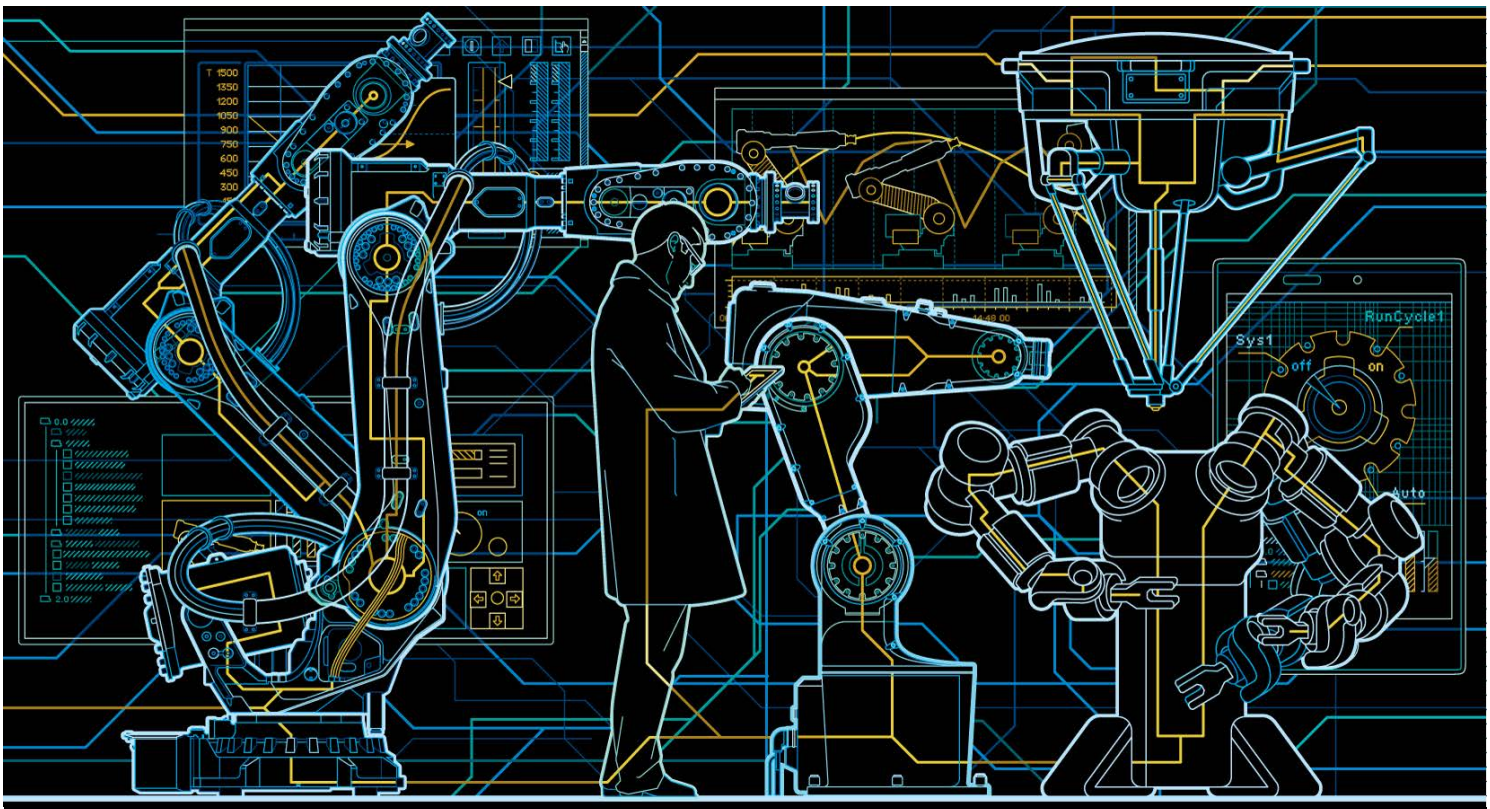


Application manual
Torch services

Power and productivity
for a better world™

Trace back information:
Workspace R16-1 version a2
Checked in 2016-02-26
Skribenta version 4.6.209

Application manual
Torch services

Document ID: 3HAC050981-001
Revision: A

The information in this manual is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this manual.

Except as may be expressly stated anywhere in this manual, nothing herein shall be construed as any kind of guarantee or warranty by ABB for losses, damages to persons or property, fitness for a specific purpose or the like.

In no event shall ABB be liable for incidental or consequential damages arising from use of this manual and products described herein.

Additional copies of this manual may be obtained from ABB.

The original language for this publication is English. Any other languages that are supplied have been translated from English.

ABB AB
Robotics Products
Se-721 68 Västerås
Sweden

Table of contents

This page is intentionally left blank

Overview of this manual

About this manual

This manual explains the basics of when and how to use the following Torch Service options:

- Product overview
- Operation overview
- Requirements overview
- Software set-up
- Software reference, instructions

Usage

This manual can be used either as a reference to find out if an option is the right choice for solving a problem, or as a description of how to use an option. Detailed information regarding syntax for RAPID routines, and similar, is not described here, but can be found in the respective reference manual.

