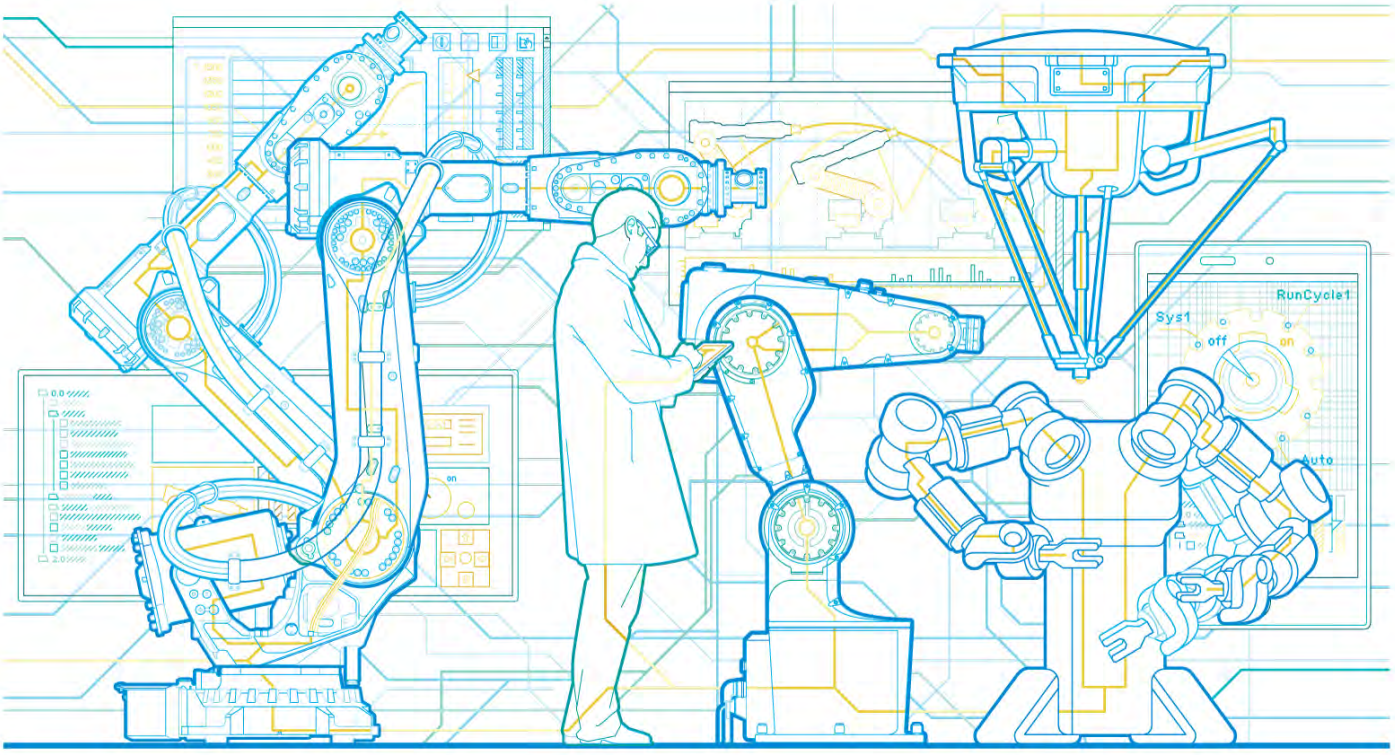



# Application manual
# Spot options

Trace back information:
Workspace R17-1 version a7
Checked in 2017-03-22
Skribenta version 5.1.011

# Application manual

# Spot options

RobotWare 6.05

Document ID: 3HAC050979-001

Revision: E

ABB AB, Robotics
Robotics and Motion
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 *Spot* and it's configurations *Spot Pneumatic*, *Spot Servo*, and *Spot Servo Equalizing*, and contains instructions for use of the software and configuration.

## Usage

This manual should be used during installation and configuration of 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 |
|---|---|
| *Operating manual - IRC5 with FlexPendant* | *3HAC050941-001* |
| *Operating manual - RobotStudio* | *3HAC032104-001* |
| *Product manual - IRC5* | *3HAC021313-001* |
| *Product specification - Controller IRC5 with FlexPendant* | *3HAC041344-001* |
| *Technical reference manual - System parameters* | *3HAC050948-001* |
| *Technical reference manual - RAPID Instructions, Functions and Data types* | *3HAC050917-001* |
| *Technical reference manual - RAPID overview* | *3HAC050947-001* |
| *Application manual - Additional axes and stand alone controller* | *3HAC051016-001* |
| *Application manual - Servo gun tuning* | *3HAC026820-001* |

*Continues on next page*

*Continued*

| References | Document ID |
|---|---|
| *Application manual - SoftMove* | *3HAC050977-001* |
| *Application manual - RobotWare Add-Ins* | *3HAC051193-001* |
| *Application manual - DeviceNet Master/Slave* | *3HAC050992-001* |
| *Application manual - PROFINET Controller/Device* | *3HAC050969-001* |

## Revisions

| Revision | Description |
|---|---|
| - | Released with RobotWare 6.0. |
| A | Released with RobotWare 6.01. |
| A | Released with RobotWare 6.01.<br>• Updated the path to the template files, see *782-11 Bosch PROFINET MFDC on page 58*. |
| B | Released with RobotWare 6.02.<br>• Possibility to configure supervision task SW_SUP, *Supervision task SW_SUP on page 94*.<br>• Added possibility to use sensor search in MeasureWearL, see *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118*.<br>• Added possibility to perform a test weld in the CalibL and CalibJ instructions, see *CalibL/CalibJ - Calibrate a servo gun during robot movement on page 108*.<br>• Added possibility to use dual forces in the SetForce instruction, see *SetForce - Close and Open a gun with desired force and time on page 104*<br>• Added support for Spot in a MultiProcess configuration on 2 robots. |
| C | Released with RobotWare 6.03.<br>• Changed template I/O naming for MultiGun and MultiMove configurations, see *Spot I/O configuration on page 41*. |
| D | Released with RobotWare 6.04.<br>• Changed template gun names in the SWUSER modules, see *gundata - Equipment specific weld data on page 135*.<br>• Improved information on how to define TCP, see *How to define the TCP on page 171*.<br>• Improved handling of independent gun mode in SetForce, see *SetForce - Close and Open a gun with desired force and time on page 104*.<br>• Improved gun force gravity compensation, see *Servo gun force gravity compensation on page 204*<br>• General improvements. |
| E | Released with RobotWare 6.05.<br>• Added possibility to handle tip change and tip wear supervision errors in the `MeasureWearL` and `ReCalcTCP` instructions in the user defined error handling.<br>• Added supervision of force calibration status of used gun, and also a possibility to create a default force table in the `ManualForceCalib` and `ManualServiceCalib` service routines. |

# 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*

*Continued*

### Application manuals

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

An application manual generally contains information about:

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

### Operating manuals

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

The group of manuals includes (among others):

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

# Safety

## Safety of personnel

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.

This page is intentionally left blank

# 1 Introduction to RobotWare Spot

## 1.1 Spot option and features

### The Spot option

The Spot option is a general and flexible software platform for creation of customized and easy to use function packages for different types of spot welding systems and process equipment.

There are three different main **Spot** configurations supporting spot welding, two for servo guns and one for pneumatic guns.

- The **Spot Pneu** configuration provides support for sequential welding with one gun equipment. For this configuration it is also possible to select a multiple gun configuration when configuring the system, with support for welding with four pneumatic guns at the same time.
- The **Spot Servo** configuration provides support for sequential welding with one gun equipment. For this configuration it is also possible to select a multiple gun configuration when configuring the system, with support for welding with four servo guns at the same time.
- The **Spot Servo Equalizing** configuration has the same functionality as the basic **Spot Servo** configuration, but can also be used for guns without mechanical equalizing systems.

All Spot configurations provides dedicated spot welding instructions for fast and accurate positioning combined with gun manipulation, process start and supervision of the different gun equipment.

Communication with the external welding equipment is done with digital signals.

The Spot option is general and can be extensively customized. The three main configurations have a default "template" functionality after installation, that can easily be customized to fit the surrounding equipment(s) by changing I/O signals, configuration data, RAPID data, and RAPID routines from RobotStudio for example.

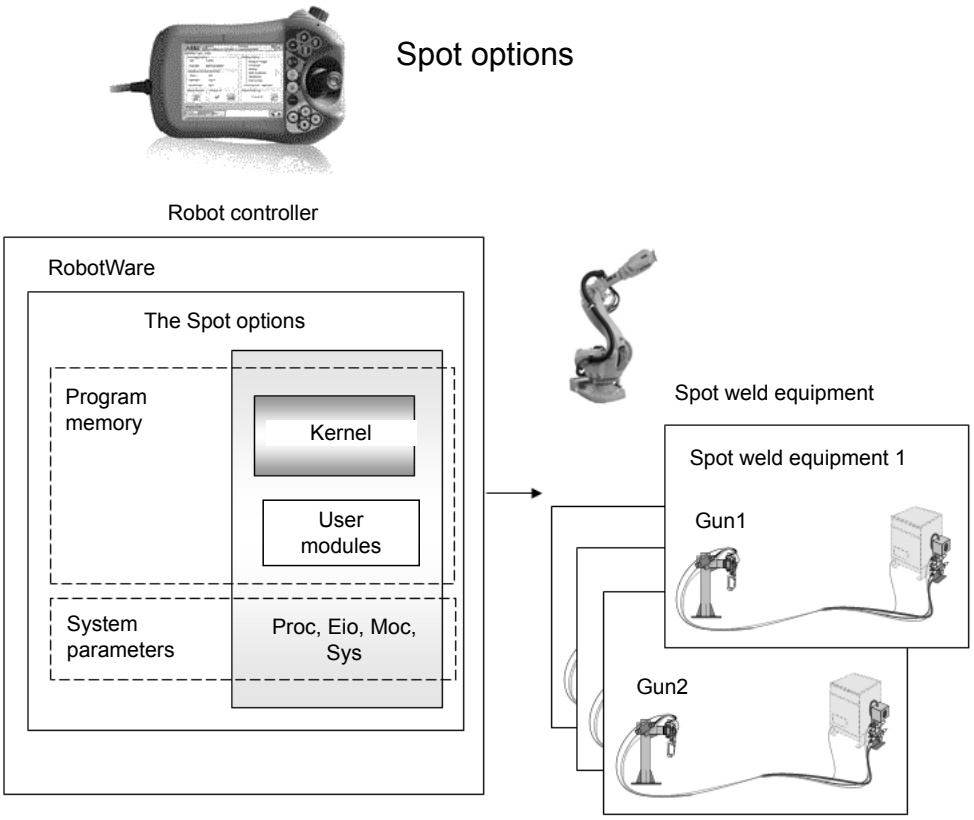
**Note**

Some functionality may not be available from one software version to another, e.g 5.15 vs 6.0.

*Continues on next page*

### Overview of the Spot option

As mentioned above it is possible to customize the Spot option to handle different types of equipment, and this picture below shows a schematic example of a spot welding system and the Spot option software.




xx1200000682

### Spot features

The Spot option package contains the following features:

- Fast and accurate positioning using the unique *QuickMove* and *TrueMove* concept.
- Gun pre-closing, gun closing will be synchronized with robot reaching the weld position to save cycle time.
- Support for mechanical gun equalizing systems.
- Software equalizing functions (if the Spot Servo Equalizing configuration is installed).
- Constant or variable tip force during welding for servo guns.
- Detect missing or improper plates for servo guns.
- Reverse execution with gun control
- Gun calibration functions for servo guns.
- Manual actions for welding and gun control.
- Welding with up to four guns at the same time.
- Several simulation possibilities for test purposes.

*Continues on next page*

- Weld error recovery with automatic rewelding.
- Default "ready to use" functionality directly after installation if SpotPack configuration is selected.
- Wide customizing possibilities, process data types, spotdata, gundata, such as weld counters and tip wear data, for each used gun.
- User-defined supervision and error recovery.
- Fast switch between two servo guns with a tool changer. Note that this requires the option *Servo Tool Change*.
- A dedicated Spot operator interface on the FlexPendant.
- Integrated weld timer fault management (for the options *782-x Bosch Weld timer*)
- RobotWare Spot can be applied on two robots robots in a MultiMove system, and run semi coordinated using WaitSyncTask. Option 634-1 MultiProcess is required for more than one robot.

## 1.2 Principles of the Spot option

**Process tasks**

The spot welding process will be controlled by separate internal hidden semi static tasks, which will run independently from the motion task.

The robot movements, the spot welding process and the continuous supervision will be handled in different independent tasks. This means that if for example the program execution and thus the robot movements is stopped, then the welding and supervision will continue until they come to a well defined process stop.

For example, the welding process will carry on and finish the weld and open the gun, although the program has been stopped during the weld phase.

The tasks running the spot weld processes are hidden and will not be visible on the FlexPendant or in RobotStudio. Only the motion task and the default supervision task are visible.

**User routines and modules**

At well defined places in the welding sequence, customizable user routines (hooks) will be executed. This offers the possibility to adapt the software to different preconditions and environments other than the default behaviour.

A number of customizable data types are also available to shape the behavior of the spot weld instructions.

## 1.3 Programming principles

### Introduction

Both the robot movement and the control of the spot weld equipment are embedded in the basic spot weld instructions `SpotL` and `SpotJ`. These are used for sequential welding and are available in all spot welding options. If there is a need to weld with several guns simultaneously then the instructions `SpotML` or `SpotMJ` are availble for that purpose. See *RAPID references on page 95*.

- Each spot welding process is specified by:
- `spotdata`: spot weld process data. See *spotdata - Spot weld data on page 140*.
- `gundata`: spot weld equipment data. See *gundata - Equipment specific weld data on page 135*.
- The system module SWUSRM: Process data and RAPID routines for data transfer between user code and kernel code. See *SWUSRM on page 164*.
- The system module SWUSER: RAPID routines for customization of the **process behavior**, checking the additional external equipments etc. See *SWUSER on page 158*.
- System parameters: the I/O signal configuration and the manipulator configuration. See *Configuration on page 21* and *Installation and service on page 202*.
- See *Operating manual - IRC5 with FlexPendant* and *Technical reference manual - System parameters*.

