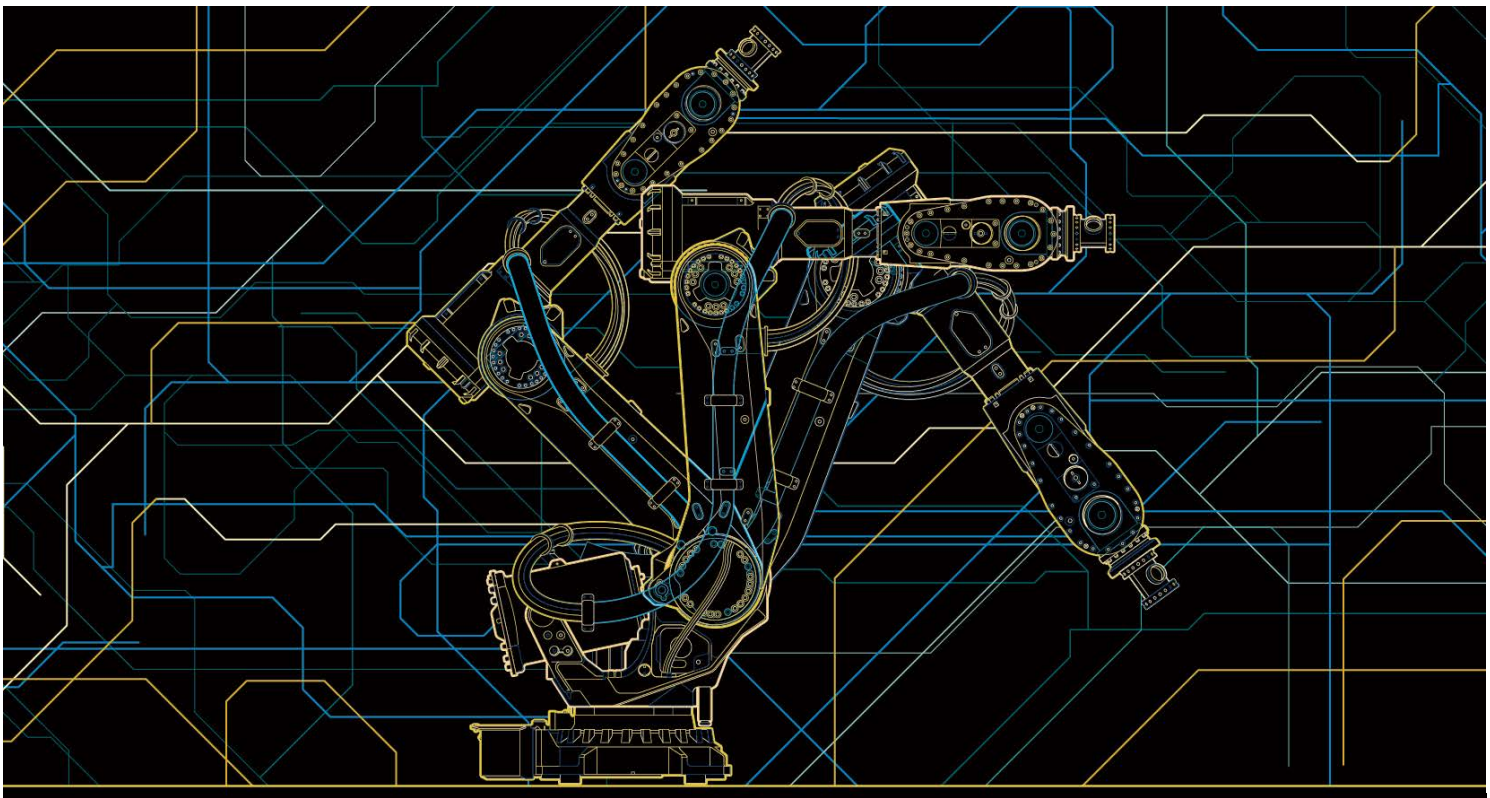


Application manual
Tool change support

Power and productivity
for a better world™

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

Application manual

Tool change support

RobotWare 6.0

Document ID: 3HAC050980-001

Revision: -

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 describes the RobotWare option *Tool change support*, which is included in the RobotWare option *Spot*.

Usage

This manual should be used when configuring tool changing applications based on the RobotWare option *Spot*.

Who should read this manual?

This manual is intended for:

- Commissioning personnel
- Service engineers
- Robot programmers
- Personnel responsible for installations and configurations of fieldbus hardware/software
- Personnel responsible for system configuration
- System integrators

Prerequisites

The reader should have the required knowledge of:

- IRC5 programming and usage
- System parameter configuration
- Mechanical installation work
- Electrical installation work
- System parameters and be used to editing these, either via RobotStudio or via cfg-files

References

References	Document ID
Application manual - Spot options	*3HAC050979-001*
Technical reference manual - System parameters	*3HAC050948-001*
Technical reference manual - RAPID Instructions, Functions and Data types	*3HAC050917-001*

Revisions

Revision	Description
-	Released with RobotWare 6.0.

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 - Introduction to RAPID*
- *Operating manual - IRC5 with FlexPendant*
- *Operating manual - RobotStudio*
- *Operating manual - Trouble shooting IRC5*, for the controller and manipulator.

Safety

Safety of personnel

When working inside the robot controller it is necessary to be aware of voltage-related risks.

A danger of high voltage is associated with the following parts:

- Devices inside the controller, for example I/O devices, can be supplied with power from an external source.
- The mains supply/mains switch.
- The power unit.
- The power supply unit for the computer system (230 VAC).
- The rectifier unit (400-480 VAC and 700 VDC). Capacitors!
- The drive unit (700 VDC).
- The service outlets (115/230 VAC).
- The power supply unit for tools, or special power supply units for the machining process.
- The external voltage connected to the controller remains live even when the robot is disconnected from the mains.
- Additional connections.

Therefore, it is important that all safety regulations are followed when doing mechanical and electrical installation work.

Safety regulations

Before beginning mechanical and/or electrical installations, ensure you are familiar with the safety regulations described in *Operating manual - General safety information*[1].

[1] This manual contains all safety instructions from the product manuals for the manipulators and the controllers.

1 Introduction

Tool change support

The *Tool Change Support* is a general and flexible example code package to create customized and easy to use tool change functionality for different types of tool change systems.

The *Tool Change Support* option provides a set of predefined tool change example instructions for safe and accurate tool changing, including built-in error handling of the used tool equipment.

The *Tool Change Support* option is general and can be extensively customized. It has a default “ready-to-use” function directly after installation, but is intended that some configuration data, RAPID data and RAPID routines be changed during the customization to fit the specific tool change equipment.

Note

The *Tool Change Support* option requires the RobotWare option *World zones*.

Tool change features

The *Tool Change Support* option package contains the following features:

- Support for up to four equipments on up to four tool stands in the default configuration.
- Support for up to 4 I/O units per tool.
- Support for pneumatic or servo tools.
- User-defined supervision and error recovery.
- Wide customizing possibilities.
- Default ready-to-use function directly after installation.