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
- the RAPID programming language
- system parameters and how to configure them.

Reference documents

References	Document ID
Technical reference manual - RAPID overview	*3HAC050947-001*
Technical reference manual - RAPID Instructions, Functions and Data types	*3HAC050917-001*
Operating manual - Getting started, IRC5 and RobotStudio	*3HAC027097-001*
Operating manual - IRC5 with FlexPendant	*3HAC050941-001*
Technical reference manual - System parameters	*3HAC050948-001*
Operating manual - RobotStudio	*3HAC032104-001*
Application manual - GAP	*3HAC024844-001*

Revisions

-	First revision.
A	Minor corrections

This page is intentionally left blank

1 Introduction

1.1 Product overview

About Torch services

Torch Services is a set of functions for maintaining the welding gun of a MIG/MAG welding system.

Available services

The available services consist of:

Automatic mechanical cleaning	Automatic mechanical cleaning of the contact tip and gas nozzle to remove welding spatter.
Automatic spraying	Automatic spraying of the contact tip and gas nozzle with a liquid weld spatter release agent.
Automatic cut-off of the welding wire	Automatic cut-off of the welding wire.

1.2 Operation overview

RAPID instructions

Torch Services consist of a number of Move instructions in RAPID. The instructions are programmed in traditional RAPID programming manner. Each instruction moves the welding gun to the service location and starts the service equipment.

1.3 Requirements

System requirements

This Torch Services version is intended for use in arc welding systems incorporating IRB 1400, 2400, etc. robots.

- RobotWare requirements: 5.06
- Controller requirements: IRC5

Torch services package

The Torch Services package includes software that is loaded into all arc welding motion tasks, when the Torch Services option is purchased. Process configuration parameters are used to connect real I/O signals and to modify the default settings.

User requirements

Any competent robot programmer (RAPID language) may be self-taught to program and use Torch Services.

This page is intentionally left blank

2 Installation

2.1 Software set-up

2.1.1 System parameters

Introduction

Torch Services I/O connections together with additional settings for torch services are configured in the process configuration (PROC).

Actual I/O assignments to real I/O boards are not made by the Torch Services installation. These definitions must be added to the EIO configuration by the user or system designer.

Default process configuration

Below is the default process configuration loaded by Torch Services. There is a separate configuration for each task.

```
PROC:CFG_1.0::
# TCMC proc.cfg file

#
TS_MECHCLEAN_PROP:
-name "TSMC_1" -ReamTime 3 -ClampOpen "diTS1_ClOp" \
-Start "doTS1_St" -AirBlast "doTS1_AiBl"

#
TS_SPRAY_PROP:
-name "TSSP_1" -SprayTime 2 -SprayOn "doTS1_SpOn"

#
TS_WIRECUT_PROP:
-name "TSWC_1" -ShearWidth 20 -ShearDirection 0 \
-Cut "doTS1_St" -CutterOpen "diTS1_ClOp"
```

To change settings

To change settings, RobotStudio is preferably used. It is also possible to load an altered proc.cfg, with the **Add or Replace** feature to override the existing fields with the new settings.

Extend the ream time

For example, a user could extend the ream time by loading a file like this:

```
PROC:CFG_1.0:
# TCMC proc.cfg file

#
TS_MECHCLEAN_PROP:
-name "TSMC_1" -ReamTime 5 -ClampOpen "diTS1_ClOp" \
-Start "doTS1_St" -AirBlast "doTS1_AiBl"
```

Continues on next page

2.1.1 System parameters
Continued

Note

Torch Services does not install any I/O signals in the EIO configuration. It provides only a mechanism to connect to existing signals in the system. If the robotic system is not a turnkey system, I/O signals will need to be installed in the system.

I/O configuration file

Below is an example of an I/O configuration file that could be used to load I/O signals for the default case:

```
EIO:CFG_1.0:5.0:
#
EIO_SIGNAL:
  -Name " diTS1_ClOp" -SignalType "DI" -Unit "Board_A" -UnitMap 0
  -Name " doTS1_St" -SignalType "DO" -Unit "Board_A" -UnitMap 0
  -Name " doTS1_AiBl" -SignalType "DO" -Unit "Board_A" -UnitMap 1
  -Name " doTS1_SpOn " -SignalType "DO" -Unit "Board_A"  -UnitMap
       2
```

2.1.2 Loading software

Loading software

The software is loaded automatically when the option Torch Services is purchased for RobotWare.

> **Note**
>
> Torch Services is a separate Arc option.

This page is intentionally left blank

3 RAPID reference

3.1 MoveMechCleanL - Move linear mechanical clean

Usage

`MoveMechCleanL` is an instruction used to ream (clean) the tool mechanically. The movement to reach the cleaning station is performed linearly.

Basic examples

```
MoveMechCleanL rtApp, rtMechClean, v200, fine, tWeldGun;
```

The tool center point (TCP) of the tool, `tWeldGun`, is moved along a linear path to the position, `rtApp`, with speed data `v200` and zone data `fine`. At `rtApp` the TCP is moved along a linear path to the position `rtMechClean`. At `rtMechClean` the mechanical cleaning equipment is started. After the ream operation the TCP is moved back linearly to position `rtApp`.