### Spot instructions

Both the robot movement and the control of the spot weld equipment are embedded in the basic spot weld instructions `SpotL` and `SpotJ`. These are used for sequential welding and are available in all spot welding options. If welding with several guns simultaneously then `SpotML` or `SpotMJ` has to be used.

| Instruction | Used to |
|---|---|
| `SpotL` | Control the motion, gun closure/opening and the welding process. Move the TCP along a linear path and perform a spot welding at the end position. |
| `SpotJ` | Control the motion, gun closure/opening and the welding process. Move the TCP along a non-linear path and perform a spot welding at the end position. |
| `SpotML` | Control the motion, gun closure/opening and 1 - 4 welding processes. Move the TCP along a linear path and perform spot welding with 1 - 4 gun equipments at the end position. Only available if *MultiGun Configuration* is installed. |
| `SpotMJ` | Control the motion, gun closure/opening and 1 - 4 welding processes. Move the TCP along a non-linear path and perform spot welding with 1 - 4 gun equipments at the end position. Only available if *MultiGun Configuration* is installed. |
| `IndGunMove` | Set the servo gun in independent mode and thereafter move the gun to a specific independent position. |
| `IndGunMoveReset` | Reset the independent mode for servo gun. |

*Continues on next page*

| Instruction | Used to |
|---|---|
| SetForce | Close the gun a predefined time then open the gun. |
| OpenHighLift | Open the pneumatic gun to the highlift position (large gap). |
| CloseHighLift | Close the pneumatic gun to the work stroke position (small gap). |
| CalibL | Calibrate the servo gun during linear movement to the programmed position. |
| CalibJ | Calibrate the servo gun during non-linear movement to the programmed position. |
| Calibrate | Calibrate the servo gun in current position without movement. |
| STTune | Tune motion parameters for the servo gun. |
| STTuneReset | Reset tuned motion parameters for the servo gun. |
| MeasureWearL | Measure the tip wear and recalculates the TCP. Only available if *Spot Servo Equalizing* is installed. |
| ReCalcTCP | Calculates the tip wear and recalculates the TCP. Only available if *Spot Servo Equalizing* is installed. |

### Spot welding data types

| Data type | Used to define |
|---|---|
| spotdata | The spot welding process, weld program number, gun force etc. |
| gundata | The spot welding equipment, gun name, weld counters etc. |
| forcedata | The SetForce process, gun force etc. |
| simdata | Simulation modes, controller simulation, timer simulation etc. |

# 2 Configuration

## 2.1 Spot process configuration

### Introduction

This chapter describes the process configuration options that are available for the Spot options, and information about the signals and parameters that are used.

The parameters that are used for the Spot options are configured in the system parameters. From the ABB menu on the FlexPendant:

1 Tap **Control Panel**.
2 Tap **Configuration**.
3 Tap **Topics** and select **Process**.

The Process configuration can also be accessed from RobotStudio.

The Spot option can be configured for different equipment setups, but the default setup for a basic Spot configuration is for one gun equipment.

Not used equipments can also be removed completely if not needed, in the Spot Equipments type.

### Spot process configuration types

The system parameters for RobotWare Spot are divided in the following types.

- Spot System
- Spot Error Handling
- Spot Equipments
- Spot Weld Equipment
- Spot Gun Equipment
- Spot Media Equipment
- Spot Equalizing
- Spot GUI

### Configuration types

| Configuration types | Definitions |
|---|---|
| Spot System | Configuration of Spot system specific data. |
| Spot Error Handling | Configuration of Spot error handling. |
| Spot Equipments | Defines the number of used spot welding equipment and equipment specific data. |
| Spot Weld Equipment | Configuration of the spot welding equipment, setup data, and the signals needed in the process and in the Spot GUI. |
| Spot Gun Equipment | Configuration of the spot gun equipment, setup data, and the signals needed in the process and in the Spot GUI. |
| Spot Media Equipment | Configuration of the spot media equipment, setup data, and the signals needed in the process and in the Spot GUI. |

*Continues on next page*

| Configuration types | Definitions |
|---|---|
| Spot Equalizing | Configuration of Software Equalizing, setup data. |
| Spot GUI | Configuration of the Spot GUI. |

### Configuration files

**Note**

Configuration files and backups shall not be loaded into systems running an older RobotWare version than in which they were created.

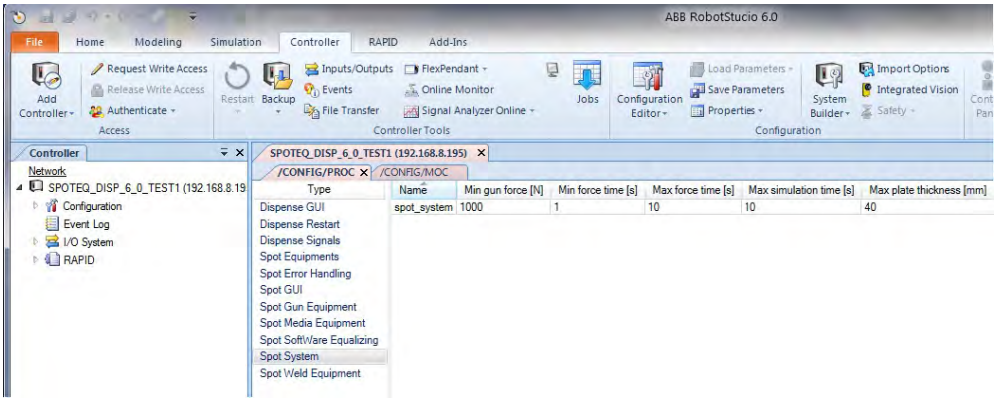
Configuration files and backups are not guaranteed to be compatible between major releases of RobotWare and may need to be migrated after a RobotWare upgrade.

*Continues on next page*

## 2.1.1 The type Spot System

### Description

The top level of configuration parameters for RobotWare Spot is *Spot System*. The settings of *Spot System* is valid for the whole robot system.




xx1400002066

### Parameters

The following parameters are used to define the system settings in RobotWare Spot.

**Note**

Settings are dependent on actual spot configuration.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | spot_system | string | The name of the system. |
| Min gun force | 1000 N | num | The minimum allowed gun force used for welding. (1-1000 N) |
| Min force time | 1 s | num | The minimum allowed gun force time in *SetForce*. (0-2 s) |
| Max force time | 10 s | num | The maximum allowed gun force time in *SetForce*. (0-15 s) |
| Max simulation time | 10 s | num | The maximum allowed simulation time if simulated weld mode. (0-15 s) |
| Max plate thickness | 40 mm | num | The maximum allowed plate thickness in the system. (0-100 mm) |
| Max plate tolerance | 1 mm | num | The maximum allowed plate tolerance in the system. (0-10 mm) |
| Min allowed tool weight | 10 kg | num | The minimum allowed tool weight when running spot instructions. (0.5-800 kg) |
| No. of force calibration measurements | 2 | num | The number of force calibration measurements in the *ManualForceCalib* service routine. (2-10) |

*Continues on next page*

### 2.1.1 The type Spot System
*Continued*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Sensor thickness for force calibration | 10 mm | num | The thickness of the force sensor used when performing a force calibration with the *ManualForceCalib* service routine. (0-50 mm) |
| Squeeze time for force calibration | 2 s | num | The squeeze time when when performing a force calibration with the *ManualForceCalib* service routine. (1-10 s) |
| Motion task user module name | SWUSRM.SYS | string | The name of the user module running in the motion task only. |
| All task user module name | SWUSER.SYS | string | The name of the user module running in all tasks. |
| Spot user modules file path | HOME:/Spot | string | The location and file path of the spot user modules. Used when saving user data in the calibration and measurement routines, e.g CalibL, MeasureWearL etc. |
| Use Spot Equipment1 - 10 | spotequipment1 | string | The name of the used spot equipment(s) used in the system. Max number of spot equipments are 10. |

## 2.1.2 The type Spot Error Handling

### Description

The *Spot Error Handling* contains parameters for the global error handling settings.

### Parameters

The following parameters and signals are used to define the error handling in RobotWare Spot.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | spot_error_handling | string | The name of the error handling instance. |
| Number of automatic rewelds | no reweld | num | Automatic reweld: Number of automatic tries to reweld after weld complete timeout, 0-3 are possible to configure. |
| Show 'skip' button in auto mode | No | bool | Show skip button in the error dialogs on the TPU in automatic oprating mode, Yes/No. |
| Show 'skip' button in manual mode | Yes | bool | Show skip button in the error dialogs on the TPU in manual oprating mode, Yes/No. |
| Show 'ignore' button in auto mode | No | bool | Show ignore button in the error dialogs on the TPU in automatic oprating mode, Yes/No. |
| Show 'ignore' button in manual mode | Yes | bool | Show ignore button in the error dialogs on the TPU in manual oprating mode, Yes/No. |
| Error dialog ack. for 'Skip' | skip_proc | signaldi | The digital input signal that may interrupt the operator dialog, connected to the 'Skip' button. |
| Error dialog ack. for 'Retry' | reweld_proc | signaldi | The digital input signal that may interrupt the operator dialog, connected to the 'Retry' button. |
| Error dialog ack. for 'Skip' or 'Retry' | ext_override | signaldi | The digital input signal that may interrupt the operator dialog, connected to the 'Skip' and 'Retry' button. |
| User defined error handling | No | bool | User defined error handling. The error handling routine `SwErrorRecover` in `SWUSER` is called instead of the built in error handling if this parameter is set to Yes. |

## 2.1.3 The type Spot Equipments

### Description

The *Spot Equipment*s defines the number of spot equipments defined in the system. Max number of instances are 10.

### Parameters

The following parameters and signals are used to define the equipments in RobotWare Spot.

**Note**

Settings are dependent on actual spot configuration.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | Spot Equipment1 | string | The name of the spot equipment instance. |
| Use Weld Equipment | weldtimer1 | string | Pointer to the used weld eqipment. |
| Use Gun Equipment | servogun1 | string | Pointer to the used gun equipment. |
| Use Media Equipment | mediapanel1 | string | Pointer to the used media equipment. |
| Spot GUI equipment OK | diEquipmentOk | signaldi | Spot GUI equipment OK status signal. |
| Spot process running | doProcessRun | signaldo | Spot process running signal, set high during the spot process. |
| Spot process fault | doProcessFault | signaldo | Spot process fault signal, set high if a process fault occur during process.. |

## 2.1.4 The type Spot Weld Equipment

### Description

The *Spot Weld Equiment* contains parameters for the connected weld equipment(s). This instance can be multiplied.

### Parameters

The following parameters and signals are used to define the weld equipments in RobotWare Spot and signals visible in the Spot GUI.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | weldtimer1 | string | The name of the weld timer instance. |
| Weld process start [DO] | doStartWeld | signaldo | Used to configure the weld start signal for the timer. Signal will be set when gun is closed and the robot is in position.<br>This signal is required. |
| Weld process complete [DI] | diWeldComplete | signaldi | Used to configure the weld complete signal for the timer. Signal will be set by the timer when the current weld is ready.<br>This signal is required. |
| Weld timeout | 2 s | num | The max time waiting for the weld process complete signal, after this time the error handling is activated. (0-10 s)<br>**Note**<br>If the auto reweld function is activated, the weld will be restarted automatically the specified times before the error handling is activated, see *The type Spot Error Handling on page 25* |
| Enable current [DO] | doEnableCurrent | signaldo | Used to configure the weld current enable signal. This signal will only be set if no simulation is activated.<br>If no signal name is specified here, there will be no function. |
| Timer ready to weld [DI] | diTimerReady | signaldi | Used to configure the weld timer ready signal. This signal will only be set if no simulation is activated. Signal will be set by the timer if no error is present.<br>If no signal name is specified here, there will be no function. |
| Timer ready timeout | 2 s | num | The max time waiting for the weld timer ready signal, after this time the error handling is activated. (0-5 s) |

*Continues on next page*

2.1.4 The type Spot Weld Equipment
*Continued*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Stop weld process [DI] | diWeldFault | signaldi | Used to configure the stop weld process signal. This signal can be set by the timer if an error is detected during the weld sequence. If set the current weld will be stopped.<br>If no signal name is specified here, there will be no function. |
| Weld program group [GO] | goWeldProgram | signalgo | Used to configure the weld program group signal. This signal will be set at the beginning of a spot instruction.<br>If no signal name is specified here, there will be no function. |
| New program selection [DO] | doNewProgram | signado | Used to configure the new weld program selection signal. This signal will be set just after the weld program group signal is set.<br>If no signal name is specified here, there will be no function. |
| Weld program valid [DI] | diProgSelectValid | signaldi | Used to configure the weld program vaild signal. This signal will be set by the timer when a vaild weld program is selected. It will be checked during the process before the weld is started.<br>If no signal name is specified here, no check will be done. |
| Weld program valid timeout | 2 s | num | The max time waiting for the valid weld program signal, after this time the error handling is activated. (0-5 s) |
| Reset timer fault [DO] | doResetFault | signaldo | Used to configure the reset timer fault signal. This signal can reset timer faults before a reweld is done.<br>If no signal name is specified here, there will be no function. |
| Reset fault time | 2 s | num | The length of the reset fault signal pulse to reset the timer. (0-2 s) |
| Wait time after reset fault | 2 s | num | Wait time after the reset fault pulse before program execution continues. (0-2 s) |
| Weld contactor on [DO] | doWeldPowerContact | signaldo | Used to configure the weld contactor signal. This signal will be set by a cross connection in the I/O configuration, as a result of the motor_on, doEnableCurrent and doProcessFault inverted. |
| Weld contactor on [DI] | diWeldContact | signaldi | Used to configure the weld contactor activated signal. This signal is normally set by the weld contactor if a contactor is used. If not set before the weld, the error handling will be activated.<br>If no signal name is specified here, there will be no function. |

