of 6mm. `Lin2_135_o2` defines the leadin/leadout path as a linear leadin path with a distance of 2 mm and at angle 135, and overlap of 2 mm. And the cut direction is clockwise (`iCW`).Cutting process data can be obtained from parameter tables by `TableConf21`, including for example, cut speed, cut power and piercing time, etc. Robot moves only axis 5 and 6 during the cutting process. The `tooldata` used is `tLaser` and work object is `wobjPart`.

Arguments

```
LsWristCutCircleJ FeatureId, ToPoint, [\RefVectX], [\RefVectZ],
     [\BiasCut]|[\BevelAngle], speed, Diameter, LeadParam,
     CutDirection, LsTableConf, [\LsCutParams], [\switch
     Wrist45]|[\switch Wrist46]|[\switch Wrist56], [\FrcL], [\Ilc],
     [\Offset], Tool, [\WObj], [\FeatureName]
```

`FeatureId`

Data type: `shpno`

`FeatureID` is a unique integer value that is used as an array pointer for feature specific data stored in the system. Its data range is from 1 to 600.

`ToPoint`

Data type: `robtarget`

`ToPoint` defines the reference point of the shape and the default cut frame. The x and z axes of the default cut frame is in the reversed direction of the x and z axes of TCP. And Y axis is in the same direction of Y axis of TCP. For more information, refer to the `ToPoint` argument of *CutCircleL/CutCircleJ/ LsCutCircleJ/ LsCutCircleL on page 75*.

`[\RefVectX]`

Data type:`pos`

`[\RefVectX]` is a unit vector that represents the X axis direction of the cut frame which is related to the `wobj` coordination system. The x direction of the default cut frame is neglected. It is useful to adjust the TCP without changing the shape direction. If it is not used, the x-direction will be derived from the negative tool x-axis. For more information, refer to the [\RefVectX] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\RefVectZ]`

Data type:`pos`

`[\RefVectZ]` is a unit vector that represents the z axis direction of the cut frame which is related to the wobj coordination system. The z direction of the default cut frame is overwrote. If it is not used, the z-direction will be derived from the reverse

Continues on next page

direction of tool z-axis. For more information, refer to the [\RefVectZ] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\BiasCut]`

Data type:`switch`

BiasCut is only used when RefVectZ is defined. If BiasCut is used, the z direction of the TCP is no parallel with the z direction of the cut frame (defined by RefVectZ). With BiasCut, it is allowed to cut not perpendicularly to the surface. The maximum angle between z axis of TCP and Cut Frame is 20 degree. For more information, refer to the [\BiasCut] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

`[\BevelAngle]`

Data type:`num`

`[\BevelAngle]` produces a given number of bevel angle in the cutting face. Then user can get a better correction result from friction tuning and ILC. For detailed information, see *`[\BevelAngle]` on page 77*.

`Speed`

Data type:`speeddata`

`Speed` applies to approaching movements. It defines the velocity of the TCP during the approach to the ToPoint or the start (piercing) point, as well as the speed of the tool's reorientation and the speed of any uncoordinated additional axes.

`Diameter`

Data type:`num`

`Diameter` defines the radius of the circle in mm. The data range is 3~6mm.

`LeadParam`

Data type: `leaddata`

`LeadParam` defines the lead-in/lead-out and overlap parameters.

`CutDirection`

Data type:`cwdirection`

`CutDirection` defines the cut direction: Inside/outside cut clockwise / counter-clockwise. (`iCW`/`iCCW`/`oCW`/`oCCW`)

`LsTableConf`

Data type: `LsTableConf`

`LsTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `WristCutCircleL`/`WristCutCircleJ`. Argument LsCutParams defines the laser cutting specific process data for `LsWristCutCircleL`/`LsWristCutCircleJ`,

Continues on next page

including cut speed and parameters for controlling head, laser, cutting gas during piercing and cutting phases. When both `LsCutParams` and `LsTableConf` are used in the instruction, the data of `LsCutParams` would be used.

`[\Wrist4 5]`

Data type:`switch`

Cutting is processed with the robot wrist 4 & 5 if this argument is selected.

`[\Wrist4 6]`

Data type:`switch`

Cutting is processed with the robot wrist 4 & 6 if this argument is selected.

`[\Wrist5 6]`

Data type:`switch`

Cutting is processed with the robot wrist 5 & 6 if this argument is selected.

`[\FrcL]`

Data type:`fricdata`

`[\FrcL]` is the friction level of 6 joints of robot which is used during cutting this shape. Its value is auto calculated to improve the path accuracy of cutting the shape. It should be a unique persistent variable for each cut shape instruction. The value is larger than 0 and less than 300 and varies at different locations in robot work range. It is very important for cutting a good circle.

`[\Ilc]`

Data type:`switch`

`[\Ilc]` is used for switching iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3. iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3.

`[\Offset]`

Data type:`shapeoffsetdata`

`[\Offset]` is an optionalparameter that defines the shape offset specific parameters. It is used for small deviation compensation in location according to the measurement in production. The dimension of the customized shape can not to adjusted.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool and it should be define precisely to achieve good cut accuracy.

Continues on next page

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (`wobj0`).

`[\FeatureName]`

Data type: `string`

`[\FeatureName]` is an optional parameter that is used to enter the string of the feature name.

Program execution

Shape calculation

System calculates the shape path according to the shape parameters. It takes some time to complete the calculation. This procedure is performed during running program for the first cycle in Auto mode and recalculated for every cycle in Manual mode. The result is stored in path memory. It will be performed again after the tuning or reset path memory by moving PP to Main or from HMI.

Controlling process equipment

The entire cutting process and each of its phases are coordinated with the robot movements. The robot moves to the start position of the lead-in path. The cutting process is initiated (surface search, laser ramping, piercing, etc.). The robot cuts to the end position (including lead-out). The cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Tuning execution

If the arguments of `Frcl` and `ILC` are used, The robot will move along the shape path in cut direction and in the reverse direction repeatedly, until system finds the optimized result for the shape. It takes much longer time (25x) than cutting a shape in a normal way. Only running tuning after all the shape and cut data are correctly set. And it can be run in auto mode.

The turning is activated and deactivated by the cutting state blocking.

Execution in manual mode

The robot moves to the center of the shape (`ToPoint`), stop, calculates the shape, and then moves to the start (piercing) point, stops, starts the cutting process and further movements.

Execution in auto mode

For the first cycle, the program is run with shape calculation, which is the same as in manual mode. Since the 2nd cycle, the robot moves to the start (pierce) point directly, and then starts the cutting process and further movements.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and the shape is skipped.

Continues on next page

Backward

The robot moves to the ToPoint position and the shape is skipped.

Error handling

The following recoverable errors are generated and can be handled in an error handler. The system variable `ERRNO` will be set to:

ERRNO	Argument	Error/Warning	Description
5100	`FeatureId`	Feature ID is out of Range	• FeatureId: -1 out of range. • FeatureId: 601 out of range.
5101		Feature ID is conflicting	FeatureId: 10 Has been used
5050	`Diameter`	Shape calculation error	Diameter< 1.4 mm
5102	`RefVectZ`	Align angle too large	The angle difference between Vectz and negtive z axis of ToPoint >= 20

Syntax

```
LsWristCutCircleL
  [FeatureId ’:=’] < expression (IN) of shpno> ‘,’
  [ ToPoint ’:=’] < expression (IN) of robtarget >
  [‘\’RefVectX ‘:=’ <expression (IN) of pos>]
  [‘\’RefVectZ ‘:=’ <expression (IN) of pos>]
  [‘\’BiasCut]
  |['\'BevelAngle]‘,’
  [Speed ‘:=’] < expression (IN) of speeddata > ‘,’
  [Diameter ‘:=’] < expression (IN) of num > ‘,’
  [LeadParam ’:=’] < expression (IN) of leaddata> ‘,’
  [CutDirection ’:=’] < expression (IN) of cwdirection> ‘,’
  [LsTableConf ’:=’] < expression (IN) of LSTableConf> ‘,’
  [‘\’LsCutParams‘:=’] < expression (IN) of LsCutParams >
  ['\'Wrist45]
  |['\'Wrist46]
  |['\'Wrist56] ‘,’
  [‘\’FrcL‘:=’ <persistent (PERS) of fricdata>]
  [‘\’Ilc]
  [‘\’Offset ‘:=’< expression (IN) of shapeoffsetdata>]
  [Tool ‘:=’] < persistent (PERS) of tooldata >
  [‘\’WObj ‘:=’ <persistent (PERS) of wobjdata>]
  [‘\’FeatureName ‘:=’ <expression (IN) of string>]
```

5.1.5 CutHexL/CutHexJ/LsCutHexL/LsCutHexJ

Usage

`CutHexL/CutHexJ/LsCutHexL/LsCutHexJ` is used to cut a hexagon hole in the workpiece.This argument provides tuning switches to improve the shape accuracy and the offset argument to adjust the deviation. The instruction has the following functions:

- Start the cutting process at start (piercing) point
- Cuts the shape with the specific lead-in and lead-out path
- Stops the cutting process at specific position on the lead-out path
- Auto tunes the friction level of robot and reference points to improve the shape accuracy

The difference between `CutHexL` and `CutHexJ` is they use different movement type(MoveL/MoveJ) to approach the start point. `CutHexL/CutHexJ` and `LsCutHexL/LsCutHexJ` use different datatypes of `CutParam` which are defined for different cutting processes. `CutHexL/CutHexJ` and `Cutdata` are for general cutting. `LsCutHexL/LsCutHexJ` and `LsCutParams` are for laser cutting process.

Continues on next page

Although `LsCutHexJ` is used in the examples, CutHexL/J or LsCutHexL could also be used.

Continues on next page

Basic example

The following example illustrates the instruction `LsCutHexJ`

Example 1

```
LsCutHexJ 4, p4, v1000, 10, 1, Arc3_r3_o3, iCW, TableConf21,
     tLaser\Wobj:=wobjPart;
```

The robot moves to the start (piercing) point with a speed of `v1000`. A hexagon is cut that is referenced to the default cut frame with the center `p4`, height of 10mm, and the corner radius of 1mm. `Arc3_r3_o3` defines the leadin/leadout path as an arc leadin path with radius of 3 mm and an overlap of 3 mm. And the cut direction is clockwise (`iCW`). Cutting process data can be obtained from parameter tables by `TableConf21`, including, for example, cut speed, cut power and piercing time, etc. The `tooldata` used is `tLaser` and work object is `wobjPart`.

Arguments

```
LsCutHexJ FeatureId, ToPoint, [\RefVectX], [\RefVectZ],
     [\BiasCut]|[\BevelAngle], Speed, Y, Radius, LeadParam,
     CutDirection, LsTableConf, [\LsCutParams], [\FrcL], [\Ilc],
     [\Offset], Tool, [\WObj], [\FeatureName]
```

`FeatureId`

Data type: `shpno`

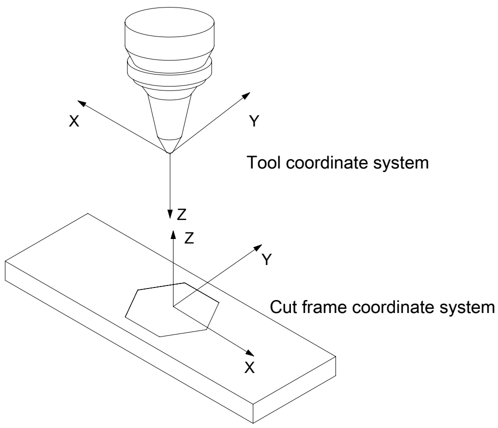
`FeatureID` is a unique integer value that is used as an array pointer for feature specific data stored in the system. Its data range is from 1 to 600.

`ToPoint`

Data type:`robtarget`

Continues on next page

ToPoint defines the reference point of the shape and the default cut frame. The x and z axes of the default cut frame is in the reversed direction of the x and z axes of TCP. And Y axis is in the same direction of Y axis of TCP.



[\RefVectX]

Data type:pos

[\RefVectX] is a unit vector that represents the X axis direction of the cut frame which is related to the wobj coordination system. The x direction of the default cut frame is neglected. It is useful to adjust the TCP without changing the shape direction. If it is not used, the x-direction will be derived from the negative tool x-axis. For more information, refer to the [\RefVectX] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

[\RefVectZ]

Data type: pos

[\RefVectZ] is a unit vector that represents the z axis direction of the cut frame which is related to the wobj coordination system. The z direction of the default cut frame is overwrote. If it is not used, the z-direction will be derived from reverse direction of tool z-axis. For more information, refer to the [\RefVectZ] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

[\BiasCut]

Data type:switch

BiasCut is only used when RefVectZ is defined. If BiasCut is used, the z direction of the TCP is no parallel with the z direction of the cut frame (defined by RefVectZ). With BiasCut, it is allowed to cut not perpendicularly to the surface. The maximum angle between the z axis of TCP and the Cut Frame is 20 degree. For more

Continues on next page

information, refer to the [\BiasCut] argument of *CutRectL/CutRectJ/LsCutRectL/LsCutRectJ on page 96*.

[\BevelAngle]

Data type:num

[\BevelAngle] produces a given number of bevel angle in the cutting face. Then user can get a better correction result from friction tuning and ILC. For detailed information, see *[\BevelAngle] on page 77*.

Speed

Data type:speeddata

Speed data applies to approaching movements. It defines the velocity of the TCP during the approach to the ToPoint or the start (piercing) point. Additionally, it defines the speed of the tool's reorientation and the speed of any uncoordinated additional axes.

