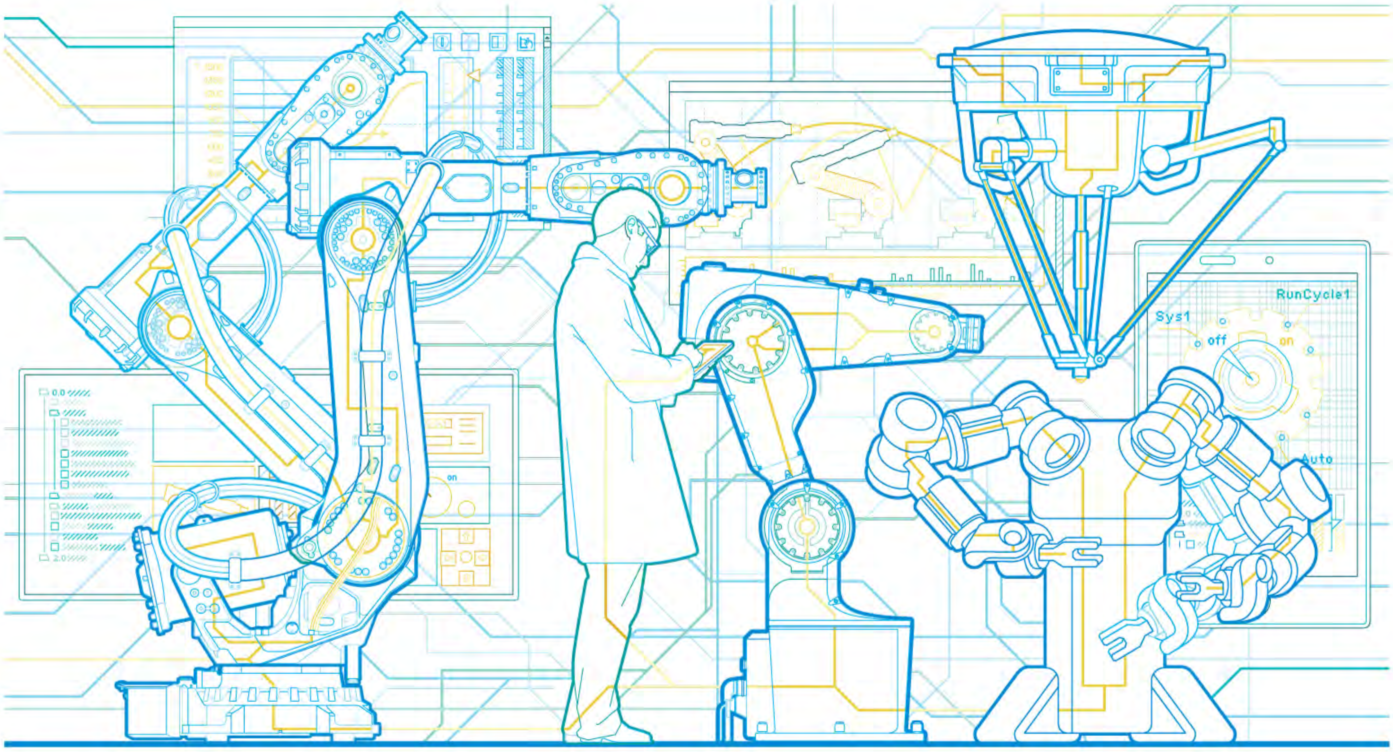


Application manual
SoftMove

ABB

Trace back information:
Workspace R17-1 version a8
Checked in 2017-03-29
Skribenta version 5.1.011

Application manual

SoftMove

RobotWare 6.05

Document ID: 3HAC050977-001

Revision: C

Keep for future reference.

Additional copies of this manual may be obtained from ABB.

Original instructions.

ABB AB, Robotics
Robotics and Motion
Se-721 68 Västerås
Sweden

Table of contents

Overview of this manual

About this manual

This manual contains information about the RobotWare option SoftMove.

Usage

This manual can be used to find out what SoftMove is and how to use it. The manual also provides information about RAPID components and system parameters related to SoftMove, and examples of how to use them.

Who should read this manual?

This manual is mainly intended for robot programmers.

Prerequisites

The reader should be familiar with:

- Industrial robots and their terminology
- The RAPID programming language
- System parameters and how to configure them

Organization of chapters

The manual is organized in the following chapters:

	Contents
1	Introduction to SoftMove.
2	Descriptions of necessary main parameters when using this option.
3	Information on how to program SoftMove. RAPID instructions and data types, including code examples.
4	SoftMove specific RAPID components.
5	SoftMove specific system parameters.

References

Reference	Document ID
Application manual - Controller software IRC5	*3HAC050798-001*
Operating manual - IRC5 with FlexPendant	*3HAC050941-001*
Operating manual - RobotStudio	*3HAC032104-001*
Technical reference manual - RAPID overview	*3HAC050947-001*
Technical reference manual - RAPID Instructions, Functions and Data types	*3HAC050917-001*
Technical reference manual - System parameters	*3HAC050948-001*

Revisions

Revision	Description
-	Released with RobotWare 6.0.

Continues on next page

Revision	Description
A	Released with RobotWare 6.02. • Warning about breaking distance is added to *Limitations on page 14* and *CSSAct - Cartesian Soft Servo activation on page 33*. • Added section *Example of parameter settings for high friction situations on page 30*.
B	Released with RobotWare 6.04. • Added information for the IRB 14000 robot (YuMi) throughout the manual.
C	Released with RobotWare 6.05. • Added a limitation for SafeMove, see *Limitations on page 14*. • Added the RAPID instruction *CSSDeact - Deactivates Cartesian Soft Servo on page 36*. • Added the system parameter *Activation during movement smoother on page 53*.

Product documentation, IRC5

Categories for user documentation from ABB Robotics

The user documentation from ABB Robotics is divided into a number of categories. This listing is based on the type of information in the documents, regardless of whether the products are standard or optional.

All documents listed can be ordered from ABB on a DVD. The documents listed are valid for IRC5 robot systems.

Product manuals

Manipulators, controllers, DressPack/SpotPack, and most other hardware is delivered with a **Product manual** that generally contains:

- Safety information.
- Installation and commissioning (descriptions of mechanical installation or electrical connections).
- Maintenance (descriptions of all required preventive maintenance procedures including intervals and expected life time of parts).
- Repair (descriptions of all recommended repair procedures including spare parts).
- Calibration.
- Decommissioning.
- Reference information (safety standards, unit conversions, screw joints, lists of tools).
- Spare parts list with exploded views (or references to separate spare parts lists).
- Circuit diagrams (or references to circuit diagrams).

Technical reference manuals

The technical reference manuals describe reference information for robotics products.

- *Technical reference manual - Lubrication in gearboxes*: Description of types and volumes of lubrication for the manipulator gearboxes.
- *Technical reference manual - RAPID overview*: An overview of the RAPID programming language.
- *Technical reference manual - RAPID Instructions, Functions and Data types*: Description and syntax for all RAPID instructions, functions, and data types.
- *Technical reference manual - RAPID kernel*: A formal description of the RAPID programming language.
- *Technical reference manual - System parameters*: Description of system parameters and configuration workflows.

Continues on next page

Application manuals

Specific applications (for example software or hardware options) are described in **Application manuals**. An application manual can describe one or several applications.

An application manual generally contains information about:

- The purpose of the application (what it does and when it is useful).
- What is included (for example cables, I/O boards, RAPID instructions, system parameters, DVD with PC software).
- How to install included or required hardware.
- How to use the application.
- Examples of how to use the application.

Operating manuals

The operating manuals describe hands-on handling of the products. The manuals are aimed at those having first-hand operational contact with the product, that is production cell operators, programmers, and trouble shooters.

The group of manuals includes (among others):

- *Operating manual - Emergency safety information*
- *Operating manual - General safety information*
- *Operating manual - Getting started, IRC5 and RobotStudio*
- *Operating manual - IRC5 Integrator's guide*
- *Operating manual - IRC5 with FlexPendant*
- *Operating manual - RobotStudio*
- *Operating manual - Trouble shooting IRC5*

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*.

This page is intentionally left blank

1 Introduction

1.1 SoftMove - Cartesian Soft Servo

General

SoftMove is a Cartesian Soft Servo movement. SoftMove is used to lower the stiffness of the robot in a predefined direction, while mainly keeping the original behavior in the other directions. When using SoftMove the behavior of the robot will be modified. The robot may not always follow a programmed path and some functions, for example collision detection, are deactivated.

The functionality can be used on its own or in combination with move instructions.

Examples of typical applications

A typical application area when using the functionality on its own is ejector machines, where the robot will follow the movements of the pushing machine.

When the functionality is used in combination with a move instruction the robot will mainly follow the ordered path in the non soft directions while allowing a larger position deviation in the soft direction. This is useful if there are variations in the work piece position in the specified direction, thereby making the robot following these variations and avoiding a hard collision.

For detailed examples of applications, see *Application examples on page 21*.

What is included?

The RobotWare option SoftMove gives you access to:

- RAPID instructions for activating/deactivating soft mode.
- RAPID instructions for activating/deactivating a force offset in a predefined direction to easily overcome static friction.
- RAPID instruction to identify the static friction level.
- System parameters for setting up behavior of SoftMove.
- System parameters for setting up supervision of SoftMove.

For general description of the main system parameters, see *General description on page 17*.

Behavior of SoftMove

Note the following behavior of SoftMove:

- SoftMove is deactivated if the program pointer is moved.
- Move instructions and jogging are only allowed during SoftMove if the argument `\AllowMove` is selected at activation.
- The execution time may increase.
- Normal position and speed supervision are replaced with a dedicated supervision that can be modified in the configuration, for example to allow a position deviation from the programmed position of 0.1 meters in a specific direction. In manual mode, there is still a max speed of 250 mm/s.

Continues on next page

- When pressing program stop the robot remains soft - that is, SoftMove is still activated. However, any force applied by `CSSForceOffsetAct` will be removed. This is also the case when the RAPID instruction `Stop` is used.
- When going to Motors Off state, the brakes are applied and the robot is not soft. SoftMove is resumed when going back to Motors On state, unless the system parameter *Automatic reactivation of css disabled* is set to *Yes*.

Note

When using SoftMove it is extremely important that the tooldata is defined correctly, especially the mass. Errors in the load definition will be interpreted as external forces, which in turn can cause the robot to move. Hence, an incorrect definition can cause robot movements.