*Continues on next page*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Gun force from timer [GI] | giGunForce | signalgi | Used to configure the gun force group signal. This signal can be used if external gun force data from the weld timer is required.<br>If no signal name is specified here, there will be no function.<br>**Note**<br>To activate the use of the timer input signals the corresponding parameter in spotdata must be set to -1, e.g. my_spot.tip_force := -1; |
| Force calculation factor | 40 | num | Gun force factor when using external force from weld timer. (0-100)<br>Example with 8 bit group input, 255 * 39.2 ~ 10000 N Max in timer. |
| Plate thickness from timer [GI] | giPlateThickness | signalgi | Used to configure the plate thickness group signal. This signal can be used if external plate thickness data from the weld timer is required.<br>If no signal name is specified here, there will be no function.<br>**Note**<br>To activate the use of the timer input signals the corresponding parameter in spotdata must be set to -1, e.g. my_spot.plate_thickness := -1; |
| Thickness calculation factor | 0.1 | num | Plate thickness factor when using external data from weld timer. (0-100)<br>Example with 8 bit group input, 255 * 0.1 = 25.5mm max thickness. |
| Plate tolerance from timer [GI] | giPlateTolerance | signalgi | Used to configure the plate tolerance group signal. This signal can be used if external plate tolerance data from the weld timer is required.<br>If no signal name is specified here, there will be no function.<br>**Note**<br>To activate the use of the timer input signals the corresponding parameter in spotdata must be set to -1, e.g. my_spot.plate_tolerance := -1; |
| Tolerance calculation factor | 0.1 | num | Plate tolerance factor when using external data from weld timer. (0-100)<br>Example with 8 bit group input, 255 * 0.1 = 25.5 max tolerance. |

*Continues on next page*

2.1.4 The type Spot Weld Equipment
*Continued*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Timer status [GI] | giTimerStatus | signalgi | Used to configure the timer status group signal. This signal is used to check the timer status when a weld fault has occured.<br>If no signal name is specified here, there will be no function. |
| Reset fault with reweld [DO] | | signaldo | Used to configure the reset timer fault with reweld functionality.<br>This signal can be used to enable a reweld in KSR mode in a Bosch weld timer when performing a re-weld in adaptive weld mode. If used the normal start weld signal will be set to 1 during the reweld sequence, and this signal will reset the timer fault and perform the reweld. The normal reset fault signal will not be used.<br>If no signal name is specified here, there will be no function. |
| User defined gui signal1-10 | | string | Used to configure user defined signals that should be visible in the Spot GUI process cabinet view. Not used in process. |

## 2.1.5 The type Spot Gun Equipment

### Description

The *Spot Gun Equipment* contains parameters for the connected gun equipment(s). This instance can be multiplied.

### Parameters

The following parameters and signals are used to define the gun equipments in RobotWare Spot and signals visible in the Spot GUI.

**Note**

Settings are dependent on actual spot configuration.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | servogun1 | string | The name of the gun instance. |
| Selected gun type | Servo gun | string | Used to configure the gun type, possible values are Servo gun or Pneu gun. |
| Gun trafo over-temperature | diTrafoTempOk | signaldi | Used to configure the transformer temperature sensor signal connected to the gun. Signal will be checked during the spot process.<br>If no signal name is specified here, no check will be done. |
| Gun opened | diGunOpen | signaldi | Used to configure the gun opened sensor signal for a pneumatic gun. Signal will be checked during process.<br>If no signal name is specified here, no check will be done during process. |
| Gun open timeout | 2 s | num | The max time[s] waiting for the gun open signal after a pneumatic gun has been opened, after this time the error handling is activated. (0-10 s) |
| Gun highlift open | diHighLiftOpen | signaldi | Used to configure the highlift open sensor signal for a pneumatic gun. Signal will be checked during the spot process.<br>If no signal name is specified here, no check will be done during process. |
| Gun equalizing | doEqualize | signaldo | Used to configure the gun equalizing signal for the gun. This signal will be set when closing the gun and reset when opening the gun after weld.<br>If no signal name is specified here, there will be no function. |

*Continues on next page*

2.1.5 The type Spot Gun Equipment
*Continued*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Gun pre equalizing time | 0 s | num | Time before gun in weld position, when the equalizing signal is set for activation of a mechanical equalizing system in the gun, if used. (0-0.5 s) |
| Gun close | doCloseGun | signaldo | Used to configure the gun close signal for a pneumatic gun. This signal will be set a predefined time before the weld position, and reset when the weld is completed.<br>If no signal name is specified here, there will be no function. |
| Gun pre closing time | 0.1 s | num | Time before gun in weld position, when the asynchronous gun closure is started. For pneumatic guns, when the gun close signal is set. (0-0.5 s)<br>This data is not used if Software Equalizing is active. In this case the preclosing is handled automatically during the movement from the release distance to the weld position. |
| Gun close timeout | 2 s | num | The max time waiting for gun open signal before closing a pneumatic gun, after this time the error handling is activated. (0-10 s) |
| Gun open highlift | doOpenHighLift | signaldo | Used to configure the open highlift signal for a pneumatic gun. Signal will be set during the spot process if the optional argument \OpenHighLift is selected in the spot instruction. (Pneumatic guns only)<br>If no signal name is specified here, there will be no function. |
| Gun close highlift | doCloseHighLift | signaldo | Used to configure the close highlift signal for a pneumatic gun. Signal will be set during the spot process if the optional argument \CloseHighLift is selected in the spot instruction. (Pneumatic guns only)<br>If no signal name is specified here, there will be no function. |
| Gun pressure group | | signalgo | Used to configure the gun pressure group signal for a pneumatic gun. Will be set to the value specified in the tip force parameter in spotdata.<br>If no signal name is specified here, there will be no function. |
| Gun pressure OK | diPressureOk | signaldi | Used to configure the gun pressure ok signal for a pneumatic gun.<br>If no signal name is specified here, no check will be done during process. |
| Gun pressure timeout | 2 s | num | The max time waiting for gun pressure ok signal, after this time the error handling is activated. (Pneumatic guns only). (0-10 s) |

*Continues on next page*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Force complete | force_complete | signaldi | Used to configure a gun force complete signal that can be used in the `SetForce` instruction.<br>If no signal name is specified here, there will be no function. |
| Max allowed gun force | 5000 N | num | Maximum allowed tip force for each gun. This value can be set from the `ManualForceCalib` service routine when performing a force calibration of the gun. (0-10000 N). See *Available service routines on page 74*.<br>**Note**<br>Normally this data is supplied by the gun manufacturer and the max allowed gun force value should be entered here. |
| Gun force normal orientation | Deactivated | num | Reference gun force when moveable gun arm is working with gravity.<br>This parameter can be set from the `ManualForceCalib` service routine.<br>Example: Applied gun force 5000N normal gun orientation, measured force 5000N.<br>Deactivated (-1) = Not used.<br>**Note**<br>The max gun force will be used as reference for the two gun positions. |
| Gun force inverted orientation rotated 180° | Deactivated | num | Reference gun force in when moveable gun arm is working against gravity.<br>This parameter can be set from the `ManualForceCalib` service routine.<br>Example: Applied gun force 5000N in inverted gun orientation, measured force 4500N.<br>If these parameters are used the gun force will be compensated before weld depending on the current angle of the gun in the specific position.<br>Deactivated, (-1) = Not used.<br>**Note**<br>The max gun force will be used as reference for the two gun positions. |

*Continues on next page*

2.1.5 The type Spot Gun Equipment
*Continued*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Use SoftWare equalizing | Yes | bool | Soft equalizing specific data.<br>This data has to be set to Yes to activate the software equalizing functions release of the fixed gun arm and gun arm deflection compensation. Yes/No |
| MeasureWearL search I/O | | signaldi | Soft equalizing specific data.<br>Used to configure an input signal that can be used instead of the reference plate for the search sequence in the `MeasureWearL` instruction.<br>If this signal is specified the search will be done against a sensor signal instead of a fixed reference surface. |
| Tip change supervision value | 3 mm | num | Soft equalizing specific data.<br>Tip change supervision value.<br>Max allowed digression [mm] in positive and negative direction from stored reference values. Default value 3 mm.<br>This data is used to supervise a missing tip or wrong size of the tip and is used in the `MeasureWearL` and `Calibrate/CalibL/J` instructions. (Max 10 mm) |
| Tip wear supervision value | 0.2 mm | num | Soft equalizing specific data.<br>Tip wear supervision value.<br>Max allowed digression [mm] in positive and negative direction since last tip wear compensation. Default value 0.2 mm.<br>This data is used to supervise the tip wear and is used in the `MeasureWearL`, `ReCalcTCP` and `Calibrate/CalibL/J`instructions. (Max 2 mm) |
| Tip wear ratio, fixed vs total wear | Deactivated | num | Soft equalizing specific data.<br>The expected ratio [%] between the tip wear for the fix tip related to the total tip wear. This value has to be set to a permitted value (between 0-100) if the calculation method (`ReCalcTCP`) is used for the tip wear compensation. The value can be set in predefined steps of 10%.<br>Example: Fixed tip 60% of total wear Indicates that the wear of the fixed tip is 60% of the total tip wear, which leaves 40% for the moving tip, 60/40.<br>**Note**<br>If the measuring method is used, (`MeasureWearL`) this value has to be set to Deactivated. |

*Continues on next page*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Opposite z-direction | No | bool | Soft equalizing specific data.<br>Defined z-direction for the TCP, gives move direction for search and compensations movements. **Yes**/**No**.<br>**No** = positive z-direction out from the fixed tip (Normal setting).<br>**Yes** = positive z-direction into the fixed tip (Setting for stationary tools to achieve the same jogging behavior as with a robot held tool).<br>This parameter also influences the direction of the gun arm deflection compensation.<br>For more information, see *How to define the TCP on page 171*. |
| MeasureWearL TouchUp force override | Deactivated | num | Soft equalizing specific data.<br>Contact force (in N) during tip measurement in the `MeasureWearL` instruction, (typically between 50 - 150N).<br>This parameter will override the global parameter in the `Spot Equalizing` instance if used.<br>Deactivated = Not used. |
| Tip change fault | | signaldo | Soft equalizing specific data.<br>Used to configure the tip change fault signal. This signal will be set when a tip wear error is detected in the `CalibL/J, MeasureWearL, RecalcTCP` instructions.<br>If no signal name is specified here, there will be no function. |
| Tip wear fault | | signaldo | Soft equalizing specific data.<br>Used to configure the tip wear fault signal. This signal will be set when a tip wear error is detected in the `MeasureWearL` and `ReCalcTCP` instructions.<br>If no signal name is specified here, there will be no function. |
| User defined gui signal1-10 | | string | Used to configure user defined signals that should be visible in the Spot GUI gun equipment view. Not used in process. |

## 2.1.6 The type Spot Media Equipment

### Description

The *Spot Media Equipment* contains parameters for the connected media equipment(s). This instance can be multiplied.

### Parameters

The following parameters and signals are used to define the media equipment settings in RobotWare Spot and signals visible in the Spot GUI.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | mediapanel1 | string | The name of the media panel instance. |
| Water flow sensor1 | diWaterFlow1Ok | signaldi | Used to configure the water flow sensor1 in the media panel. Signal will be checked during the spot process and from the supervision task if configured.<br>If no signal name is specified here, no check will be done. |
| Water flow sensor2 | diWaterFlow2Ok | signaldi | Used to configure the water flow sensor2 in the media panel. Signal will be checked during the spot process and from the supervision task if configured.<br>If no signal name is specified here, no check will be done. |
| Water sensor flow timeout | 2 s | num | Used to configure the water flow timeout. If no water flow is detected by the water flow sensors within the specified time an operator error dialog will take focus. (0-10 s) |
| Continuous water supervision | No = FALSE | bool | Used to configure the continuous water supervision in the SW_SUP task.<br>If set to Yes, the SW_SUP task will supervise the water flow continuously if the system is in motors on state, if an error is detected the robot movement will stop. |
| Air flow sensor | diAirOk | signaldi | Used to configure the air flow signal in the media panel. Signal will be checked during the spot process.<br>If no signal name is specified here, no check will be done. |
| Water flow start | doStartWater | signaldo | Used to configure the water start signal in the media panel. This signal will be reset when a flow error is detected. If a delay is configured the signal is reset after the time has passed.<br>If no signal name is specified here, there will be no function. |

*Continues on next page*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Water turn off delay | 1 s | num | Used to configure a delay before the water start signal is reset, can be used as a filter to prevent air bubbles causing false alarms. (0-2 s) |
| Water saver activated | No | bool | Used to configure water saver function in the SW_SUP task. If set to Yes, water will only start and be supervised if the system is in cycle on state and executing a program. |
| Media equipment OK | diWaterOk | signaldi | Used to configure the media equipment ok summary signal. This signal will be checked during the spot process and from the supervision task if configured.<br>If no signal name is specified here, no check will be done. |
| User defined gui signal1-10 | | string | Used to configure user defined signals that should be visible in the Spot GUI water and air unit view. Not used in process. |

## 2.1.7 The type Spot SoftWare Equalizing

### Description

The *Spot Equalizing* contains parameters for global software equalizing settings.

### Parameters

The following parameters are used to define the software equalizing settings in RobotWare Spot.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | spot_equalizing | string | The name of the software equalizing instance. |
| MeasureWearL search speed | 5 mm/s | num | Search speed during tip measurement in the `MeasureWearL`instruction, (between 1 - 5 mm/s) |
| MeasureWearL TouchUp force | 100 N | num | Contact force (in N) during tip measurement in the `MeasureWearL` instruction, (typically between 50 - 150N).<br>-1 will deactivate this parameter and use the override parameter in the `Spot Gun Equipment` instance instead. |
| MeasureWearL movein distance | 10 mm | num | Maximal distance from programmed point to search for reference surface in the `MeasureWearL` instruction. |
| Max allowed release distance | 15 mm | num | Maximum allowed release distance. (0-20 mm) |
| Max allowed deflection value | 15 mm | num | Maximum allowed deflection distance. (0-20 mm) |
| Min allowed TouchUp step | 0.1 mm | num | Minimum allowed touch up step. (0-1 mm) |
| Max allowed TouchUp step | 10 mm | num | Maximum allowed touch up step. (1-15 mm) |
| SoftMove offset distance | 3 mm | num | Offset distance from the nominal plate position when using SMEQ (between 1 - 5 mm).<br>**Note**<br>When using SoftMove Equalizing the teach_dist cannot be used. |
| SoftMove approach speed | 50 mm/s | num | Search speed (v_tcp) into nominal position when using SoftMove Equalizing (between 20 - 200 mm/s).<br>**Note**<br>A too high speed will influence the "search" result negatively. Excessive force may deform the plate. |

*Continues on next page*

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| SoftMove gun close speed | 200 mm/s | num | Gun close speed (v_leax) to target position (plate thickness) when using SoftMove Equalizing (between 20 - 200 mm/s).<br>This parameter can be used to change the gun closing speed if the gun is impacting the plates, for example if the location of the plates are higher than the nominal position. |
| SoftMove force offset auto tuning | Yes | bool | Enable or disable `force_offset` auto tuning. |

## 2.1.8 The type Spot GUI

### Description

The *Spot GUI* contains parameters for the spot gui settings.