Y

Data type:num

Y defines the height of the hexagon in mm. The minimum value is 4mm.

Radius

Data type:num

Radius defines the corner radius of the hexagon in mm. The minimum value is 0.2mm. In case 0.2mm < Radius < 1mm, the corner radius is 0.2mm.

LeadParam

Data type:leaddata

LeadParam defines the lead-in/lead-out and overlap parameters.

CutDirection

Data type:cwdirection

Argument cwdirection CutDirection defines the cut direction: Inside/outside cut clockwise / counter-clockwise. (iCW/iCCW/oCW/oCCW)

LsTableConf

Data type:LsTableConf

LSTableConf defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

[\LsCutParams]

Data type:cutdata/LsCutParams

[\LsCutParams] defines the general cutting process data for CutHexL/CutHexJ. LsCutParams defines the laser cutting specific process data for LsCutHexL/LsCutHexJ, including cut speed and parameters for controlling head, laser, cutting gas during piercing and cutting phases. When both LsCutParams and LsTableConf are used in the instruction, the data of LsCutParams would be used.

Continues on next page

`[\FrcL]`

Data type:`fricdata`

`[\FrcL]` is the friction level of 6 joints of the robot which is used during cutting this shape. Its value is auto calculated to improve the path accuracy of cutting the shape.Up to 5 global tuning data can be used in different cutting instructions.Normal cutting data generated by users should be a unique and persistent variable for each cut shape instruction. The value is larger than 0 and less than 300 and varies at different locations in robot work range. It is very important for cutting a good circle.

`[\Ilc]`

Data type:`switch`

`[\Ilc]` is used for switching iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3.

`[\Offset]`

Data type:`shapeoffsetdata`

`[\Offset]` is an optional parameter that defines the shape offset specific parameters. It is used for small deviation compensation in location according to the measurement in production. The dimension of the customized shape can not to adjusted.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool, and it should be defined precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (wobj0).

`[\FeatureName]`

Data type:`string`

`[\FeatureName]` is an optional parameter that is used to enter the string of the feature name.

Continues on next page

Program execution

Shape calculation

System calculates the shape path according to the shape parameters. It takes some time to complete the calculation. This procedure is performed during running program for the first cycle in Auto mode and recalculated for every cycle in Manual mode. The result is stored in path memory. It will be performed again after the tuning or reset path memory by moving PP to Main or from HMI.

Controlling process equipment

The entire cutting process and each of its phases are coordinated with the robot movements. The robot moves to the start position of the lead-in path. The cutting process is initiated (surface search, laser ramping, piercing, etc.). The robot cuts to the end position (including lead-out) and then the cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Tuning execution

If the arguments of `Frcl` and `ILC` are used, the robot will move along the shape path in the cut direction and in the reverse direction repeatedly, until the system finds the optimal result for the shape. It takes much longer time (25x) than cutting a shape in a normal way. Only running tuning after all the shape and cut data are correctly set. And it can be run in auto mode.

The turning is activated and deactivated by the cutting state blocking.

Execution in manual mode

The robot moves to the center of the shape (`ToPoint`), stops, calculates the shape, and then moves to the start (piercing) point, stops, starts the cutting process and further movements.

Execution in auto mode

For the first cycle, the program is run with shape calculation, which is the same as in manual mode. Since the 2nd cycle, the robot moves to the start (piercing) point directly, and then starts the cutting process and further movements.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and the shape is skipped.

Backward

The robot moves to the ToPoint position and the shape is skipped.

Continues on next page

Error handling

The following recoverable errors are generated and can be handled in an error handler. The system variable `ERRNO` will be set to:

ERRNO	Argument	Error/Warning	Description
5100	`FeatureId`	Feature ID is out of Range	• FeatureId: -1 out of range. • FeatureId: 601 out of range.
5101		Feature ID is conflicting	FeatureId: 10 Has been used
5050	`Y` or `Radius`	Shape calculation error	Y< 4 mm, Radius< 1mm
5102	`RefVectZ`	Align angle too large	the angle difference between Vectz and negtive z axis of ToPoint >= 20

Syntax

```
LsCutHexL
  [FeatureId ':='] < expression (IN) of shpno> ','
  [ ToPoint ':='] < expression (IN) of robtarget >
  ['\'RefVectX ':=' <expression (IN) of pos>]
  ['\'RefVectZ ':=' <expression (IN) of pos>]
  ['\'BiasCut]
  |['\'BevelAngle]','
  [Speed ':='] < expression (IN) of speeddata > ','
  [Y ':='] < expression (IN) of num > ','
  [Radius ':='] < expression (IN) of num > ','
  [LeadParam ':='] < expression (IN) of leaddata > ','
  [CutDirection ':='] < expression (IN) of cwdirection > ','
  [LsTableConf':='] < expression (IN) of LsTableConf> ','
  ['\'LsCutParams':='] < expression (IN) of cuddata | LsCutParams
       > ','
  ['\'FrcL':=' <persistent (PERS) of fricdata>]
  ['\'Ilc]
  ['\'Offset ':='< expression (IN) of shapeoffsetdata>]','
  [Tool ':='] < persistent (PERS) of tooldata > ','
  ['\'WObj ':=' <persistent (PERS) of wobjdata>]
  ['\'FeatureName ':=' <expression (IN) of string>]
```

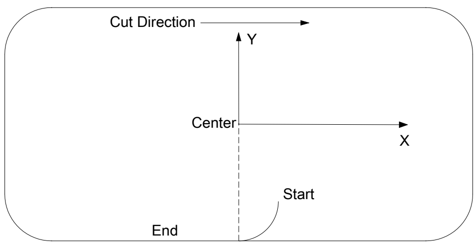
5.1.6 CutRectL/CutRectJ/LsCutRectL/LsCutRectJ

Usage

`CutRectL`/`CutRectJ`/`LsCutRetcL`/`LsCutRetcJ` is used to cut a rectangle in the work piece. It provides tuning switches to improve the shape accuracy, and the offset argument to adjust the deviation. The instruction has the following functions:

- Start the cutting process at start (piercing) point
- Cut the shape with the specific lead-in and lead-out path
- Stop the cutting process at specific position on the lead-out path
- Auto tune the friction level of robot and reference points to improve the shape accuracy

The difference between `CutRectL` and `CutRectJ` is they use different movement type (MoveL/MoveJ) to approach the start point. `CutRectL`/`CutRectJ` and `LsCutRectL`/`LsCutRectJ` use different datatypes of cutting parameters which are defined for different cutting processes. `CutRectL`/`CutRectJ` and cutdata are for general cutting. `LsCutRectL`/`LsCutRectJ` and LsCutParams are for laser cutting process. Although LsCutRectJ is used in the examples, CutRectL/J or LsCutRectL could also be used.



Basic examples

The following example illustrates the instruction `LsCutRectJ`.

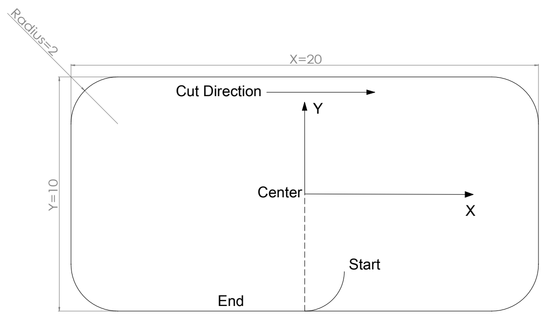
Example 1

```
LsCutRectJ 5, p5, v1000, 20, 10, 2, Arc3_r3_o3, iCW, TableConf21,
     tLaser\Wobj:=wobjPart;
```

The robot moves to the start (piercing) point with a speed v1000. A rectangle is cut that is referenced to the default cut frame with the center p5, width of 20mm,

Continues on next page

height of 10mm, and the corner radius of 2mm. `Arc3_r3_o3` defines the leadin/leadout path as an arc leadin path with radius of 3 mm and overlap of 3 mm. And the cut direction is clockwise (`iCW`). Cutting process data is contained in the TableConf21, including, for example, cut speed, cut power and pierce time, etc. The tooldata used is tLaser and work object is wobjPart.



Arguments

```
LsCutRectJ FeatureId, ToPoint, [\RefVectX], [\RefVectZ],
     [\BiasCut]|[\BevelAngle], Speed, X, Y, Radius, LeadParam,
     CutDirection, LsTableConf, [\LsCutParams] , [\FrcL], [\Ilc],
     [\Offset], Tool, [\WObj], [\FeatureName]
```

`FeatureId`

Data type: `shpno`

`FeatureID` is a unique integer value that is used as an array pointer for feature specific data stored in the system. Its data range is from 1 to 600.

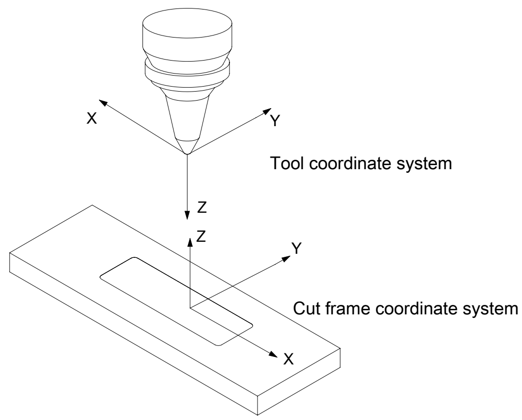
`ToPoint`

Data type:`robtarget`

`ToPoint` defines the reference point of the shape (typically the center) and the default cut frame. The x and z axes of the default cut frame is in the reversed

Continues on next page

direction of x and z axes of TCP. And Y axis is in the same direction of Y axis of TCP.



`[\RefVectX]`

Data type:`robtarget`

`[\RefVectX]` is a unit vector that represents the X axis direction of the cut frame that is related to the wobj coordination system. The x direction of the default cut frame is neglected. It is useful to adjust the TCP without changing the shape

Continues on next page

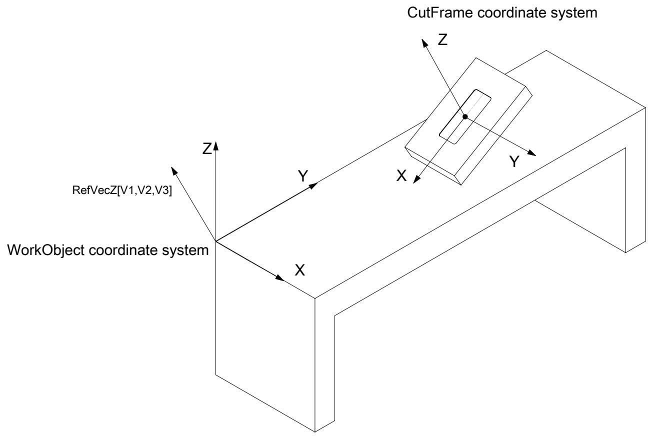
direction. If it is not used, the x-direction will be derived from the negative tool x-axis.

Continues on next page

`[\RefVectZ]`

Data type:`robtarget`

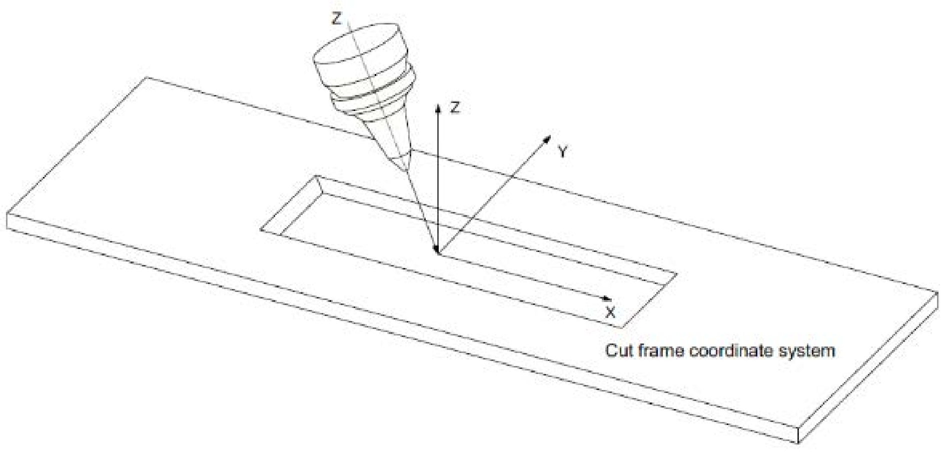
`[\RefVectZ]` is a unit vector that represents the z axis direction of the cut frame which is related to the wobj coordination system. The z direction of the default cut frame is neglected. If it is not used, the z-direction will be derived from the negative direction of tool z-axis.



`[\BiasCut]`

Data type:`switch`

`BiasCut` is used only when `RefVectZ` is defined. If `BiasCut` is used, the z direction of the TCP is not parallel with the z direction of the cut frame (defined by `RefVectZ`). When using `BiasCut`, it is allowed to cut not perpendicularly to the surface. The maximum angle between the z axis of TCP and the cut frame is 20 degrees.



Continues on next page

`[\BevelAngle]`

Data type:`num`

`[\BevelAngle]` produces a given number of bevel angle in the cutting face. Then user can get a better correction result from friction tuning and ILC. For detailed information, see *[\BevelAngle] on page 77*.

`Speed`

Data type:`speeddata`

`Speed` data applies to approaching movements. It defines the velocity of the TCP during the approach to the ToPoint or start (piercing) poin.Additionally, it defines the speed of the tool's reorientation and the speed of any uncoordinated additional axes.