Arguments

```
MoveMechCleanL ApproachPoint CleanPoint Speed Zone Tool [\WObj]
    [\TLoad]
```

`ApproachPoint`

Data type: `robtarget`

The approach point of the robot and external axes directly above the clean station. It is defined as a named position or stored directly in the instruction (marked with an * in the instruction).

`CleanPoint`

Data type: `robtarget`

The destination point of the robot and external axes. It is defined as a 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 external axes.

`Zone`

Data type: `zonedata`

Zone data for the movement. Zone data describes the size of the generated corner path.

`Tool`

Data type: `tooldata`

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

`[\WObj]`

Data type: `wobjdata`

Continues on next page

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

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

`[\TLoad]`

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. For a complete description of the `TLoad` argument, see `MoveL` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Program execution

Automatic or continuous mode

When executed, the robot makes a linear movement to the approach position, `ApproachPoint`. Then, the robot makes a linear movement to the cleaning position, `CleanPoint` with fixed low speed. At the cleaning position the mechanical cleaner is started. The cleaner will run the timed specified in the system configuration. When the cleaning operation is done, the robot will move back to the approach position with fixed low speed.

Forward step mode

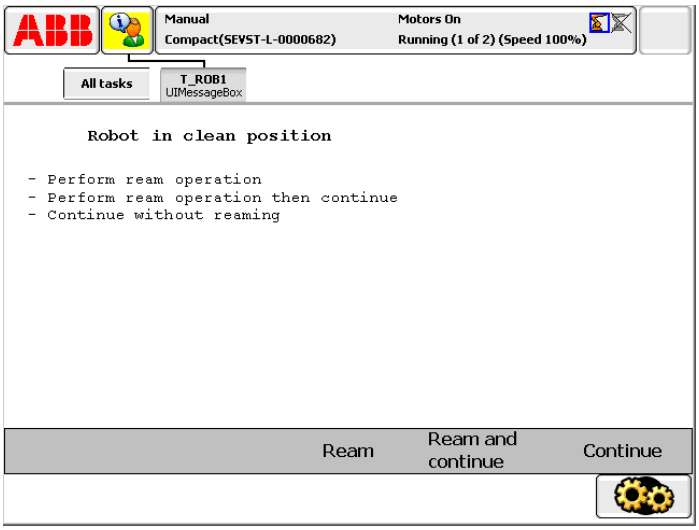
When executed, the robot makes a linear movement to the approach position, `ApproachPoint`.

Then, the robot makes a linear movement to the cleaning position, `CleanPoint` with fixed low speed.

Continues on next page

User dialog

At the cleaning position a user dialog is shown.



xx1400001927

Ream	The reamer will run for one cycle and thereafter the user dialog is shown again.
Ream and continue	The reamer will run for one cycle thereafter the robot will move back to the approach position with fixed low speed.
Continue	The robot will move back to the approach position with fixed low speed.

More examples

```
MoveMechCleanL rtApp, rtMC, vMax, fine, tWeldGun \WObj:=fixture;
```

The TCP of the tool, `tWeldGun`, is moved linearly to the position, `rtApp`. This position is specified in the object coordinate system for fixture.

At `rtApp` the TCP is moved along a linear path to the position `rtMC`. This position is also specified in the object coordinate system for fixture. At `rtMC` the mechanical cleaning equipment is started. After the ream operation the TCP is moved back linearly to position `rtApp`.

Syntax

```
MoveMechCleanL
  [ ApproachPoint ':=' ] < expression (IN) of robtarget > ','
  [ CleanPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ Zone ':=' ] < expression (IN) of zonedata > ','
  [ Tool ':=' ] < persistent (PERS) of tooldata >
  [ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
  [ '\' TLoad ':=' ] < persistent (PERS) of loaddata > ] ';'
```

Related information

	Described in:
`MoveMechCleanJ`	*MoveMechCleanJ - Move joint mechanical clean on page 21*

Continues on next page

3.1 MoveMechCleanL - Move linear mechanical clean

Continued

	Described in:
`MoveSprayL`	*MoveSprayL - Move linear spray on page 25*
`MoveSprayJ`	*MoveSprayJ - Move spray on page 29*
`MoveWireCutL`	*MoveWireCutL - Move linear wire cut on page 33*
`MoveWireCutJ`	*MoveWireCutJ - Move wire cut on page 37*
`MoveL`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of `loaddata`	*Technical reference manual - RAPID Instructions, Functions and Data types*

3.2 MoveMechCleanJ - Move joint mechanical clean

Usage

`MoveMechCleanJ` is an instruction used to ream (clean) the tool mechanically. The movement to reach the cleaning station does not have to be in a straight line.

Basic examples

```
MoveMechCleanJ rtApp, rtMechClean, v200, fine, tWeldGun;
```

The tool center point (TCP) of the tool, `tWeldGun`, is moved to the position, `rtApp`, with speed data `v200` and zone data `fine`. At `rtApp` the TCP is moved along a linear path to the position `rtMechClean`. At `rtMechClean` the mechanical cleaning equipment is started. After the ream operation the TCP is moved back linearly to position `rtApp`.