Tip

Some general tips when using SoftMove:

- Use SoftMove only for the part of the program where it is needed as the robot will not always follow the programmed path and this can increase the cycle time.
- Avoid singular robot configurations.
- The RAPID instruction `CSSOffsetTune` should be seen as a guideline to find the value of the static friction. Since the static friction may drop when the robot is warm, the value identified by this instruction may be too high to use at activation with the instruction `CSSForceOffsetAct`.
- If the robot is not soft enough, no matter how you tune, then try to use another arm configuration and test if the robot behavior is better.
- Before running SoftMove, make sure the load definition of the tool is correct.

Limitations

Limitations when using SoftMove:

- The SoftMove instructions does not activate motion control. A move instruction must be executed before `CSSAct` or `CSSOffsetTune`.
- SoftMove is not available for 4-axis robots.
- SoftMove does not work together with the option MultiMove Coordinated (option 604-1).
- SoftMove cannot be combined with joint soft servo (instruction `SoftAct`).
- When SafeMove is used together with SoftMove there is a risk for servo lag problems. The recommended action is to add a Contact Application Tolerance (CAP) in the area where SoftMove is active.

When SoftMove is active, that is, between a `CSSAct` and a `CSSDeactMoveL` instruction, the following functionality is *not* accessible:

- Collision Detection (option 613-1).
- Force Control (option 661-2).

Continues on next page

- Tracking functionality like Conveyor Tracking (option 606-1), Optical Tracking (660-1) and Weldguide (815-2).
- World Zones (option 608-1).

WARNING

The braking distance for category 1 stops will be longer when SoftMove is active.

This page is intentionally left blank

2 Configuration

2.1 General description

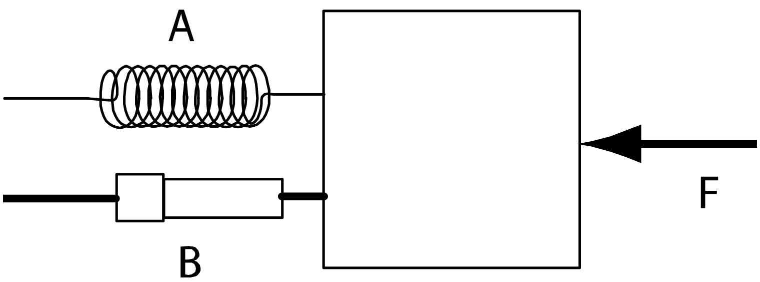
Parameters for basic behavior

SoftMove is used to set up softness in one of the following directions in relation to either the tool or the work object:

- one of the Cartesian directions (x, y, or z)
- one of the Cartesian planes (xy, xz or yz)
- all directions (xyz)
- the plane xy and rotational around the z axis
- the arm angle (only for 7-axis robots)

The behavior of the softness is controlled by two parameters - *Stiffness* and *Damping*.

Parameter	Description
Stiffness	Describes how strongly the robot tries to move back to the reference point (see *Reference position on page 18*). The value set in the RAPID program is a percentage of a configured value. A higher value gives a stronger spring effect. Setting the value to zero gives no spring effect and the robot will be floating in the chosen direction. **Note** If the `\AllowMove` argument is used in the `CSSAct` instruction the robot will always have a spring effect, even if *Stiffness* is set to zero.
Damping	Describes how much resistance there is to push the robot. The resistance does not increase with the distance from the reference point but usually with the speed of the robot. Damping is set as ratio of the stiffness in the system parameters. See *Stiffness to damping ratio on page 67*.



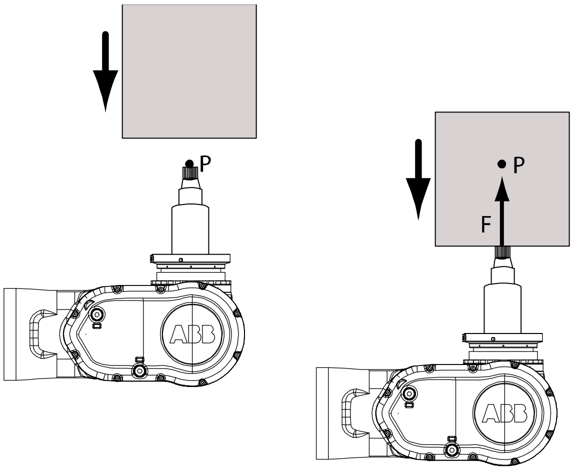
xx1000001046

A	Stiffness can be compared to a spring.
B	Damping can be compared to a hydraulic shock absorber.

Continues on next page

Reference position

If SoftMove is activated but no move instruction is used, the robot position at activation will act as reference position. If an external force pushes the robot away from this position, the robot acts as a spring and applies a force that is proportional to the distance to the reference position.



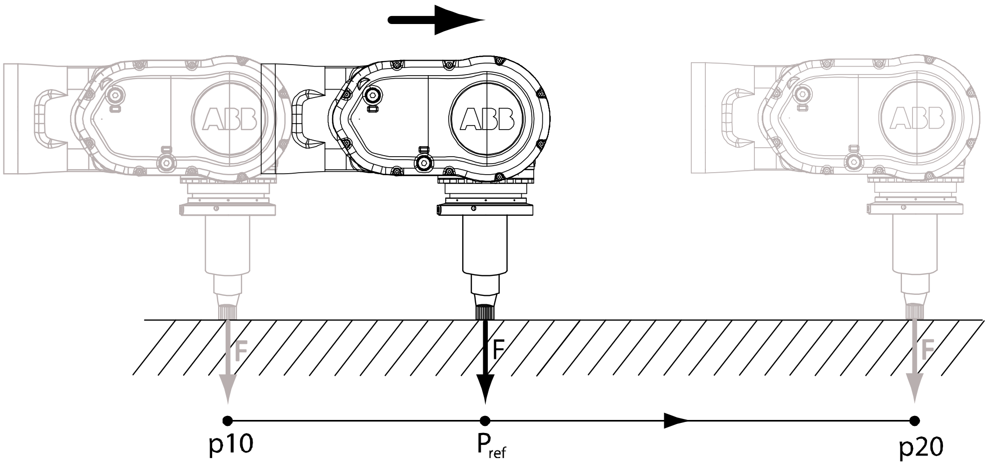
xx1000001084

P	Reference position
F	Force (proportional to distance between robot TCP and reference position)

If a move instruction is used, the robot will try to move to its programmed position. If there is an obstacle in the soft direction, the robot applies a force towards the reference position. The reference position in each instant is the point where the robot TCP should have been if there was no obstacle. In the non-soft directions the robot follows the reference path closely, but in the soft direction there can be a distance to the reference position. This distance generates a force pushing

Continues on next page

towards the reference position. When the movement is finished, the programmed target is the new reference position.



xx1000001085

p10	Programmed position before the move instruction.
p20	Programmed target of the move instruction.
P	Reference position for the current robot position, located on the reference path between p10 and p20.
F	Force

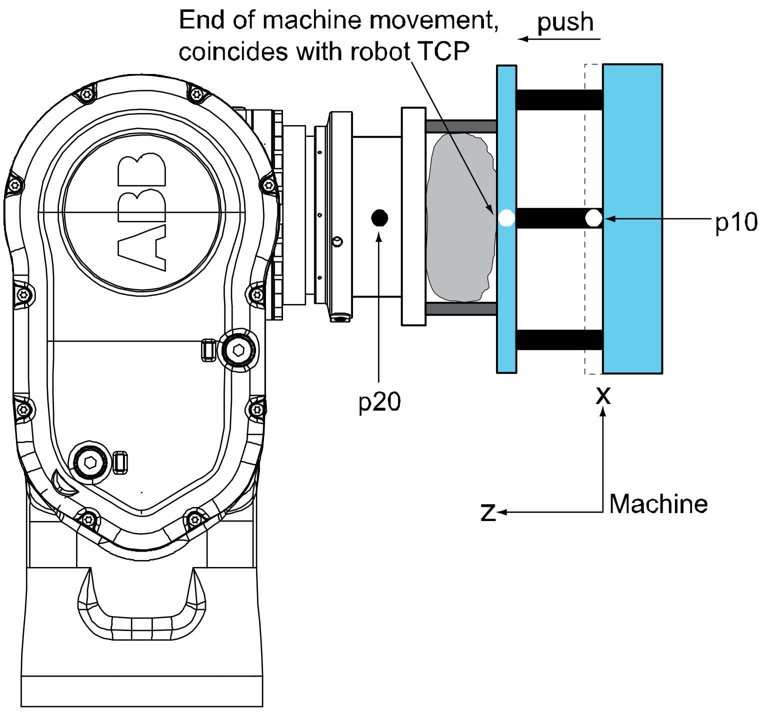
Note

Due to varying inner friction in the robot, the exact same force cannot be guaranteed trough out a robot movement.

Continues on next page

Illustration, example

The graphic below illustrates an example of a robot activating SoftMove in the horizontal direction, work object z, in the position p10. By changing stiffness and damping it is possible to control the behavior of the robot. A high stiffness value forces the robot to go back to the position p10, while setting stiffness to zero makes the robot only float with the external push. The damping controls how much the robot will resist the pushes of the external machine.



en0700000676

push	The external machine pushes the robot away from the position p10.

Friction offset compensation

Very small external forces to a soft robot may not cause the robot to move, due to the internal friction of the robot. By using friction offset compensation in a predefined direction, the robot will apply a small force to help overcome this internal friction. If x is the soft direction and friction offset compensation is used in positive x direction, a very small force is enough to make the robot move in positive x direction. The consequence is that the robot becomes less soft in the negative x direction.

The friction offset to be used in a certain position for a certain direction can be identified and stored to be used later on. Friction offset is activated with the instruction `CSSForceOffsetAct`. See *CSSForceOffsetAct - Cartesian Soft Servo force offset activation on page 40*.

3 Programming

3.1 Application examples

3.1.1 Part extraction

Description

The robot compliance can be used in a typical machine tending application where a machine, for example Injection Moulding Machine or Die Casting Machine ejects a part out of a mould.

This example shows a case where the direction is known and the robot needs to be as soft as possible in that direction. The robot has to cooperate with the machine to extract a finished part. For illustration, see *Illustration, example on page 20*.