`X`

Data type:`num`

`X` defines the width of the shape in millimeters. The minimum value is 3mm.

`Y`

Data type: `num`

`Y` defines the height of the shape in millimeters. The minimum value is 3mm.

`Radius`

Data type: `num`

`Radius` defines the corner radius of the rectangle in mm. The minimum value is 0.2mm. In case 0.2mm < Radius < 0.7mm, the corner radius is 0.2mm.

`LeadParam`

Data type:`leaddata`

`LeadParam` defines the lead-in/lead-out and overlap parameters.

`CutDirection`

Data type:`cwdirection`

`CutDirection` defines the cut direction: Inside/outside cut clockwise / counter-clockwise. (`iCW/iCCW/oCW/oCCW`)

`LsTableConf`

Data type:`LsTableConf`

`LSTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type: `cutdata/LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutRectL/CutRectJ`. Argument LsCutParams defines the laser cutting specific process data for `LsCutRectL/LsCutRectJ`, including cut speed and parameters for controlling head, laser, cutting gas during piercing and cutting phases.When

Continues on next page

both `LsCutParams` and `LsTableConf` are used in the instruction, the data of `LsCutParams` would be used.

`[\FrcL]`

Data type:`fricdata`

`[\FrcL]` is the friction level of 6 joints of the robot which is used during cutting this shape. Its value is auto calculated to improve the path accuracy of cutting the shape.Up to 5 global tuning data can be used in different cutting instructions.Normal cutting data generated by users should be a unique and persistent variable for each cut shape instruction. The value is larger than 0 and less than 300 and varies at different locations in robot work range. It is very important for cutting a good circle.

`[\Ilc]`

Data type:`switch`

`[\Ilc]` is used for switching iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3. for iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3.

`[\Offset]`

Data type: `shapeoffsetdata`

`[\Offset]` is an optional parameter that defines the shape offset specific parameters. It is used for small deviation compensation in location and dimension according to the measurement in production.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool and it should be define precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (wobj0).

`[\FeatureName]`

Data type:`string`

`[\FeatureName]` is an optional parameter that defines the feature name.

Continues on next page

Program execution

Shape calculation

System calculates the shape path according to the shape parameters. It takes some time to complete the calculation. This procedure is performed during running program for the first cycle in Auto mode and recalculated for every cycle in Manual mode. The result is stored in path memory. It will be performed again after the tuning or reset path memory by moving PP to Main or from HMI.

Controlling process equipment

The entire cutting process and each of its phases are coordinated with the robot movements. The robot moves to the start position of the lead-in path. The cutting process is initiated (surface search, laser ramping, piercing, etc.). The robot cuts to the end position (including lead-out). The cutting process is terminated. See the process signals.TThe robot cuts to the end position

The cutting process is influenced by the cutting state blocking.

Tuning execution

If the arguments of `Frcl` and `ILC` are used, the robot will move along the shape path in the cut direction and in the reverse direction repeatedly, until system finds the optimal result for the shape. It takes much longer time (25x) than cutting a shape in normal way. Only running tuning after all the shape and cut data are correctly set. And it can be run in auto mode.

The turning is activated and deactivated by the cutting state blocking.

Execution in manual mode

The robot moves to the center of the shape (`ToPoint`), stops, calculates the shape, and then moves to the start (piercing) point, stops, starts the cutting process and further movements.

Execution in auto mode

For the first cycle, the program is run with shape calculation, which is the same as in manual mode. Since the 2nd cycle, the robot moves to the start (pierce) point directly, and then starts the cutting process and movement.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and the shape is skipped.

Backward

The robot moves to the ToPoint position and the shape is skipped.

Continues on next page

Error handling

The following recoverable errors are generated and can be handled in an error handler. The system variable `ERRNO` will be set to:

ER-RNO	Argument	Error/Warning	Description
5100	`FeatureId`	Feature ID is out of Range	• FeatureId: -1 out of range. • FeatureId: 601 out of range.
5101		Feature ID is conflicting	FeatureId: 10 Has been used
5050	`X,Y` or `Radius`	Shape calculation error	X,Y< 2.5 mm, Radius< 0.7mm
5102	`RefVectZ`	Align angle too large	the angle difference between Vectz and negtive z axis of ToPoint >= 20

Syntax

```
LsCutRectL
  [FeatureId ’:=’] < expression (IN) of shpno> ‘,’
  [ ToPoint ’:=’] < expression (IN) of robtarget >
  [‘\’RefVectX ‘:=’ <expression (IN) of robtarget>]
  [‘\’RefVectZ ‘:=’ <expression (IN) of robtarget>]
  [‘\’BiasCut]
  |['\'BevelAngle]‘,’
  [Speed ‘:=’] < expression (IN) of speeddata > ‘,’
  [X ‘:=’] < expression (IN) of num > ‘,’
  [Y ‘:=’] < expression (IN) of num > ‘,’
  [Radius ‘:=’] < expression (IN) of num > ‘,’
  [LeadParam ‘:=’] < expression (IN) of leaddata > ‘,’
  [CutDirection ‘:=’] < expression (IN) of cwdirection > ‘,’
  [LsTableConf’:=’] < expression (IN) of LsTableConf> ‘,’
  [‘\’LsCutParams‘:=’] < expression (IN) of cuddata | LsCutParams
       > ‘,’
  [‘\’FrcL‘:=’ <persistent (PERS) of fricdata>]
  [‘\’Ilc]
  [‘\’Offset ‘:=’< expression (IN) of shapeoffsetdata>]’,’
  [Tool ‘:=’] < persistent (PERS) of tooldata > ‘,’
  [‘\’WObj ‘:=’ <persistent (PERS) of wobjdata>]
  [‘\’FeatureName ‘:=’ <expression (IN) of string>]
```

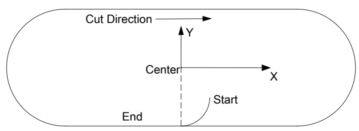
5.1.7 CutSlotL/CutSlotJ/LsCutSlotL/LsCutSlotJ

Usage

`CutSlotL`/`CutSlotJ`/`LsCutSlotL`/`LsCutSlotJ` is used to cut a slot in the work piece. It provides tuning switches to improve the shape accuracy, and the offset argument to adjust the deviation. The instruction has the following functions:

- Start the cutting process at start (piercing) point
- Cut the shape with the specific lead-in and lead-out path
- Stop the cutting process at specific position on the lead-out path
- Auto tune the friction level of robot and reference points to improve the shape accuracy

The difference between `CutSlotL` and `CutSlotJ` is they use different movement types (MoveL/MoveJ) to approach the start point. `CutSlotL`/`CutSlotJ` and `LsCutSlotL`/`LsCutSlotJ` use different datatypes of cutting parameters which are defined for different cutting processes. `CutSlotL`/`CutSlotJ` and `Cutdata` are for general cutting. `LsCutSlotL`/`LsCutSlotJ` and `LsCutParams` are for laser cutting process. Although `LsCutSlotJ` is used in the examples, `CutSlotL`/`CutSlotJ` or `LsCutSlotL` could also be used.



Basic examples

The following example illustrates the instruction `LsCutSlotJ`.

Example 1

```
LsCutSlotJ 6, p6, v1000, 20, 5, Arc3_r3_o3, iCW, TableConf21,
     tLaser\Wobj:=wobjPart;
```

The robot moves to the start (piercing) point with a speed v1000. A slot is cut that is referenced to the default cut frame with the center p6, width of 20mm, height of

Continues on next page

5mm. `Arc3_r3_o3` defines the leadin/leadout path as an arc leadin path with radius of 3 mm and overlap of 3 mm. And the cut direction is clockwise (`iCW`). Cutting proces data can be obtained from parameter tables by `TableConf21`, including, for example, cut speed, cut power and pierce time, etc. The `tooldata` used is `tLaser` and work object is `wobjPart`.

Arguments

```
LsCutSlotJ FeatureId, ToPoint, [\RefVectX], [\RefVectZ],
     [\BiasCut]|[\BevelAngle], Speed, X, Y, LeadParam,
     CutDirection, LsTableConf, [\LsCutParams], [\FrcL], [\Ilc],
     [\Offset], Tool, [\WObj], [\FeatureName]
```

`FeatureId`

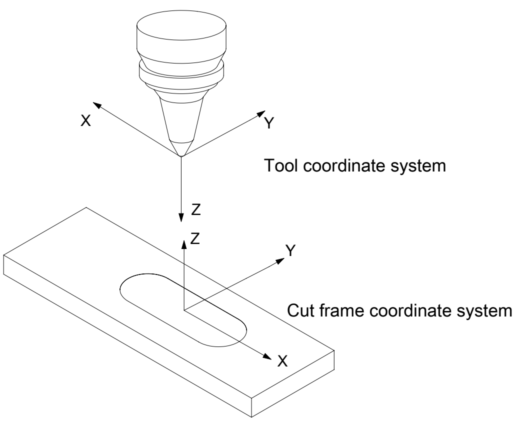
Data type: `shpno`

`FeatureID` is a unique integer value that is used as an array pointer for feature specific data stored in the system. Its data range is from 1 to 600.

`ToPoint`

Data type:`robtarget`

`ToPoint` defines the reference point of the shape (typically the center) and the default cut frame. The x and z axes of the default cut frame is in the reversed direction of x and z axes of TCP. And Y axis is in the same direction of Y axis of TCP.



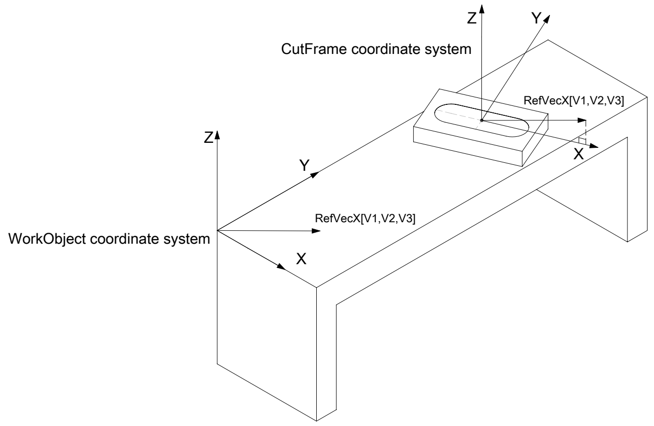
`[\RefVectX]`

Data type:`robtarget`

`[\RefVectX]` is a unit vector that represents the X axis direction of the cut frame that is related to the wobj coordination system. The x direction of the default cut frame is neglected. It is useful to adjust the TCP without changing the shape

Continues on next page

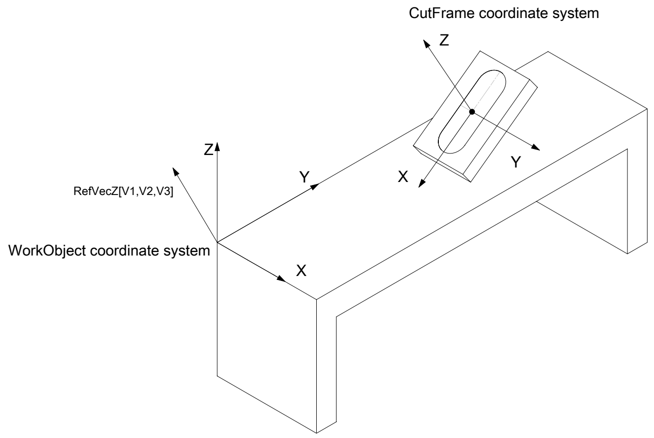
direction. If it is not used, the x-direction will be derived from the negative tool x-axis.



`[\RefVectZ]`

Data type:`robtarget`

`[\RefVectZ]` is a unit vector that represents the z axis direction of the cut frame which is related to the wobj coordination system. The z direction of the default cut frame is neglected. If it is not used, the z-direction will be derived from the negative direction of tool z-axis.



Continues on next page

`[\BiasCut]`

Data type:`switch`

`BiasCut` is only used when `RefVectZ` is defined. If `BiasCut` is used, the z direction of the TCP is no parallel with the z direction of the cut frame (defined by `RefVectZ`). With `BiasCut`, it is allowed to cut no perpendicularly to the surface. The maximum angle between z axis of TCP and Cut Frame is 20 degree.

`[\BevelAngle]`

Data type:`num`

`[\BevelAngle]` produces a given number of bevel angle in the cutting face. Then user can get a better correction result from friction tuning and ILC. For detailed information, see *[\BevelAngle] on page 77*.

`Speed`

Data type:`speeddata`

`Speed` applies to approaching movements. It defines the velocity of the TCP during the approach to the ToPoint or start (piercing) point, as well as the speed of the tool's reorientation and the speed of any uncoordinated additional axes.

`X`

Data type:`num`

`X` defines the width of the shape in mm. The minimum value is 3mm.

`Y`

Data type:`num`

`Y` defines the height of the shape in mm. The minimum value is 3mm.Note: X should bigger than Y.

`LeadParam`

Data type:`leaddata`

`LeadParam` defines the lead-in/lead-out and overlap parameters.

`CutDirection`

Data type:`cwdirection`

`CutDirection` defines the cut direction: Inside/outside cut clockwise / counter-clockwise. (`iCW`/`iCCW`/`oCW`/`oCCW`)

`LsTableConf`

Data type:`LsTableConf`

`LSTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutSlotL`/`CutSlotJ`. `LsCutParams` defines the laser cutting specific process data for `LsCutSlotL`/`LsCutSlotJ`, including cut speed and parameters for

Continues on next page

controlling head, laser, cutting gas during piercing and cutting phases.When both `LsCutParams` and `LsTableConf` are used in the instruction, the data of `LsCutParams` would be used.