Note

Tool changing between servo tools requires the RobotWare option *Servo Tool Change*.

Programming principles

The robot movements and control of the tool changing equipment are embedded in the basic tool change instructions `TcPickupTool` and `TcDropOffTool`. These are used to pickup and drop off the selected tool and can be accessed from the FlexPendant.

Error handling is embedded in the instructions but can be handled on the user level if configured, see *TCDEFINE on page 39*.

The tool change process is specified by:

- `TcToolData`: tool changer configuration data
- `TcIOUnits`: tool changer I/O unit configuration data.
- The system module `TCDEFINE`: tool changer definitions, global data for customization purposes e.g. adaptations for a specific process equipment.

Continues on next page

Continued

- The system module TCBASE: tool changer base module, public shell routines, all tool changer interface routines are located here. Error handling is included in all the routines.
- The system module TCDEVICE: tool changer device module, RAPID routines specific to the configured tool changer equipment(s).
- The system module TCUSER: the tool stands path routines and tool stand positions and the .
- System parameters: the I/O signal configuration. See *I/O Configuration on page 45*.

Tool change instructions

Instructions	Used to:
TcPickupTool	Move the robot to the tool stand and pickup the tool specified in the instruction. The equipment will be checked before and after the pickup. A timeout can be specified by selecting an optional switch.
TcDropOffTool	Move the robot to the tool stand and drop off the tool specified in the instruction. The equipment will be checked before and after the drop off. A timeout can be specified by selecting an optional switch.
TcOpenCover	Open the tool stand cover specified in the instruction. A timeout can be specified by selecting a optional switch. The I/O supervision can be ignored by setting an optional switch
TcCloseCover	Close the tool stand cover specified in the instruction. A timeout can be specified by selecting a optional switch. The I/O supervision can be ignored by setting an optional switch
TcLockTool	Lock the tool to the toolchanger. A timeout can be specified by selecting an optional switch and the configured I/O units can be left disabled if a switch is selected.
TcUnlockTool	Unlock the tool from the toolchanger. A timeout can be specified by selecting an optional switch.

For more information see *TCBASE on page 35*.

Tool change functions

Functions	Used to:
TcChkToolOnRobot	Check that the tool is mounted on the robot and that it is not present in the tool stand.

For more information see *TCBASE on page 35*.

Tool change data

Data types	Used to define:
ToolInfo	The tool configuration data.

2 Programming

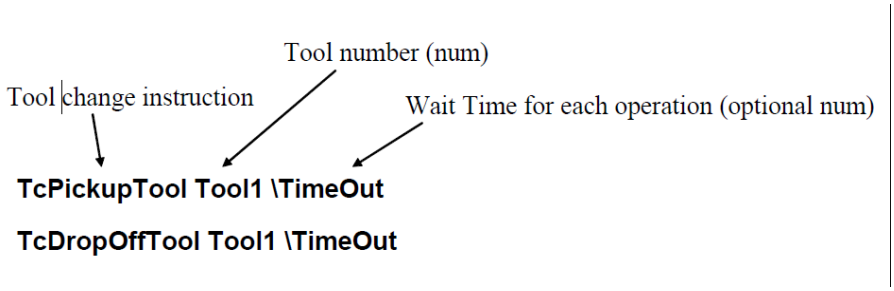
Overview

This chapter describes the basic functions steps to create, test, and run tool change instructions with *Tool Change Support*.

Note that the option *Tool Change Support* only provides the software functionality. Hardware, such as a tool changer, is not included.

The tool change instructions

`TcPickupTool` and `TcDropOffTool` are the basic tool change instructions in the Tool Change Support package. The instructions includes a movement to a predefined home position, a movement to a position above the tool at the toolstand, performing the desired activity and afterwards a movement back to the home position. These instructions are used to pickup or drop off a tool at a stand position.



xx1400001534

Defining tool change data

Before starting to program the instructions, the tool change data to be used should be defined.

- `ToolInfo`; setup data for all tool change equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` witch is located in the system module `TCDEFINE`. Change the default setup so it corresponds to the tool change equipment used. For more information, see *ToolInfo - Tool configuration data on page 32*.

Programming tool change instructions

	Action
1	Open the **Program Editor** and press the **Add Instruction** button.
2	Select the motion and process picklist, press **Next** if the instruction is not visible in the first window.
3	Select the instruction `TcPickupTool` or `TcDropOffTool`.

The instruction will be added directly to the program. The arguments are set in relation to the last programmed instruction.

- Change the arguments if necessary.

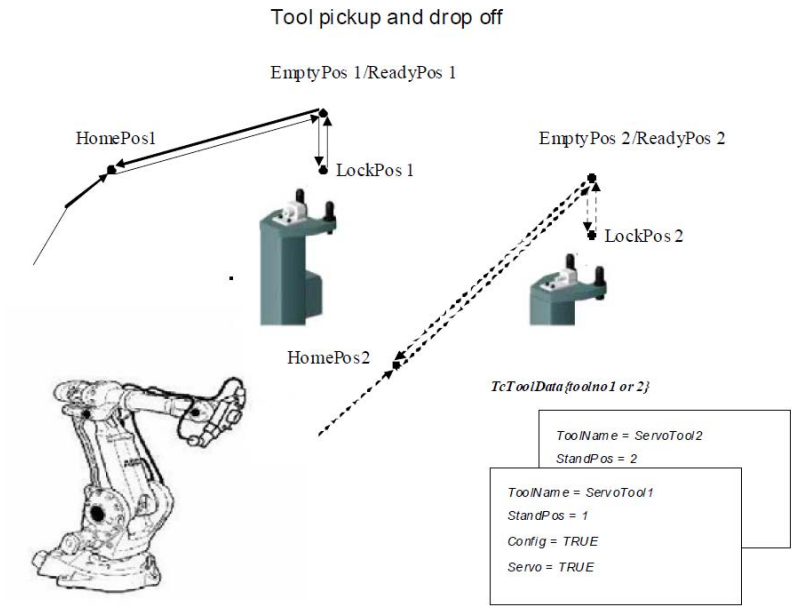
Other tool change instructions are programmed in a similar way.

Continues on next page

Continued

Programming example 1 tool changing between two tools

In this example `ServoTool1` (`toolno1`) will be picked up at stand position 1 and used for production, and when the production sequence is ready the tool will be dropped off. First the robot will move to the home position for tool 1 and then the robot will move to the tool stand position defined in the `TcToolData`. When the tool is picked up the robot will move back to the home position for tool 1 and from there start the production. After the production sequence is ready the tool will be dropped off at the same stand position where it was picked up. When the robot is ready with the first tool, it will move to the home position for tool 2 and then pickup that tool and use it for production and later drop the tool off at it's tool stand position.



xx1400001536

RAPID code sequence:

```
TcPickupTool toolno1;
DoWorkSequence1;
CheckEquipment;
TcDropOffTool toolno1;
TcPickupTool toolno2;
DoWorkSequence2;
CheckEquipment;
TcDropOffTool toolno2;
```

Editing tool change instructions

Change to another tool number.