### Parameters

The following parameters are used to define the spot gui settings in RobotWare Spot.

| Parameter | Default value | Data type | Note |
|---|---|---|---|
| Name | spot_gui | string | The name of the spot gui instance. |
| Show Spot GUI at startup | No | bool | Used to define i the spot gui should be started automatically at system startup, Yes/No. |
| Spot system type in Manual Actions | servo | string | Used to define the system type in the Manual Action view, pneumatic/servo/combined. |
| Show simdata in Process Data | No | bool | Used to define if the simdata should be visible in Process Data view. |
| Show forcedata in Process data | Yes | bool | Used to define if the forcedata should be visible in the Process Data view. |

## 2.2 Spot I/O configuration

### Introduction

This chapter describes the different predefined I/O configurations that are available for the Spot options, and information about the use of the signals.

The Spot package can be configured for different equipment setups. The default I/O configurations should be seen as default templates, and the physical connections and names can be changed freely and signals not in use can be connected to simulated devices.

The signals used are configured in the system parameters. From the ABB menu on the FlexPendant:

1. Tap **Control Panel**.
2. Tap **Configuration**.
3. Tap **Topics** and select **I/O**.

The I/O configuration can also be accessed from RobotStudio.

**Note**

The used signals names are also used in the process configuration. If the names are changed, the corresponding names in the process configuration must be changed also. For more information, see *Spot process configuration on page 21*.

*Continues on next page*

## 2.2.1 Spot I/O configuration for single gun system

### Introduction

If a basic configuration is selected when building a spot welding system, the system will be prepared with signals for one spot welding equipment on simulated I/O. It is then easy to adapt the configuration to the surrounding equipment, e.g weld timer, media panel etc.

### Default configuration

The I/O configuration is prepared for one weld equipment. A set of customized user modules are also installed if this configuration is selected.

The signals are connected to LOCAL devices by default, and can be changed to any unit type if needed, for example PROFINET, DeviceNet, and so on.

**Note**

Some of the signals are only used in a Spot Pneumatic configuration.

### Predefined I/O devices

There are three predefined I/O devices:

- One local device, named *SW_TIMER*, with signals for the weld timer.
- One local device, named *SW_BOARD*, with signals for the media panel and the gun.
- One local device, named *SW_SIM_BOARD* with some internal or normally not connected signals.

The devices are configured on a virtual network by default.

### Weld timer signals

| Name | Type | Information |
|---|---|---|
| diWeldComplete | input | Weld complete signal from the weld timer. |
| diWeldFault | input | Fault signal from the weld timer. If this signal is activated during the weld process the weld error handling in Spot is started without waiting for weld time out. |
| diTimerReady | input | The timer is ready to weld. |
| doTimerOn | output | 24V safety signal cross connected to motors on, can be used to turn off the timer logic in motor off state. |
| doStartWeld | output | Start signal to the weld timer. |
| doEnableCurrent | output | Signal used for the weld simulation function (simtype = 2). See *Simulation modes on page 72*. |
| doResetFault | output | Reset signal. Can be used to reset the welding controller after a weld error. The signal is pulsed with a user defined pulse length before manual or automatic rewelding. |
| goWeldProgram | output group | Selected weld program number in weld timer, see *spotdata - Spot weld data on page 140*. Default size is 8 bits, 0 - 255, 256 different programs. The maximum group size can be configured up to 32 bits, see*spotdata - Spot weld data on page 140* |

*Continues on next page*

| Name | Type | Information |
|---|---|---|
| doNewProgram | output | This signal can be used as handshaking to let the timer know a new program has been selected in the *goWeldProgram* group.<br>**Note**<br>For some timers this signal must be set after a new program number is set in *goWeldProgram* in order for the timer to set the corresponding inputs *giGunForce*, *giPlateThickness*, and *giPlateTolerance*. |
| diProgSelectValid | input | This signal can be used as handshaking to let spot know that a valid program has been read by the timer and to continue the execution and read the timer input groups, force, thickness, etc.<br>**Note**<br>Some timers has the possibility to check if a valid weld program selection has been done.<br>Valid program = 1, Not valid = 0<br>This signal is used by spot. If not set, a program valid timeout will occur during execution. |
| giGunForce | input group | Gun force from weld timer if `tip_force` in `spotdata` is -1. |
| giPlateThickness | input group | Plate thickness from weld timer if `plate_thickness` in `spotdata` is -1. |
| giPlateTolerance | input group | Plate tolerance from weld timer if `plate_tolerance` in `spotdata` is -1. |
| giTimerStatus | input group | Weld timer status code. |

## Gun and media signals

| Name | Type | Information |
|---|---|---|
| diGunOpen | input | Signal indicating that a pneumatic gun is opened. |
| diHighLiftOpen | input | Signal indicating that a pneumatic gun has reached the highlift position. |
| diPressureOk | input | Signal indicating that the right gun pressure is reached for a pneumatic gun. |
| diTrafoTempOk | input | Signal indicating that the temperature is too high. |
| diWaterFlow1Ok | input | Signal that can be used to indicate problems with the water supply in pipe 1. |
| diWaterFlow2Ok | input | Signal that can be used to indicate problems with the water supply in pipe 2. |
| diAirOk | input | Signal indicating low air pressure in the equalize cylinder. |
| diWeldContact | input | Signal indicating the state of the weld contactor.<br>(0 = deactivated) |
| diEquipmentOk | input | Signal indicating the total gun status. A number of input signals from the gun is cross connected to this signal. |

*Continues on next page*

2.2.1 Spot I/O configuration for single gun system
*Continued*

| Name | Type | Information |
|---|---|---|
| doEqualize | output | Gun equalizing signal if mechanical equalizing system are used. |
| doCloseGun | output | Gun close signal for a pneumatic gun. |
| doOpenHighLift | output | Signal used to open a pneumatic gun to the highlift position. |
| doCloseHighLift | output | Signal used to close apneumatic gun from the highlift position. |
| doStartWater | output | Signal used to activate the water cooling system.<br>This signal is set depending on status of several cross-connected signals.<br>See *Cross-connected signals on page 48*. |
| doWeldPowerContact | output | Signal used to pull the weld contactor.<br>This signal is set depending on status of several cross-connected signals<br>See *Cross-connected signals on page 48*. |

**Process status signals**

| Name | Type | Information |
|---|---|---|
| doProcessRun | output | Is set at motion start and is reset when the weld process is ready and motion is released. |
| doProcessFault | output | Is set when an error situation occurs and the process is interrupted. |

**Other signals**

| Name | Type | Information |
|---|---|---|
| force_complete | input | Can be used to interrupt the `SetForce` instruction before the programmed force time is elapsed for a servo gun. |
| reweld_proc | input | Can be used to answer a weld error dialog on the FlexPendant with an input signal. The same as tapping **Reweld**. |
| skip_proc | input | Can be used to answer an error dialog on the FlexPendant with an input signal. The same as tapping **Skip**. Only in manual operating mode. |

**Cross-connected signals**

| Resultant | Activator(s) | Information |
|---|---|---|
| doStartWater | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Water start signal, if any of the activators are not set the water start signal will be reset. |
| doWeldPower-Contact | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset. |
| diEquipmentOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok AND<br>diAirOk AND<br>diTempOk AND<br>diTimerReady | Equipment status signal, if any of the activators are not set the equipment ok signal will be reset. |

*Continues on next page*

| Resultant | Activator(s) | Information |
|---|---|---|
| diWaterOk | diWaterFlow1Ok AND diWaterFlow2Ok | Water status signal, if any of the activators are not set the water ok signal will be reset. |
| ext_override | skip_proc AND reweld_proc | Reset signal. Can be used to reset the operator error dialog on the FlexPendant when an error occurs.<br>See *Other signals on page 48*. |

## Limitations

This configuration is only availible for single robot Spot systems, not in MultiProcess systems with more than one Spot robot.

## 2.2.2 Spot I/O configuration multiple guns system

### Introduction

If a multiple gun configuration is selected when building a spot welding system, the system will be prepared with signals for four spot welding equipments on simulated I/O's. It is then easy to adapt the configuration to the surrounding equipment, e.g weld timers, media panels etc.

### Default configuration

The I/O configuration is prepared for four weld equipments. A set of customized user modules are also installed if this configuration is selected.

The signals are connected to LOCAL devices by default, and can be changed to any unit type if needed, for example PROFINET, DeviceNet, and so on.

**Note**

Some of the signals are only used in a Spot Pneumatic configuration.

**Note**

Signal names for gun equipment 2 are the same as for gun 1 but with the ending G2, e.g. doStartWeldG2.

### Predefined I/O devices

There are five predefined I/O devices if a multiple gun configuration is selected:

- One local device, named **SW_BOARD1**, with signals for gun equipment 1.
- One local device, named **SW_BOARD2** with signals for gun equipment 2.
- One local device, named **SW_BOARD3** with signals for gun equipment 3.
- One local device, named **SW_BOARD4** with signals for gun equipment 4.
- One local device, named **SW_SIM_BOARD** with some internal or normally not connected signals.

The devices are configured on a virtual network by default.

### Weld timer signals for eqipment 1

| Name | Type | Information |
|---|---|---|
| diWeldComplete | input | Weld complete signal from the weld timer. |
| diWeldFault | input | Fault signal from the weld timer. If this signal is activated during the weld process the weld error handling in Spot is started without waiting for weld time out. |
| diTimerReady | input | The timer is ready to weld. |
| doTimerOn | output | 24V safety signal cross connected to motors on, can be used to turn off the timer logic in motor off state. |
| doStartWeld | output | Start signal to the weld timer. |
| doEnableCurrent | output | Signal used for the weld simulation function (simtype = 2). See *Simulation modes on page 72*. |

*Continues on next page*

| Name | Type | Information |
|---|---|---|
| doResetFault | output | Reset signal. Can be used to reset the welding controller after a weld error. The signal is pulsed with a user defined pulse length before manual or automatic rewelding. |
| goWeldProgram | output group | Selected weld program number in weld timer, see *spotdata - Spot weld data on page 140*. Default size is 8 bits, 0 - 255, 256 different programs. The maximum group size can be configured up to 32 bits, see*spotdata - Spot weld data on page 140* |
| doNewProgram | output | This signal can be used as handshaking to let the timer know a new program has been selected in the *goWeldProgram* group.<br>**Note**<br>For some timers this signal must be set after a new program number is set in *goWeldProgram* in order for the timer to set the corresponding inputs *giGunForce*, *giPlateThickness*, and *giPlateTolerance*. |
| diProgSelectValid | input | This signal can be used as handshaking to let spot know that a valid program has been read by the timer and to continue the execution and read the timer input groups, force, thickness, etc.<br>**Note**<br>Some timers has the possibility to check if a valid weld program selection has been done.<br>Valid program = 1, Not valid = 0<br>This signal is used by spot. If not set, a program valid timeout will occur during execution. |
| giGunForce | input group | Gun force from weld timer if `tip_force` in `spotdata` is -1. |
| giPlateThickness | input group | Plate thickness from weld timer if `plate_thickness` in `spotdata` is -1. |
| giPlateTolerance | input group | Plate tolerance from weld timer if `plate_tolerance` in `spotdata` is -1. |
| giTimerStatus | input group | Weld timer status code. |

### Gun and media signals for equipment 1

| Name | Type | Information |
|---|---|---|
| diGunOpen | input | Signal indicating that a pneumatic gun is opened. |
| diHighLiftOpen | input | Signal indicating that a pneumatic gun has reached the highlift position. |
| diPressureOk | input | Signal indicating that the right gun pressure is reached for a pneumatic gun. |
| diTrafoTempOk | input | Signal indicating that the temperature is too high. |
| diWaterFlow1Ok | input | Signal that can be used to indicate problems with the water supply in pipe 1. |
| diWaterFlow2Ok | input | Signal that can be used to indicate problems with the water supply in pipe 2. |

*Continues on next page*

2.2.2 Spot I/O configuration multiple guns system
*Continued*

| Name | Type | Information |
|---|---|---|
| diAirOk | input | Signal indicating low air pressure in the equalize cylinder. |
| diWeldContact | input | Signal indicating the state of the weld contactor.<br>(0 = deactivated) |
| diEquipmentOk | input | Signal indicating the total gun status. A number of input signals from the gun is cross connected to this signal. |
| doEqualize | output | Gun equalizing signal if mechanical equalizing system are used. |
| doCloseGun | output | Gun close signal for a pneumatic gun. |
| doOpenHighLift | output | Signal used to open a pneumatic gun to the highlift position. |
| doCloseHighLift | output | Signal used to close apneumatic gun from the highlift position. |
| doStartWater | output | Signal used to activate the water cooling system.<br>This signal is set depending on status of several cross-connected signals.<br>See *Cross-connected signals on page 48*. |
| doWeldPowerContact | output | Signal used to pull the weld contactor.<br>This signal is set depending on status of several cross-connected signals<br>See *Cross-connected signals on page 48*. |

### Process status signals for equipment 1

| Name | Type | Information |
|---|---|---|
| doProcessRun | output | Is set at motion start and is reset when the weld process is ready and motion is released. |
| doProcessFault | output | Is set when an error situation occurs and the process is interrupted. |

### Other signals

| Name | Type | Information |
|---|---|---|
| force_complete | input | Can be used to interrupt the `SetForce` instruction before the programmed force time is elapsed for a servo gun. |
| reweld_proc | input | Can be used to answer a weld error dialog on the FlexPendant with an input signal. The same as tapping **Reweld**. |
| skip_proc | input | Can be used to answer an error dialog on the FlexPendant with an input signal. The same as tapping **Skip**. Only in manual operating mode. |

### Cross-connected signals

| Resultant | Activator(s) | Information |
|---|---|---|
| doStartWater | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Water start signal, if any of the activators are not set the water start signal will be reset. |
| doWeldPower-Contact | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset. |

*Continues on next page*

| Resultant | Activator(s) | Information |
|---|---|---|
| diEquipmentOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok AND<br>diAirOk AND<br>diTempOk AND<br>diTimerReady | Equipment status signal, if any of the activators are not set the equipment ok signal will be reset. |
| diWaterOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok | Water status signal, if any of the activators are not set the water ok signal will be reset. |
| ext_override | skip_proc AND<br>reweld_proc | Reset signal. Can be used to reset the operator error dialog on the FlexPendant when an error occurs.<br>See *Other signals on page 48*. |

## Limitations

This configuration alternative can not be used together with the option 634-1 MultiProcess, not in MultiMove systems with more than one Spot robot.