`[\FrcL]`

Data type:`fricdata`

`[\FrcL]` is the friction level of 6 joints of the robot which is used during cutting this shape. Its value is auto calculated to improve the path accuracy of cutting the shape.Up to 5 global tuning data can be used in different cutting instructions.Normal cutting data generated by users should be a unique and persistent variable for each cut shape instruction. The value is larger than 0 and less than 300 and varies at different locations in robot work range. It is very important for cutting a good circle.

`[\Ilc]`

Data type:`switch`

`[\Ilc]` is used for switching iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3. iteration learning control. Fine tune of robot performance after all the other tuning methods are used (including friction tuning). Small corrections are auto introduced into reference points of the cut shape perimeter according to the deviation. Default iteration times of ILC is 3.

`[\Offset]`

Data type:`shapeoffsetdata`

`[\Offset]` is an optional parameter that defines the shape offset specific parameters. It is used for small deviation compensation in location and dimension according to the measurement in production.

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (wobj0).

`[\FeatureName]`

Data type:`string`

`[\FeatureName]` is an optional parameter that defines the feature name.

Program execution

Shape calculation

System calculates the shape path according to the shape parameters. It takes some time to complete the calculation. This procedure is performed during running program for the first cycle in Auto mode and recalculated for every cycle in Manual mode. The result is stored in path memory. It will be performed again after the tuning or reset path memory by moving PP to Main or from HMI.

Continues on next page

Controlling process equipment

The entire cutting process and each of its phases are coordinated with the robot movements. The robot moves to the start position of the lead-in path. The cutting process is initiated (surface search, laser ramping, piercing, etc.). The robot cuts to the end position (including lead-out). The cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Tuning execution

If the arguments of `Frcl` and `ILC` are used, the robot will move along the shape path in the cut direction and in the reverse direction repeatedly, until system finds the optimal result for the shape. It takes much longer time (25x) than cutting a shape in normal way. Only running tuning after all the shape and cut data are correctly set. And it can be run in auto mode.

The turning is activated and deactivated by the cutting state blocking.

Execution in manual mode

The robot moves to the center of the shape (`ToPoint`), stops, calculates the shape, and then moves to the start (piercing) point, stops, starts the cutting process and further movements.

Execution in auto mode

For the first cycle, the program is run with shape calculation, which is the same as in manual mode. Since the 2nd cycle, the robot moves to the start (pierce) point directly, and then starts the cutting process and movement.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and the shape is skipped.

Backward

The robot moves to the ToPoint position and the shape is skipped.

Error handling

The following recoverable errors are generated and can be handled in an error handler. The system variable `ERRNO` will be set to:

ERRNO	Argument	Error/Warning	Description
5100	`FeatureId`	Feature ID is out of Range	• FeatureId: -1 out of range. • FeatureId: 601 out of range.
5101		Feature ID is conflicting	FeatureId: 10 Has been used
5050	`X,Y`	Shape calculation error	Y< 1.4mm, X<=Y
5102	`RefVectZ`	Align angle too large	the angle difference between Vectz and negtive z axis of ToPoint >= 20

Syntax

```
LsCutSlotL
  [FeatureId ’:=’] < expression (IN) of shpno> ‘,’
  [ ToPoint ’:=’] < expression (IN) of robtarget >
```

Continues on next page

```
[‘\’RefVectX ‘:=’ <expression (IN) of pos>]
[‘\’RefVectZ ‘:=’ <expression (IN) of pos>]
[‘\’BiasCut]
|['\'BevelAngle]‘,’
[Speed ‘:=’] < expression (IN) of speeddata > ‘,’
[X ‘:=’] < expression (IN) of num > ‘,’
[Y ‘:=’] < expression (IN) of num > ‘,’
[LeadParam ‘:=’] < expression (IN) of leaddata > ‘,’
[CutDirection ‘:=’] < expression (IN) of cwdirection > ‘,’
[LsTableConf’:=’] < expression (IN) of LsTableConf> ‘,’
[‘\’LsCutParams‘:=’] < expression (IN) of cuddata | LsCutParams
     > ‘,’
[‘\’FrcL‘:=’ <persistent (PERS) of fricdata>]
[‘\’Ilc]
[‘\’Offset ‘:=’< expression (IN) of shapeoffsetdata>]’,’
[Tool ‘:=’] < persistent (PERS) of tooldata > ‘,’
[‘\’WObj ‘:=’ <persistent (PERS) of wobjdata>]
[‘\’FeatureName ‘:=’ <expression (IN) of string>]
```

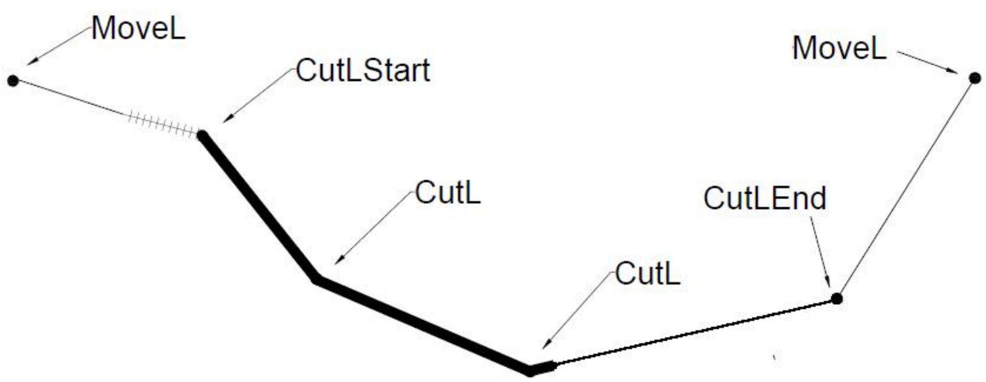
5.1.8 CutLStart/LsCutLStart

Usage

`CutLStart`/`LsCutLStart` is used to begin cutting at the start point. This argument should be used to start a free form cut path. The instruction has the following functions:

- Move to the start point
- Set the purge gas signal before reach the start point
- Stop at the start point and start the cut process

`CutLStart` and `LsCutLStart` use different datatypes of cut parameters which are defined for different cutting processes. `CutLStart` and `cutdata` are for general cutting. `LsCutLStart` and `LsCutParams` or `LsTableConf` are for laser cutting process. Although `LsCutLStart` is used in the examples, `CutLStart` could also be used.



Basic example

The following example illustrates the instruction `LsCutLstart`.

Example 1

```
LsCutLStart pStart, v1000, TableConf21, fine, tLaser\Wobj:=wobjPart;
```

The robot moves to `pStart` point with a speed v1000. The cut begins at `pStart` after robot stop. Cutting process data can be obtained from parameter tables by `TableConf21`, including, for example, cut speed, cut power and pierce time, etc.

Arguments

```
LsCutLStart ToPoint, [\ID], Speed, LsTableConf, [\LsCutParams],
     Zone, Tool, [\WObj], [\SeamName]
```

`ToPoint`

Data type:`robtarget`

ToPoint is the start point of the free form cut path.

`[\ID]`

Data type:`identno`

Used to control the synchronization of two or more coordinated synchronized movements.

Continues on next page

The data type identno can only be used in a MultiMove system with the option Coordinated Robots, and can be used only in program tasks defined as Motion Task.

`Speed`

Data type:`speeddata`

`Speed` defines the velocity of the robot and of the external axes while approaching `ToPoint`.

`LsTableConf`

Data type:`LsTableConf`

`LSTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`cutdata` defines the general cutting process data for `CutLStart`. `LsCutParams` defines the laser cutting specific process data for `LsCutLStart`, including cut speed and parameters for controlling head, laser, cutting gas during piercing and cutting phases. The cut speed will be used for following movement along the free form path.When both `LsCutParams` and `LsTableConf` are used in the instruction, the data of `LsCutParams` would be used.

`Zone`

Data type:`zonedata`

`Zone` defines how close the axes must be to the programmed position before they can start moving towards the next position. Fly-by points should be used for all cutting positions except the start point. A corner path is generated past the `ToPoint`.

`Tool`

Data type:`tooldata`/`LsCutParams`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of workpiece. The cut frame is related to the tool and it should be define precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`/`LsCutParams`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (wobj0).

`[\SeamName]`

Data type:`string`/`LsCutParams`

Optional parameter that is used to enter the string of the Seam name.

Continues on next page

Program execution

Controlling process equipment

The entire cutting process and each of its phases are coordinated with the robot movements. The robot moves to the start position of the lead-in path. The cutting process is initiated (surface search, laser ramping, piercing, etc.). The robot cuts to the end position (including lead-out). The cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Execution in stepwise mode

Forward

The robot moves to the `ToPoint` position and cut process is skipped.

Backward

The robot moves to the `ToPoint` position and cut process is skipped.

Syntax

```
LsCutLStart
  [ ToPoint ':='] < expression (IN) of robtarget >
  ['\'ID ':=' < expression (IN) of identno > ] ','
  [Speed ':='] < expression (IN) of speeddata >
  [LsTableConf':='] < expression (IN) of LsTableConf> ','
  ['\'LsCutParams':='] < expression (IN) of LsCutParams > ] ','
  [Zone':='] < expression (IN) of zonedata > ','
  [Tool ':='] < persistent (PERS) of tooldata >
  ['\'WObj ':=' <persistent (PERS) of wobjdata>]
  ['\SeamName ':=' < expression (IN) of string>]
```

5.1.9 CutC/LsCutC

Usage

CutC/LsCutC is used to cut a circular curve from the current position to ToPoint. It can also trigger I/O signals during movement.

CutC and LsCutC use different datatypes of cut parameters which are defined for different cutting processes. CutC and cutdata are for general cutting. LsCutC and LsCutParams or LsTableConf are for laser cutting process. Although LsCutC is used in the examples, CutC could also be used.

Basic example

The following example illustrates the instruction LsCutLstart.

Example 1

```
LsCutLStart pStart, v1000, TableConf21, fine,
     tLaser\Wobj:=wobjPart;LsCutC p1, p2, v100, z1,
     tLaser\Wobj:=wobjPart;
```

The cut starts at pStart. Then cut continues in a circular path through p1 to p2. Cutting process data can be obtained from parameter tables by TableConf21.

Arguments

```
LsCutC CirPoint, ToPoint, [\ID], Speed, [\T1], [\T2], [\T3], [\T4],
     LsTableConf, [\LsCutParams], Zone, Tool, [\WObj]
```

CirPoint

Data type:robtarget

Argument CirPoint is the circle point of the circular path. The circle point is a position on the circle between the start point and the destination point. To obtain the best accuracy, it should be placed about halfway between the start and destination points. If it is placed too close to the start or destination point, the robot may give a big movement.

ToPoint

Data type:robtarget

ToPoint is the destination position of the robot and external axes.

[\ID]

Data type:identno

\ID is used to control synchronizing of two or more coordinated synchronized movements with each other.

The data type identno can only be used in a MultiMove system with option Coordinated Robots and only in program tasks defined as Motion Task.

Speed

Data type:speeddata

Speed data applies to step forward/backward movements. The speed of the cut process is given by LsCutParams or be obtained by LsTableConf.

Continues on next page

`[\T1],[\T2],[\T3],[\T4]`

Data type:`triggdata`

T1,T2,T3,T4 is the triggdata for periphery equipment. Totally four triggdata are for customized behavior. No priority between four triggdata.

`LsTableConf`

Data type:`LsTableConf`

`LsTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutC`. `LsCutParams` defines the laser cutting specific process data for `LsCutC`, including cut speed and parameters for controlling head and laser during cutting phases. Only use it when the cut process data has to be changed. E.g. at the corner of the trim path. If no `LsCutParams`, the previous `LsCutParams.CutSpeed` will be used.When both `LsCutParams` and `LsTableConf` are used in the instruction, the data of `LsCutParams` would be used.

`Zone`

Data type:`zonedata`

`Zone` defines how close the axes must be to the programmed position before they can start moving towards the next position. Fly-by points should be used for all cutting positions except the start point. A corner path is generated past the ToPoint.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool and it should be define precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (wobj0).

Program execution

Controlling process equipment

When argument `udTableConf` or `LsCutParams` is used, the signal cwdoDataChange will be set at the beginning of the movement. The change of process data, e.g. power, will be sent to related equipment.

The cutting process is influenced by the cutting state blocking.

Continues on next page

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and cut process is skipped.

Backward

The robot moves to the ToPoint position and cut process is skipped.

Syntax

```
LsCutC
  [CirPoint ':='] < expression (IN) of robtarget >','
  [ ToPoint ':='] < expression (IN) of robtarget >','
  ['\'ID ':=' < expression (IN) of identno > ]
  [Speed ':='] < expression (IN) of speeddata > ','
  ['\' T1 ':=' <persistent (VAR) of triggdata>]
  ['\' T2 ':=' <persistent (VAR) of triggdata>]
  ['\' T3 ':=' <persistent (VAR) of triggdata>]
  ['\' T4 ':=' <persistent (VAR) of triggdata>]
  [LsTableConf':='] < expression (IN) of LsTableConf> ','
  ['\'LsCutParams':='] < expression (IN) of LsCutParams >]','
  [Zone':='] < expression (IN) of zonedata > ','
  [Tool ':='] < persistent (PERS) of tooldata >','
  ['\'WObj ':=' <persistent (PERS) of wobjdata>]
```

5.1.10 CutL/LsCutL

Usage

CutL/LsCutL is used to cut a linear path from the current position ToPoint. Meanwhile, it can trigger signals during movement.