	Action
1	Mark the current `toolno` in the instruction.
2	Call up the data by choosing **Edit** and then **Change Selected**.
3	Select another tool number, for example `toolno2`.
4	Press **OK**

Continues on next page

Manual actions

Some useful service routines are predefined to be used for manual actions during programming and test.

`ToolOpenCover`	Opens the tool cover for the selected tool.
`ToolCloseCover`	Close the tool cover for the selected tool.
`ToolLock`	Lock the selected tool to the toolchanger.
`ToolUnlock`	Unlock the selected tool from the toolchanger.
`ToolPickup`	Pickup the selected tool.
`ToolDropOff`	Drop off the selected tool.
`ToolModifyLockPos`	Used to modify the selected tool changer lock position. Jog the robot to the desired position and run this routine, the selected position will be stored.
`ToolModifyHomePos`	Used to modify the home positions. Jog the robot to the desired position and run this routine, the selected position will be stored.

If several tools are used then a dialog will appear asking for the tool number of the tool to be handled.

Setup the home and lock positions

	Action
1	Jog the robot to the desired home position and modify the position by running the service routine `ToolModifyHomePos` and follow the instructions.
2	Jog the robot to the desired tool stand position/lock position and modify the position by runnig the service routine `ToolModifyLockPos` and follow the instructions.

This page is intentionally left blank

3 Rapid references

3.1 Instructions

3.1.1 TcCheckToolOnRobot - Check that tool is mounted on robot

Usage

`TcCheckToolOnRobot` is used to check if the tool is connected to the robot and not present in the tool stand anymore.

Example

```
PROC MyProc()
  ! Lock the tool
  TcLockTool tool1;

  ! Do something

  ! Check if tool is connected to robot
  IF TcCheckToolOnRobot(tool1) THEN
    ! Do something
  ELSE
    ! Do something else
  ENDIF
ENDPROC
```

The tool changer is locked to the specified tool in the instruction, and later a check is done to verify that the tool is connected to the robot.

The parameter `tool1` is a num corresponding to the used tool equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

The tool stand position is a part of this data.

Arguments

```
TcUnlockTool ToolNo
```

`ToolNo`

Data type: `alias num toolno`

Used tool equipment number. Corresponding to the element number in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

Program execution

Internal sequence when the `TcChkToolOnRobot` function is executed:

- The group output `goTC_LockTool` is checked and group input `gi_TC_ToolCode` is checked for tool connection to tool changer.
- The tool stand I/O is checked for tool presence.
- If tool is connected the function will return TRUE.
- If tool is not connected the function will return FALSE.

Continues on next page

3.1.1 TcCheckToolOnRobot - Check that tool is mounted on robot
Continued

Error handling

No error handling is present.

Syntax

```
TcChkToolOnRobot
  [ ToolNo ':='] < expression (IN) of num > ';'
```

3.1.2 TcCloseCover - Close a tool stand cover

Usage

`TcCloseCover` is used to close a tool stand cover.

Example

```
PROC MyProc()
  ! Open the cover
  TcCloseCover tool1;
  ! Do something
  ! Close the cover
  TcOpenCover tool1;
ENDPROC
```

The tool stand cover is closed, and later opened.

The parameter `tool1` is a num corresponding to the used tool equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

The tool stand position is a part of this data.

Arguments

```
TcCloseCover ToolNo [\TimeOut] [\NoCheck]
```

`ToolNo`

Data type: `alias num toolno`

Used tool equipment number. Corresponding to the element number in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

`[\TimeOut]`

Data type: `num`

Wait time for each operation, if not specified a default timeout of 5 seconds will be used.

`[\NoCheck]`

Data type: `switch`

If this switch is used no supervision will be done on the tool stand I/O.

Program execution

Internal sequence when the `TcOpenCover` instruction is executed:

The tool stand cover is opened.

Error handling

The following error situations can occur:

- Instruction parameter supervision.
- The tool is not configured.
- Media signals are off.
- The tool stand cover can not be closed.

Continues on next page

3.1.2 TcCloseCover - Close a tool stand cover
Continued

Common handling for all errors

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

Limitations

The range for the number of tools that can be used and the tool stand positions are defined in the `ToolInfo` array `TcToolData`. See the system module *TCDEFINE.SYS*.

Syntax

```
TcCloseCover
  [ ToolNo ':='] < expression (IN) of num >
  [ '\'TimeOut ':=' < variable or persistent (IN) of num > ]
  [ '\'NoCheck ':=' < switch > ]';'
```

3.1.3 TcDropOffTool - Drop off a tool at a stand position

Usage

`TcDropOffTool` is used to drop off a tool at a predefined tool stand position.

Example

```
PROC MyProc()
  ! Drop off the tool
  TcDropOffTool tool1;

  ! Do something

  ! Pickup the tool
  TcPickupTool tool1;
ENDPROC
```

The tool is dropped off up at a specified tool stand position, and later picked up at the same position.

The parameter `tool1` is a num corresponding to the used tool equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

The tool stand position is a part of this data.

Arguments

```
TcLockTool ToolNo [\TimeOut] [\NoIO]
```

`ToolNo`

Data type: `alias num toolno`

Used tool equipment number. Corresponding to the element number in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

`[\TimeOut]`

Data type: `num`

Wait time for each operation, if not specified a default timeout of 5 seconds will be used.

Program execution

Internal sequence when the `TcDropOffTool` instruction is executed:

- The tool changer is checked for locked status.
- The tool position is checked, and the tool not present in stand is checked.
- The tool stand cover is opened.
- The robot moves to the tool changer home position.
- The robot moves to the tool changer ready position.
- The robot moves to the tool changer lock position.
- The tool changer is unlocked.
- The robot moves to the tool changer ready position.
- The tool unlocked is checked, and the tool present in stand is checked.
- The robot moves to the tool changer empty position.

Continues on next page

3.1.3 TcDropOffTool - Drop off a tool at a stand position
Continued

- The tool stand cover is closed.
- The robot moves to the tool changer home position.

Error handling

The following error situations can occur:

- Instruction parameter supervision.
- The tool is present in stand.
- The tool stand cover can not be opened.
- The tool changer can not be unlocked.
- The tool is still mounted on the robot, or is not present in stand
- The tool stand cover can not be closed.

Instruction parameter supervision

The error occurs when the `TcDropOffTool` is called with faulty parameters.

- The program stops with an error text.
- The error is logged.
- The parameter must be changed.
- When the program is restarted the current instruction is restarted from the beginning.

The tool is present in stand

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

The tool stand cover an not be opened

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

The tool charger cannot be unlocked

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

The tool is still mounted on the robot

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

Continues on next page

The tool stand cover cannot be closed

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

Syntax

```
TcDropOffTool
  [ ToolNo ':='] < expression (IN) of num > ','
  [ '\'TimeOut ':=' < variable or persistent (IN) of num > ]';'
```

3.1.4 TcLockTool - Lock the tool changer to the tool

Usage

`TcLockTool` is used to lock the tool changer to the specified tool at a predefined tool stand position.