Program example

The program example begins with the robot approaching the machine to the position p10. The first time the program runs, an offset force needed to overcome the static friction in the z-direction is identified. After the work piece is gripped, SoftMove is activated to make the robot soft in the machine's z-direction. Here stiffness zero is used to prevent the robot moving back to p10. To make the robot even easier to push, the earlier identified force offset is applied. At this point, the machine can start pushing the robot. When the push is finished, SoftMove is deactivated with a linear movement to the position p20.

Program code

```
CONST robtarget p10 := [[0,0,0], [0,0,1,0], [1,1,0,0],
     [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
CONST robtarget p20 := [[0,0,400], [0,0,1,0], [-1,1,0,0],
     [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
PERS loaddata piece1 := […];
PERS num ForceOffset := 0;
PERS bool IsOffsetTuned := FALSE;
PERS wobjdata Machine := […];
PERS tooldata Gripper := […];

MoveL p10, v100, fine, Gripper \WObj := Machine;

! Before the first activation the static friction is identified as
     a force offset
! This force offset is later applied to make it easier to push the
     robot
IF NOT IsOffsetTuned THEN
  CSSOffsetTune \RefFrame:=CSS_REFFRAME_WOBJ, CSS_POSZ, ForceOffset
       \StiffnessNonSoftDir:=30;
  IsOffsetTuned:=TRUE;
ENDIF

! Grip the work piece and increase the tool load accordingly
SetDO do_gripper,0;
```

Continues on next page

```
GripLoad piece1;

! Activate CSS with zero stiffness to make it "free-floating"
CSSAct \RefFrame:= CSS_REFFRAME_WOBJ, CSS_Z \StiffnessNonSoftDir:=30
     \Stiffness:=0;

! Apply the earlier identified offset to make the robot easier to
     push
CSSForceOffsetAct CSS_POSZ, ForceOffset;

! Wait while the machine pushes robot
! Here we would use some other condition for determining that the
     robot is in the new position and has zero speed, for example
     a digital signal between the machine and the robot
WaitTime 1;

! Deactivate CSS while doing a linear motion away from the machine
CSSDeactMoveL p20,v100,Gripper \WObj:=Machine;
```

WARNING

Make sure that the Cartesian Soft Servo supervision parameters are set according to the application to be used. Otherwise there is a risk the supervision will trigger and possibly lead to equipment damage.

For applications like the example above, the critical supervision parameters are:

- *Max pos error in x*
- *Max pos error in y*
- *Max pos error in z*
- *Max pos error around z*
- *Max pos error in arm angle*
- *Max speed error in x*
- *Max speed error in y*
- *Max speed error in z*
- *Max speed error around z*
- *Max speed error in arm angle*

See *System parameters reference information on page 51*.

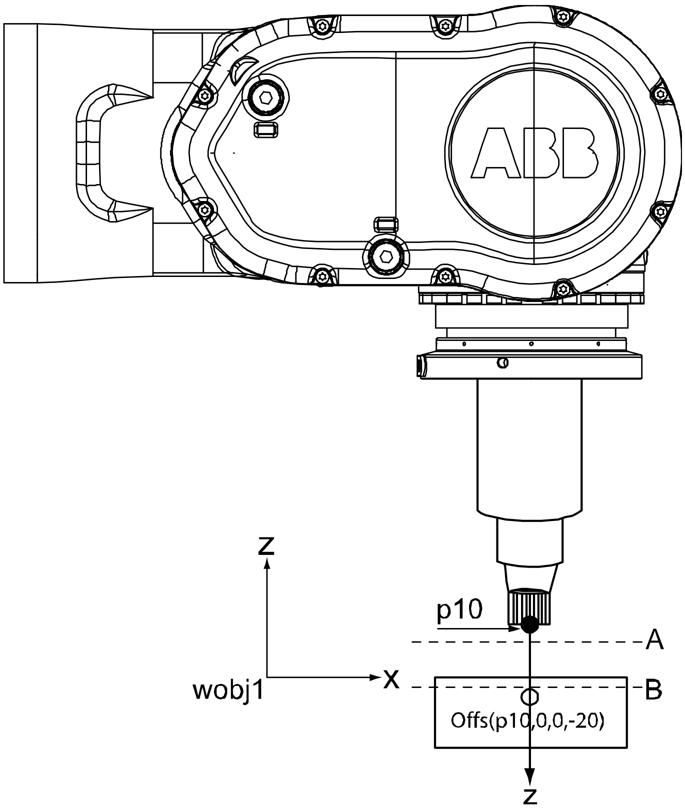
3.1.2 Work piece variation

Description

This example shows a case where the robot moves with a tool into a work piece. There is a variation of the surface. Hence, the exact z-coordinate of the surface is not known in advance.

Illustration, example

This picture illustrates a robot that will hit the surface somewhere between A and B. By activating Cartesian Soft Servo above A and make the last movement, that can be programmed to be below B, the robot will now go into the surface smoothly without triggering any collision. The applied contact force will increase with the distance between contact position and the programmed position.



en0700000678

Note

When the argument `\AllowMove` is used the stiffness and damping will have a stiffer basic setup. This will make the robot less soft but it will follow the planned movement better. As a result, the robot will always behave like a spring, even when the stiffness is set to zero.

Continues on next page

Program code

```
PERS wobjdata wobj1:=[...];
PERS robtarget p10:=[...];

! Move to a position above A
MoveL p10,v100,fine,tool1 \WObj:=wobj1;

! Make the robot soft in the tool’s z-direction (tool is the default
     frame)
! Must use \AllowMove to be able to execute move instructions in
     CSS mode
CSSAct CSS_Z \StiffnessNonSoftDir:=50 \Stiffness:=1 \AllowMove;

! This position is known to be below B and the robot will now push
     into the surface
! The contact force will increase with the distance between the
     programmed position and the contact position
MoveL Offs(p10,0,0,-20),v100,fine,tool1 \Wobj:=wobj1;

! Do the work: close the gripper, polish the surface, etc.

! Deactivate CSS while moving back to the starting position
CSSDeactMoveL p10,v100,tool1 \WObj:=wobj1;
```

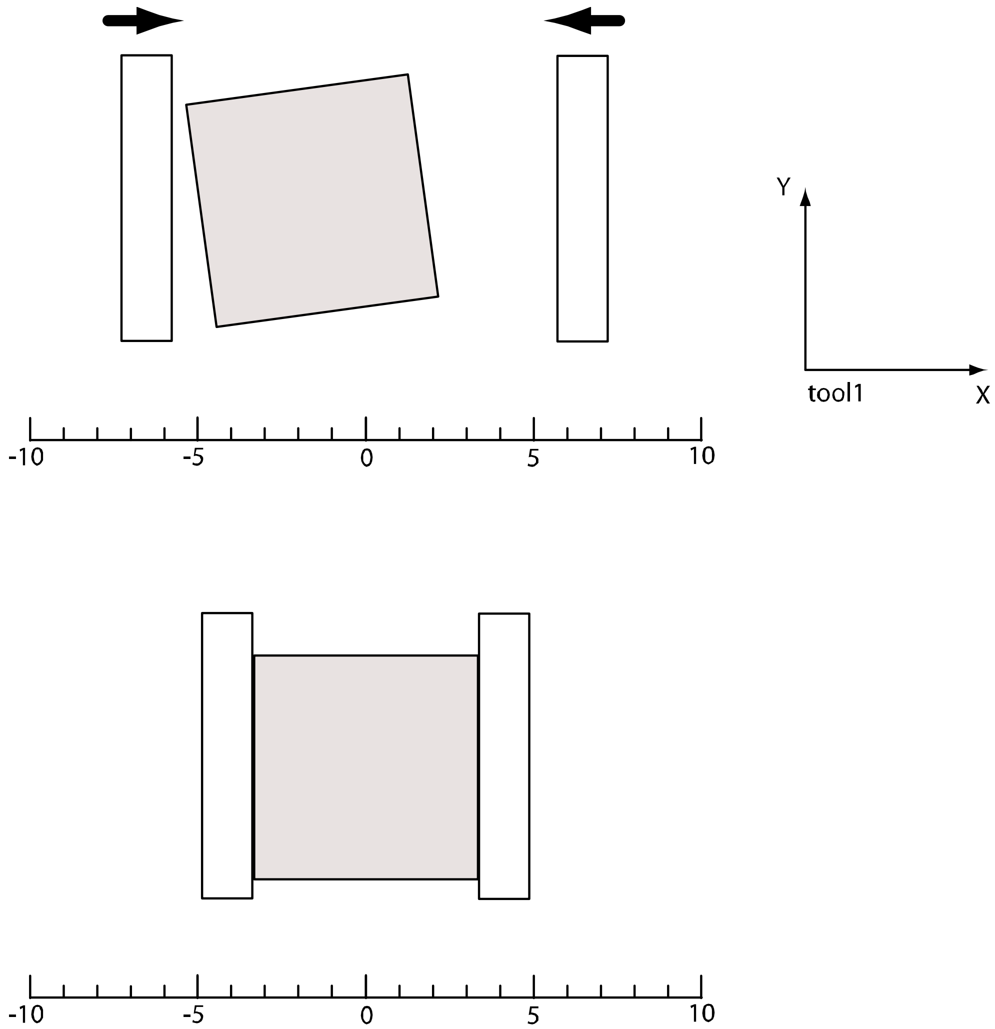
3.1.3 Softness in rotational direction

Description

This example shows that a work piece, held by the robot, can be both pushed in linear direction and forced to rotate by external equipment.