## 2.2.3 Spot I/O configuration for MultiMove / MultiProcess systems

### Introduction

If a Spot MultiMove and MultiProcess configuration for more than one robot is selected when building a spot welding system, the system will be prepared with signals for two spot welding equipments on simulated I/O's, one equipment per robot. It is then easy to adapt the configuration to the surrounding equipment, e.g weld timer, media panel etc.

### Default configuration

The I/O configuration is prepared for two weld equipments. A set of customized user modules are also installed if this configuration is selected.

The signals are connected to LOCAL devices by default, and can be changed to any unit type if needed, for example PROFINET, DeviceNet, and so on.

**Note**

Some of the signals are only used in a Spot Pneumatic configuration.

**Note**

Signal names for gun equipment 2 are the same as for gun 1 but with the ending G2, e.g. doStartWeldG2.

### Predefined I/O devices

There are five predefined I/O devices if a MultiMove / MultiProcess configuration is selected:

- One local device, named *SW_TIMER1*, with signals for weld timer1 (Robot 1).
- One local device, named *SW_BOARD1*, with signals for media panel and gun1 (Robot 1).
- One local device, named *SW_TIMER2*, with signals for weld timer2 (Robot 2).
- One local device, named *SW_BOARD2*, with signals for media panel and gun2 (Robot 2).
- One local device, named *SW_SIM_BOARD* with some internal or normally not connected signals.

The devices are configured on a virtual network by default.

### Weld timer signals

| Name | Type | Information |
|---|---|---|
| diWeldComplete | input | Weld complete signal from the weld timer. |
| diWeldFault | input | Fault signal from the weld timer. If this signal is activated during the weld process the weld error handling in Spot is started without waiting for weld time out. |

*Continues on next page*

| Name | Type | Information |
|---|---|---|
| diTimerReady | input | The timer is ready to weld. |
| doTimerOn | output | 24V safety signal cross connected to motors on, can be used to turn off the timer logic in motor off state. |
| doStartWeld | output | Start signal to the weld timer. |
| doEnableCurrent | output | Signal used for the weld simulation function (simtype = 2). See *Simulation modes on page 72*. |
| doResetFault | output | Reset signal. Can be used to reset the welding controller after a weld error. The signal is pulsed with a user defined pulse length before manual or automatic rewelding. |
| goWeldProgram | output group | Selected weld program number in weld timer, see *spotdata - Spot weld data on page 140*. Default size is 8 bits, 0 - 255, 256 different programs. The maximum group size can be configured up to 32 bits, see*spotdata - Spot weld data on page 140* |
| doNewProgram | output | This signal can be used as handshaking to let the timer know a new program has been selected in the *goWeldProgram* group.<br>**Note**<br>For some timers this signal must be set after a new program number is set in *goWeldProgram* in order for the timer to set the corresponding inputs *giGunForce*, *giPlateThickness*, and *giPlateTolerance*. |
| diProgSelectValid | input | This signal can be used as handshaking to let spot know that a valid program has been read by the timer and to continue the execution and read the timer input groups, force, thickness, etc.<br>**Note**<br>Some timers has the possibility to check if a valid weld program selection has been done.<br>Valid program = 1, Not valid = 0<br>This signal is used by spot. If not set, a program valid timeout will occur during execution. |
| giGunForce | input group | Gun force from weld timer if `tip_force` in `spotdata` is -1. |
| giPlateThickness | input group | Plate thickness from weld timer if `plate_thickness` in `spotdata` is -1. |
| giPlateTolerance | input group | Plate tolerance from weld timer if `plate_tolerance` in `spotdata` is -1. |
| giTimerStatus | input group | Weld timer status code. |

### Gun and media signals

| Name | Type | Information |
|---|---|---|
| diGunOpen | input | Signal indicating that the pneumatic gun is opened. |
| diHighLiftOpen | input | Signal indicating that the pneumatic gun has reached the highlift position. |

*Continues on next page*

2.2.3 Spot I/O configuration for MultiMove / MultiProcess systems
*Continued*

| Name | Type | Information |
|---|---|---|
| diPressureOk | input | Signal indicating that the right gun pressure is reached for a pneumatic gun. |
| diTrafoTempOk | input | Signal indicating that the temperature is too high. |
| diWaterFlow1Ok | input | Signal that can be used to indicate problems with the water supply in pipe 1. |
| diWaterFlow2Ok | input | Signal that can be used to indicate problems with the water supply in pipe 2. |
| diAirOk | input | Signal indicating low air pressure in the equalize cylinder. |
| diWeldContact | input | Signal indicating the state of the weld contactor.<br>(0 = deactivated) |
| diEquipmentOk | input | Signal indicating the total gun status. A number of input signals from the gun is cross connected to this signal. |
| doEqualize | output | Gun equalizing signal. |
| doCloseGun | output | Gun close signal for a the pneumatic gun. |
| doOpenHighLift | output | Signal used to open the pneumatic gun to the highlift position. |
| doCloseHighLift | output | Signal used to close the pneumatic gun from the highlift position. |
| doStartWater | output | Signal used to activate the water cooling system.<br>This signal is set depending on status of several cross-connected signals.<br>See *Cross-connected signals on page 48*. |
| doWeldPowerContact | output | Signal used to pull the weld contactor.<br>This signal is set depending on status of several cross-connected signals<br>See *Cross-connected signals on page 48*. |

### Process status signals

| Name | Type | Information |
|---|---|---|
| doProcessRun | output | Is set at motion start and is reset when the weld process is ready and motion is released. |
| doProcessFault | output | Is set when an error situation occurs and the process is interrupted. |

### Other signals

| Name | Type | Information |
|---|---|---|
| force_complete | input | Can be used to interrupt the `SetForce` instruction before the programmed force time is elapsed for a servo gun. |
| reweld_proc | input | Can be used to answer a weld error dialog on the FlexPendant with an input signal. The same as tapping **Reweld**. |
| skip_proc | input | Can be used to answer an error dialog on the FlexPendant with an input signal. The same as tapping **Skip**. Only in manual operating mode. |

*Continues on next page*

### Cross-connected signals

| Resultant | Activator(s) | Information |
|---|---|---|
| doStartWater | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Water start signal, if any of the activators are not set the water start signal will be reset. |
| doWeldPower-Contact | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset. |
| diEquipmentOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok AND<br>diAirOk AND<br>diTempOk AND<br>diTimerReady | Equipment status signal, if any of the activators are not set the equipment ok signal will be reset. |
| diWaterOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok | Water status signal, if any of the activators are not set the water ok signal will be reset. |
| ext_override | skip_proc AND<br>reweld_proc | Reset signal. Can be used to reset the operator error dialog on the FlexPendant when an error occurs.<br>See *Other signals on page 48*. |

## 2.3 Spot Weld timer configuration options

### 2.3.1 782-1 Bosch DeviceNet AC and 782-7 Bosch DeviceNet MFDC

**Introduction**

If either of the additional spot weld timer options *782-1 Bosch DeviceNet AC* or *782-7 Bosch DeviceNet MFDC* are selected when building a spot welding system, the system will be prepared with signals for one spot welding equipment.

**Default configuration**

The default configuration is for one spot welding equipment. A set of customized user modules are also installed if this option is selected.

**Note**

Some of the signals are only used in a Spot Pneumatic configuration.

**Note**

This option requires the option *DeviceNet*, see *Application manual - DeviceNet Master/Slave*.

**Predefined I/O devices**

There are three pre-defined I/O devices configured by default:

- One DeviceNet device, named *SW_BOARD*, with signals for gun equipment, media panel etc. This device is configured on DeviceNet address 10 by default.
- One DeviceNet device, named *BOSCH_TIMER*, with signals for the weld timer. This device is configured on DeviceNet address 21 by default.
- One simulated device, named *SW_SIM_BOARD* with some internal or normally not connected signals.

**Weld timer signals**

| Name | Type | Information |
|---|---|---|
| diWeldComplete | input | Weld complete signal from the weld timer. |
| diWeldFault | input | Fault signal from the weld timer. If this signal is activated during the weld process the weld error handling in Spot is started without waiting for weld time out. |
| diTimerReady | input | The timer is ready to weld. |
| doTimerOn | output | 24V safety signal cross connected to motors on, used to turn off the timer logic in motor off state. |
| doStartWeld | output | Start signal to the weld timer. |
| doEnableCurrent | output | Signal used for the weld simulation function (simtype = 2). See *Simulation modes on page 72*. |

*Continues on next page*

| Name | Type | Information |
|---|---|---|
| doResetFault | output | Reset signal. Can be used to reset the welding controller after a weld error. The signal is pulsed with a user defined pulse length before manual or automatic rewelding. |
| goWeldProgram | output group | Selected weld program number in weld timer, see *spotdata - Spot weld data on page 140*. Default size is 8 bits, 0 - 255, 256 different programs. It is possible to use up to 20 bits for this timer. |
| doNewProgram | output | This signal is used as handshaking to let the timer know a new program has been selected by the robot.<br>**Note**<br>This signal must be set after a new program number is set in *goWeldProgram* in order for the timer to set the corresponding inputs *giGunForce*, *giPlateThickness*, and *giPlateTolerance*. |
| diProgSelectValid | input | This signal is used as handshaking to let spot know that a valid program has been read by the timer and to continue the execution.<br>**Note**<br>For this timer type the valid program selection is not implemented, but the signal is used by spot. If this signal is not set, a program valid timeout will occur during execution.<br>Valid program = 1, Not valid = 0 |
| diProgComplete | input | Signal used to let the robot know that a new program has been selected. Will be set just after the *doNewProgram* has been set.<br>**Note**<br>This signal will be set by the timer when a new program selection has been done, and it is cross-connected to the *diProgSelectValid* signal used to check if a valid weld program is selected. |
| giGunForce | input group | Gun force from weld timer if `tip_force` in `spotdata` is -1. |
| giPlateThickness | input group | Plate thickness from weld timer if `plate_thickness` in `spotdata` is -1. |
| giPlateTolerance | input group | Plate tolerance from weld timer if `plate_tolerance` in `spotdata` is -1. |
| giTimerStatus | input group | Weld timer status code. |

### Gun and media signals

| Name | Type | Information |
|---|---|---|
| diGunOpen | input | Signal indicating that a pneumatic gun is opened. |
| diHighLiftOpen | input | Signal indicating that a pneumatic gun has reached the highlift position. |

*Continues on next page*

2.3.1 782-1 Bosch DeviceNet AC and 782-7 Bosch DeviceNet MFDC
*Continued*

| Name | Type | Information |
|---|---|---|
| diPressureOk | input | Signal indicating that the right gun pressure is reached for a pneumatic gun. |
| diTrafoTempOk | input | Signal indicating that the temperature is too high. |
| diWaterFlow1Ok | input | Signal that can be used to indicate problems with the water supply in pipe 1. |
| diWaterFlow2Ok | input | Signal that can be used to indicate problems with the water supply in pipe 2. |
| diAirOk | input | Signal indicating low air pressure in the equalize cylinder. |
| diWeldContact | input | Signal indicating the state of the weld contactor.<br>(0 = deactivated) |
| diEquipmentOk | input | Signal indicating the total gun status. A number of input signals from the gun is cross connected to this signal. |
| doEqualize | output | Gun equalizing signal if mechanical equalizing system are used. |
| doCloseGun | output | Gun close signal for a a pneumatic gun. |
| doOpenHighLift | output | Signal used to open a pneumatic gun to the highlift position. |
| doCloseHighLift | output | Signal used to close a pneumatic gun from the highlift position. |
| doStartWater | output | Signal used to activate the water cooling system.<br>This signal is set depending on status of several cross-connected signals.<br>See *Cross-connected signals on page 48*. |
| doWeldPowerContact | output | Signal used to pull the weld contactor.<br>This signal is set depending on status of several cross-connected signals<br>See *Cross-connected signals on page 48*. |

### Process status signals

| Name | Type | Information |
|---|---|---|
| doProcessRun | output | Is set at motion start and is reset when the weld process is ready and motion is released. |
| doProcessFault | output | Is set when an error situation occurs and the process is interrupted. |

### Other signals

| Name | Type | Information |
|---|---|---|
| force_complete | input | Can be used to interrupt the `SetForce` instruction before the programmed force time is elapsed for a servo gun. |
| reweld_proc | input | Can be used to answer a weld error dialog on the FlexPendant with an input signal. The same as tapping **Reweld**. |
| skip_proc | input | Can be used to answer an error dialog on the FlexPendant with an input signal. The same as tapping **Skip**. Only in manual operating mode. |

*Continues on next page*

### Cross-connected signals

| Resultant | Activator(s) | Information |
|---|---|---|
| doStartWater | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Water start signal, if any of the activators are not set the water start signal will be reset. |
| doWeldPower-Contact | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset. |
| diEquipmentOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok AND<br>diAirOk AND<br>diTempOk AND<br>diTimerReady | Equipment status signal, if any of the activators are not set the equipment ok signal will be reset. |
| diWaterOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok | Water status signal, if any of the activators are not set the water ok signal will be reset. |
| ext_override | skip_proc AND<br>reweld_proc | Reset signal. Can be used to reset the operator error dialog on the FlexPendant when an error occur.<br>See *Other signals on page 48*. |
| diProgSelectValid | diProgComplete | Valid program selection signal, signal used to check if a valid weld program is selected in the timer. |

### Limitations

This configuration is only availible for single robot Spot systems, not in MultiProcess systems with more than one Spot robot.