Example

```
PROC MyProc()
  ! Lock the tool
  TcLockTool tool1;
  ! Do something
  ! Unlock the tool
  TcUnlockTool tool1;
ENDPROC
```

The tool changer is locked to the specified tool in the instruction, and later unlocked.

The parameter tool1 is a num corresponding to the used tool equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

The tool stand position is a part of this data.

Arguments

```
TcLockTool ToolNo [\TimeOut] [\NoIO]
```

`ToolNo`

Data type: `alias num toolno`

Used tool equipment number. Corresponding to the element number in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

`[\TimeOut]`

Data type: `num`

Wait time for each operation, if not specified a default timeout of 5 seconds will be used.

`[\NoIO]`

Data type: `switch`

If this switch is used the configured I/O units can be left disabled.

Program execution

Internal sequence when the `TcLcokTool` instruction is executed:

- The tool changer is locked to the tool.
- The tool I/O is dynamically enabled based on the tool configuration.
- The tool I/O is not enabled when the `\NoIO` switch is present.

Error handling

The following error situations can occur:

- Instruction parameter supervision.
- The tool is not configured.

Continues on next page

- Media signals are off.
- The tool changer can not be locked to the tool.

Common handling for all errors

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

Limitations

The range for the number of tools that can be used and the tool stand positions are defined in the `ToolInfo` array `TcToolData`. See the system module `TCDEFINE.SYS`.

Syntax

```
TcLockTool
  [ ToolNo ':='] < expression (IN) of num > ';'
  [ '\'TimeOut ':=' < variable or persistent (IN) of num > ]
  [ '\'NoCheck ':=' < switch > ] ';'
```

3.1.5 TcOpenCover - Open a tool stand cover

Usage

`TcOpenCover` is used to open a tool stand cover.

Example

```
PROC MyProc()
  ! Open the cover
  TcOpenCover tool1;

  ! Do something

  ! Close the cover
  TcCloseCover tool1;
ENDPROC
```

The tool stand cover is opened, and later closed.

The parameter `tool1` is a num corresponding to the used tool equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

The tool stand position is a part of this data.

Arguments

```
TcUnlockTool ToolNo
```

`ToolNo`

Data type: `alias num toolno`

Used tool equipment number. Corresponding to the element number in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

`[\TimeOut]`

Data type: `num`

Wait time for each operation, if not specified a default timeout of 5 seconds will be used.

`[\NoCheck]`

Data type: `switch`

If this switch is used no supervision will be done on the tool stand I/O.

Program execution

Internal sequence when the `TcOpenCover` instruction is executed:

- The tool stand cover is opened.

Error handling

The following error situations can occur:

- Instruction parameter supervision.
- The tool is not configured.
- Media signals are off.
- The tool stand cover can not be opened.

Continues on next page

Common handling for all errors

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

Limitations

The range for the number of tools that can be used and the tool stand positions are defined in the `ToolInfo` array `TcToolData`. See the system module *TCDEFINE.SYS*.

Syntax

```
TcChkToolOnRobot
  [ ToolNo ':='] < expression (IN) of num > ';'
  [ '\'TimeOut ':=' < variable or persistent (IN) of num > ]
  [ '\'NoCheck ':=' < switch > ]';'
```

3.1.6 TcPickupTool - Pickup a tool at a stand position

Usage

`TcPickupTool` is used to pickup a tool at a predefined tool stand position.

Example

```
PROC MyProc()
  ! Pickup the tool
  TcPickupTool tool1;

  ! Do something

  ! Drop off the tool
  TcDropOffTool tool1;
ENDPROC
```

The tool is picked up at a specified tool stand position, and later dropped off at the same position.

The parameter `tool1` is a num corresponding to the used tool equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

The tool stand position is a part of this data.

Arguments

```
TcPickupTool ToolNo [\TimeOut]
```

`ToolNo`

Data type: `alias num toolno`

Used tool equipment number. Corresponding to the element number in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

`[\TimeOut]`

Data type: `num`

Wait time for each operation, if not specified a default timeout of 5 seconds will be used.

Program execution

Internal sequence when the `TcPickupTool` instruction is executed:

- The tool changer is checked for unlocked status.
- The tool position is checked, and the tool present in stand is checked.
- The robot moves to the tool changer home position
- The robot moves to the tool changer empty position.
- The tool stand cover is opened.
- The robot moves to the tool changer lock position.
- The tool changer is locked.
- The robot moves to the tool changer ready position.
- The tool locked is checked, and the tool present on robot is checked.
- The robot moves to the tool changer home position.

Continues on next page

Error handling

The following error situations can occur:

- Instruction parameter supervision.
- The tool is not present in stand.
- The tool stand cover can not be opened.
- The tool changer can not be locked.
- The tool is not mounted on the robot, or is still present in stand

Instruction parameter supervision

The error occurs when the `TcPickupTool` is called with faulty parameters.

- The program stops with an error text.
- The error is logged.
- The parameter must be changed.
- When the program is restarted the current instruction is restarted from the beginning.

The tool is not present in stand

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

The tool stand cover cannot be opened

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

The tool charger cannot be locked

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

The tool is not mounted on the robot

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged.

Syntax

```
TcPickupTool
  [ ToolNo ':='] < expression (IN) of num > ';'
  [ '\'TimeOut ':=' < variable or persistent (IN) of num > ] ';'
```

3.1.7 TcUnlockTool - Unlocks the tool changer from the tool

Usage

`TcUnlockTool` is used to unlock the tool changer from the specified tool at a predefined tool stand position.

Example

```
PROC MyProc()
  ! Lock the tool
  TcLockTool tool1;

  ! Do something

  ! Unlock the tool
  TcUnlockTool tool1;
ENDPROC
```

The tool changer is unlocked from the specified tool in the instruction, and later locked.

The parameter `tool1` is a num corresponding to the used tool equipment. All equipment used are defined in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

The tool stand position is a part of this data.

Arguments

```
TcUnlockTool ToolNo
```

`ToolNo`

Data type: `alias num toolno`

Used tool equipment number. Corresponding to the element number in the `ToolInfo` array `TcToolData` in `TCDEFINE`.

`[\TimeOut]`

Data type: `num`

Wait time for each operation, if not specified a default timeout of 5 seconds will be used.

Program execution

Internal sequence when the `TcUnlockTool` instruction is executed:

- The tool I/O is dynamically disabled based on the tool configuration.
- The tool changer is unlocked from the tool.

Error handling

The following error situations can occur:

- Instruction parameter supervision.
- The tool is not configured.
- Media signals are off.
- The tool changer can not be unlocked from the tool.

Continues on next page

Common handling of errors

If an error occurs then:

- The signal `doErrActive` will be set. The program stops.
- An error message is displayed on the FlexPendant with retry possibilities.
- The error message is logged

Limitations

The range for the number of tools that can be used and the tool stand positions are defined in the `ToolInfo` array `TcToolData`. See the system module *TCDEFINE.SYS*.