Illustration, example

The robot holds a work piece that is grabbed by external equipment. The external equipment acts like a vise, clamping the work piece in the middle. The robot does not hold the work piece exactly in the middle and not exactly strait. The robot must therefore allow the work piece to be pushed in the X direction and rotated around the Z axis.



xx1000001079

Program code

```
PERS wobjdata tool1:=[...];
PERS wobjdata wobj1:=[...];
PERS robtarget p10:=[...];
PERS loaddata piece1:=[...];
```

Continues on next page

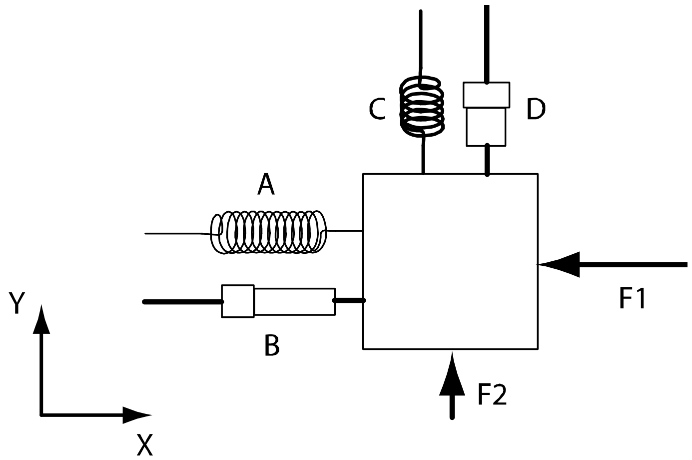
```
! Grab the work piece
SetDo do_gripper, 1;

! Adjust the load.
! Correct load is important when activating SoftMove
gripload piece1;

! Move the work piece into the external eqpment
MoveL p10,v500,fine,tool1;

! Make the robot soft in the tool’s xy-plane and rotational z axis
CSSAct \RefFrame:= CSS_REFFRAME_TOOL, CSS_XYRZ
     \StiffnessNonSoftDir:=50 \Stiffness:=0;

! Wait until the external equipment has clamped the work piece
WaitDI di_viseClosed, 1;

! Adjust the load so the robot does not drift upwards when releasing
     the work piece
gripload load0;

! Open the gripper to release the work piece
SetDo do_gripper, 0;

! Deactivate CSS while moving out of the external equipment
CSSDeactMoveL p20,v500,tool1;
```

4 Stiffness and damping calculations for advanced users

4.1 Stiffness and damping in the different directions

About stiffness and damping

SoftMove is used to make the robot soft in one or several directions (soft direction). The other directions are stiffer with higher control gains (non-soft direction).



xx1000001047

X	Soft direction
Y	Non-soft direction
A	Stiffness in soft direction
B	Damping in soft direction
C	Stiffness in non-soft direction
D	Damping in non-soft direction
F1	Force in soft direction
F2	Force in non-soft direction

Setting the stiffness value to zero gives no spring effect and the robot will be floating in that direction.

Min and max values

The minimum stiffness is configured by four different system parameters, depending on direction (soft or non-soft) and if movement is allowed or not.

	No movement	Movement allowed
Soft direction	*Stiffness min on page 65*	*Stiffness min move on page 70*
Non-soft direction	*Stiffness in non soft dir. min on page 63*	*Stiffness in non soft dir. min move on page 68*

Continues on next page

The maximum stiffness is configured by four different system parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness max on page 66*	*Stiffness max move on page 71*
Non-soft direction	*Stiffness in non soft dir. max on page 64*	*Stiffness in non soft dir. max move on page 69*

Setting stiffness percentage for CSSAct

When activating the softness with `CSSAct`, the stiffness is set as a percentage value between the configured min. and max. values.

Example:

```
CSSAct CSS_X \StiffnessNonSoftDir:=80 \Stiffness:=25;
```

Calculating the actual stiffness

The actual stiffness is calculated as:

Stiffness = *Stiffness min* + (Percentage /100)*(*Stiffness max* - *Stiffness min*)

where *Stiffness min* and *Stiffness max* are system parameters and Percentage is the value set in the instruction `CSSAct`.

Example

System parameter values:

Parameter	Value
Stiffness min move	2
Stiffness max move	10
Stiffness in non soft dir. min move	10
Stiffness in non soft dir. max move	20

Activation instruction:

```
CSSAct CSS_X \StiffnessNonSoftDir:=80 \Stiffness:=25 \AllowMove;
```

Stiffness in soft direction (X in this example):

2 + (25/100)*(10-2) = 4

Stiffness in non-soft direction (all directions except X in this example):

10+ (80/100)*(20-10) = 18

Calculating the actual damping

The damping is calculated from the stiffness values and the system parameter *Stiffness to damping ratio* or *Stiffness to damping ratio move*.

Damping = Stiffness / Ratio

Example

System parameter values:

Parameter	Value
Stiffness to damping ratio	0.5
Stiffness to damping ratio move	0.5

Continues on next page

If the stiffness is as previously calculated, 4 in soft direction and 18 in non-soft direction, the damping is calculated as follows.

Damping in soft direction:

4 / 0.5 = 8

Damping in non-soft direction:

18 / 0.5 = 36

Limitations

The actual stiffness in the soft direction must be lower than the stiffness in the non-soft direction. Also, the actual damping in the soft direction must be lower than the damping in the non-soft direction. If this is not the case, there will be an error message.

4.2 Example of parameter settings for high friction situations

Example of parameter settings for high friction

When the robot does not seem to follow the direction set to soft it can be due to high friction, causing a non soft direction to move before a soft direction. To increase stiffness and improve the soft direction accuracy, set the following values for the system parameters for the type *CSS*.

System parameter	Value
Stiffness in non soft dir. max	100
Stiffness max	50
Stiffness in non soft dir. max move	100
Stiffness max move	50
Damping stability limit	0.3
Stiffness to damping ratio move	2
Stiffness to damping ratio	2

Then, in RAPID, start with `StiffnessNonSoftDir` 50 and increase in small steps until robot behaves as desired.

Note

Changing parameters in too big steps can cause robot to be unstable. Decrease the parameters if small oscillations occur for the robot.

4.3 Conversion from damping settings in previous versions

About the changes in the settings

Stiffness and damping for soft direction are set the same way as earlier versions of RobotWare. However, for non-soft direction the values are set and calculated differently than versions prior to RobotWare 5.14.

Conversion table

To convert SoftMove programs made in earlier versions of RobotWare, with maintained stiffness and damping values in the non-soft direction, use the following table.

Conversion of system parameters:

New system parameter	Calculation from previous parameters
Stiffness to damping ratio	*Stiffness to damping ratio*
Stiffness in non soft dir. min	*Damping min * Stiffness to damping ratio*
Stiffness in non soft dir. max	*Damping max * Stiffness to damping ratio*
Stiffness to damping ratio move	*Stiffness to damping ratio move*
Stiffness in non soft dir. min move	*Damping min move * Stiffness to damping ratio move*
Stiffness in non soft dir. max move	*Damping max move * Stiffness to damping ratio move*

The argument `\StiffnessNonSoftDir` in the instruction `CSSAct` should have the same value as the previous argument `\Damping`.

This page is intentionally left blank

5 RAPID reference information

5.1 Instructions

5.1.1 CSSAct - Cartesian Soft Servo activation

Usage

`CSSAct` is used to activate Cartesian Soft Servo in a given direction.

This instruction can only be used in the main task `T_ROB1` or, if in a MultiMove system, in motion tasks.

Basic examples

Basic example of the instruction `CSSAct` is illustrated below.

Example 1

```
CSSAct CSS_X;
! Robot is now soft in the tool's X direction
```

Cartesian Soft Servo will be activated in the current tool x-direction.

Arguments

```
CSSAct [\RefFrame] [\RefOrient] SoftDir [\StiffnessNonSoftDir] |
    [\Damping] [\Stiffness] [\AllowMove] [\Ramp]
```

`[\RefFrame]`

Reference Frame

Data type: `cssframe` (see *cssframe - Soft direction coordinate system on page 49*)

The coordinate system the soft direction is related to.

Default: Tool coordinate system.

`[\RefOrient]`

Reference Orientation

Data type: `orient`

This argument gives the possibility to rotate the coordinate system described by `RefFrame`.

Default: No rotation [1,0,0,0]

`SoftDir`

Soft Direction

Data type: `css_soft_dir` (see *css_soft_dir - Soft direction on page 48*)

The Cartesian direction in which the robot will be soft. Soft direction is in relation to `RefFrame`.

`[\StiffnessNonSoftDir]`

Stiffness in non-soft directions

Data type: `num`

Continues on next page

This argument sets the softness for all directions that are not defined as soft by the argument `SoftDir`.

Default: 50%

`[\Damping]`

Damping

Data type: `num`

This argument is obsolete. For details about converting to `StiffnessNonSoftDir`, see *Conversion from damping settings in previous versions on page 31*.

`[\Stiffness]`

Stiffness

Data type: `num`

This argument describes how strongly the robot tries to move back to the reference point when it is pushed away from that point. It is a percentage of a configured value where 0 gives no spring effect of going back to the reference point.

Default: 50%

`[\AllowMove]`

Allow Movement

Data type: `switch`

When this switch is used movement instructions will be allowed during the activated soft mode. Note that using `\AllowMove` will internally increase the value of the stiffness parameter. For example, it is not possible to make the robot free-floating if `\AllowMove` is used.

`[\Ramp]`

Ramp

Data type: `num`

This argument defines how fast the softness is implemented, as a percentage of the value set by the system parameter *Activation smoothness time*. Can be set to between 1 and 500%.

Default: 100%

Limitations

`CSSAct` is only allowed in a motion controlling task.

`CSSAct` does not activate motion control. A move instruction must be executed before `CSSAct`.

The actual stiffness in the soft direction must be lower than the stiffness in the non-soft direction. Also, the actual damping in the soft direction must be lower than the damping in the non-soft direction. If this is not the case, there will be an error message. For details about calculation of stiffness and damping, see *Stiffness and damping in the different directions on page 27*.

If the argument `\AllowMove` is used, it is allowed to activate SoftMove while the robot is moving (`CSSAct` called after a move instruction with a fly-by point). In this

Continues on next page

case, it is not possible to determine exactly where along the path SoftMove is activated, unless a `trigg` instruction is used.

WARNING

The braking distance for category 1 stops will be longer when SoftMove is active.