## 2.3.2 782-11 Bosch PROFINET MFDC

### Introduction

If the additional spot weld timer option *782-11 Bosch PROFINET MFDC* is selected when building a spot welding system, the system will be prepared with signals for one spot welding equipment.

### Default configuration

The default configuration is for one spot welding equipment. A set of customized user modules are also installed if this option is selected.

**Note**

Some of the signals are only used in a Spot Pneumatic configuration.

**Note**

This option requires the option *PROFINET Master and Slave*, see *Application manual - PROFINET Controller/Device*.

### Predefined PROFINET bus and I/O devices

The pre-installed PROFINET I/O bus *Profinet1* will be configured on address 192.168.5.1 by default if this option is selected, and the necessary network configuration file `IPPNIO.xml` is copied to the home directory of the system.

**Tip**

The current KW-Software PROFINET Configurator project is available in the Spot option utility directory in the RobotWare installation. This can be use if there is a need to add new units or modify the configuration, see:

*...\RobotPackages\RobotWare_RPK_<version>\utility\Spot\BoschPnet\KWPnetProj*.

Navigate to the RobotWare installation folder from the RobotStudio **Add-Ins** tab, by right-clicking on the installed RobotWare version in the **Add-Ins** browser and selecting **Open Package Folder**.

For more information about PROFINET configuration, see *Application manual - PROFINET Controller/Device*.

There are two predefined PROFINET I/O devices and one virtual device configured by default:

- One PROFINET device, named *sw_board*, with signals for gun equipment, media panel etc. This device is configured on address *192.168.5.5* by default.
- One PROFINET device, named *bosch_timer*, with signals for the weld timer. This device is configured on address *192.168.5.7* by default.
- One simulated device, named *sw_sim_board* with some internal or normally not connected signals.

*Continues on next page*

### Weld timer signals

| Name | Type | Information |
|---|---|---|
| diWeldComplete | input | Weld complete signal from the weld timer. |
| diWeldFault | input | Fault signal from the weld timer. If this signal is activated during the weld process the weld error handling inside Spot is started without waiting for weld time out. |
| diTimerReady | input | The timer is ready to weld. |
| doTimerOn | output | 24V safety signal cross connected to motors on, used to turn off the timer logic in motor off state. |
| doStartWeld | output | Start signal to the weld timer. |
| doEnableCurrent | output | Signal used for the weld simulation function (simtype = 2). See *Simulation modes on page 72*. |
| doResetFault | output | Reset signal. Can be used to reset the welding controller after a weld error. The signal is pulsed with a user defined pulse length before manual or automatic rewelding. |
| goWeldProgram | output group | Selected weld program number in weld timer, see *spotdata - Spot weld data on page 140*. Default size is 8 bits, 0 - 255, 256 different programs. It is possible to use up to 20 bits for this timer |
| doNewProgram | output | This signal can be used as handshaking to let the timer know a new program has been selected.<br>**Note**<br>This timer requires this signal to be set after a new program number is set in *goWeldProgram* in order for the timer to set the corresponding inputs *giGunForce*, *giPlateThickness*, and *giPlateTolerance*. |
| diProgSelectValid | input | This signal can be used as handshaking to let spot know that a valid program has been read by the timer.<br>**Note**<br>This timer has the possibility to check if a valid weld program selection has been done.<br>Valid program = 1, Not valid = 0<br>This signal is used by spot. If not set, a program valid timeout will occur during execution. |
| giGunForce | input group | Gun force from weld timer if `tip_force` in `spotdata` is -1. |
| giPlateThickness | input group | Plate thickness from weld timer if `plate_thickness` in `spotdata` is -1. |
| | | |
| giTimerStatus | input group | Weld timer status code. |

### Gun and media signals

| Name | Type | Information |
|---|---|---|
| diGunOpen | input | Signal indicating that a pneumatic gun is opened. |
| diHighLiftOpen | input | Signal indicating that a pneumatic gun has reached the highlift position. |

*Continues on next page*

2.3.2 782-11 Bosch PROFINET MFDC
*Continued*

| Name | Type | Information |
|---|---|---|
| diPressureOk | input | Signal indicating that the right gun pressure is reached for a pneumatic gun. |
| diTrafoTempOk | input | Signal indicating that the temperature is too high. |
| diWaterFlow1Ok | input | Signal that can be used to indicate problems with the water supply in pipe 1. |
| diWaterFlow2Ok | input | Signal that can be used to indicate problems with the water supply in pipe 2. |
| diAirOk | input | Signal indicating low air pressure in the equalize cylinder. |
| diWeldContact | input | Signal indicating the state of the weld contactor.<br>(0 = deactivated) |
| diEquipmentOk | input | Signal indicating the total gun status. A number of input signals from the gun is cross connected to this signal. |
| doEqualize | output | Gun equalizing signal if mechanical equalizing system are used. |
| doCloseGun | output | Gun close signal for a pneumatic gun. |
| doOpenHighLift | output | Signal used to open a pneumatic gun to the highlift position. |
| doCloseHighLift | output | Signal used to close a pneumatic gun from the highlift position. |
| doStartWater | output | Signal used to activate the water cooling system.<br>This signal is set depending on status of several cross-connected signals.<br>See *Cross-connected signals on page 48*. |
| doWeldPowerContact | output | Signal used to pull the weld contactor.<br>This signal is set depending on status of several cross-connected signals<br>See *Cross-connected signals on page 48*. |

### Process status signals

| Name | Type | Information |
|---|---|---|
| doProcessRun | output | Is set at motion start and is reset when the weld process is ready and motion is released. |
| doProcessFault | output | Is set when an error situation occurs and the process is interrupted. |

### Other signals

| Name | Type | Information |
|---|---|---|
| force_complete | input | Can be used to interrupt the `SetForce` instruction before the programmed force time is elapsed for a servo gun. |
| reweld_proc | input | Can be used to answer a weld error dialog on the FlexPendant with an input signal. The same as tapping **Reweld**. |
| skip_proc | input | Can be used to answer an error dialog on the FlexPendant with an input signal. The same as tapping **Skip**. Only in manual operating mode. |

*Continues on next page*

### Cross-connected signals

| Resultant | Activator(s) | Information |
|---|---|---|
| doStartWater | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Water start signal, if any of the activators are not set the water start signal will be reset. |
| doWeldPower-Contact | motor_on AND<br>doEnableCurrent AND<br>doProcessFault (Inverted) | Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset. |
| diEquipmentOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok AND<br>diAirOk AND<br>diTempOk AND<br>diTimerReady | Equipment status signal, if any of the activators are not set the equipment ok signal will be reset. |
| diWaterOk | diWaterFlow1Ok AND<br>diWaterFlow2Ok | Water status signal, if any of the activators are not set the water ok signal will be reset. |
| ext_override | skip_proc AND<br>reweld_proc | Reset signal. Can be used to reset the operator error dialog on the FlexPendant when an error occur.<br>See *Other signals on page 48*. |

### Limitations

This configuration is only availible for single robot Spot systems, not in MultiProcess systems with more than one Spot robot.

This page is intentionally left blank

# 3 Programming

## Introduction to programming

This chapter describes the basic functions and steps to take when creating, testing, and running spot weld programs with the Spot options.

It is assumed that a servo gun is installed and tuned at this stage. If not, see *Servo gun motion control on page 195*, and *Application manual - Servo gun tuning*.

*Continues on next page*

## 3.1 Quick start for servo gun

**Install servo gun parameters**

If the system is cold started, the servo gun parameters are probably not loaded. See *Install servo gun parameters on page 202*

**Set the servo gun name**

After the gun parameters are installed and the system is restarted, the `gundata` needs to be updated with the servo gun name (mechanical unit name) so the spot instructions will work correctly. See *Set the servo gun name on page 203*.

**Servo gun force calibration**

To protect the gun from to high forces there is a RAPID service routine to calibrate the motor torque vs tip force characteristics, `ManualForceCalib`. See *Servo gun force calibration on page 203*.

**Servo gun init calibration**

Before running any spot instructions, the gun must be synchronized by a fine calibration. Apart from other kinds of additional axes, this action also requires running a RAPID service routine,
**`ManualServiceCalib`** to initialize or synchronize the gun position. See *Servo gun init calibration on page 205*.

## 3.2 Spot weld instructions and data

### Defining spot welding data

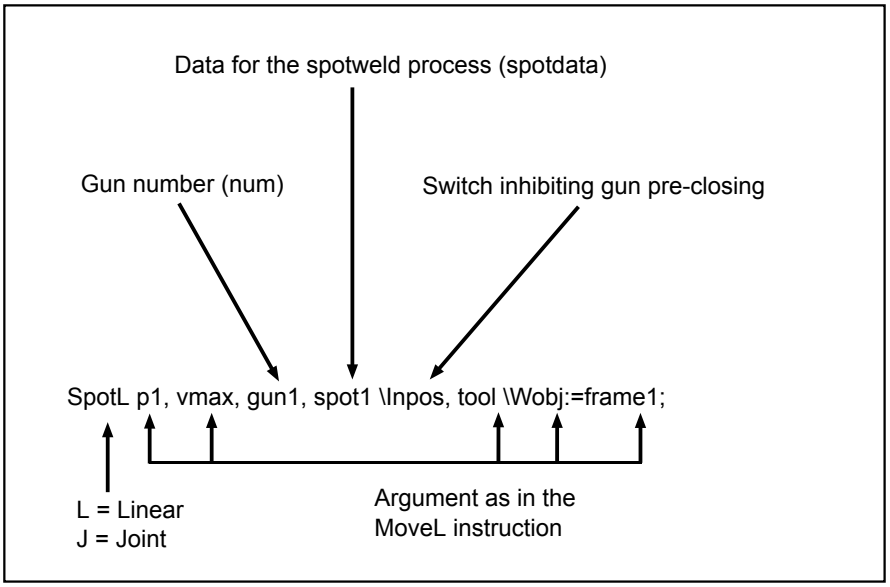
Before starting to program the instructions, define the spot welding data to be used. This data is divided into two types:

- `spotdata`; describes the spot welding process specific data for a specific spot. See *spotdata - Spot weld data on page 140*.
- `gundata`; describes spot welding gun specific data, weld counters, tip wear data etc. All gun equipment used are defined in the `gundata` array `curr_gundata` in SWUSER. See *gundata - Equipment specific weld data on page 135*.

### Spot weld instructions for sequential welding

`SpotL` and `SpotJ` are the basic spot welding instructions in the Spot options. The instructions includes a movement to the weld position and performing the desired weld process. They contains basically the same type of information as a positioning instruction, but also arguments that serve as data for the spot welding process. These instructions are used for welding with one gun or welding with several guns in sequence.

For further details, see section *SpotL/SpotJ - The basic spot welding instructions on page 95*.



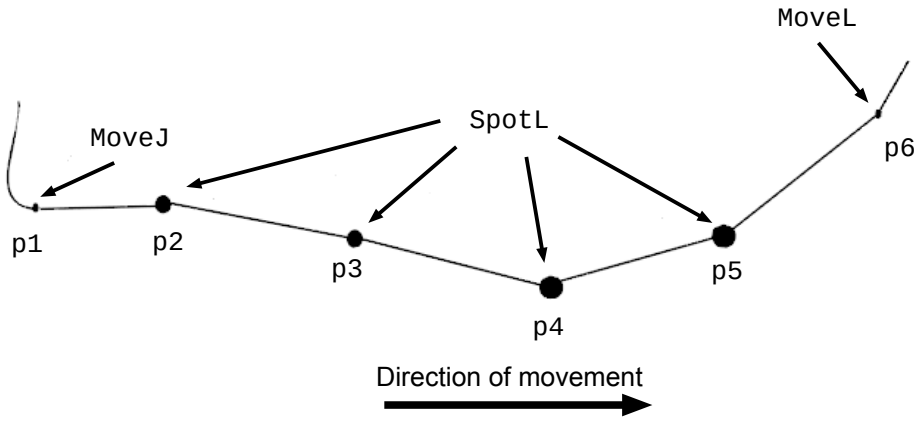

en1200000250

*Continues on next page*

### Programming example for one servo gun

In this example a single servo gun (gun1) is used, held by the robot. Four spots are to be welded with two different `spotdata` used, *spot10* and *spot20*. The data is created in advance. The gun parameters are set in the first `gundata` in the *curr_gundata* array located in the SWUSER module.

The robtargets p2 and p3 will be welded with weld program number 10 and the tip force 2000N. Thickness is set to 1.6mm and the supervision tolerance is set to 0.5mm. If the tolerance is exeeded the execution will be stopped with an operator dialog. The next two robtargets p4, p5 will be welded with weld program number 20 and the selected tip force is set to 3000N, the plate thickness 3mm, will not be supervised since the tolerance is set to 0.




xx1200000245

```
spot10
  prog_num = 10
  tip_force = 2000
  plate_thickness = 1.6
  plate_tolerance = 0.5

spot20
  prog_num = 20
  tip_force = 3000
  plate_thickness = 3
  plate_tolerance = 0
```

```
curr_gundata{1}
  gun_name = "SGUN_1"
  weld_counter = 112
  max_nof_welds = 1000
  curr_tip_wear = 5.2
  max_tip_wear = 8
```

**RAPID code sequence:**

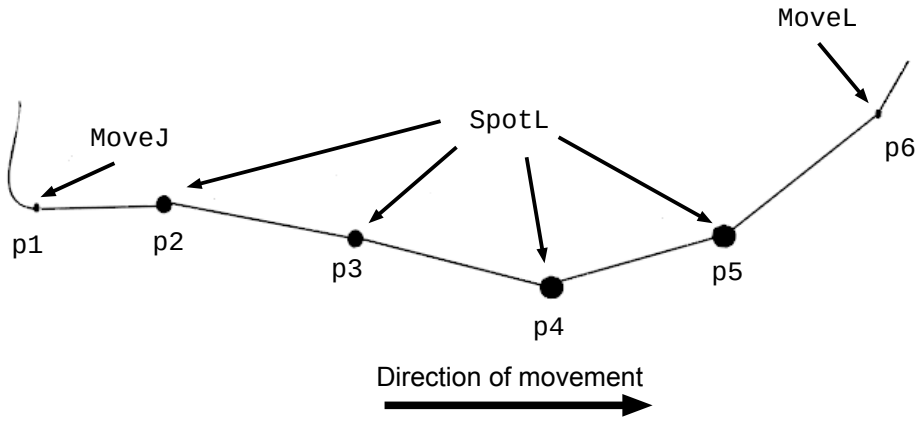
```
MoveJ p1, v600, z50, toolGun1;
SpotL p2, vmax, gun1, spot10, toolGun1;
SpotL p3, vmax, gun1, spot10, toolGun1;
SpotL p4, vmax, gun1, spot20, toolGun1;
SpotL p5, vmax, gun1, spot20, toolGun1;
MoveL p6, v600, z50, toolGun1;
```

*Continues on next page*

### Programming example for one pneumatic gun

Same example as above but with parameters for a pneumatic gun.