Syntax

```
TcUnlockTool
  [ ToolNo ':='] < expression (IN) of num > ';'
  [ '\'TimeOut ':=' < variable or persistent (IN) of num > ] ';'
```

3.2 Data types

3.2.1 ToolInfo - Tool configuration data

Usage

`ToolInfo` is used to define tool change equipment specific data, to setup each tool equipment. Each `toolinfo` index defines one tool equipment.

Description

`ToolInfo` has the following default structure:

- `ToolName` - the tool name
- `StandPos` - the stand position.
- `Config` - should be true if the tool is used.
- `Servo` - should be true if the tool is a servo tool.
- `IOUnits` - the number of I/O units connected to the tool.

Components

`ToolName`

Data type: `string`

The name of the tool. This name must be identical with the name of the mechanical unit defined in the motion servo gun parameters if the tool is a servo tool.

`StandPos`

Data type: `num`

The tool stand position number used for the tool.

`Config`

Data type: `bool`

Defines if the tool is configurated or not. TRUE = tool is configured, FALSE = tool is not configured.

`Servo`

Data type: `bool`

Defines if the tool is a servo tool or not. TRUE = servo tool, FALSE = pneumatic tool. If the tool is configured as an servo tool it will be activated after it has been locked to the tool changer, and deactivated after it has been unlocked from the tool changer.

`IOUnits`

Data type: `num`

Defines how many I/O units that are connected to each tool. If no I/O units are used, set `IOUnits` to 0. Max number of configurable I/O units per tool is 4 by default, but if more are needed increase the `TcIOUnits` array in `TCDEFINE` accordingly.

Continues on next page

Default structure

```
<dataobject of ToolInfo>
  <ToolName of string>
  <StandPos of num>
  <Config of bool>
  <Servo of bool>
  <IOUnits of num>
```

Predefined data

```
CONST ToolInfo TcToolData{4} :=
  [[“Tool1Name”, TC_STANDPOS1, FALSE, FALSE, 0],
  [“Tool2Name”, TC_STANDPOS2, FALSE, FALSE, 0],
  [“Tool3Name”, TC_STANDPOS3, FALSE, FALSE, 0],
  [“Tool4Name”, TC_STANDPOS4, FALSE, FALSE, 0]];
```

`TcToolData` is an array with `ToolInfo` parameters for each used tool. These parameters must be changed by the user during the installation and programming phase to be in agreement with the equipment in use. In the default package, `TcToolData` is defined in the module `TCDEFINE`.

Customizing

The *Tool Change Support* package provides opportunities for the user to customize the functionality to adapt to different types of tool change equipment and user defined standards. For this data type it is possible to delete components if they are not used. It is also possible to give the components own user defined names. However, the main subject of this description is the default setup.

For more information, see *Programming on page 13*.

This page is intentionally left blank

4 System modules

4.1 TCBASE

Description

TCBASE is running in the motion task and contains the public tool changing RAPID instructions and funtions.

Normally there is no need to make any modification in this module, but it can be done if needed. This chapter describes the default functionality.

Tool change procedures

The following predefined tool change procedures are installed with the application. These procedures have a default functionality but can easily be changed.

Error handling is encapsulated in all tool changing procedures.

```
PROC TcPickupTool(toolno ToolNum, \num TimeOut)
```
This procedure is used to pickup a tool at it's specified tool stand position.

```
PROC TcDropOffTool(toolno ToolNum, \num TimeOut)
```
This procedure is used to drop off a tool at it's specified tool stand position.

```
PROC TcOpenCover(toolno ToolNum, \num TimeOut \switch NoCheck)
```
This procedure is used to open a tools tool stand cover.

```
PROC TcCloseCover(toolno ToolNum, \num TimeOut \switch NoCheck)
```
This procedure is used to close a tools tool stand cover.

```
PROC TcLockTool(toolno ToolNum, \num TimeOut \switch NoIO)
```
This procedure is used to lock a tool to the robot.

```
PROC TcUnlockTool(toolno ToolNum, \num TimeOut \switch NoIO)
```
This procedure is used to unlock a tool to the robot.

```
FUNC bool TcChkToolOnRobot(toolno ToolNum)
```
This procedure can be used to check that the tool is mounted on robot.

TRUE = Tool is mounted on the robot.

FALSE = Tool is present in stand.

```
PROC TcRetryCycle(toolno ToolNum, num ErrAction)
```
This procedure can be used to perform optional actions if Retry is pressed on the FlexPendant or if diRetryProc is pulsed.

No default functionality.

```
PROC TcAbortCycle(toolno ToolNum, num ErrAction)
```
This procedure can be used to perform optional actions if Abort is pressed on the FlexPendant or if diAbortProc is pulsed.

No default functionality.

4.2 TCDEVICE

Description

TCDEVICE runs in the motion task and contains the procedures and functions for the specific tool changer system and the tool stands.

This module is intended to be customized to fit the desired tool change equipment.

This chapter describes the default functionality.

Data definitions

The following local data are predefined.

Name	Declaration	Description
di_tool_present1	LOCAL VAR signaldi	User signal (DI) to check if tool is present in stand, sensor1
di_tool_present1	LOCAL VAR signaldi	User signal (DI) to check if tool is present in stand, sensor2
di_cover_opened	LOCAL VAR signaldi	User signal (DI) to check if cover is opened.
di_cover_closed	LOCAL VAR signaldi	User signal (DI) to check if cover is closed.
do_cover_close	LOCAL VAR signaldo	User signal (DO) to close the cover.
do_cover_open	LOCAL VAR signaldo	User signal (DO) to open the cover.
do_tc_unlock	LOCAL VAR signaldo	User signal (DO) to unlock the tool.
do_tc_lock	LOCAL VAR signaldo	User signal (DO) to lock the tool.
di_tc_locked	LOCAL VAR signaldi	User signal (DI) to check if tool is locked.
di_tc_unlocked	LOCAL VAR signaldi	User signal (DI) to check if tool is unlocked.
go_tc_lock_tool	LOCAL VAR signalgo	User signal (GO) to check what tool is mounted on the robot.
gi_tc_tool_code	LOCAL VAR signalgi	User signal (GI) to check what tool is mounted on the robot.
di_io_power_on	LOCAL VAR signaldi	User signal (DI) to check if I/O unit is on.
di_air_on	LOCAL VAR signaldi	User signal (DI) to check if air is on.
do_err_active	LOCAL VAR signaldo	User signal (DO) to check if a error is active.

I/O definition routines

The signal names used in this module are connected to the physical used signals. As default, signals for four tool stands and one tool changer are defined.

All procedures in this module have a default functionality but can easily be changed to fit a specific tool changer system.

```
LOCAL PROC InitCommonIO()
```

This procedure is called in the beginning of each tool change instruction.

Default functionality: Init commonly used I/O signals for all equipment

```
LOCAL PROC InitTcStandIO(num StandPos)
```

This procedure is called in the beginning of each tool change instruction

Default functionality: Init tool stand I/O signals.

Continues on next page

```
LOCAL PROC InitTcIO(toolno ToolNum)
```

This procedure is called in the beginning of each tool change instruction

Default functionality: Init tool changer I/O signals.