Arguments

```
MoveMechCleanJ ApproachPoint CleanPoint Speed Zone Tool [\WObj]
    [\TLoad]
```

`ApproachPoint`

Data type: `robtarget`

The approach point of the robot and external axes directly above the spray station. It is defined as a named position or stored directly in the instruction (marked with an * in the instruction).

`CleanPoint`

Data type: `robtarget`

The destination point of the robot and external axes. It is defined as a 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 external axes.

`Zone`

Data type: `zonedata`

Zone data for the movement. Zone data describes the size of the generated corner path.

`Tool`

Data type: `tooldata`

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

`[\WObj]`

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 it is, the position is related to the world coordinate system. If, on the other hand, a stationary tool or coordinated external axes are used, this argument must be specified in order to perform a linear movement relative to the work object.

`[\TLoad]`

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. For a complete description of the `TLoad` argument, see `MoveL` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Program execution

Automatic or continuous mode

When executed, the robot makes a linear movement to the approach position, `ApproachPoint`. Then, the robot makes a linear movement to the cleaning position, `CleanPoint` with fixed low speed. At the cleaning position the mechanical cleaner is started. The cleaner will run the timed specified in the system configuration. When the cleaning operation is done, the robot will move back to the approach position with fixed low speed.

Forward step mode

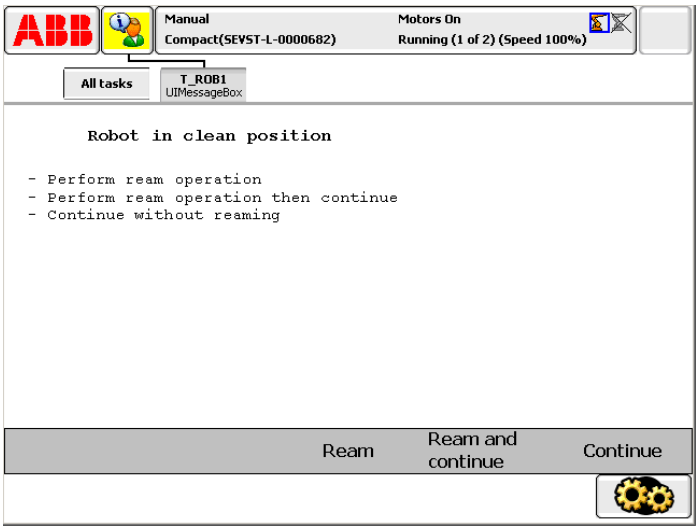
When executed, the robot makes a movement to the approach position, `ApproachPoint`.

Then, the robot makes a linear movement to the cleaning position, `CleanPoint` with fixed low speed.

Continues on next page

User dialog

At the cleaning position a user dialog is shown.



xx1400001927

Ream	The reamer will run for one cycle and thereafter the user dialog is shown again.
Ream and continue	The reamer will run for one cycle thereafter the robot will move back to the approach position with fixed low speed.
Continue	The robot will move back to the approach position with fixed low speed.

More examples

```
MoveMechCleanJ rtApp, rtMC, vMax, fine, tWeldGun \WObj:=fixture;
```

The TCP of the tool, `tWeldGun`, is moved linearly to the position, `rtApp`. This position is specified in the object coordinate system for fixture.

At `rtApp` the TCP is moved along a linear path to the position `rtSP`. This position is also specified in the object coordinate system for fixture. At `rtSP` the mechanical cleaning equipment is started. After the ream operation the TCP is moved back linearly to position `rtApp`.

Syntax

```
MoveMechCleanJ
  [ ApproachPoint ':=' ] < expression (IN) of robtarget > ','
  [ CleanPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ Zone ':=' ] < expression (IN) of zonedata > ','
  [ Tool ':=' ] < persistent (PERS) of tooldata >
  [ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
  [ '\' TLoad ':=' ] < persistent (PERS) of loaddata > ] ';'
```

Related information

	Described in:
`MoveMechCleanL`	*MoveMechCleanL - Move linear mechanical clean on page 17*

Continues on next page

3.2 MoveMechCleanJ - Move joint mechanical clean
Continued

	Described in:
`MoveSprayL`	*MoveSprayL - Move linear spray on page 25*
`MoveSprayJ`	*MoveSprayJ - Move spray on page 29*
`MoveWireCutL`	*MoveWireCutL - Move linear wire cut on page 33*
`MoveWireCutJ`	*MoveWireCutJ - Move wire cut on page 37*
`MoveL`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of `loaddata`	*Technical reference manual - RAPID Instructions, Functions and Data types*

3.3 MoveSprayL - Move linear spray

Usage

`MoveSprayL` is an instruction used to move the weld gun to a station for automatic spraying of the contact tip and gas nozzle with a liquid weld spatter release agent. The movement to reach the cleaning station is performed linearly.

Basic examples

```
MoveSprayL rtApp, rtSpray, v200, fine, tWeldGun;
```

The tool center point (TCP) of the tool, `tWeldGun`, is moved along a linear path to the position, `rtApp`, with speed data `v200` and zone data `fine`. At `rtApp` the TCP is moved along a linear path to the position `rtSpray`. At `rtSpray` the sprayer is started. After the spraying operation the TCP is moved back linearly to position `rtApp`.