Predefined data

```
CONST cssframe CSS_REFFRAME_TOOL := 1;
CONST cssframe CSS_REFFRAME_WOBJ := 2;

CONST css_soft_dir CSS_X := 1;
CONST css_soft_dir CSS_Y := 2;
CONST css_soft_dir CSS_Z := 3;
CONST css_soft_dir CSS_XY := 4;
CONST css_soft_dir CSS_XZ := 5;
CONST css_soft_dir CSS_YZ := 6;
CONST css_soft_dir CSS_XYZ := 7;
CONST css_soft_dir CSS_XYRZ := 8;
CONST css_soft_dir CSS_ARM_ANGLE := 9;
```

Syntax

```
CSSAct
  [ '\' RefFrame ':=' < expression (IN) of cssframe > ',']
  [ '\' RefOrient ':=' < expression (IN) of orient > ',']
  [ SoftDir ':=' ] < expression (IN) of css_soft_dir >
  [ '\' StiffnessNonSoftDir ':=' < expression (IN) of num > ] |
  [ '\' Damping ':=' < expression (IN) of num > ]
  [ '\' Stiffness ':=' < expression (IN) of num > ]
  [ '\' AllowMove ]
  [ '\' Ramp ':=' < expression (IN) of num > ] ';'
```

Related information

For information about	See
Deactivation of Cartesian Soft Servo	*CSSDeact - Deactivates Cartesian Soft Servo on page 36* *CSSDeactMoveL - Moves the robot linearly and deactivates Cartesian Soft Servo on page 37*
Identification of static friction for Cartesian Soft Servo	*CSSOffsetTune - Force offset identification on page 43*
Activation of force offset for static friction	*CSSForceOffsetAct - Cartesian Soft Servo force offset activation on page 40*

5.1.2 CSSDeact - Deactivates Cartesian Soft Servo

Usage

`CSSDeact` is used to shift the robot back to stiff control. Any force offset applied by `CSSForceOffsetAct` is also deactivated.

This instruction can only be used in the main task `T_ROB1` or, if in a MultiMove system, in motion tasks.

Basic examples

The following example illustrates the instruction `CSSDeact`.

Example 1

```
CSSDeact;
```

Limitations

- Deactivation can only be done in stop points.
- `CSSDeact` in only allowed in a motion controlling task.

Program execution

The stiffness of the robot is gradually increased and the robot will be stiff when it returns from the instruction.

Syntax

```
CSSDeact ';'
```

Related information

For information about	See
Deactivation of Cartesian Soft Servo	*CSSDeactMoveL - Moves the robot linearly and deactivates Cartesian Soft Servo on page 37*
Identification of static friction for Cartesian Soft Servo	*CSSOffsetTune - Force offset identification on page 43*
Activation of force offset for static friction	*CSSForceOffsetAct - Cartesian Soft Servo force offset activation on page 40*

5.1.3 CSSDeactMoveL - Moves the robot linearly and deactivates Cartesian Soft Servo

Usage

`CSSDeactMoveL` is used to move the tool center point (TCP) linearly to a given stop point destination while shifting the robot back to stiff control. Any force offset applied by `CSSForceOffsetAct` is also deactivated.

This instruction can only be used in the main task `T_ROB1` or, if in a MultiMove system, in motion tasks.

Basic examples

Basic examples of the instruction `CSSDeactMoveL` are illustrated below.

Example 1

```
CSSDeactMoveL p1,v200,tool2 \WObj:=fixture;
```

The TCP of the tool, `tool2`, is moved linearly to a stop point with speed data `v200`. This position is specified in the object coordinate system for `fixture`.

Example 2

```
CSSDeactMoveL *,v100,grip3;
```

The TCP of the tool, `grip3`, is moved linearly to a stop point stored in the instruction (marked with an *).

Arguments

```
CSSDeactMoveL ToPoint Speed Tool [\WObj] [\TLoad]
```

`ToPoint`

Data type: `robtarget`

The destination point of the robot and additional axes. It is defined as named position or stored directly in the instruction (marked with an * in the instruction).

`Speed`

Data type: `speeddata`

The speed data that applies to movements. Speed data defines the velocity for the tool center point, the tool reorientation, and additional axes. The TCP speed allowed for `CSSDeactMoveL` is limited to 500 mm/s.

`Tool`

Data type: `tool`

The tool in use when the robot moves. The tool center point is the point moved to the specified destination position.

`[ \WObj]`

Work Object

Data type: `wobjdata`

The work object (coordinate system) to which the robot position in the instruction is related.

Continues on next page

This argument can be omitted and if so then the position is related to the world coordinate system. If, on the other hand, a stationary tool or coordinated external axes are used then this argument must be specified to perform a linear movement relative to the work object.

`[ \TLoad ]`

Total load

Data type: `loaddata`

The `\TLoad` argument describes the total load used in the movement. The total load is the tool load together with the payload that the tool is carrying. If the `\TLoad` argument is used, then the `loaddata` in the current `tooldata` is not considered.

If the `\TLoad` argument is set to `load0`, then the `\TLoad` argument is not considered and the `loaddata` in the current `tooldata` is used instead.

To be able to use the `\TLoad` argument it is necessary to set the value of the system parameter `ModalPayLoadMode` to 0. If `ModalPayLoadMode` is set to 0, it is no longer possible to use the instruction `GripLoad`.

The total load can be identified with the service routine LoadIdentify. If the system parameter `ModalPayLoadMode` is set to 0, the operator has the possibility to copy the `loaddata` from the tool to an existing or new `loaddata` persistent variable when running the service routine.

It is possible to test run the program without any payload by using a digital input signal connected to the system input `SimMode` (Simulated Mode). If the digital input signal is set to 1, the `loaddata` in the optional argument `\TLoad` is not considered, and the `loaddata` in the current `tooldata` is used instead.

Note

The default functionality to handle payload is to use the instruction `GripLoad`. Therefore the default value of the system parameter `ModalPayLoadMode` is 1.

Limitations

There are the following limitations:

- Deactivation can only be done in stop points.
- The programmed TCP speed is not allowed to exceed 500 mm/s.
- Only allowed in a motion controlling task.

Program execution

The robot and additional axes are moved to the destination position as follows:

- The TCP of the tool is moved linearly at constant programmed velocity.
- The tool is reoriented at equal intervals along the path.
- Uncoordinated additional axes are executed at a constant velocity in order for them to arrive at the destination point at the same time as the robot axes.
- The stiffness of the robot is gradually increased during the movement. When the stop point is reached the robot will be stiff.

Continues on next page

If it is not possible to attain the programmed velocity for the reorientation or the additional axes then the velocity of the TCP will be reduced.

Syntax

```
CSSDeactMoveL
  [ ToPoint ':='] < expression (IN) of robtarget > ','
  [ Speed ':='] < expression (IN) of speeddata > ','
  [ Tool ':='] < persistent (PERS) of tooldata>
  [ '\' WObj ':=' < persistent (PERS) of wobjdata >]
  [ '\' TLoad ':=' < persistent (PERS) of loaddata > ] ';'
```

Related information

For information about	See
Deactivation of Cartesian Soft Servo	*CSSDeact - Deactivates Cartesian Soft Servo on page 36*
Other positioning instructions, motion summary	*Technical reference manual - RAPID overview*
Definition of velocity, `speeddata`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of stop point data, `stoppointdata`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of tools, `tooldata`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of work objects, `wobjdata`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Motion in general	*Technical reference manual - RAPID overview*
Coordinate systems	*Technical reference manual - RAPID overview*

5.1.4 CSSForceOffsetAct - Cartesian Soft Servo force offset activation

Usage

`CSSForceOffsetAct` is used whenever a force offset is needed to compensate for static friction. If the force value is too low it will be hard to push the robot. On the other hand, if the force value is too high the robot will float away by itself. By using the instruction `CSSOffsetTune` a good starting point for tuning is achieved.

This instruction can only be used in the main task `T_ROB1` or, if in a MultiMove system, in motion tasks.

Note

A warning message appears if the force is applied in a non-soft direction.

Basic examples

Basic example of the instruction `CSSForceOffsetAct` is illustrated below.

Example 1

```
CSSForceOffsetAct CSS_POSZ,10;
```

This example will activate the force offset with a value of 10 Newton in the positive z-direction.

Arguments

```
CSSForceOffsetAct OffsetDir Force [\ForceRamp]
```

`OffsetDir`

Offset Direction

Data type: `css_offset_dir` (see *css_offset_dir - Cartesian Soft Servo offset direction on page 47*)

The direction of the force offset to achieve the desired softness or movement.

`Force`

Force

Data type: `num`

The value of the force reference in Newton. Note that this is only a rough reference value and in most cases it will *not* be very accurate. A suitable start value can be identified with the instruction `CSSOffsetTune`. If the value of this force is too high the robot will start to move by itself, without any external force pushing it.

`[\ForceRamp]`

Force Ramp

Data type: `num`

Value in percent (between 0 and 100) of the default value in the configuration. Determines how fast the force is ramped up to the target value. The default value is 50%, of which 100% corresponds to the value of the system parameter *Ramping for force offset change* (`force_offset_ramp`).

Continues on next page

Limitations

It is only possible to compensate for friction in one direction (positive or negative x, y or z). It is not possible to combine two directions.

Predefined data

```
CONST css_offset_dir CSS_POSX := 1;
CONST css_offset_dir CSS_NEGX := 2;
CONST css_offset_dir CSS_POSY := 3;
CONST css_offset_dir CSS_NEGY := 4;
CONST css_offset_dir CSS_POSZ := 5;
CONST css_offset_dir CSS_NEGZ := 6;
```

Syntax

```
CSSForceOffsetAct
  [ OffsetDir ':=' ] < expression (IN) of css_offset_dir > ','
  [ Force ':=' ] < expression (IN) of num >
  [ '\' ForceRamp ´:=´ < expression (IN) of num > ] ';'
```

Related information

For information about	See
Deactivation of force offset	*CSSForceOffsetDeact - Cartesian Soft Servo force offset deactivation on page 42*
Identify static friction for Cartesian Soft Servo	*CSSOffsetTune - Force offset identification on page 43*

5.1.5 CSSForceOffsetDeact - Cartesian Soft Servo force offset deactivation

Usage

`CSSForceOffsetDeact` is used to deactivate the force offset.

This instruction can only be used in the main task `T_ROB1` or, if in a MultiMove system, in motion tasks.

> **Note**
>
> The following combination is redundant because `CSSDeactMoveL` will also deactivate any active force offset:
>
> ```
> CSSForceOffsetDeact;
> CSSDeactMoveL *,v100,tool1;
> ```

Basic examples

Basic example of the instruction `CSSForceOffsetDeact` is illustrated below.

Example 1

```
CSSForceOffsetDeact;
```

Force offset will be deactivated, and no compensation for static friction will be applied.

Syntax

```
CSSForceOffsetDeact ';'
```

Related information

For information about	See
Activation of force offset	*CSSForceOffsetAct - Cartesian Soft Servo force offset activation on page 40*
Identification of static friction	*CSSOffsetTune - Force offset identification on page 43*

5.1.6 CSSOffsetTune - Force offset identification

Usage

`CSSOffsetTune` is used to identify the static friction level in a certain position. This value can later be used as input to the instruction `CSSForceOffsetAct` in the same position.