Tool change procedures

These procedures and functions are used by the tool change instructions and are called from the TCBASE module during the process.

These procedures have a default functionality but can easily be changed. The procedures cannot be deleted since they are called from other modules. Any errors that occur in these procedures will be returned to TCBASE and handled there.

```
LOCAL PROC TcOpenCover(INOUT num ErrStatus, toolno ToolNum, \num TimeOut \switch
NoCheck)
```

This procedure is called from the TcOpenCover procedure in the TCBASE module.

Default functionality: Open the specified tool stand cover.

```
LOCAL PROC TcCloseCover(INOUT num ErrStatus, toolno ToolNum, \num TimeOut \switch
NoCheck)
```

This procedure is called from the TcCloseCover procedure in the TCBASE module.

Default functionality: Close the specified tool stand cover.

```
LOCAL PROC TcLockTool(INOUT num ErrStatus, toolno ToolNum, \num TimeOut \switch
NoIO)
```

This procedure is called from the TcLockTool procedure in the TCBASE module.

Default functionality: Lock the specified tool to the robot.

```
LOCAL PROC TcUnlockTool(INOUT num ErrStatus, toolno ToolNum, \num TimeOut)
```

This procedure is called from the TcUnlockTool procedure in the TCBASE module.

Default functionality: Unlock the specified tool from the robot.

```
LOCAL PROC TcChkNoToolAtStand(INOUT num ErrStatus, toolno ToolNum)
```

This procedure is called from the TcPickupTool and TcDropOffTool procedures in the TCBASE module.

Default functionality: Check that the tool is not present in the stand and that the tool position is right.

```
LOCAL PROC TcChkToolAtStand(INOUT num ErrStatus, toolno ToolNum)
```

This procedure is called from the TcPickupTool and TcDropOffTool procedures in the TCBASE module.

Default functionality: Check that the tool is present in the stand and that the tool position is right.

```
LOCAL PROC TcChkToolOnRobot(INOUT num ErrStatus, toolno ToolNum)
```

This procedure is called from the function TcChkToolOnRobot in the TCBASE module.

Default functionality: Returns TRUE if the tool is mounted on robot and is not present in the stand, returns FALSE if tool is present in stand.

Continues on next page

`LOCAL FUNC num TcControlEquip(toolno ToolNum, \num TimeOut \switch Disable)`

This function is used locally in this module and is called when the tool is locked and unlocked.

Default functionality: Enable or disable I/O units.

If the switch `\Disable` is present all configured I/O units for that tool will be disabled.

If the switch `\TimeOut` is not present a default timeout will be used, 5 seconds.

`LOCAL FUNC num CheckIOAndMedia(string Caller, toolno ToolNum)`

This procedure is used locally in this module and is called in the beginning of all tool change procedures.

Default functionality: Check if I/O power and air media are ok.

4.3 TCDEFINE

Description

This module is intended for the person who creates and sets up the tool changing system. TCDEFINE is run in the motion task and contains global process and configuration data used in the tool change support option.

This chapter describes the default setup.

Data definitions

The names are predefined and used internally when tool change instructions are used. Therefore, they must not be deleted.

The following global data are predefined:

Name	Declaration	Description
ToolInfo	RECORD of type CONST	Tool configuration data: This record contains all the setup data for the tools. string ToolName, the name of the tool. num StandPos, the stand position of the tool. bool Config, set to TRUE if this tool should be used. bool Servo, set to TRUE if the tool is a servo tool. num IOUnits, number of I/O units connected to the tool.
ErrorInfo	RECORD of type PERS	Error information structure: This record contains the error information, and is updated when an error occurs in the tool change routines in the TCDEVICE module. string device_mod, the name of the device module. string routine, the name of the failing routine. string error_type, the type of error. string description, the reason for the error. string error_info, additional error information. string action, error recovery information.
TC_MAX_TOOLS	CONST num	Max number of configurable tools, default value 4.
TC_MAX_IOUNIT	CONST num	Max number of configurable I/O units per tool, default value 4.
TcToolData{4}	CONST ToolInfo	Tool configuration data array for all tools.
TcIOUnits{4, 4}	CONST string	I/O unit configuration data array for all tools. If the toolchanger should disconnect I/O units the corresponding I/O unit names should be added in this data.

Continues on next page

Name	Declaration	Description
TcErrInfo{4}	PERS ErrorInfo	Error information structure, contains error information, and is used in error handling from the TCDEVICE module(s)
DEVICE_MOD_NAME{4}	CONST string	The name of the device module if different toolchanger systems are used, eg. in a multimove system. E.g. TC_DEVICE_XXX, TCDEVICE_YYY etc. Default name: "TCDEVICE"
TcUserRecover	PERS bool	User error recovery, TRUE - All errors are raised to user level from the tool change routines in TCBASE, no internal error handling and elog message is done. FALSE - Error handling is done inside the tool change routines in the TCBASE module.
toolno1 to 4	PERS toolno	Alias num for the tool used in the tool change routines. Default values: 1 to 4

Global procedures

The following predefined function is installed with the application. It is used to retrieve error texts from the *tctext.xml* file that is installed with the application.

```
FUNC string TcTextGet(num Index)
```

This function returns a text string from the *tctext.xml* file based on the index number.

Example:

```
TcErrInfo{ToolNum}.action := TcTextGet(25);
```

4.4 TCUSER

Description

This module is intended for the person who creates and tests the user program.

TCUSER is run in the motion task and contains data and routines that should be customized regarding the tool changer positions and the tool data that are used.

It contains the different path routines (for example GoToTcEmptyPos, GoToTcLockPos, and GoToTcReadyPos). This chapter describes the default functionality.

Data

The names are predefined and used internally when tool change instructions are used. They must therefore not be deleted.

The following global data are predefined:

Name	Declaration	Description
TcNoTool0	PERS tooldata	Tool data used with no tool on robot.
TcTool{4}	PERS tooldata	Tool data used with all configured tools.
TcLockPos{4}	PERS robtarget	Tool stand positions, lock positions for all tools. The positions can be modified by running the service routine ToolModifyLockPos.
TcLockPosOffset	PERS num	Offset from the tool stand lock position, default 50mm.
TcMoveSpeed	PERS speddata	The robot speed while tool changing.
TcHomePos{4}	PERS jointarget	Tool changing home position, home positions for all tools. The positions can be modified be running the service routine ToolModifyHomePos.
TcCurrentTool	PERS tooldata	The current tool when moving to home position. The current tool will be stored in this variable when running the routine GoToTcHomePos.

Tool changing user procedures

The following predefined procedures are installed with the application. They are used by the toolchange instructions TcPickupTool and TcDropOffTool during the process.

The parameter ToolNum is the tool equipment number.

```
PROC GoToTcEmptyPos(toolno ToolNum)
```

This procedure moves the robot to a configurable position, for example 50 mm, above the lock position suitable for checking that the tool is present in the tool stand and not mounted in the robot.

The robot is empty.

Continues on next page

`PROC GoToTcLockPos(toolno ToolNum)`

This procedure moves the robot to the tool stand lock position specified by `ToolNum`. By default this position is a dummy position, this `robtarget` has to be modified. The positions in `TcLockPos` can be modified by running the service routine `ToolModifyLockPos`.