CutL and LsCutL use different datatypes of LsCutParams which are defined for different cutting processes. CutL and cutdata are for general cutting. LsCutL and LsCutParams are for laser cutting process. Although LsCutL is used in the examples, CutL could also be used.

Basic examples

The following example illustrates the instruction LsCutL.

Example 1

```
LsCutLStart pStart, v1000, LsCutParams1, fine,
     tLaser\Wobj:=wobjPart;
LsCutL p3, v100, z1, tLaser\Wobj:=wobjPart;
```

The cut starts at pStart. Then the cut continues to p3 along a straight line. The cut speed is defined in LsCutParams1.

Arguments

```
LsCutL ToPoint, [\ID], Speed, [\T1], [\T2], [\T3], [\T4],
     LsTableConf, [\LsCutParams], Zone, Tool, [\WObj],
```

ToPoint

Data type:robtarget

ToPoint is the destination position of the robot and external axes.

[\ID]

Data type:identno

\ID is used to control synchronizing of two or more coordinated synchronized movements with each other.

The data type identno can only be used in a MultiMove system with option Coordinated Robots and only in program tasks defined as Motion Task.

Speed

Data type:speeddata

Speed data applies to step forward/backward movements. The speed of the cut process is given by LsCutParams or be obtained by LsTableConf.

[\T1], [\T2], [\T3], [\T4]

Data type:triggdata

T1,T2,T3,T4 is the triggdata for periphery equipment. Totally four triggdata are for customized behavior. No priority between four triggdata.

LsTableConf

Data type:LsTableConf

Continues on next page

`LsTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutL`. `LsCutParams` defines the laser cutting specific process data for `LsCutL`, including cut speed and parameters for controlling head and laser during cutting phases. Only use it when the cut process data has to be changed. E.g. at the corner of the trim path. If there is no `LsCutParams`, the previous `LsCutParams.CutSpeed` will be used.When both LsCutParams and `LsTableConf` are used in the instruction, the data of LsCutParams would be used.

`Zone`

Data type:`zonedata`

`Zone` defines how close the axes must be to the programmed position before they can start moving towards the next position. Fly-by points should be used for all cutting positions except the start point. A corner path is generated past the ToPoint.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of work piece. The cut frame is related to the tool and it should be define precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (wobj0).

Program Execution

Controlling process equipment

When argument `LsTableConf` or `LsCutParams` is used, the signal cwdoDataChange will be set at the beginning of the movement. The change of process data, e.g. power, will be sent to related equipment.

The cutting process is influenced by the cutting state blocking.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and cut process is skipped.

Backward

The robot moves to the ToPoint position and cut process is skipped.

Syntax

```
LsCutL
  [ ToPoint ':='] < expression (IN) of robtarget >','
```

Continues on next page

```
['\'ID ':=' < expression (IN) of identno > ]
[Speed ':='] < expression (IN) of speeddata > ','
['\' T1 ':=' <persistent (VAR) of triggdata>]
['\' T2 ':=' <persistent (VAR) of triggdata>]
['\' T3 ':=' <persistent (VAR) of triggdata>]
['\' T4 ':=' <persistent (VAR) of triggdata>]
[LsTableConf’:=’] < expression (IN) of LsTableConf> ‘,’
['\' LsCutParams':='] < expression (IN) of LsCutParams >]
[Zone':='] < expression (IN) of zonedata > ','
[Tool ':='] < persistent (PERS) of tooldata >
['\'WObj ':=' <persistent (PERS) of wobjdata>]
```

5.1.11 CutCEnd/LsCutCEnd

Usage

CutCEnd/LsCutCEnd is used for stopping the cutting process. It has the following functions.

- Move a circular path to ToPoint.
- Stop the cutting process on-fly at the current position or at the fine point according to the LsCutParams used on the path.

CutCEnd and LsCutCEnd use different datatypes of LsCutParams which are defined for different cutting processes. CutCEnd and cutdata are for general cutting. LsCutCEnd and LsCutParams are for laser cutting process. Although LsCutCEnd is used in the examples, CutCEnd could also be used.

Basic examples

The following example illustrates the instruction LsCutCEnd.

Example 1

```
LsCutLStart pStart, v1000, LsCutParams1, fine,
     tLaser\Wobj:=wobjPart;
LsCutC p1, p2, v100, z1, tLaser\Wobj:=wobjPart;
LsCutCEnd p4, p5, v100,, z1, tLaser\Wobj:=wobjPart;
```

The cut starts at pStart. Then cut continues in a circular path through p1 to p2. Then stop the cutting process on-fly. The robot moves along a circular path through P4 to p5. The cut speed is defined in LsCutParams1.

Arguments

```
LsCutCEnd CirPoint, ToPoint, [\ID], Speed, LsTableConf,
     [\LsCutParams], Zone, Tool,[\WObj]
```

CirPoint

Data type:robtarget

CirPoint is the circle point of the circular path. The circle point is a position on the circle between the start point and the destination point. To obtain the best accuracy, it should be placed about halfway between the start and destination points. If it is placed too close to the start or destination point, the robot may give a big movement.

ToPoint

Data type:robtarget

ToPoint is the destination position of the robot and external axes.

[\ID]

Data type:identno

\ID is used to control synchronizing of two or more coordinated synchronized movements with each other.

The data type identno can only be used in a MultiMove system with option Coordinated Robots and only in program tasks defined as Motion Task.

Continues on next page

`Speed`

Data type:`speeddata`

`Speed` data applies to step forward/backward movements. The speed of the cut process is given by `LsCutParams` or be obtained by `LsTableConf`.

`LsTableConf`

Data type:`LsTableConf`

`LsTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutCEnd`. `LsCutParams` defines the laser cutting specific process data for `LsCutCEnd`, including cut speed and parameters for controlling head and laser during cutting phases. If there is no `LsCutParams`, the previous `LsCutParams.CutSpeed` will be used. When both `LsCutParams` and `LsTableConf` are used in the instruction, the data of `LsCutParams` would be used.

`Zone`

Data type:`zonedata`

`Zone` defines how close the axes must be to the programmed position before they can start moving towards the next position. Fly-by points should be used for all cutting positions except the start point. A corner path is generated past the `ToPoint`.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool and it should be define precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (`wobj0`).

Program Execution

Controlling process equipment

The cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Execution in stepwise mode

Forward

The robot moves to the ToPoint position and cut process is skipped.

Continues on next page

Backward

The robot moves to the ToPoint position and cut process is skipped.

Syntax

```
LsCutCEnd
  [CirPoint ':='] < expression (IN) of robtarget >
  [ ToPoint ':='] < expression (IN) of robtarget >
  ['\'ID ':=' < expression (IN) of identno > ] ','
  [Speed ':='] < expression (IN) of speeddata > ','
  ['\'LsCutParams':='] < expression (IN) of LsCutParams > ','
  [Zone':='] < expression (IN) of zonedata > ','
  [Tool ':='] < persistent (PERS) of tooldata >
  ['\'WObj ':=' <persistent (PERS) of wobjdata>]
```

5.1.12 CutLEnd/LsCutLEnd

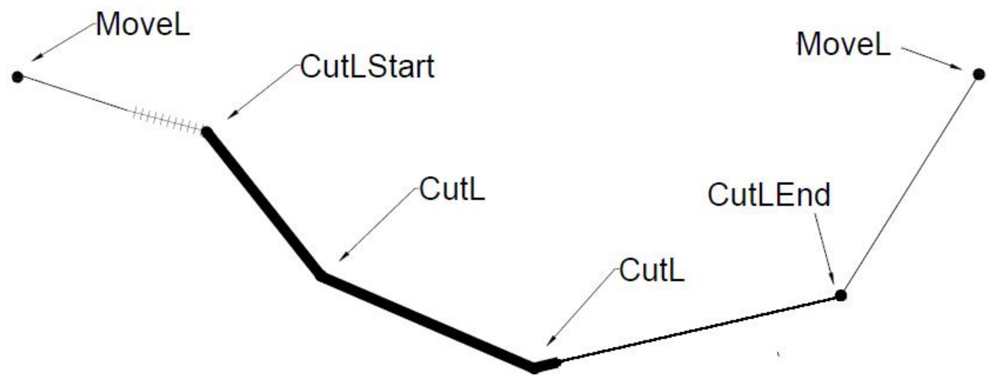
Usage

`CutLEnd`/`LsCutLEnd` is used for stopping the cutting process. It has following functions.

- Move a linear path to ToPoint.
- Stop the cutting process on-fly at the current position or at the fine point according to the LsCutParams used on the path.

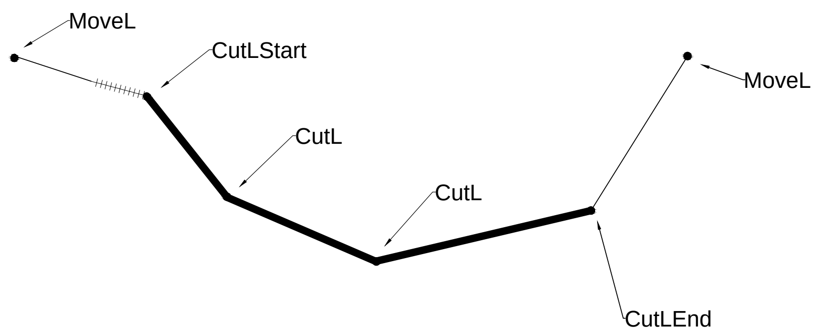
`CutLEnd` and `LsCutLEnd` use different datatypes of cut parameters which are defined for different cutting processes. `CutLEnd` and `cutdata` are for general cutting. `LsCutLEnd` and `LsCutParams` are for laser cutting process. Although `LsCutLEnd` is used in the examples, `CutLEnd` could also be used.

If the end point of the cutting path is a zone point, the cutting path would be following:



Continues on next page

If the end point of the cutting path is a fine point, the cutting path would be following:



Basic examples

The following example illustrates the instruction `LsCutLEnd`.

Example 1

```
LsCutLStart pStart, v1000, LsCutParams1, fine,
     tLaser\Wobj:=wobjPart;
LsCutL 3, v100, z1, tLaser\Wobj:=wobjPart;
LsCutLEnd p4, v100, z1, tLaser\Wobj:=wobjPart;
```

The cut starts at `pStart`. Then the cut continues to `p3` along a straight line. Then stop the cutting process on-fly. The robot moves to `p4`. The cut speed is defined in `LsCutParams1`.

Arguments

```
LsCutLEnd ToPoint, [\ID], Speed, LsTableConf, [\LsCutParams], Zone,
     Tool, [\WObj]
```

`ToPoint`

Data type:`robtarget`

Argument ToPoint is the destination position of the robot and external axes.

`[\ID]`

Data type:`identno`

\ID is used to control synchronizing of two or more coordinated synchronized movements with each other.

Continues on next page

The data type identno can only be used in a MultiMove system with option Coordinated Robots and only in program tasks defined as Motion Task.

`Speed`

Data type:`speeddata`

`Speed` data applies to step forward/backward movements. The speed of the cut process is given by `LsCutParams` or be obtained by `LsTableConf`.

`LsTableConf`

Data type:`LsTableConf`

`LsTableConf` defines the configuration of the current active LS cutting parameter table. i.e. the combination of the cutting group and the piercing group of the LS cutting parameter table, which are used in the cutting instruction.

`[\LsCutParams]`

Data type:`cutdata`/`LsCutParams`

`[\LsCutParams]` defines the general cutting process data for `CutLEnd`. `LsCutParams` defines the laser cutting specific process data for `LsCutLEnd`, including cut speed and parameters for controlling head and laser during cutting phases. If there is no `LsCutParams`, the previous `LsCutParams.CutSpeed` will be used. When both `LsCutParams` and `LsTableConf` are used in the instruction, the data of `LsCutParams` would be used.

`Zone`

Data type:`zonedata`

`Zone` defines how close the axes must be to the programmed position before they can start moving towards the next position. Fly-by points should be used for all cutting positions except the start point. A corner path is generated past the ToPoint.

`Tool`

Data type:`tooldata`

`Tool` is the tool used in the course of movement. The z-axis of the tool should be perpendicular to the surface of the work piece. The cut frame is related to the tool and it should be define precisely to achieve good cut accuracy.

`[\WObj]`

Data type:`wobjdata`

`[\WObj]` is the work object coordinate system to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system (wobj0).

Program Execution

Controlling process equipment

The cutting process is terminated. See the process signals.

The cutting process is influenced by the cutting state blocking.

Execution in stepwise mode

Forward

Continues on next page

The robot moves to the ToPoint position and cut process is skipped.

Backward

The robot moves to the ToPoint position and cut process is skipped.

Syntax

```
LsCutLEnd
  [ ToPoint ':='] < expression (IN) of robtarget >
  ['\'ID ':=' < expression (IN) of identno > ] ','
  [Speed ':='] < expression (IN) of speeddata > ','
  [LsTableConf':='] < expression (IN) of LsTableConf> ','
  ['\'LsCutParams':='] < expression (IN) of LsCutParams > ','
  [Zone':='] < expression (IN) of zonedata > ','
  [Tool ':='] < persistent (PERS) of tooldata >
  ['\'WObj ':=' <persistent (PERS) of wobjdata>]
```

5.1.13 SetCutWareState

Usage

`SetCutWareState` is used to activate and deactivate different functions of cutting. It has the same function as the state blocking window of Cutting HMI.