`CSSOffsetTune` will ramp up a force until movement is detected and then return the value that corresponds to the static friction level.

This instruction can only be used in the main task `T_ROB1` or, if in a MultiMove system, in motion tasks.

Basic examples

Basic example of the instruction `CSSOffsetTune` is illustrated below.

Example 1

```
PERS num force_offset:=0;
...
MoveL p10, v50, fine, tool1;
CSSOffsetTune CSS_POSX,force_offset;
```

The static friction level is calculated and stored in the variable `force_offset`. This value can later be used in `ForceOffsetAct` for the same position (`p10`).

Arguments

```
CSSOffsetTune [\RefFrame] [\RefOrient] OffsetDir ForceOffset
     [\StiffnessNonSoftDir] | [\Damping] [\MoveDetected]
     [\ForceRamp] [\DeltaAbsForce] [\MaxTestForce]
```

`[\RefFrame]`

Reference Frame

Data type: `cssframe` (see *cssframe - Soft direction coordinate system on page 49*)

The coordinate system the soft direction is related to.

Default: Tool coordinate system

`[\RefOrient]`

Reference Orientation

Data type: `orient`

This argument gives the possibility to rotate the coordinate system described by `RefFrame`.

Default: No rotation [1, 0, 0, 0]

`OffsetDir`

Offset Direction

Data type: `css_offset_dir` (see *css_offset_dir - Cartesian Soft Servo offset direction on page 47*)

The direction of the offset to achieve the desired softness or movement.

Continues on next page

`ForceOffset`

Force Offset

Data type: `num`

The result of the identification and the force, in Newton, corresponding to the static friction in the chosen direction.

`[\StiffnessNonSoftDir]`

Stiffness in non-soft directions

Data type: `num`

This argument sets the softness for all directions that are not defined as soft by the argument `SoftDir`. Set as percentage of the configured value.

Default: 50%

`[\Damping]`

Damping

Data type: `num`

This argument is obsolete. For details about converting to `StiffnessNonSoftDir`, see *Conversion from damping settings in previous versions on page 31*.

`[\MoveDetected]`

Movement Detected

Data type: `num`

The small distance the robot should have moved to consider it being a movement (in mm).

Default: 0.1 mm

`[\ForceRamp]`

Force Ramp

Data type: `num`

Value in percent (between 0 and 100) of the default value in the configuration. Determines how fast the force is ramped up between the increment steps. The default value is 100%, which corresponds to the value of the system parameter *Ramping for force offset change*.

`[\DeltaAbsForce]`

Delta Absolute Force

Data type: `num`

The force is increased stepwise with a value set by the argument `DeltaAbsForce`. A higher value will lower the identification time but give a coarser result. The identification will continue to stepwise increase the force until movement is detected or the `\MaxTestForce` value is achieved.

`[\MaxTestForce]`

Maximum Test Force

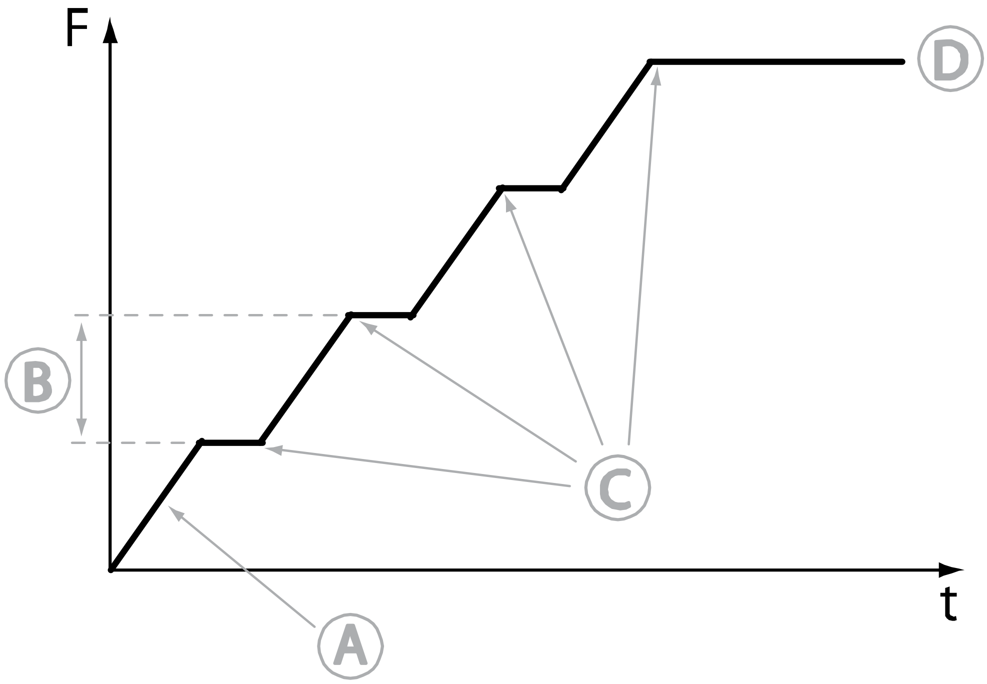
Data type: `num`

Continues on next page

The identification will continue to stepwise increase the force until movement is detected or if `MaxTestForce` is tested without movement.

Default: 500 N

Illustration



xx1000001402

A	The gradient of the force ramp is specified by `ForceRamp`.
B	The size of the increment steps is specified by `DeltaAbsForce`.
C	At every force increment step, a test is done to see if the robot moves.
D	The first force level that makes the robot move will be the identified friction offset level that is returned in the argument `ForceOffset`.

Limitations

`CSSOffsetTune` does not activate motion control. A move instruction must be executed before `CSSOffsetTune`.

Predefined data

```
CONST cssframe CSS_REFFRAME_TOOL := 1;
CONST cssframe CSS_REFFRAME_WOBJ := 2;

CONST css_offset_dir CSS_POSX := 1;
CONST css_offset_dir CSS_NEGX := 2;
CONST css_offset_dir CSS_POSY := 3;
CONST css_offset_dir CSS_NEGY := 4;
CONST css_offset_dir CSS_POSZ := 5;
CONST css_offset_dir CSS_NEGZ := 6;
```

Continues on next page

Syntax

```
CSSOffsetTune
  [ '\' RefFrame ':=' < expression (IN) of cssframe > ',']
  [ '\' RefOrient ':=' < expression (IN) of orient > ',']
  [ OffsetDir ':=' ] < expression (IN) of css_offset_dir > ','
  [ ForceOffset ':=' ] < variable or persistent (INOUT) of num >
       ','
  [ '\' StiffnessNonSoftDir ':=' < expression (IN) of num > ] |
  [ '\' Damping ':=' < expression (IN) of num > ]
  [ '\' MoveDetected ':=' < expression (IN) of num > ]
  [ '\' ForceRamp ':=' < expression (IN) of num > ]
  [ '\' DeltaAbsForce ':=' < expression (IN) of num > ]
  [ '\' MaxTestForce ':=' (IN) of num > ] ';'
```

Related information

For information about	See
Activation of force offset	*CSSForceOffsetAct - Cartesian Soft Servo force offset activation on page 40*

5.2 Data types

5.2.1 css_offset_dir - Cartesian Soft Servo offset direction

Usage

`css_offset_dir` is used as an input to find out the level of static friction in a Cartesian direction.

Description

This data type is used as input for the instructions `CSSOffsetTune` and `CSSForceOffsetAct` to specify a certain Cartesian direction.

Examples

Basic examples of the `css_offset_dir` data type is illustrated below.

Example 1

```
VAR css_offset_dir offdir1;
...
offdir1 := CSS_POSX;
```

The variable `offdir1` is set to `CSS_POSX`.

Predefined data

Constant	Value	Comment
CSS_POSX	1	Force reference will be in positive X direction.
CSS_NEGX	2	Force reference will be in negative X direction
CSS_POSY	3	Force reference will be in positive Y direction
CSS_NEGY	4	Force reference will be in negative Y direction.
CSS_POSZ	5	Force reference will be in positive Z direction.
CSS_NEGZ	6	Force reference will be in negative Z direction.

Characteristics

`css_offset_dir` is an alias data type for `num` and consequently inherits its characteristics.

Related information

For information about	See
Identification of static friction	*CSSOffsetTune - Force offset identification on page 43*
Activation of force offset	*CSSForceOffsetAct - Cartesian Soft Servo force offset activation on page 40*
Alias data types	*Technical reference manual - RAPID overview*

5.2.2 css_soft_dir - Soft direction

Usage

`css_soft_dir` is an alias data type that specifies the soft direction.

Description

Soft direction describes the direction(s) where the robot moves softly. This is used by the instruction `CSSAct`.

Examples

A basic example of the `css_soft_dir` data type is illustrated below.

Example 1

```
VAR css_soft_dir mydir;
...
mydir := CSS_Z;
```

The variable `mydir` is set to `CSS_Z`.

Predefined data

Constant	Value	Comment
CSS_X	1	X direction
CSS_Y	2	Y direction
CSS_Z	3	Z direction
CSS_XY	4	XY plane
CSS_XZ	5	XZ plane
CSS_YZ	6	YZ plane
CSS_XYZ	7	all linear directions (but no rotational)
CSS_XYRZ	8	XY plane and rotational around the Z axis
CSS_ARM_ANGLE	9	Arm angle (only for 7-axis robots)

Characteristics

`css_soft_dir` is an alias data type for `num` and consequently inherits its characteristics.

Related information

For information about	See
Activation of Cartesian Soft Servo	*CSSAct - Cartesian Soft Servo activation on page 33*
Alias data types	*Technical reference manual - RAPID overview*

5.2.3 cssframe - Soft direction coordinate system

Usage

`cssframe` is the reference frame in which the soft direction is specified.

Description

The reference frame can either be CSS_REFFRAME_TOOL or CSS_REFFRAME_WOBJ.

Examples

Basic example of the `cssframe` data type is illustrated below.

Example 1

```
VAR cssframe refframe1;
...
refframe1 := CSS_FREFRAME_TOOL;
```

Predefined data

Constant	Value	Comment
CSS_REFFRAME_TOOL	1	Softness direction will be in relation to current tool.
CSS_REFFRAME_WOBJ	2	Softness direction will be in relation to current work object.

Characteristics

`cssframe` is an alias data type for `num` and consequently inherits its characteristics.