Arguments

```
MoveMechCleanL ApproachPoint CleanPoint Speed Zone Tool [\WObj]
     [\TLoad]
```

`ApproachPoint`

Data type: `robtarget`

The approach point of the robot and external axes directly above the clean station. It is defined as a named position or stored directly in the instruction (marked with an * in the instruction).

`SprayPoint`

Data type: `robtarget`

The destination point of the robot and external axes. It is defined as a 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 external axes.

`Zone`

Data type: `zonedata`

Zone data for the movement. Zone data describes the size of the generated corner path.

`Tool`

Data type: `tooldata`

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

`[\WObj]`

Data type: `wobjdata`

Continues on next page

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

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

`[\TLoad]`

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. For a complete description of the `TLoad` argument, see `MoveL` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Program execution

Automatic or continuous mode

When executed, the robot makes a linear movement to the approach position, `ApproachPoint`. Then, the robot makes a linear movement to the spraying position, `SprayPoint` with fixed low speed. At the spraying position the sprayer is started. The sprayer will run the timed specified in the system configuration. When the spraying operation is done, the robot will move back to the approach position with fixed low speed.

Forward step mode

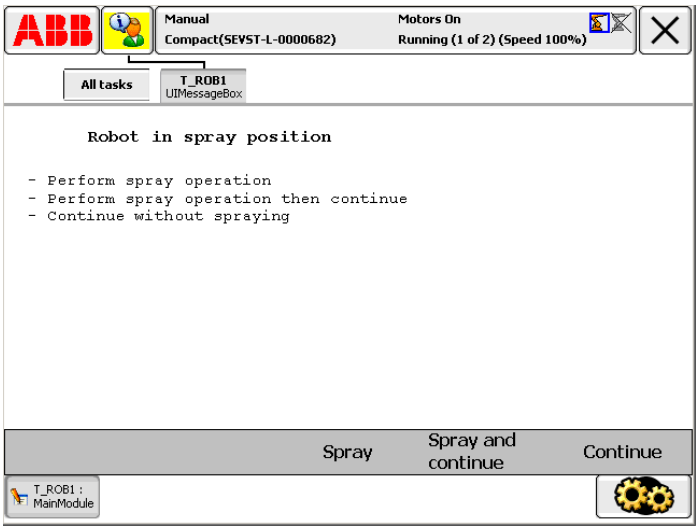
When executed, the robot makes a linear movement to the approach position, `ApproachPoint`.

Then, the robot makes a linear movement to the spraying position, `SprayPoint` with fixed low speed.

Continues on next page

User dialog

At the spraying position a user dialog is shown.



xx1400001928

Spray	The sprayer will run for one cycle and thereafter the user dialog is shown again.
Spray and continue	The sprayer will run for one cycle thereafter the robot will move back to the approach position with fixed low speed.
Continue	The robot will move back to the approach position with fixed low speed.

More examples

```
MoveSprayL rtApp, rtSP, vMax, fine, tWeldGun \WObj:=fixture;
```

The TCP of the tool, `tWeldGun`, is moved linearly to the position, `rtApp`. This position is specified in the object coordinate system for fixture.

At `rtApp` the TCP is moved along a linear path to the position `rtSP`. This position is also specified in the object coordinate system for fixture. At `rtSP` the spraying equipment is started. After the spraying operation the TCP is moved back linearly to position `rtApp`.

Syntax

```
MoveSprayL
  [ ApproachPoint ':=' ] < expression (IN) of robtarget > ','
  [ SprayPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ Zone ':=' ] < expression (IN) of zonedata > ','
  [ Tool ':=' ] < persistent (PERS) of tooldata >
  [ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
  [ '\' TLoad ':=' ] < persistent (PERS) of loaddata > ] ';'
```

Related information

	Described in:
`MoveMechCleanL`	*MoveMechCleanL - Move linear mechanical clean on page 17*

Continues on next page

3.3 MoveSprayL - Move linear spray
Continued

	Described in:
`MoveMechCleanJ`	*MoveMechCleanJ - Move joint mechanical clean on page 21*
`MoveSprayJ`	*MoveSprayJ - Move spray on page 29*
`MoveWireCutL`	*MoveWireCutL - Move linear wire cut on page 33*
`MoveWireCutJ`	*MoveWireCutJ - Move wire cut on page 37*
`MoveL`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of `loaddata`	*Technical reference manual - RAPID Instructions, Functions and Data types*

3.4 MoveSprayJ - Move spray

Usage

`MoveSprayJ` is an instruction used to move the weld gun to a station for automatic spraying of the contact tip and gas nozzle with a liquid weld spatter release agent. The movement to reach the cleaning station does not have to be in a straight line.

Basic examples

```
MoveSprayJ rtApp, rtSpray, v200, fine, tWeldGun;
```

The tool center point (TCP) of the tool, `tWeldGun`, is moved to the position, `rtApp`, with speed data `v200` and zone data `fine`. At `rtApp` the TCP is moved along a linear path to the position `rtSpray`. At `rtSpray` the sprayer is started. After the spraying operation the TCP is moved back linearly to position `rtApp`.