Basic examples

The following example illustrates the instruction `SetCutWareState`.

Example 1

```
SetCutWareState \CutProcess;
```

All the process and equipment signals are activated. Cut instructions will start the cutting process. The result of last tuning are used for each shape instruction, however, the friction tuning and Ilc fine tuning won't be executed.

Arguments

```
SetCutWareState [\switch Process]|[\switch CutProcess],[\switch
     FrictionTune], [\switch IlcInit]|[\switch
     IlcContinue],[\switch ResetPathMemory]
```

`[\Process]|[CutProcess]`

Data type:`switch`

`[\Process]` is to only activate part of the process control but not to cut the workpiece. E.g. enable the cut head control, but not activate the laser power source and gas. It is used to verify the path and IO signals during a dry run of the program.

`[\CutProcess]` is to activate the all the process controls. Cutting instructions will cut the workpiece. It is used in production.

`[\FrictionTune]`

Data type:`switch`

`[\FrictionTune]` is used to activate the advance shape tuning. Shape cut instructions will run the tuning motion if the `/FrcL` argument is used.

`[\IlcInit]|[\IlcContinue]`

Data type:`switch`

`[\IlcInit] |[\ IlcContinue]` is to activate the iterative learning control as fine tuning of the shape. Shape cut instructions will run the tuning motion if the `/Ilc` argument is used. `IlcInit` start the calculation from zero and `IlcContinue` continues the calculation based on the result of last time.

`[\ResetPathMemory]`

Data type:`switch`

`[\ResetPathMemory]` cleans the results of shape generation. Normally, the path memory will be reset automatically in manual mode. It is a way to trigger recalculation of the shape generation in Auto mode. E.g. recalculate the shape after the offset data is changed in production.

Continues on next page

Syntax

```
SetCutWareState
  ['\'Process]
  |['\'CutProcess]
  ['\'FrictionTune]
  ['\'IlcInit]
  |['\'IlcContinue]
  ['\'ResetPathMemory] ','
```

5.2 Data types

5.2.1 LsTableHead

Usage

`LsTablehead` is used in laser cutting parameter tables to indicate the major characteristics as thickness, cutting power, focal length and so on.

> **Note**
>
> It is a built in data type for the laser table. It is not recommended for users to create their own variables in Rapid programs, because these variables could not be managed by the laser table.

Components

`CutPower`

Data type: `num`

The max power of the laser equipment. Only this argument would directly affect the cutting instruction. Other arguments are used for user's reference to select the suitable cutting table.Unit: w.

`FocalLength`

Data type: `num`

The focal length of the selected laser optic system. Unit: mm

`FocalPoint`

Data type: `num`

Focal Position of the lens with respect to the nozzle. Unit:mm

`LaserType`

Data type: `string`

Type and brand information of the laser device.

`Material`

Data type: `String`

The material that the cutting work object is made of, for example steel, Fe, and so on.

`Nozzle`

Data type: `num`

The nozzle diameter to be installed on the cutting head. Unit:mm

`Thickness`

Data type: `num`

The thickness of the work object to be cut. Unit: mm

Continues on next page

Structure

```
< dataobject of LsTableHead >
  < CutPower of num >
  < FocalLength of num >
  < FocalPoint of num >
  < LaserType of string >
  < Material of string >
  < Nozzle of num >
  < Thickness of num >
```

5.2.2 LsCuttingData

Usage

Used to control the laser-cutting process, and includes parameters for controlling the head, laser, and also cutting gas during piercing and cutting phases.

Note

It is a built in data type for the laser table. It is not recommended for users to create their own variables in Rapid programs, because these variables could not be managed by the laser table.

Components

`CutSpeed`

Data type: `num`

TCP speed during the cutting phase. The cut speed is proportional to the cut power.

Unit: mm/s, usually 10-300

`PowerLevel`

Data type: `num`

Sets the power level of the laser during the cutting phase. The actual cutting power equals to this value multiplying the max cut power. Use the 100% cut power of the laser source to achieve the highest cutting speed.

Unit: %

`MinPowerLevel`

Data type: `num`

Minimum laser power level for speed modulation power control.The minimum cutting power equals to this value multiplying the max cut power.

Unit: %

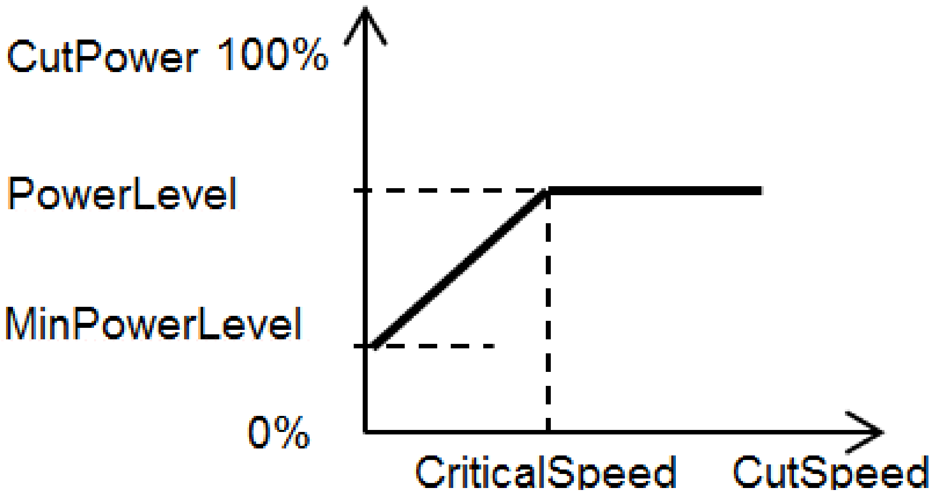
`CriticalSpeed`

Data type: `num`

In case the current speed is lower than it, the average power will be updated. If the critical speed equals to 0, then speed modulation power control is disabled.

Continues on next page

Unit: mm/s



`Acceleration`

Data type: `num`

Acceleration of the cutting speed. On fast machines, in some cases, the accelearation is reduced during cut.

Unit: mm/s^2

`CutProgram`

Data type: `num`

Defines the gas type (valve selection) to be active during the cutting phase. It depends on hardware setting. If only use one type gas during cutting, this data is ignored.

`GasFlow`

Data type: `num`

Depending on the equipment, GasFlow defines the gas flow or pressure to be used during the cutting phase.

- The part is thicker, the pressure should be higher.
- The pressure of oxygen is lower than that of nitrogen.

`HC_StandOff`

Data type: `num`

The distance between the nozzle and the material.

Unit: mm

`HC_LockOn`

Data type: `bool`

If the value is set to TRUE, the cutting head will remain locked during the entire cutting phase. It is used in 2D shape cutting processes to improve the cut quality. Should not be used in 3D free form cut instruction.

Continues on next page

`HC_CharCurve`

Data type: `num`

Selects the laser head characteristic curve. Different sensors have different, non-linear characteristic curves. Selects the characteristic curve remains linear throughout the entire work area.

`Kerf`

Data type: `num`

Kerf width compensates for material that is removed by the cutting process. The value entered should be the width of the cut path. The cut path is offset by half the Kerf width according to the cut direction. There is not offset for the path, if the `kerf` = 0. And not compensation for free form cut instructions (`LsCutL`/`LsCutC`).

Structure

```
< dataobject of LsCuttingData >
  < CutSpeed of num >
  < PowerLevel of num >
  < MinPowerLevel of num >
  < CriticalSpeed of num >
  < Acceleration of num >
  < CutProgram of num >
  < GasType of num >
  < GasFlow of num >
  < HC_StandOff of num >
  < HC_LockOn of bool >
  < HC_CharCurve of num >
  < Kerf of num >
```

5.2.3 LsPierData

Usage

Used to control the laser-cutting process, and includes parameters for controlling the head, laser, and also cutting gas during piercing and cutting phases.

> **Note**
>
> It is a built in data type for the laser table. It is not recommended for users to create their own variables in Rapid programs, because these variables could not be managed by the laser table.

Components

`GasType`

Data type: `num`

Defines the gas type (valve selection) to be active during the cutting phase. It depends on hardware setting. If only use one type gas during cutting, this data is ignored.

`GasFlow`

Data type: `num`

Depending on the equipment, this argument defines the gas flow or pressure to be used during the piercing phase.

- The part is thicker, the pressure should be higher.
- The pressure of oxygen is lower than that of nitrogen.

`PurgeTime`

Data type: `num`

Time that gas flows before the robot reaches the pierce point. Set to make the air stable enough for protecting the cutting head before cutting begins. Default value is 0.

Unit: s

`PowerLevel`

Data type: `num`

The proportion of current cutting power to the maximum cutting power. It is to be used in the piercing phase.

Unit: %

`PierceTime`

Data type: `num`

Time to pierce through the work piece. It is depends on the material thickness and the value that is set for the above piercing power level. Usually it is less than 1s for thin metal.

Unit: s

Continues on next page

`CutProgram`

Data type: `num`

Program (schedule) of the laser system used during the piercing process. The value corresponds to the program numbers in the laser equipment, e.g. IPG.

`HC_StandOff`

Data type: `num`

The distance between the nozzle and the material.

Unit: mm

`HC_LockOn`

Data type: `bool`

If the value is set to True, the cutting head will remain locked during the entire piercing phase.

Structure

```
< dataobject of LsPierData >
  < GasType of num >
  < GasFlow of num >
  < PurgeTime of num >
  < PowerLevel of num >
  < PierceTime of num >
  < CutProgram of num >
  < HC_StandOff of num >
  < HC_LockOn of bool >
```

5.2.4 LsCutParams

Usage

Used to control the laser-cutting process, and includes parameters for controlling the head, laser, and also cutting gas during piercing and cutting phases.

It contains three part: Cut power, CuttingParams, PierParams.

Note

It is a built in data type for the laser table. It is not recommended for users to create their own variables in Rapid programs, because these variables could not be managed by the laser table.

Components

`CutPower`

Data type: `num`

Sets the power of the laser during the cutting phase. It is proportional to the cut speed. Use the max cut power of the laser source to achieve the highest cutting speed.

Unit: watt

`LsCuttingData`

Data type: `CuttingParams`

For detailed information about CuttingParams, see *LsCuttingData on page 132*.

`LsPierData`

Data type: `PierParams`

For detailed information about PierParams, see *LsPierData on page 135*.

Structure

```
< dataobject of LsCutParams >
  < CutPower of num >
  < GasFlow of num >
  < GasType of num >
  < HC_StandOff of num >
  < HC_LockOn of bool >
  < MinPowerLevel of num >
  < PowerLevel of num >
  < CutSpeed of num >
  < CriticalSpeed of num >
  < Acceleration of num >
  < Delay of num >
  < HC_CharCurve of num >
  < Duty_max of num >
  < Duty_min of num >
  < CutProgram of num >
  < GasFlow of num >
```

Continues on next page

```
< GasType of num >
< HC_StandOff of num >
< HC_LockOn of bool >
< PowerLevel of num >
< PurgeTime of num >
< PierceTime of num >
< Duty_max of num >
< Duty_min of num >
< CutProgram of num >
```

5.2.5 LsTableConf

Usage

The data type `LsTableConf` is used to configure the cutting and piercing parameter groups used in laser cutting parameter tables.It contains two index value representing the indexes of the cutting group and the piercing group which are selected to be used in the current active laser cutting parameter table. Up to 5 cutting groups and 5 piercing groups can be created in a parameter table, so the maximum value of the indexes are 5.

Components

`IndexCutting`

Data type: `num`

The index of cutting group to be used in cutting instructions. data range 1- 5.

`IndexPiercing`

Data type: `num`

The index of piercing group to be used in cutting instructions. data range 1- 5.

Structure

```
< dataobject of LsTableConf >
  < IndexCutting of num >
  < IndexPiercing of num >
```

5.2.6 cutdata

Usage

The data type cutdata is used to control the general cutting process. It includes parameters for cutting speed, and it also synchronizes movements with process signals.

Components

`CutSpeed`

Data type: `num`

The TCP speed during the cutting phase. Unit: mm/s, datarange 1-400.

`StartFly`

Data type: `bool`

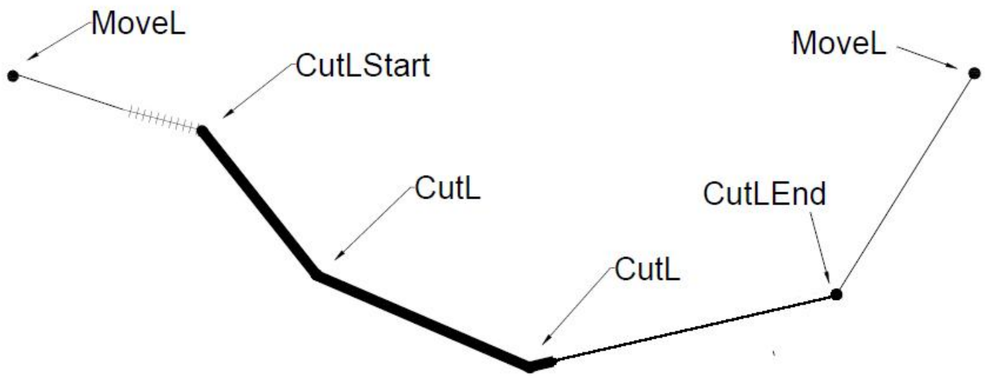
Reserved. Flying start is not supported in this version.