Related information

For information about	See
Activation of Cartesian Soft Servo	*CSSAct - Cartesian Soft Servo activation on page 33*
Identification of static friction	*CSSOffsetTune - Force offset identification on page 43*
Alias data types	*Technical reference manual - RAPID overview*

This page is intentionally left blank

6 System parameters reference information

6.1 Type CSS

6.1.1 The CSS type

Overview

This section describes the type *CSS* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CSS

Type description

The *CSS* type describes the common parameters for *Cartesian Soft Servo*. There is one set of parameters for each robot.

Related information

Application manual - Controller software IRC5

6.1.2 Name

Parent

Name belongs to the type *CSS*, in the topic *Motion*.

Cfg name

name

Usage

Name is used to define the name for the Cartesian Soft Servo setup.

Default value

The default value is robX where X is the robot number.

Allowed values

A string with maximum 32 characters.

6.1.3 Activation during movement smoother

Parent

Activation during movement smoother belongs to the type *CSS*, in the topic *Motion*.

Cfg name

smooth_activation_during_movement

Description

This parameter can prevent false sensitive supervision, for example SafeMove events, when SoftMove is activated during a movement with `\AllowMove`.

Usage

Setting this value prevents false alarm from sensitive supervision when activating CSS while the robot is moving.

The parameter will make linear directions less accurate when robot has moved far away from activation point and should not be used when application needs robot to move linearly for more than a few centimeters.

Parameter will only have effect when `\AllowMove` is set in the `CSSAct` instruction.

Allowed values

Yes or No.

Default value is No.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.4 Activation smoothness time

Parent

Activation smoothness time belongs to the type *CSS*, in the topic *Motion*.

Cfg name

act_smoothness_time

Description

Activation smoothness time controls the time needed for activation.

Usage

Activation smoothness time is used to control the smoothness behavior at activation of Cartesian Soft Servo. It sets the time for ramping to less stiffness.

A higher value gives smoother activation but higher cycle time.

A lower value might cause the robot to jerk but will lower the cycle time.

Use a value higher than 0 when activating SoftMove while the robot is moving (`CSSAct` after a move instruction with a fly-by point).

Default value

The default value is 0.1 second.

Allowed values

A value between 0 and 10 seconds.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.5 Deactivation smoothness time

Parent

Deactivation smoothness time belongs to the type *CSS*, in the topic *Motion*.

Cfg name

deact_smoothness_time

Description

Deactivation smoothness time controls the time needed for deactivation.

Usage

Deactivation smoothness time is used to control the smoothness behavior at deactivation of Cartesian Soft Servo. It sets the time for ramping up from the lower stiffness.

A higher value gives smoother deactivation but higher cycle time.

A lower value might cause robot to jerk but will lower the cycle time.

Default value

The default value is 0.1 seconds.

Allowed values

A value between 0 and 10 seconds.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.6 Max pos error in x, Max pos error in y, Max pos error in z

Parent

The parameters *Max pos error in x*, *Max pos error in y* and *Max pos error in z* belong to the type *CSS*, in the topic *Motion*.

Cfg name

max_pos_error_x

max_pos_error_y

max_pos_error_z

Description

Max pos error in x defines the position supervision limit in x direction.

Max pos error in y defines the position supervision limit in y direction.

Max pos error in z defines the position supervision limit in z direction.

The position supervision limit is the maximum distance between the actual TCP position and the reference position (see *Reference position on page 18*).

Usage

These parameters are used to set up a supervision limit for position in the respective direction. The position is supervised in the current CSS_REF_FRAME (the frame used in the RAPID instruction `CSSAct`). If this limit is exceeded, the robot's brakes are activated.

WARNING

Make sure that these parameters are set according to the application to be used. Otherwise there is a risk that the supervision will trigger and possibly lead to equipment damage.

Default value

The default value is 0.5 meters.

Allowed values

A value between 0.001 and 5 meters.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.7 Max pos error around x, Max pos error around y, Max pos error around z, Max pos error in arm angle

Parent

The parameters *Max pos error around x*, *Max pos error around y*, *Max pos error around z*, and *Max pos error in arm angle* belong to the type *CSS*, in the topic *Motion*.

Cfg name

max_pos_error_rx

max_pos_error_ry

max_pos_error_rz

max_pos_error_ra

Description

Max pos error around x defines the position supervision limit around the x direction.

Max pos error around y defines the position supervision limit around the y direction.

Max pos error around z defines the position supervision limit around the z direction.

Max pos error in arm angle defines the position supervision limit for the arm angle (only for 7-axis robots).

The position supervision limit is the maximum distance between the actual TCP position and the reference position (see *Reference position on page 18*).

Usage

These parameters are used to set up a supervision limit for position around z direction. The position is supervised in the current CSS_REF_FRAME (the frame used in the RAPID instruction `CSSAct`).

WARNING

Make sure that this parameter is set according to the application to be used. Otherwise there is a risk that the supervision will trigger and possibly lead to equipment damage.

Default value

The default value is 0.5 radians.

Allowed values

A value between 0.001 and 5 radians.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.8 Max speed error in x, Max speed error in y, Max speed error in z

Parent

The parameters *Max speed error in x*, *Max speed error in y* and *Max speed error in z* belong to the type *CSS*, in the topic *Motion*.

Cfg name

max_speed_error_x

max_speed_error_y

max_speed_error_z

Description

Max speed error in x defines the speed supervision limit in x direction.

Max speed error in y defines the speed supervision limit in y direction.

Max speed error in z defines the speed supervision limit in z direction.

Usage

These parameters are used to set up a supervision limit for speed in the respective direction. The position is supervised in the current CSS_REF_FRAME (the frame used in the RAPID instruction `CSSAct`). If this limit is exceeded, the robot's brakes are activated.

WARNING

Make sure that these parameters are set according to the application to be used. Otherwise there is a risk that the supervision will trigger and possibly lead to equipment damage.

Additional information

In manual mode, there is still a max speed of 250 mm/s. That means that total speed must not exceed 250 mm/s and also that the speed in the respective directions must not exceed *Max speed error in x*, *Max speed error in y* and *Max speed error in z*.

Default value

The default value is 1 meter per second.

Allowed values

A value between 0.001 and 10 meters per second.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.9 Max speed error around x, Max speed error around y, Max speed error around z, Max speed error in arm angle

Parent

Max speed error around x, *Max speed error around y*, *Max speed error around z*, and *Max speed error in arm angle* belong to the type *CSS*, in the topic *Motion*.

Cfg name

max_speed_error_rx

max_speed_error_ry

max_speed_error_rz

max_speed_error_ra

Description

Max speed error around x defines the speed supervision limit around the x direction.

Max speed error around y defines the speed supervision limit around the y direction.

Max speed error around z defines the speed supervision limit around the z direction.

Max pos error in arm angle defines the speed supervision limit for the arm angle (only for 7-axis robots).

Usage

These parameters are used to set up a supervision limit for rotational speed around the z axis. The position is supervised in the current CSS_REF_FRAME (the frame used in the RAPID instruction `CSSAct`). If this limit is exceeded, the robot's brakes are activated.

WARNING

Make sure that this parameter is set according to the application to be used. Otherwise there is a risk that the supervision will trigger and possibly lead to equipment damage.

Default value

The default value is 0.5 radians per second.

Allowed values

A value between 0.001 and 10 radians per second.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.10 Damping stability limit

Parent

Damping stability limit belongs to the type *CSS*, in the topic *Motion*.

Cfg name

damping_stability_limit

Description

Damping stability limit defines a lower limit value for control stability.

Usage

Damping stability limit is used if the robot stops with error message 50387, which means that Cartesian Soft Servo is close to unstable. If this happens the value should be increased in small steps (0.02 is a recommendation).

Damping stability limit can also be used to increase damping in non-stiff directions.

Default value

The default value for IRB 14000 (YuMi) is 0.15.

The default value for all other robots is 0.01.

Allowed values

A value between -1 and 1.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.11 Ramp for force offset change

Parent

Ramp for force offset change belongs to the type *CSS*, in the topic *Motion*.

Cfg name

force_offset_ramp

Description

Ramp for force offset change defines the ramp for the force offset change in N/s.

Usage

Ramp for force offset change is used to control the force change ramp while using force offset. The unit is N/s.

Default value

The default value is 1000 N/s.

Allowed values

A value between 10 N/s and 10 000 N/s.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.12 Dynamic to static friction ratio

Parent

Dynamic to static fric ratio belongs to the type *CSS*, in the topic *Motion*.

Cfg name

dynamic_to_static_fric_ratio

Description

Dynamic to static fric ratio defines the ratio between static and dynamic friction.

Usage

Dynamic to static fric ratio is used to set up as estimation of the static/dynamic friction ratio. When the instruction `CSSForceOffsetTune` is used and it detects movement, the force that is needed to create the movement is approximately the static friction level.

If the robot is too stiff the value *Dynamic to static fric ratio* of should be increased. If the robot drifts by itself this value should be decreased.

Default value

The default value is 0.7.

Allowed values

A value between 0 and 1.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.13 Stiffness in non soft dir. min

Parent

Stiffness in non soft dir. min belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_non_soft_dir_min

Description

Stiffness in non soft dir. min defines the minimum stiffness level in all non-soft directions when move instructions are not allowed.

Usage

This stiffness will be used when the argument `\StiffnessNonSoftDir` in the instruction `CSSAct` is set to 0% (and the argument `\AllowMove` is not present).

The argument `\StiffnessNonSoftDir` sets a percentage of the span between *Stiffness in non soft dir. min* and *Stiffness in non soft dir. max*. The parameter *Stiffness in non soft dir. min* therefore affects all percentages lower than 100%, but affects lower values more.

Default value

The default value is 5.

Allowed values

A value between 0 and 100.

Note

The value must be lower than the value for the system parameter *Stiffness in non soft dir. max*.

Related information

The minimum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness min on page 65*	*Stiffness min move on page 70*
Non-soft direction	*Stiffness in non soft dir. min on page 63*	*Stiffness in non soft dir. min move on page 68*

6.1.14 Stiffness in non soft dir. max

Parent

Stiffness in non soft dir. max belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_non_soft_dir_max

Description

Stiffness in non soft dir. max defines the maximum stiffness level in all non-soft directions when move instructions are not allowed.

Usage

This stiffness will be used when the argument `\StiffnessNonSoftDir` in the instruction `CSSAct` is set to 100% (and the argument `\AllowMove` is not present).