The robtargets p2 and p3 will be welded with weld program number 10 and the tip force output group is set to 2. The next two robtargets p4, p5 will be welded with weld program number 20 and the selected tip force output group is set to 3 It is more common to control the the tip force from the weld timer, and in those cases the tip force parameter can be ignored or removed. See *How to change the Spot data types on page 255*.




xx1200000245

```
spot10
  prog_num = 10
  tip_force = 2
spot20
  prog_num = 20
  tip_force = 3
```

```
curr_gundata{}
  gun_name = "PNEU_G1"
  weld_counter = 112
  max_nof_welds = 1000
```

**RAPID code sequence:**

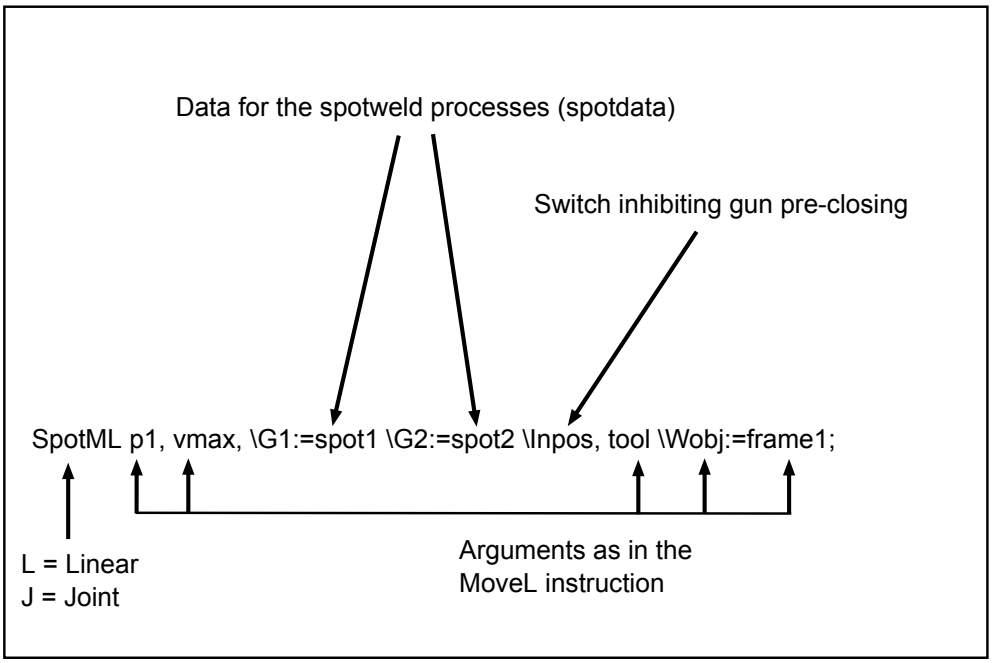
```
MoveJ p1, v600, z50, toolGun1;
SpotL p2, vmax, gun1, spot10, toolGun1;
SpotL p3, vmax, gun1, spot10, toolGun1;
SpotL p4, vmax, gun1, spot20, toolGun1;
SpotL p5, vmax, gun1, spot20, toolGun1;
MoveL p6, v600, z50, toolGun1;
```

*Continues on next page*

### Spot weld instructions for simultaneous welding with multiple guns

`SpotML` and `SpotMJ` has to be used if welding with several guns at the same time is desired. It is possible to use four guns simultaneously. The instruction includes a movement to the weld position and performing the desired weld processes. It contains basically the same type of information as a positioning instruction but also arguments that serve as data for the different spot welding processes.

See *SpotML/SpotMJ - Spot welding with multiple guns on page 100*.




en1200000251

### Programming example for two servo guns

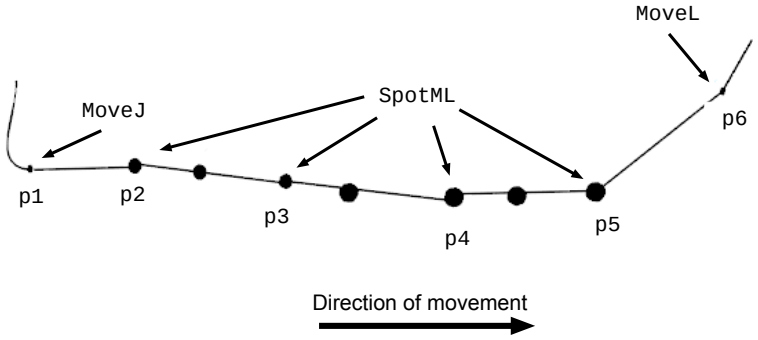
In this example two different stationary guns are used, mounted close to each other. The robot is holding the work piece. Seven spots are to be welded with two different `spotdata` used, `spot10` and `spot20`. Current gun parameters has been set up in the first and second `gundata` in the `curr_gundata` array in SWUSER.

The robtarget p2 will be welded with gun1(SGUN_1) with weld program number 10 and the tip force 2000N. Thickness is set to 1.6mm and the supervision tolerance is set to 0.5mm. If the tolerance is exeeded the execution will be stopped with an operator dialog.

The next robtargets p3 and p4 will also be welded with gun 1 but also at the same time with gun 2(SGUN_2) with weld program number 20 and the selected tip force

*Continues on next page*

is set to 3000N. No thickness supervion. The robtarget p5 will be welded with just gun 2 also with weld program 20.




xx1200000241

```
spot10                          curr_gundata{1}
  prog_num = 10                   gun_name = "SGUN_1"
  tip_force = 2000                weld_counter = 112
  plate_thickness = 1.6           max_nof_welds = 1000
  plate_tolerance = 0.5           curr_tip_wear = 5.2
                                  max_tip_wear = 8
spot20                          curr_gundata{2}
  prog_num = 20                   gun_name = "SGUN_2"
  tip_force = 3000                weld_counter = 345
  plate_thickness = 3             max_nof_welds = 1000
  plate_tolerance = 0             curr_tip_wear = 3.4
                                  max_tip_wear = 11
```

RAPID code sequence:

```
MoveJ p1, v600, z50, toolGrip1\Wobj:= frame1;
SpotML p2, vmax\G1:=spot10,toolGrip1\Wobj:= frame1;
SpotML p3, vmax\G1:=spot20\G2:=spot20,toolGrip1\Wobj:= frame1;
SpotML p4, vmax\G1:=spot20\G2:=spot20,toolGrip1\Wobj:= frame1;
SpotML p5, vmax\G2:=spot20,toolGrip1\Wobj:= frame1;
MoveL p6, v600, z50, toolGrip1\Wobj:= frame1;
```

### Programming example for two pneumatic guns

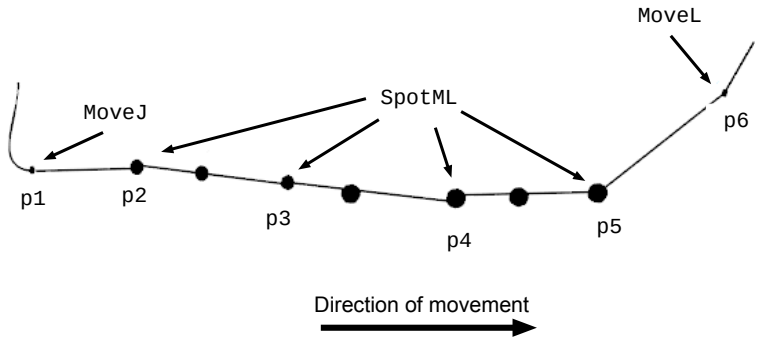
Same example as above but with parameters for pneumatic guns.

The robtarget p2 will be welded with gun1(PNEU_G1) with weld program number 10 and the tip force output group is set to 2.

The next robtargets p3 and p4 will also be welded with gun 1 but also at the same time with gun 2(PNEU_G2) with weld program number 20 and the selected tip force output group is set to 3. The robtarget p5 will be welded with only gun 2 also with weld program 20 and the tip force output group is set to 3. It is more common to control the the tip force from the weld timer, and in those cases the tip force

*Continues on next page*

parameter can be ignore or removed. See *How to change the Spot data types on page 255*.




xx1200000241

```
spot10                          curr_gundata{1}
  prog_num = 10                   gun_name = "PNEU_G1"
  tip_force = 2                   weld_counter = 112
                                  max_nof_welds = 1000
spot20                          curr_gundata{2}
  prog_num = 20                   gun_name = "PNEU_G2"
  tip_force = 3                   weld_counter = 215
                                  max_nof_welds = 1000
```

**RAPID code sequence:**

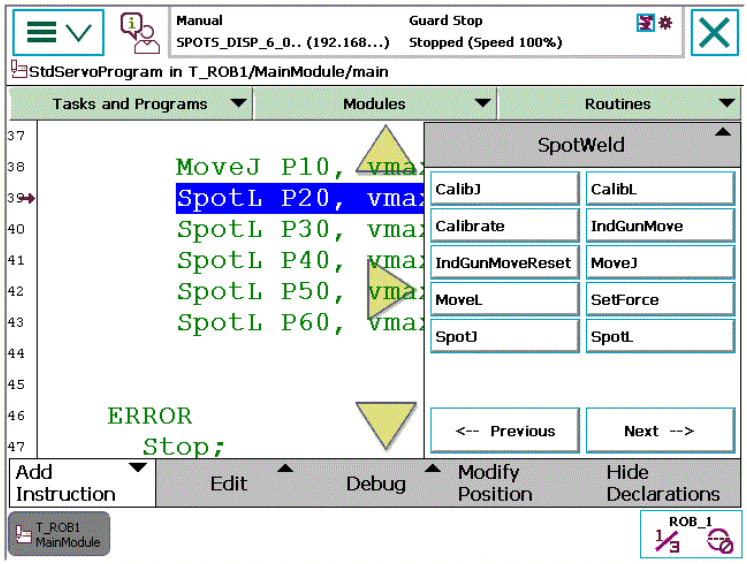
```
MoveJ p1, v600, z50, toolGrip1\Wobj:= frame1;
SpotML p2, vmax\G1:=spot10,toolGrip1\Wobj:= frame1;
SpotML p3, vmax\G1:=spot10\G2:=spot20, toolGrip1\Wobj:= frame1;
SpotML p4, vmax\G1:=spot20\G2:=spot20, toolGrip1\Wobj:= frame1;
SpotML p5, vmax\G2:=spot20, toolGrip1\Wobj:= frame1;
MoveL p6, v600, z50, toolGrip1\Wobj:= frame1;
```

**Programming spot welding instructions**

1. Jog the robot to the desired destination position and jog also the gun axis to desired preclose tip position (Only for servo guns).
2. In the Program Editor, tap **Add instruction**, and then select **SpotWeld** from the list of instructions.
3. Select the instruction **SpotL** or **SpotJ**.

   The instruction will be added directly to the program. The arguments are set in relation to the last programmed spot welding instruction.
4. Change the optional arguments if needed.
5. Jog the robot to another position and add more spot weld instructions the same way.

*Continues on next page*

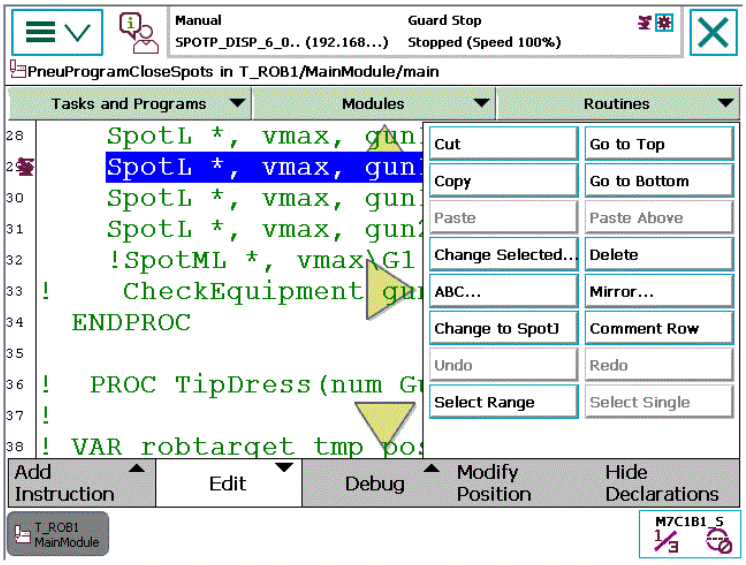



en1200000246

## Edit current used spotdata

1 Select current `spotdata` in the instruction.
2 Tap **Debug**, and then tap **View Value**.
3 Change the value.
4 Tap **OK**.

§




en1200000243

## Changing to another spotdata

1 Select current `spotdata` in the instruction.
2 Tap **Edit**, and then tap **Change Selected**.
3 Select a `spotdata` from the list of available spotdata.
4 Tap **OK**.

## 3.3 Testing spot weld instructions in simulated mode

### Simulation modes

To prevent the spot welding process executing during programming and testing, it is possible to run the program in different simulation modes. For more information about simulation modes see *Simulation modes on page 80*.

**Tip**

The fastest way to change the simulation mode is to use the FlexPendant Interface, see Simulation *Main View on page 217*.

## 3.4 Gun control

**Preclosing of gun**

The spot welding instructions have a built-in preclosing of the weld guns, that is when approaching the position the guns will start to close in advance to save time. For more information about gun control, see *Gun closing and pre closing time on page 77*.