`EndFly`

Data type: `bool`

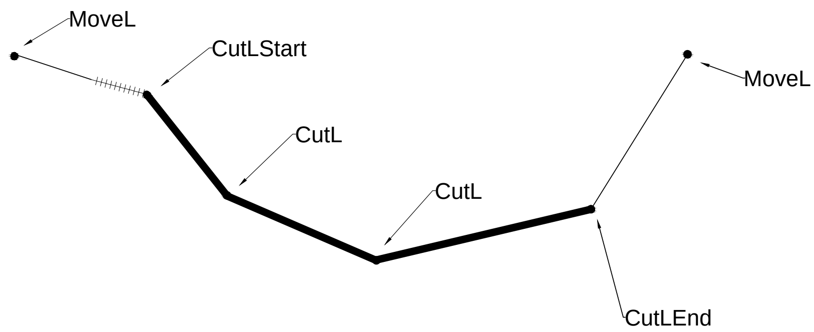
The index of piercing group to be used in cutting instructions. data range 1- 5.

Specifies whether the robot stops or flies at the end of the cut path. If the value is set to "True", cut process off task will be executed when the value starts to move at the current position.



Continues on next page

If it is False, cut process off task will be executed after CutLEnd/CutCEnd reaches the destination point and stops the robot.



`SpeedModulation`

Data type: `bool`

Specifies whether the `TcpSpeed` is monitored during cutting phase. If it is True, `cwSpeedChange` routine will be called when the change of `TcpSpeed` is larger than 2mm/s.

`Kerf`

Data type: `num`

Compensates for material that is removed by the cutting process. The value entered should be the width of the cut path. The cut path is offset by half the Kerf width according to the cut direction. There is not offset for the path, if the `kerf` = 0. No compensation for free form cut instructions (`CutL`/`CutC`).

`PurgeTime`

Data type: `num`

Time for purging gas before the robot reaches the piercing point. If the value is bigger than 0, the PurgeOn signal will be set on the fly.If it is 0, the PurgeOn signal won't be used.

Unit: s

`Data1`,`Data2`,`Data3`,`Data4`

Data type: `num`

4 data can be used in specific cutting process.

Continues on next page

Structure

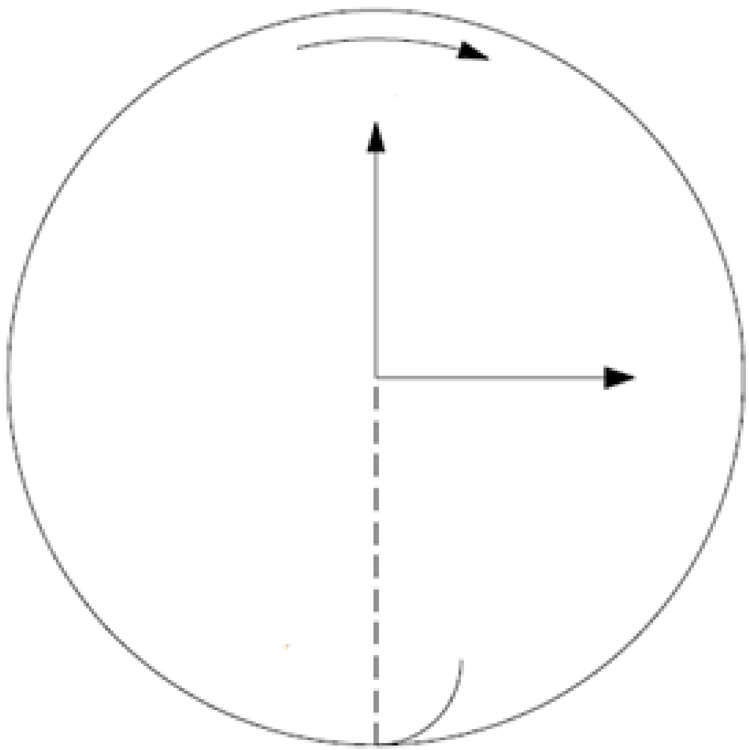
```
< dataobject of cutdata >
  < CutSpeed of num >
  < StartFly of bool >
  < EndFly of bool >
  < SpeedModulation of bool >
  < Kerf of num >
  < PurgeTime of num >
  < Data1 of num >
  < Data2 of num >
  < Data3 of num >
  < Data4 of num >
```

5.2.7 cwdirection

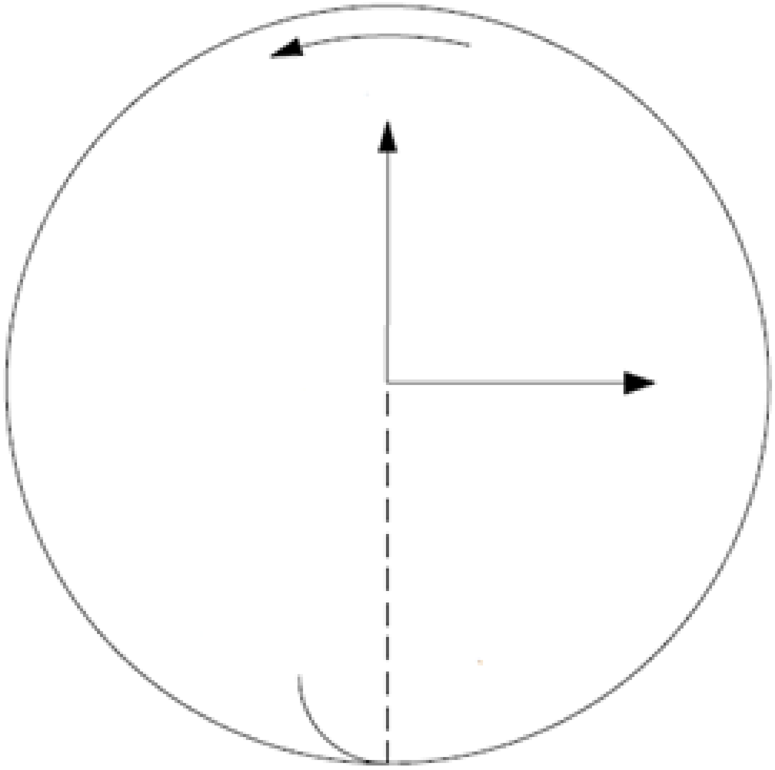
Usage

`cwdirection` is used to describe the cut direction of 2D shapes.

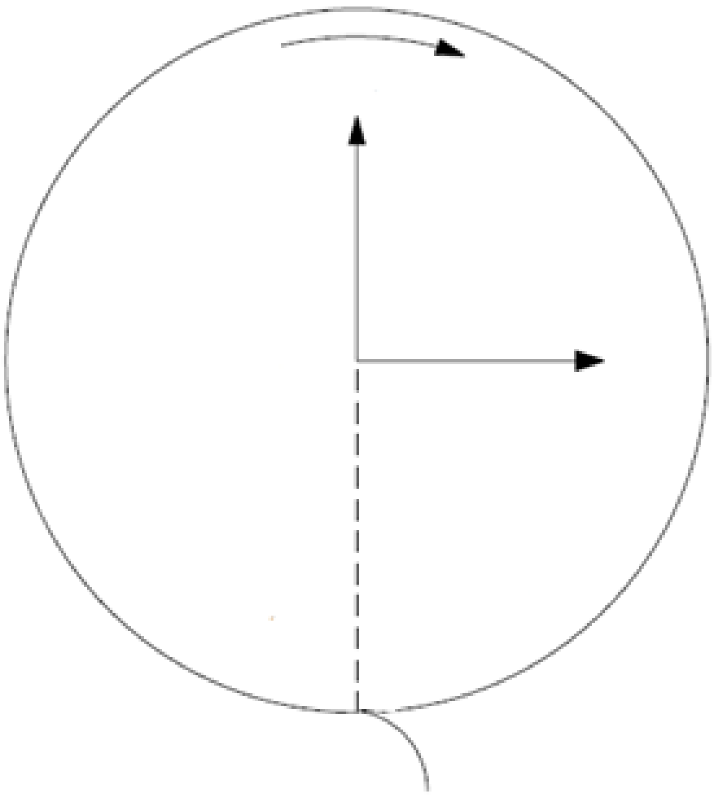
Description

Value	Description
iCW:=1	Cuts inside in clockwise direction. The pierce point is inside the shape. The tool radius compensation is to decrease the dimension. 

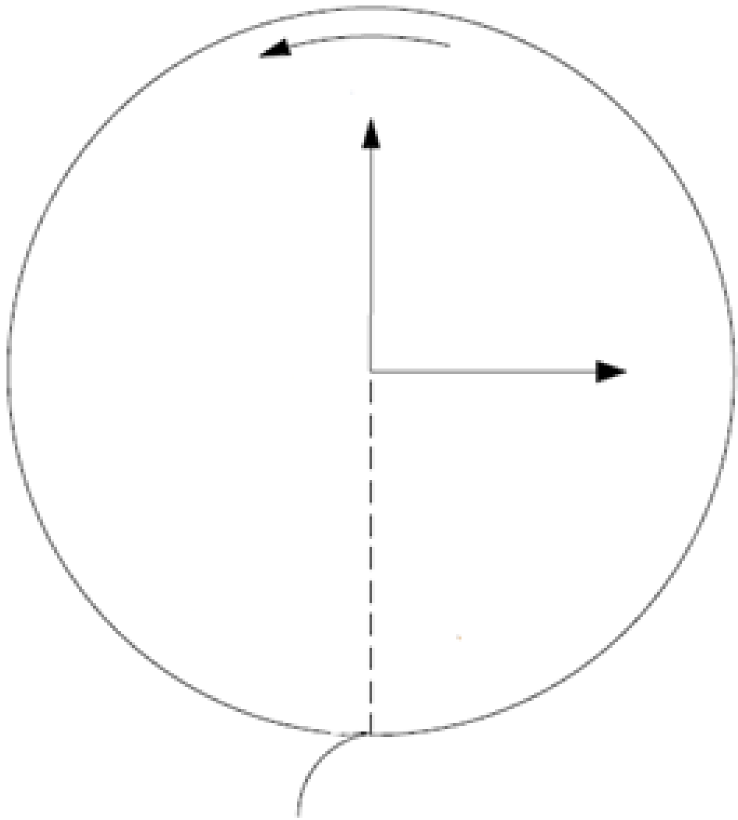
Continues on next page

Value	Description
iCCW:=-1	Cuts inside in clockwise direction. The pierce point is inside the shape. The tool radius compensation is to decrease the dimension. 

Continues on next page

Value	Description
oCW:=2;	Cuts outside in counterclockwise direction. The pierce point is outside the shape. The tool radius compensation is to increase the dimension. 

Continues on next page

Value	Description
oCCW:=-2	Cuts outside in counterclockwise direction. The pierce point is outside the shape. The tool radius compensation is to increase the dimension. 

Note

Tool radius compensation is not support for CutCadL/CutCadJ

5.2.8 fricdata

Usage

The data type fricdata contains auto friction tuning results for the robot. It is used with shape cut instructions to improve the performance of an individual cut shape. It has six elements corresponding to the six axes of the robot. Optimum friction data can be obtained by executing friction tuning for related shape instruction. The data variable should be unique for every shape cut instruction.

Components

`FricLev_Ax1`

Data type: `num`

Defines the friction compensation factor for axis 1 of the robot. Default is 100.

`FricLev_Ax2`

Data type: `num`

Defines the friction compensation factor for axis 2 of the robot. Default is 100.

`FricLev_Ax3`

Data type: `num`

Defines the friction compensation factor for axis 3 of the robot. Default is 100.

`FricLev_Ax4`

Data type: `num`

Defines the friction compensation factor for axis 4 of the robot. Default is 100.

`FricLev_Ax5`

Data type: `num`

Defines the friction compensation factor for axis 5 of the robot. Default is 100.

`FricLev_Ax6`

Data type: `num`

Defines the friction compensation factor for axis 6 of the robot. Default is 100.

Structure

```
< dataobject of fricdata >
  < FricLev_Ax1 of num >
  < FricLev_Ax2 of num >
  < FricLev_Ax3 of num >
  < FricLev_Ax4 of num >
  < FricLev_Ax5 of num >
  < FricLev_Ax6 of num >
```

5.2.9 leaddata

Usage

Used to define the leadin and leadout path for 2D shape cut instructions. The cutting process starts on the leadin path and ends on the leadout path. The design of the lead path depends on the cutting methods and the materials to cut.

Components

`Zone`

Data type: `num`

Reserved. The zone of entry point on the boundary of the shape. Only support entry point on the negative y axis direction in this version.

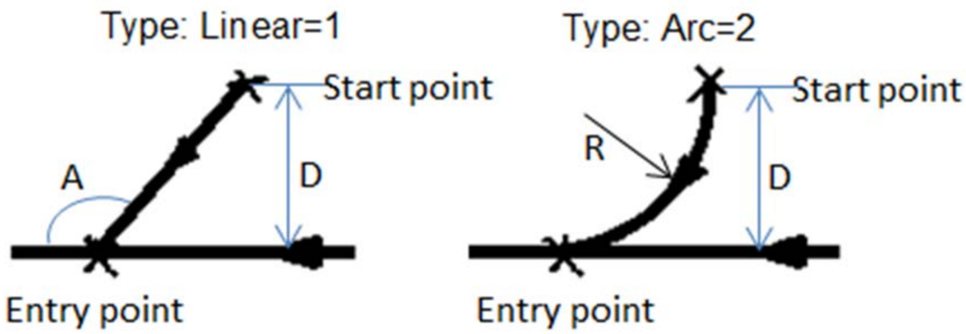
`LeadIn`

Data type: `inoutdata`

Defines the friction compensation factor for axis 5 of the robot. Default is 100.