The argument `\StiffnessNonSoftDir` sets a percentage of the span between *Stiffness in non soft dir. min* and *Stiffness in non soft dir. max*. The parameter *Stiffness in non soft dir. max* therefore affects all percentages higher than 0%, but affects higher values more.

Default value

The default value is 20.

Allowed values

A value between 0 and 100.

Note

The value must be higher than the value for the system parameter *Stiffness in non soft dir. min*.

Related information

The maximum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness max on page 66*	*Stiffness max move on page 71*
Non-soft direction	*Stiffness in non soft dir. max on page 64*	*Stiffness in non soft dir. max move on page 69*

6.1.15 Stiffness min

Parent

Stiffness min belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_min

Description

Stiffness min defines the stiffness level, in soft directions, that corresponds to 0% in the RAPID instruction `CSSAct` when move instructions are not allowed. That is, the softness is only used to follow peripheral equipment.

Usage

Stiffness min is used internally in the control of Cartesian Soft Servo when the `\AllowMove` argument is absent from the `CSSAct` instruction. Change this parameter if you need to increase the stiffness level that corresponds to 0% in the RAPID instruction `CSSAct`.

Default value

The default value is 0.

Allowed values

A value between 0 and 100.

Note

The value must be lower than the value for the system parameter *Stiffness max*.

Related information

The minimum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness min on page 65*	*Stiffness min move on page 70*
Non-soft direction	*Stiffness in non soft dir. min on page 63*	*Stiffness in non soft dir. min move on page 68*

6.1.16 Stiffness max

Parent

Stiffness max belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_max

Description

Stiffness max defines the stiffness level, in soft directions, that corresponds to 100% in the RAPID instruction `CSSAct` when move instructions are not allowed. That is, the softness is only used to follow peripheral equipment.

Usage

Stiffness max is used internally in the control of Cartesian Soft Servo when the `\AllowMove` argument is absent from the `CSSAct` instruction. Change this parameter if you need to change the stiffness level that corresponds to 100% in the RAPID instruction `CSSAct`.

Default value

The default value is 10.

Allowed values

A value between 0 and 100.

Note

The value must be higher than the value for the system parameter *Stiffness min*.

Related information

The maximum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness max on page 66*	*Stiffness max move on page 71*
Non-soft direction	*Stiffness in non soft dir. max on page 64*	*Stiffness in non soft dir. max move on page 69*

6.1.17 Stiffness to damping ratio

Parent

Stiffness to damping ratio belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_to_damping_ratio

Description

Stiffness to damping ratio defines the ratio used internally to calculate the damping based on the stiffness values, when no movement is allowed.

Usage

Stiffness to damping ratio is used internally in the control of Cartesian Soft Servo. The damping is calculated as:

Damping = Stiffness / *Stiffness to damping ratio*

Setting *Stiffness to damping ratio* to a higher value makes the robot more compliant, on the other hand oscillations and other unstable behavior may occur.

Note

Be careful when changing this ratio.

Default value

The default value is 0.5.

Allowed values

A value between 0.001 and 10.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.18 Stiffness in non soft dir. min move

Parent

Stiffness in non soft dir. min move belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_non_soft_dir_min_move

Description

Stiffness in non soft dir. min move defines the minimum stiffness level in all non-soft directions when move instructions are allowed.

Usage

This stiffness will be used when the argument `\StiffnessNonSoftDir` in the instruction `CSSAct` is set to 0% (and the argument `\AllowMove` is present).

The argument `\StiffnessNonSoftDir` sets a percentage of the span between *Stiffness in non soft dir. min move* and *Stiffness in non soft dir. max move*. The parameter *Stiffness in non soft dir. min move* therefore affects all percentages lower than 100%, but affects lower values more.

Default value

The default value is 10.

Allowed values

A value between 0 and 100.

Note

The value must be lower than the value for the system parameter *Stiffness in non soft dir. max move*.

Related information

The minimum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness min on page 65*	*Stiffness min move on page 70*
Non-soft direction	*Stiffness in non soft dir. min on page 63*	*Stiffness in non soft dir. min move on page 68*

6.1.19 Stiffness in non soft dir. max move

Parent

Stiffness in non soft dir. max move belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_non_soft_dir_max_move

Description

Stiffness in non soft dir. max move defines the maximum stiffness level in all non-soft directions when move instructions are allowed.

Usage

This stiffness will be used when the argument `\StiffnessNonSoftDir` in the instruction `CSSAct` is set to 100% (and the argument `\AllowMove` is present).

The argument `\StiffnessNonSoftDir` sets a percentage of the span between *Stiffness in non soft dir. min move* and *Stiffness in non soft dir. max move*. The parameter *Stiffness in non soft dir. max move* therefore affects all percentages higher than 0%, but affects higher values more.

Default value

The default value is 20.

Allowed values

A value between 0 and 100.

Note

The value must be higher than the value for the system parameter *Stiffness in non soft dir. min move*.

Related information

The maximum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness max on page 66*	*Stiffness max move on page 71*
Non-soft direction	*Stiffness in non soft dir. max on page 64*	*Stiffness in non soft dir. max move on page 69*

6.1.20 Stiffness min move

Parent

Stiffness min move belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_min_move

Description

Stiffness min move defines the stiffness level, in soft directions, that corresponds to 0% in the RAPID instruction `CSSAct` when move instructions are allowed.

Usage

Stiffness min move is used internally in the control of Cartesian Soft Servo when the `\AllowMove` argument is present in the `CSSAct` instruction. Change this parameter if you need to change the stiffness level that corresponds to 0% in the RAPID instruction `CSSAct` (when `\AllowMove` is used).

Default value

The default value is 2.

Allowed values

A value between 0 and 100.

Note

The value must be lower than the value for the system parameter *Stiffness max move*.

Related information

The minimum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness min on page 65*	*Stiffness min move on page 70*
Non-soft direction	*Stiffness in non soft dir. min on page 63*	*Stiffness in non soft dir. min move on page 68*

6.1.21 Stiffness max move

Parent

Stiffness max move belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_max_move

Description

Stiffness max move defines the stiffness level, in soft directions, that corresponds to 100% in the RAPID instruction `CSSAct` when move instructions are allowed.

Usage

Stiffness max move is used internally in the control of Cartesian Soft Servo when the `\AllowMove` argument is present in the `CSSAct` instruction. Change this parameter if you need to change the stiffness level that corresponds to 100% in the RAPID instruction `CSSAct` (when `\AllowMove` is used).

Default value

The default value is 10.

Allowed values

A value between 0 and 100.

Note

The value must be higher than the value for the system parameter *Stiffness min move*.

Related information

The maximum value for stiffness is set by four different parameters, depending on direction (soft or non-soft) and if movement is allowed.

	No movement	Movement allowed
Soft direction	*Stiffness max on page 66*	*Stiffness max move on page 71*
Non-soft direction	*Stiffness in non soft dir. max on page 64*	*Stiffness in non soft dir. max move on page 69*

6.1.22 Stiffness to damping ratio move

Parent

Stiffness to damping ratio move belongs to the type *CSS*, in the topic *Motion*.

Cfg name

stiffness_to_damping_ratio_move

Description

Stiffness to damping ratio move defines the ratio used internally to calculate the damping based on the stiffness values, when movement is allowed.

Usage

Stiffness to damping ratio move is used internally in the control of Cartesian Soft Servo. The damping is calculated as:

Damping = Stiffness / *Stiffness to damping ratio move*

Setting this parameter to a higher value makes the robot more compliant, on the other hand oscillations and other unstable behavior may occur.

Note

Be careful when changing this ratio.

Default value

The default value is 0.5.

Allowed values

A value between 0.001 and 10.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

6.1.23 Automatic reactivation of css disabled

Parent

Automatic reactivation of css disabled belongs to the type *CSS*, in the topic *Motion*.

Cfg name

auto_reactivation_disabled

Description

When the robot goes to Motors Off state the brakes are applied and the robot is not soft. When going back to Motors On state, SoftMove is automatically activated again unless *Automatic reactivation of css disabled* is set to *Yes*.

Usage

Setting this value prevents that the robot automatically goes back to soft mode when it goes back to Motors On state.

Automatic reactivation of css disabled can, for example, be set to No during commissioning and set to Yes in production. The robot programmer may want to modpos positions and resume the program with a soft robot. If the robot stops during production, the service engineer may want to jog the robot without the awkward behavior that a soft robot may have.

Allowed values

Yes or No.

The default value is No.

6.1.24 Damping min, Damping max, Damping min move, Damping max move

Parent

The parameters *Damping min*, *Damping max*, *Damping min move* and *Damping max move* belong to the type *CSS*, in the topic *Motion*.

Cfg name

damping_min

damping_max

damping_min_move

damping_max_move

Description

The damping parameters are obsolete. For information on how the damping is calculated, see *Stiffness and damping calculations for advanced users on page 27*.

Usage

The damping parameters can be set, for backward compatibility, but will have no effect.

6.1.25 Minimize damping off

Parent

Minimize damping off belongs to the type *CSS*, in the topic *Motion*.

Cfg name

minimize_damping_off

Description

SoftMove has been improved and continuously optimizes damping to the minimal value achieving stability according to the damping stability limit criteria.

It will never set a lower value than what is calculated from the stiffness to damping ratio. Before, this calculation was only done during activation but the needed damping will vary with the robots position.

The typical effect will be lower damping and fewer instability issues when the robot is moved far from the activation position and when some directions are set to be much stiffer than others.

Usage

Set this value to TRUE to use the old behavior.

Allowed values

TRUE or FALSE.

Default value is FALSE.

Related information

Damping stability limit on page 60

Stiffness to damping ratio on page 67

Stiffness to damping ratio move on page 72

This page is intentionally left blank

Index

Contact us

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

ABB AS, Robotics
Robotics and Motion
Nordlysvegen 7, N-4340 BRYNE, Norway
Box 265, N-4349 BRYNE, Norway
Telephone: +47 22 87 2000

ABB Engineering (Shanghai) Ltd.
Robotics and Motion
No. 4528 Kangxin Highway
PuDong District
SHANGHAI 201319, China
Telephone: +86 21 6105 6666

ABB Inc.
Robotics and Motion
1250 Brown Road
Auburn Hills, MI 48326
USA
Telephone: +1 248 391 9000

www.abb.com/robotics

3HAC050977-001, Rev C, en