Arguments

```
MoveSprayJ ApproachPoint SprayPoint Speed Zone Tool [\WObj] [\TLoad]
```

`ApproachPoint`

Data type: `robtarget`

The approach point of the robot and external axes directly above the clean station. It is defined as a named position or stored directly in the instruction (marked with an * in the instruction).

`SprayPoint`

Data type: `robtarget`

The destination point of the robot and external axes. It is defined as a 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 external axes.

`Zone`

Data type: `zonedata`

Zone data for the movement. Zone data describes the size of the generated corner path.

`Tool`

Data type: `tooldata`

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

`[\WObj]`

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 it is, the position is related to the world coordinate system. If, on the other hand, a stationary tool or coordinated external axes are used, this argument must be specified in order to perform a linear movement relative to the work object.

`[\TLoad]`

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. For a complete description of the `TLoad` argument, see `MoveL` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Program execution

Automatic or continuous mode

When executed, the robot makes a movement to the approach position, `ApproachPoint`. Then, the robot makes a linear movement to the spraying position, `SprayPoint` with fixed low speed. At the spraying position the sprayer is started. The sprayer will run the timed specified in the system configuration. When the spraying operation is done, the robot will move back to the approach position with fixed low speed.

Forward step mode

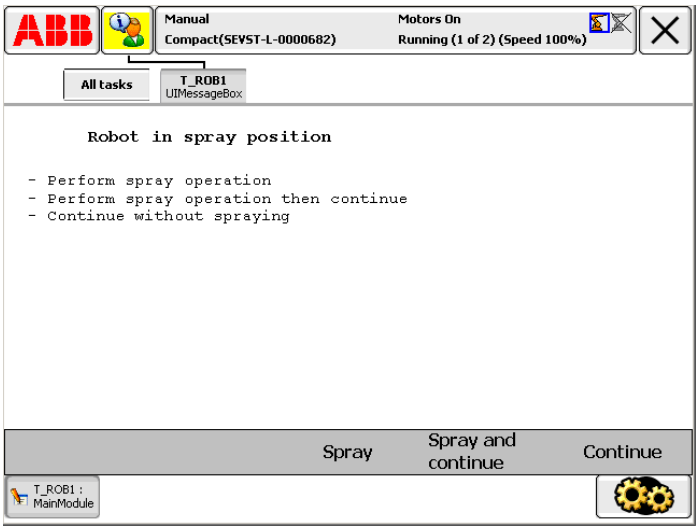
When executed, the robot makes a movement to the approach position, `ApproachPoint`.

Then, the robot makes a linear movement to the spraying position, `SprayPoint` with fixed low speed.

Continues on next page

User dialog

At the spraying position a user dialog is shown.



xx1400001928

Spray	The sprayer will run for one cycle and thereafter the user dialog is shown again.
Spray and continue	The sprayer will run for one cycle thereafter the robot will move back to the approach position with fixed low speed.
Continue	The robot will move back to the approach position with fixed low speed.

More examples

```
MoveSprayJ rtApp, rtSP, vMax, fine, tWeldGun \WObj:=fixture;
```

The TCP of the tool, `tWeldGun`, is moved to the position, `rtApp`. This position is specified in the object coordinate system for fixture.

At `rtApp` the TCP is moved along a linear path to the position `rtSP`. This position is also specified in the object coordinate system for fixture. At `rtSP` the spraying equipment is started. After the spraying operation the TCP is moved back linearly to position `rtApp`.

Syntax

```
MoveSprayJ
  [ ApproachPoint ':=' ] < expression (IN) of robtarget > ','
  [ SprayPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ Zone ':=' ] < expression (IN) of zonedata > ','
  [ Tool ':=' ] < persistent (PERS) of tooldata >
  [ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
  [ '\' TLoad ':=' ] < persistent (PERS) of loaddata > ] ';'
```

Related information

	Described in:
`MoveMechCleanL`	*MoveMechCleanL - Move linear mechanical clean on page 17*

Continues on next page

3.4 MoveSprayJ - Move spray

Continued

	Described in:
`MoveMechCleanJ`	*MoveMechCleanJ - Move joint mechanical clean on page 21*
`MoveSprayL`	*MoveSprayL - Move linear spray on page 25*
`MoveWireCutL`	*MoveWireCutL - Move linear wire cut on page 33*
`MoveWireCutJ`	*MoveWireCutJ - Move wire cut on page 37*
`MoveL`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of `loaddata`	*Technical reference manual - RAPID Instructions, Functions and Data types*

3.5 MoveWireCutL - Move linear wire cut

Usage

`MoveWireCutL` is an instruction used to move the weld gun to a station for automatic cut-off of the welding wire. The movement to reach the cutting station is performed linearly.

Basic examples

```
MoveWireCutL rtApp, rtCut, v200, fine, tWeldGun;
```

The tool center point (TCP) of the tool, `tWeldGun`, is moved along a linear path to the position, `rtApp`, with speed data `v200` and zone data `fine`. At `rtApp` the TCP is moved along a linear path to the position `rtCut`. At `rtCut` the wire is cut off. After the cutting operation the TCP is moved back linearly to position `rtApp`.