The robot starts empty, and ends docked to the tool.

`PROC GoToTcReadyPos(toolno ToolNum)`

This procedure moves the robot to a configurable position, for example 50 mm, above the lock position suitable for checking that the tool is mounted in the robot and not present in the tool stand.

The tool is mounted in the robot.

`PROC GoToTcHomePos(toolno ToolNum)`

This procedure moves the robot to the tool change home position, start and end position. The tool can be mounted in the robot or not, the current `tooldata` is stored in the persistant `tooldata TcCurrentTool`.

By default this position is a dummy position, this `jointtarget` has to be modified. The positions in `TcHomePos` can be modified by running the service routine `ToolModifyHomePos`.

4.5 TCSERVICE

Description

TCSERVICE is run in the motion task and contains service routines for manual control of the tool change equipment. This chapter describes the default functionality.

Tool change service routines

The following predefined service routines are installed with the application.

These service routines have a default functionality but can easily be changed.

Interaction with the operator will be performed when running these service routines, the operator will be promted to select tool or tool stand.

`PROC ToolOpenCover()`

This service routine is used to open the specified tool stand cover.

`PROC ToolCloseCover()`

This service routine is used to close the specified tool stand cover.

`PROC ToolLock()`

This service routine is used to lock a tool on the robot.

`PROC ToolUnlock()`

This service routine is used to unlock a tool from the robot.

It is not possible to unlock the tool unless the sensors on the toolstand indicate a present tool.

`PROC ToolPickup()`

This service routine is used to pickup a tool at a specified tool stand position.

`PROC ToolDropOff()`

This service routine is used to drop off a tool at a specified tool stand position.

`PROC ToolModifyLockPos()`

This service routine is used to modify the tool changer lock positions.

Jog the robot to the tool stand lock position and then run this service routine. The specified lock position in the TcLockPos array will be updated and the TCUSER module will be saved.

`PROC ToolModifyHomePos()`

This service routine is used to modify the home positions.

Jog the robot to the desired tool change home position and then run this service routine. The specified position in the TcHomePos array will be updated and the TCUSER module will be saved.

This page is intentionally left blank

5 I/O Configuration

Description

The digital signals used for Tool Change Support are configured in the system parameters.

The Tool Change Support package can be configured for different equipment setup.

This chapter describes the basic (default) setup for one tool changer and four tools in four tool stands but it is possible to configure up to as many as needed.

The physical connections can be changed freely. To save physical signals, signals not in usecan be connected to a virtual board (type virtual).

Basic setup - board description

There are six predefined boards:

- One virtual board, named `TCChanger`, with basic setup signals for the tool changer. Changing this board to a physical digital I/O board is enough for running the default version of the *Tool Change Support* option.
- One virtual board, named `TCStand1` with basic setup signals for tool stand 1. This board should be changed to a physical digital I/O board used if tool stand 1 is used.
- One virtual board, named `TCStand2` with basic setup signals for tool stand 2. This board should be changed to a physical digital I/O board used if tool stand 2 is used.
- One virtual board, named `TCStand3` with basic setup signals for tool stand 3. This board should be changed to a physical digital I/O board used if tool stand 3 is used.
- One virtual board, named `TCStand4` with basic setup signals for tool stand 4. This board should be changed to a physical digital I/O board used if tool stand 4 is used.
- One virtual board, named `TC_SimBoard` with some internal or normally not connected signals. Normally these signals will remain on a virtual board.

Basic setup - signal description

Tool Stand 1 signals

Name	Type	Information
`diCoverOpnd_Stnd1`	input	The signal is set when the cover is opened on tool stand 1.
`diCoverClsd_Stnd1`	input	The signal is set when the cover is closed on tool stand 1.
`diTool1_Prsnt1`	input	The signal is set when a tool is present in tool stand 1, sensor 1.
`diTool1_Prsnt2`	input	The signal is set when a tool is present in tool stand 1, sensor 2.
`doClsCover_Stnd1`	output	Signal used to close the cover on tool stand 1.
`doOpnCover_Stnd1`	output	Signal used to open the cover on tool stand 1.

Continues on next page

Continued

Name	Type	Information
doTool1_Prsnt	output	Cross connection result signal. Is set when sensor 1 and 2 is set on tool stand 1. This signal can be used to check if a tool is present in tool stand position 1.

Tip

Signal names for tool stand 2 are the same as for tool stand 1 but with the number 2 instead.

Tool Changer signals

Name	Type	Information
diIOPowerOn	input	Signal used to check if I/O unit is powered on.
diAirOn	input	Signal used to check if there is air pressure in the system.
diTC_Locked	input	Signal used to check if the tool changer is locked.
diTC_Unlocked	input	Signal used to check if the tool changer is unlocked.
doTC_UnLock	output	Signal used to unlock the tool changer.
doTC_Lock	output	Signal used to lock the tool changer to the tool.
doTC_Locked	output	Cross connection result signal. Is set when the diTC_Lock is set and when the diTC_Unlock is reset, i.e when the tool changer is locked.
giTC_ToolCode (group)	input	Tool changer tool code, should be keyed for the connected tool, i.e when tool 1 is connected this signal should be set to 1.
goTC_LockTool (group)	output	This signal group will be set to the tool number when the tool is locked to the tool changer.

Other signals

Name	Type	Information
doTC_LockPos	output	The signal is set when the robot is within the defined zone for tool changing.
doErrActive	output	The signal is set when an error situation occur.
diRetryProc	input	Can be used to answer a weld error dialog with an input signal. The same as pressing RETRY.
diAbortProc	input	Can be used to answer an error dialog with an input signal. The same as pressing ABORT.

6 Customizing

6.1 Introduction

Description

The *Tool Change Support* package provides wide opportunities to customize and adapt the functionality to different types of tool changing equipment and user standards. Another purpose of this customizing process is to reduce the amount of data and number of variables presented to the programmer or operator.

Customizing possibilities

Following customizing guides can be found in this chapter:

- How to define the number of tools to be used. Only used tools are visible and programmable on the FlexPendant.
- How to add new data components in the tool change data types to the tool changing equipment in use.
- How to adapt the functionality in the device routines which are called from the kernel during the tool change sequence.
- How to change the predefined service routines for manual actions (`ToolOpenCover`, `ToolCloseCover` etc).
- How to add equipment specific supervision and error handling.
- How to define the used I/O signals. It is possible to have user defined names for the used signals.
- How to use different types of tool changer systems.