[Type, Distance, Angle, Radius], these parameters define the leadin path. If Type = 0, there is not lead in path. The limitation is that Distance>=1.0 mm. Angle is between 90 to 170 degreen. Radius is bigger or equal to 1.0 mm.

Unit: [/,mm, degree, mm]



`Overlap`

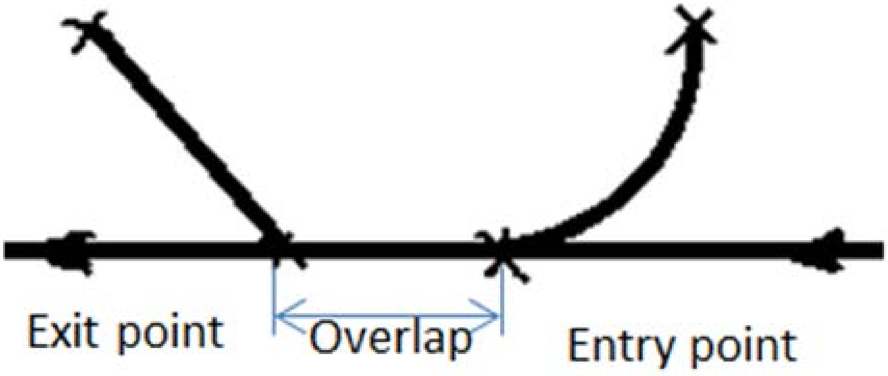
Data type: `num`

Defines the friction compensation factor for axis 5 of the robot. Default is 100.

The overlap distance between the entry point and the start point of leadout (exit point). Laser is turned off at the exit point.

Continues on next page

Unit: mm



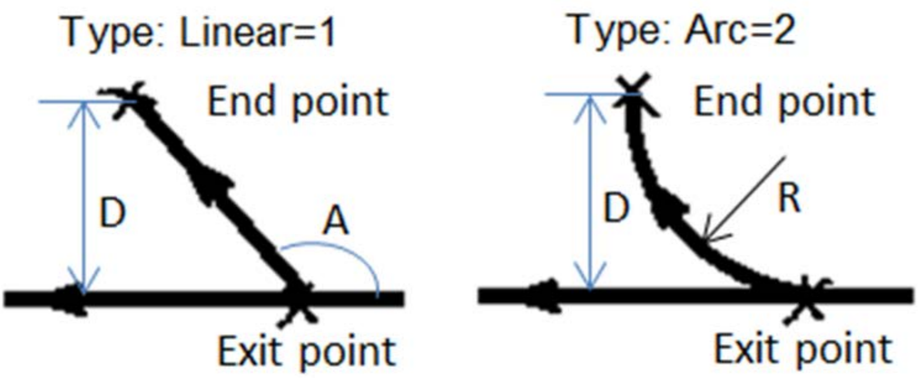
`LeadOut`

Data type: `inoutdata`

Defines the friction compensation factor for axis 5 of the robot. Default is 100.

[Type, Distance, Angle, Radius], these parameters define the leadout path. If Type = 0, there is no lead out for cutting, instead there is a short path following the boundary of the shape to keep the move direction and to close the laser on the fly. The limitation is that Distance>=1.0 mm. Angle is between 90 to 170 degreen. Radius is bigger or equal to 1.0 mm.

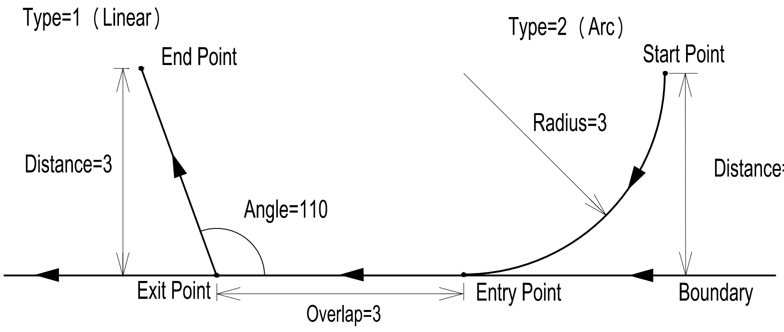
Unit: [/,mm, degree, mm]



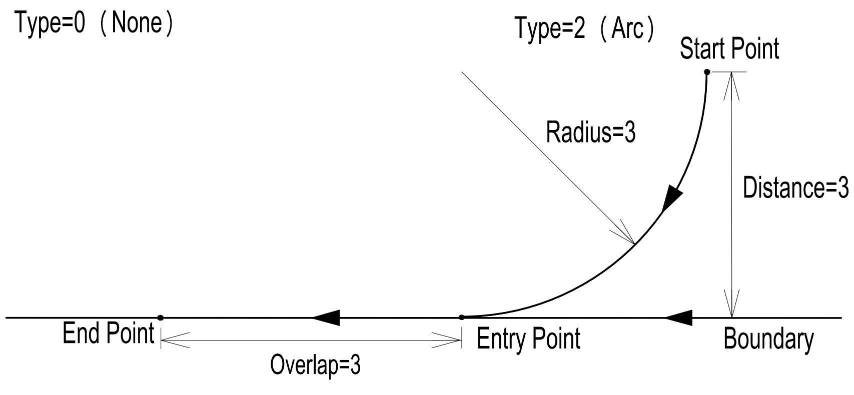
Examples

```
leaddata Arc3_r3_o3_Lin3_110:=[0,[2,3,0,3],3,[1,3,110,0]];
```

Continues on next page




```
leaddata Arc3_r3_o3:=[0,[2,3,0,3],3,[0,0,0,0]];
```



Structure

```
< dataobject of leaddata >
  < Zone of num >
```

Continues on next page

```
< LeadIn of inoutdata >
< Overlap of num >
< LeadOut of inoutdata >
< dataobject of inoutdata >
< Type of num >
< Distance of num >
< Angle of num >
< Radius of num >
```

5.2.10 pthpos

Usage

Used for defining free-form shapes. A customized 2D cut shape can be defined as an array with each element representing a position and corresponding cut type. The path point array is stored in the CadShape file which is used in cut instruction `CutCad`. The CadShape file can be generated by RobotStudio Cutting PowerPac.

Components

`Trans` (translation)

Data type: `pos`

The position (x, y and z) of the tool center point expressed in mm. The position is specified in relation to the cut frame coordinate system. Z-value will be zero for all points in the pthpos array.

`Type`

Data type: `num`

`Type` defines which type of move to perform. CutL=1, CutC=2.

Structure

```
< dataobject of pthpos >
  < Trans of pos >
  < Type of num >
```

5.2.11 ShapeOffsetData

Usage

Used to adjust the small deviation of an individual cut shape in production based on the measurement.

> **Note**
>
> ResetPathMemory is necessary after modifying the ShapeOffsetData,in order to recalculate the shape path.

Components

`Description`

Data type: `string`

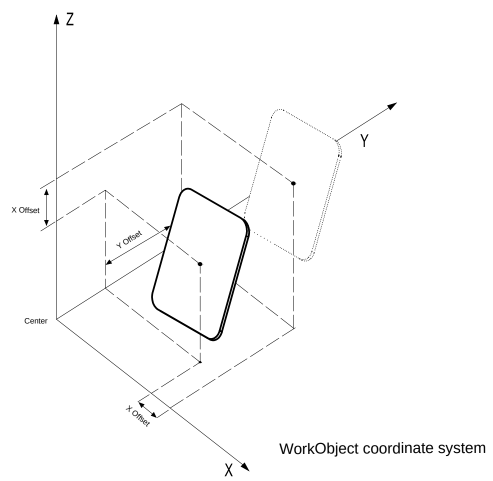
String to describe the shape.

`OffsetX`, `OffsetY`, `OffsetZ`

Data type: `num`

Offset to adjust the center of the shape in wobj coordinate system.

Unit: mm



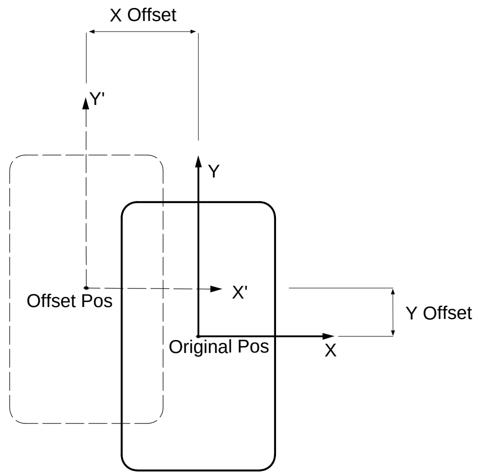
`OffsetShapeX`, `OffsetShapeY`

Data type: `num`

Offset to adjust the shape in the local cut frame coordinate system.

Continues on next page

Unit: mm



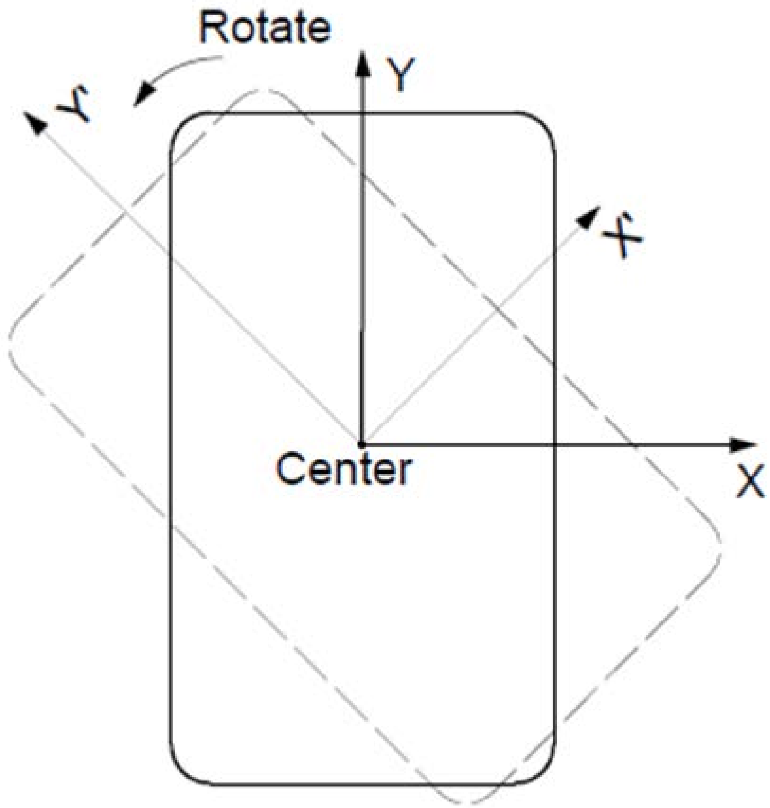
`ShapeRotation`

Data type: `num`

Rotate the shape in the local cut frame coordinate system.

Continues on next page

Unit: degree



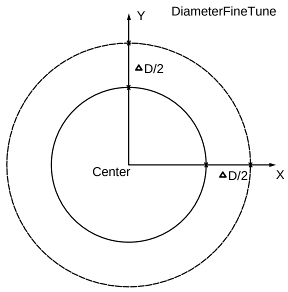
`DiameterFineTune`

Data type: `num`

Adjusts the diameter of the circle.

Continues on next page

Unit: mm



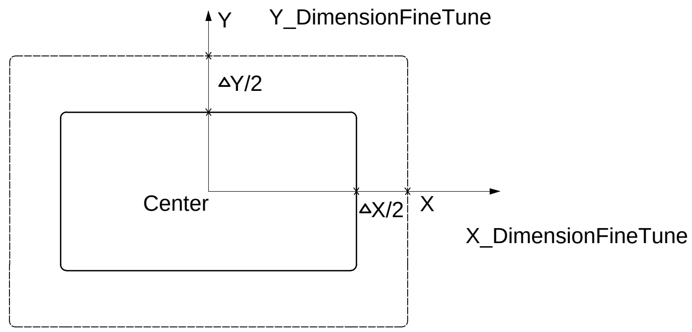
`X_DimensionFineTune/Y_DimensionFineTune`

Data type: `num`

Adjusts the X, Y dimension of the shape.

Continues on next page

Unit: mm



Structure

```
< dataobject of ShapeOffsetData >
  < Description of string >
  < OffsetX of num >
  < OffsetY of num >
  < OffsetZ of num >
  < OffsetShapeX of num >
  < OffsetShapeY of num >
  < ShapeRotation of num >
  < DiameterFineTune of num >
  < X_DimensionFineTune of num >
  < Y_DimensionFineTune of num >
```

This page is intentionally left blank

Index

Contact us

ABB AB
Discrete Automation and Motion
Robotics
S-721 68 VÄSTERÅS, Sweden
Telephone +46 (0) 21 344 400

ABB AS, Robotics
Discrete Automation and Motion
Nordlysvegen 7, N-4340 BRYNE, Norway
Box 265, N-4349 BRYNE, Norway
Telephone: +47 51489000

ABB Engineering (Shanghai) Ltd.
5 Lane 369, ChuangYe Road
KangQiao Town, PuDong District
SHANGHAI 201319, China
Telephone: +86 21 6105 6666

www.abb.com/robotics

3HAC043508-001, Rev D, en