Arguments

```
MoveWireCutL ApproachPoint CutPoint Speed Zone Tool [\WObj] [\TLoad]
```

`ApproachPoint`

Data type: `robtarget`

The approach point of the robot and external axes directly above the spray station. It is defined as a named position or stored directly in the instruction (marked with an * in the instruction).

`CutPoint`

Data type: `robtarget`

The destination point of the robot and external axes. It is defined as a 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 external axes.

`Zone`

Data type: `zonedata`

Zone data for the movement. Zone data describes the size of the generated corner path.

`Tool`

Data type: `tooldata`

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

`[\WObj]`

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 it is, the position is related to the world coordinate system. If, on the other hand, a stationary tool or coordinated external axes are used, this argument must be specified in order to perform a linear movement relative to the work object.

`[\TLoad]`

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. For a complete description of the `TLoad` argument, see `MoveL` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Program execution

Automatic or continuous mode

When executed, the robot makes a linear movement to the approach position, `ApproachPoint`. Then, the cutter opens and the robot makes a linear movement to the cutting position, `CutPoint` with fixed low speed. When the cutting operation is done, the robot will move back to the approach position with fixed low speed.

Note

If the system parameters *ShearWidth* and *ShearDirection* are used the cut position will be slightly different each time to minimize the shear wear.

Forward step mode

When executed, the robot makes a linear movement to the approach position, `ApproachPoint`.

Then, the robot makes a linear movement to the cutting position, `SprayPoint` with fixed low speed.

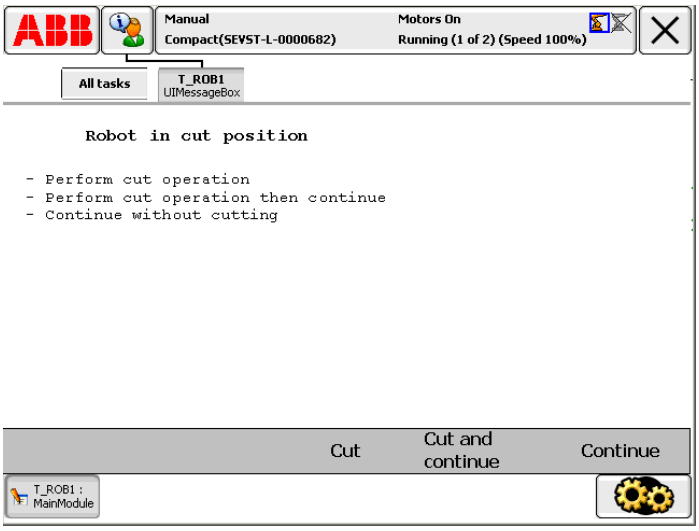
Note

If the system parameters *ShearWidth* and *ShearDirection* are used the cut position will be slightly different each time to minimize the shear wear.

Continues on next page

User dialog

At the cutting position a user dialog is shown.



xx1400001929

Cut	The cutter will close and open, thereafter the user dialog is shown again.
Cut and continue	The cutter will close and open thereafter the robot will move back to the approach position with fixed low speed.
Continue	The robot will move back to the approach position with fixed low speed.

More examples

```
MoveWireCutJ rtApp, rtCut, vMax, fine, tWeldGun \WObj:=fixture;
```

The TCP of the tool, `tWeldGun`, is moved linearly to the position, `rtApp`. This position is specified in the object coordinate system for fixture.

At `rtApp` the cutter opens and the TCP is moved along a linear path to the position `rtCut`. This position is also specified in the object coordinate system for fixture. At `rtCut` the closes and opens. After the cutting operation the TCP is moved back linearly to position `rtApp`.

Syntax

```
MoveWireCutL
  [ ApproachPoint ':=' ] < expression (IN) of robtarget > ','
  [ CutPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ Zone ':=' ] < expression (IN) of zonedata > ','
  [ Tool ':=' ] < persistent (PERS) of tooldata >
  [ '\' WObj ':=' < persistent (PERS) of wobjdata >
  [ '\' TLoad ':=' ] < persistent (PERS) of loaddata > ] ';'
```

Related information

	Described in:
`MoveMechCleanL`	*MoveMechCleanL - Move linear mechanical clean on page 17*

Continues on next page

3.5 MoveWireCutL - Move linear wire cut
Continued

	Described in:
`MoveMechCleanJ`	*MoveMechCleanJ - Move joint mechanical clean on page 21*
`MoveSprayL`	*MoveSprayL - Move linear spray on page 25*
`MoveSprayJ`	*MoveSprayJ - Move spray on page 29*
`MoveWireCutJ`	*MoveWireCutJ - Move wire cut on page 37*
`MoveL`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of `loaddata`	*Technical reference manual - RAPID Instructions, Functions and Data types*

3.6 MoveWireCutJ - Move wire cut

Usage

`MoveWireCutJ` is an instruction used to move the weld gun to a station for automatic cut-off of the welding wire. The movement to reach the cutting station does not have to be in a straight line.