6.2 Files to be changed during the customizing

Introduction

The customizing process is done by changing a number of predefined data and routines, preferably using a standard PC with RobotStudio or using a offline text editor. The following RAPID modules and configuration files are intended to be changed during the customizing process:

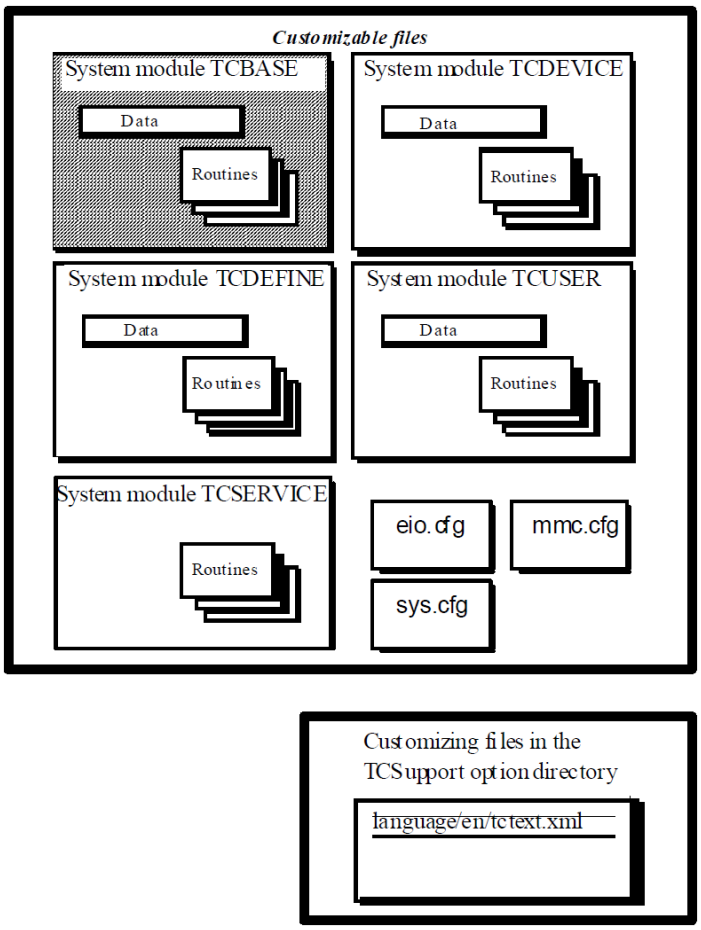


Figure 6.1: Files possible to customize

TCDEFINE(Nostepin)

This module is intended for the person who creates and sets up the tool changing system. `TCDEFINE` is run in the motion task and contains global process and configuration data used in the tool change support option. For more information, see *TCDEFINE on page 39*.

Continues on next page

TCBASE(Noview)

TCBASE is run in the motion task and contains the public tool changing RAPID instructions and functions. For more information, see *TCBASE on page 35*.

> **Note**
>
> Normally there is no need to make any modification in this module, but it can be done if needed.

TCDEVICE(Nostepin)

TCDEVICE is run in the motion task and contains the routines and functions for the specific tool changer system and the tool stands.

This module is intended to be customized to fit the desired tool change equipment. For more information, see *TCDEVICE on page 36*.

TCUSER

This module is intended for the person who creates and tests the user program.

TCUSER is run in the motion task and contains data and routines that should be customized regarding the tool changer positions and the tool datas that are used.

It contains the different path routines (for example GoToTcEmptyPos, GoToTcLockPos, and GoToTcReadyPos). For more information, see *TCUSER on page 41*.

TCSERVICE(Noview)

TCSERVICE is run in the motion task and contains service routines for manual control of the tool change equipment. For more information, see *TCSERVICE on page 43*.

> **Note**
>
> Normally there is no need to make any modification in this module, but it can be done if needed.

I/O configuration (eio.cfg)

Default tool change signals for four gun equipment are defined, and by default all signals are connected to virtual boards. The signals used must be connected to physical signals. For more information, see *I/O Configuration on page 45*

MMC configuration (mmc.cfg)

This configuration file contains information about which instructions are included in the different instruction pick lists and which routines are added to the Service menu in the programming window, to be used as manual actions.

SYS configuration (sys.cfg)

This configuration file contains information about which modules that are loaded in which tasks.

6.3 Customizing guides

How to define the number of tools to be used

As default it is possible to use up to four tools in four tool stands and is possible to use and set up data for more equipments if needed.

	Action	Note
1	Change the data TC_MAX_TOOLS in TCDEFINE to desired value.	CONST num TC_MAX_TOOLS:= 4;
2	Add signals in eio.cfg for all equipments to be used in a similar manner as the predefined signals for tool equipment 1 to 4.	
3	Increase the number of data in the arrays in TCDEFINE (The size shall be in agreement with the used number of tools):	• TcToolData • TcIOUnits • TcErrInfo
4	Increase the routines InitTcStandIO and InitTcIO in TCDEVICE with AliasIO instructions for the signals corresponding to the extra equipments.	

How to add new data components in the tool change data types

	Action	Note
1	Change the definition of the ToolInfo data type in TCDEFINE to desired.	It is possible to add more data components.
2	Change the structure and the default values of following arrays in TCDEFINE (if the corresponding data type is changed):	• TcToolData • TcIOUnits • TcErrInfo

Note

The default data component names are used in the code and can not be changed or moved unless the the code is also changed.

How to adapt the functionality in the device routines

	Action	Note
1	Add code to the tool change routines in TCDEVICE.	For more information, see *TCDEVICE on page 36*.

How to change the predefined service routines for manual actions

	Action
1	Change the routines for the manual actions (ToolOpenCover, ToolCloseCover, ToolLock etc.). The routines are found in module TCSERVICE.
2	If manual actions are added or deleted, or if manual action names are changed then the MMC configuration must be changed for the routine to be visible in the view service routines menu. Change the names under MMC_SERV_ROUT_STRUCT:

Continues on next page

How to add equipment specific supervision and error handling

	Action
1	If the supervision during the tool change sequence is changed, add or change the code in the tool change routines in `TCDEVICE`. See *TCDEVICE on page 36*.

How to define the used IO signals

	Action	Note
1	This routine call in `TCBASE` is used to define the signals used for error handling in the tool change routines.	• `TcInitIO`
2	The following routine calls in `TCDEVICE` are used to define the signals used for all equipments.	• `InitCommonIO` • `InitTcStandIO` • `InitTcIO`
3	Change the parameters in these instructions so they are in agreement with the used signal names in the *EIO.cfg* file.	

How to use different types of tool changer systems

As default it is possible to use up to four tools in four tool stands and one tool changer, and is possible to use and set up data for more equipments if needed, e.g for a MultiMove system.

	Action	Note
1	Change the data `TC_MAX_TOOLS` in `TCDEFINE` to desired value. Change the name of the Device module if different toolchanger systems are used in eg. a MultiMove system. E.g. `TC_DEVICE_XXX`, `TC_DEVICE_YYY` etc.	`CONST string DEVICE_MOD_NAME{4} := ["TCDEVICE", "TCDEVICE", "TCDEVICE", "TCDEVICE"];`
2	Add signals in *eio.cfg* for the new tool changers to be used in a similar manner as the predefined signals for tool changer 1.	
3	Increase the routines `InitTcIO` in `TCDEVICE` with `AliasIO` instructions for the signals corresponding to the extra tool changer.	

This page is intentionally left blank

Index

Contact us

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

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

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

www.abb.com/robotics

3HAC050980-001, Rev -, en