Basic examples

```
MoveWireCutJ rtApp, rtCut, v200, fine, tWeldGun;
```

The tool center point (TCP) of the tool, `tWeldGun`, is moved to the position, `rtApp`, with speed data `v200` and zone data `fine`. At `rtApp` the TCP is moved along a linear path to the position `rtCut`. At `rtCut` the wire is cut off. After the cutting operation the TCP is moved back linearly to position `rtApp`.

Arguments

```
MoveMechCleanL ApproachPoint CleanPoint Speed Zone Tool [\WObj]
     [\TLoad]
```

`ApproachPoint`

Data type: `robtarget`

The approach point of the robot and external axes directly above the spray station. It is defined as a named position or stored directly in the instruction (marked with an * in the instruction).

`CutPoint`

Data type: `robtarget`

The destination point of the robot and external axes. It is defined as a 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 external axes.

`Zone`

Data type: `zonedata`

Zone data for the movement. Zone data describes the size of the generated corner path.

`Tool`

Data type: `tooldata`

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

`[\WObj]`

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 it is, the position is related to the world coordinate system. If, on the other hand, a stationary tool or coordinated external axes are used, this argument must be specified in order to perform a linear movement relative to the work object.

`[\TLoad]`

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. For a complete description of the `TLoad` argument, see `MoveL` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Program execution

Automatic or continuous mode

When executed, the robot makes a movement to the approach position, `ApproachPoint`. Then, the cutter opens and the robot makes a linear movement to the cutting position, `CutPoint` with fixed low speed. When the cutting operation is done, the robot will move back to the approach position with fixed low speed.

Note

If the system parameters *ShearWidth* and *ShearDirection* are used the cut position will be slightly different each time to minimize the shear wear.

Forward step mode

When executed, the robot makes a linear movement to the approach position, `ApproachPoint`.

Then, the robot makes a linear movement to the cutting position, `SprayPoint` with fixed low speed.

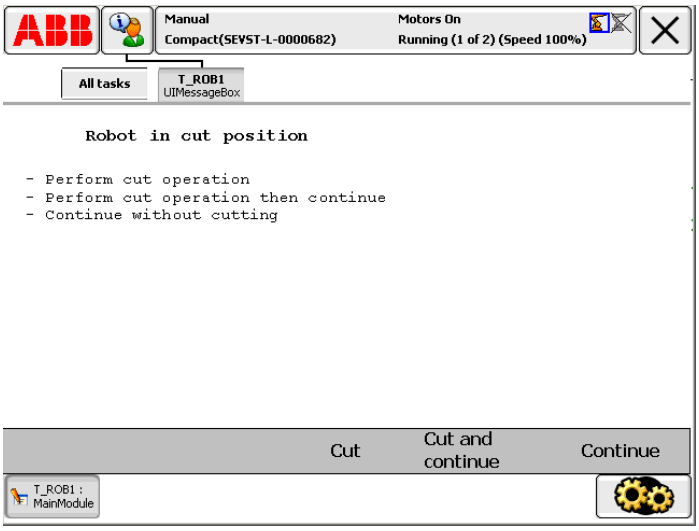
Note

If the system parameters *ShearWidth* and *ShearDirection* are used the cut position will be slightly different each time to minimize the shear wear.

Continues on next page

User dialog

At the cutting position a user dialog is shown.



xx1400001929

Cut	The cutter will close and open, thereafter the user dialog is shown again.
Cut and continue	The cutter will close and open thereafter the robot will move back to the approach position with fixed low speed.
Continue	The robot will move back to the approach position with fixed low speed.

More examples

```
MoveWireCutJ rtApp, rtCut, vMax, fine, tWeldGun \WObj:=fixture;
```

The TCP of the tool, `tWeldGun`, is moved to the position, `rtApp`. This position is specified in the object coordinate system for fixture.

At `rtApp` the cutter opens and the TCP is moved along a linear path to the position `rtCut`. This position is also specified in the object coordinate system for fixture. At `rtCut` the closes and opens. After the cutting operation the TCP is moved back linearly to position `rtApp`.

Syntax

```
MoveWireCutJ
  [ ApproachPoint ':=' ] < expression (IN) of robtarget > ','
  [ CutPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ Zone ':=' ] < expression (IN) of zonedata > ','
  [ Tool ':=' ] < persistent (PERS) of tooldata >
  [ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
  [ '\' TLoad ':=' ] < persistent (PERS) of loaddata > ] ';'
```

Related information

	Described in:
`MoveMechCleanL`	*MoveMechCleanL - Move linear mechanical clean on page 17*

Continues on next page

	Described in:
`MoveMechCleanJ`	*MoveMechCleanJ - Move joint mechanical clean on page 21*
`MoveSprayL`	*MoveSprayL - Move linear spray on page 25*
`MoveSprayJ`	*MoveSprayJ - Move spray on page 29*
`MoveWireCutL`	*MoveWireCutL - Move linear wire cut on page 33*
`MoveL`	*Technical reference manual - RAPID Instructions, Functions and Data types*
Definition of `loaddata`	*Technical reference manual - RAPID Instructions, Functions and Data types*

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.
No. 4528 Kangxin Hingway
PuDong District
SHANGHAI 201319, China
Telephone: +86 21 6105 6666

www.abb.com/robotics

3HAC050981-001, Rev A, en