**Mechanical gun equalizing**

The spot welding instructions have a function for equalizing with mechanical equalizing systems in the gun, to minimize the impact on the plates during the welding. See *Gun equalizing on page 76*.

**Software equalizing**

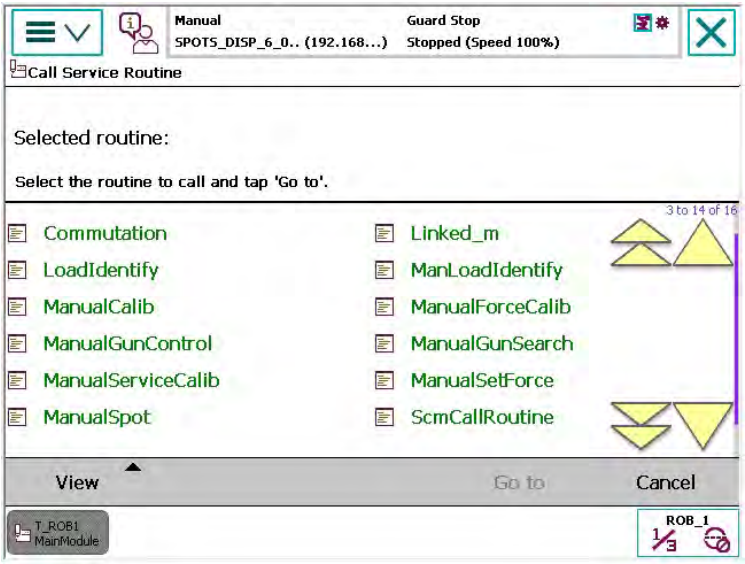
The spot welding instructions SpotL and SpotJ also has functions that make it possible to use spot welding guns **without** mechanical equalizing systems. These functions are available if the configuration *Spot Servo Equalizing* is selected. For more information see *Software Equalizing on page 169*.

## 3.5 Manual actions

### Service routines

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

- From the Spot GUI application, select **RobotWare Spot** and **Manual Actions**.
- From the Program Editor, tap **Debug**, and then tap **Call Service Routine**




en1200000242

**Tip**

It is also possible to access and run the service routines from the FlexPendant interface, see *Manual Actions on page 226*

### Available service routines

The following service routines are available in the Spot Options:

| Service routine | Description |
|---|---|
| ManualGunSearch | This routine will search the system for available servo guns and add their names to gun name in current `gundata`. |
| ManualGunControl | This routine will close or open the gun according to data in `curr_forcedata`. The gun equalize signal is also activated/deactivated. |
| ManGunPosition | This routine will open or close the pneumatic gun to the large stroke or work stroke position. |
| ManualSpot | This routine will perform a weld in current position according to data in `curr_spotdata`. |
| ManualSetForce | This routine will perform a `SetForce` action according to data in `curr_forcedata`. The gun equalize signal is also activated/deactivated. |

*Continues on next page*

| Service routine | Description |
|---|---|
| ManualCalib | This routine will perform a calibration of the servo gun, 1 Tool Change, 2 Tip Change or 3 Tip Wear calibration.<br>• Option 1. Tool Change calibration, used after changing tool if using more than one gun.<br>• Option 2: Tip Change calibration, used after worn tips has been replaced with new tips.<br>• Option 3: Tip Wear calibration, used after the tip has been dressed.<br>See *Tip management on page 200*. |
| ManualForceCalib | This routine will perform a force calibration of the servo gun. 2 - 10 forces and positions can be stored. See *Servo gun force calibration on page 203*.<br>From this routine is also possible to store two reference forces can be used to compensate for gravitational influences of gun force during welding. This function can be used if a servo gun loses force when the movable gun arm moves against gravity.<br>For more information, see *Servo gun force gravity compensation on page 204* and *The type Spot Gun Equipment on page 31*. |
| ManualServiceCalib | This routine will perform a gun init calibration of the servo gun, find the zero position.<br>• Option 1: Will synchronize the servo gun without jogging after the revolution counter has been updated.<br>The servo gun will close slowly until it reaches the contact position.<br>• Option 2: Will synchronize the servo gun without jogging after the gun has been fine calibrated.<br>The servo gun will move fast to a predefined pre-position. and then continue to close slowly until it reaches the contact position.<br>See *Servo gun init calibration on page 205*. |
| ManualCheckMeasPos | This routine can be used to verify if a robot position or gun orientation is suitable for a tip wear measurement with MeasureWearL. When this routine is run in the selected position, status information will be presented on the FlexPendant whether the position is suitable or not. The recommended touch up axis should be 4 to 6 and the touch up value should be in range between 0.25 and 1.<br>This instruction is only available if the Spot Servo Equalizing option is installed.<br>**Note**<br>For some special configurations the `MeasureWearL` measuring method is less suitable, for example very large guns and/or when an acceptable touch up position is not possible to reach for some reason. Then the `ReCalcTcp` method should be used instead. |

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

## 3.6 Process sequence and error handling

### Process sequence

This section describes the internal process sequence when a SpotL/J or a SpotML/MJ instruction is executed:

1. Data definition user routines are executed. (eg. DefineSpotData)
2. The weld program number is set. (eg. goWeldProgram)
3. The new program selection signal is set, if configured.
4. The robot and gun starts to move towards the programmed position.
5. The process will check and wait for the program valid signal from the weld controller, if configured.
6. The new program selection signal is reset, if configured (doNewProgram).
7. If valid program selection (diProgSelectValid), the process will read the weld controller groups if configured, (eg. giGunForce).
8. The new program selection signal is reset, if configured (doNewProgram).
9. User routine UpdateSpotData is executed.
10. User routine SwInitUserIO is executed.
11. User routine SwPrepare is executed.
12. User routine SwCloseGun is executed (pneumatic guns).
13. The gun will start to close before the position is reached (unless argument \InPos is used), according to the predefined gun pre closing time.
14. The equalizing signal is set according to the predefined pre equalizing time. (eg. doEqualize).
15. User routine SwPreWeld is executed when the weld position is reached. (Preweld supervision.)
16. The plate thickness is checked. (Servo guns only). The requested gun force is established if OK.
17. The start signal to the weld controller is set. (eg. doStartWeld).
18. The weld controller performs the weld, and can change the gun force during the weld sequence if configured (eg. new value on giGunForce).
19. When the weld complete signal from the weld controller is received, (eg. diWeldComplete) the start signal will be reset and the gun will start to open and the equalizing signal will be reset.
20. User routine SwOpenGun is executed (pneumatic guns).
21. User routine SwPostWeld is executed.
22. The instruction is ready.

### Gun equalizing

When approaching the position a signal is activated to be used for the gun equalizing. The signal is deactivated after the weld process before the next robot motion is released.

*Continues on next page*

The gun pre equalizing time, `Gun pre equalizing time`, is defined for each used gun in `Spot Gun Equipment` process data. See *The type Spot Gun Equipment on page 31*.

**Note**

The gun pre equalizing time in `Spot Gun Equipment` process data, `Gun pre equalizing time`, is not used when SoftWare equalizing is used. For more information see *Software Equalizing on page 169*.

### Gun closing and pre closing time

The spot welding instructions have a built-in preclosing of the weld guns, that is when approaching the position the guns will start to close in advance to save time.

The gun closure is activated at a defined time before the weld position. The gun pre closing time, `Gun pre closing time`, can be defined for each used gun in the `Spot Gun Equipment` process data. See *The type Spot Gun Equipment on page 31*.

For servo guns the movement to the weld position starts with a synchronous phase which means that the servo gun axis is moved synchronized with the robot movement. The gun closing speed is automatically adapted so the contact position is reached at the same time as the robot reaches the programmed weld position. For more information about servo gun motion control, see *Servo gun motion control on page 195*.

**Note**

The data `Gun pre closing time` is not used if *Software Equalizing* is active, see *Software Equalizing on page 169*.

**Note**

The pre closing can be disabled by using the `\InPos` argument in the instruction.

**Note**

If the pre closing time is set to high it can lead to a longer cycle time if close positions are programmed, because the gun movement will synchronize with the robot.

**Note**

If the optional data `\SMEQ` is used in the SpotL/J instruction the robot will be set into a soft state during the gun closing, for more information see *SoftMove Equalizing on page 185*.

*Continues on next page*

**Note**

If using gun open position less than 10mm for servo guns, there may be problems with sporadic "internal servo tool" errors. The reason for this is probably a too hard tuned or a very fast gun.

### Welding

When the welding position is reached the gun starts to build up the gun force and the user hook SwPreWeld is executed, see *Process hooks on page 160*.

The plate thickness is checked (Servo guns only). The weld start signal is set as soon as SwPreWeld is ready and the requested gun force is reached. After ordering weld, the system waits for weld complete from the weld equipment. If configured it is also possible to change the gun force before the weld complete is set. See *How to use spot data programmed in the weld timer on page 258*.

For pneumatic guns the start signal is set as soon as the robot has reached the weld position and a number of supervisions have been acknowledged. The start signal is high during the entire welding period. It is reset either after weld complete or after a predefined timeout time elapsed.

### Gun opening

The gun starts to open to the programmed position after the weld process is finished. At the same time the user hook `SwOpenGun` is executed. When the gun is opened enough and `SwOpenGun` is ready then the movement is released and the robot movement is started.

The gun is also opened to the programmed position after a weld error or in other error situations.

For pneumatic guns the gun opens to a small or large stroke after the welding has finished, depending on the parameter `\OpenHLift`. The opening is supervised in such a way that the gun open signal is expected.

**Note**

The gun opening gap must be large enough that the tips are free from the plates when welding.

So therefore, the software will compensate for the release distance that is used, and the plate thickness, as the opening position is the same as the tips closed with plates + release distance.

Example:

If release_dist is 10mm, the moving tip will open to 10mm even if you modify the position with the gun closed on the plate surface.

A simple recommendation is to have approximately the same distance from the plate to the movable electrode as the release_dist that has been configured.

For more information about the SoftWare Equalizing functionality, see *Software Equalizing on page 169*.

*Continues on next page*

**Program stop and restart**

Stop during the motion and restart

The robot stops on the path. If the gun closure already is started the gun will open to the programmed position.

On restart, the robot continues towards the programmed position, closes the gun again and the sequence in `SpotL/J` carries on as normal.

Stop during welding and restart

The welding is finished, validation is done after the stop and the gun opens.

On restart, the robot continues with next instruction.

**Quick stop and restart**

Quick stop during the motion and restart

The robot stops immediately probably deviated from the path. If the gun closure already is started the gun will open to the programmed or gun open position.

On restart, the robot first moves back to the path, then continues towards the programmed position, closes the gun again and the sequence in `SpotL/J` and `SpotML/MJ` carries on as normal.

Quick stop during welding and restart

The weld process is interrupted. The gun is still closed but the gun force will be reduced. (Servo guns only).

A pneumatic gun will open in this situation.

On restart, the weld error handling is executed with possibilities to reweld the last spot.

**Power failure handling**

At system restart after power failure:

- All spot welding output signals are set to the old status, except the weld start signal.

At program restart after power failure:

- The robot returns to the path and the program execution which was interrupted is continued.
- If a power failure occurred when a weld process was active, the current spot is automatically rewelded.

**Instruction by instruction execution**

Forward

The instruction is executed in two steps (Recommended setting, Step Mode = Step Over):

1 The robot will move to the weld position, an operator dialog will be shown with instructions on how to continue. After this step it is possible to modify the position if needed. It is possible to weld or skip the current position and move to next instruction.

*Continues on next page*

2 If the step forward button is pressed again, current instruction will be skipped. If start button is pressed the current instruction will be welded.

> **Note**
>
> To perform a weld in this position, the start button must be pressed. Program execution will stop after the current instruction is ready. To restart the program normally, the start button must be pressed again.

Backward

The motion is performed backwards to the programmed position with gun control, but the gun is not closed in the weld position and no weld process is activated. (Servo guns only).

For pneumatic guns the gun is set to work or highlift stroke depending on position of the `\OpenHLift` switch. The motion is performed backwards.

The gun is set to work or highlift stroke depending on the position of the `\CloseHLift` switch.

**Simulation modes**

All active simulation modes are defined in `curr_simdata` in SWUSER. See *simdata - Simulation data on page 146*.

Weld simulation in the robot controller

Activated by setting `sim_type = 1` in *curr_simdata* in SWUSER, simulated welding.

This will inhibit the weld start signal to the timer. The simulated weld time used is the time `sim_time` in *curr_simdata*. In this simulation mode the start signal is never sent to the welding timer.

No preweld supervision is performed, water air etc.

> **Note**
>
> Program valid check and reading of timer input groups will still be done in this mode, tip force, plate thickness etc.

Weld simulation in the timer

Activated by setting `sim_type = 2` in curr_simdata in SWUSER, dry welding.

This will set the enable current signal low to the timer at the next weld, the weld program in the timer will be executed normally, but without current. The timer will perform a "dry weld".

No preweld supervision is performed, water air etc.

Testing without closing the guns

When simulation is active it is also possible to run **without closing** the gun. Ativated by setting `inhib_close` to `TRUE` in *curr_simdata*. This mode can only be used when `sim_type` is set to 1 or 2.

This inhibits the gun closing and opening.

*Continues on next page*

Testing without plates

When simulation is active it is also possible to run without testing plate thickness (servo guns only).
Activated by setting `no_plates` to `TRUE` in *curr_simdata*. This mode can only be used when `sim_type` is set to 1 or 2. (Servo guns only).

This inhibits the plate thickness supervision.

Weld position Touch Up mode

If *Spot Servo Equalizing* is installed it is possible to set `sim_type = 3` to activate the weld position Touch Up function. See *Software Equalizing on page 169*

Activated by setting `sim_type = 3` in `curr_simdata`.

Disable all simulations

All simulations are disabled if `sim_type = 0` in `curr_simdata`.

## Error handling

The following error situations can occur:

- Instruction parameter supervision
- Supervision of valid program selection (If configured)
- Supervision in the beginning of the movement
- Gun closure supervision
- Detection of missing or improper plates (Servo guns only)
- Supervision before weld start
- Weld error
- Supervision after welding
- Gun opening supervision

Instruction parameter supervision

The error occurs when `SpotL/J` or a `SpotML/MJ` is called with faulty parameters.

- The signal `process_fault` for the current equipment is set. The program stops.
- An error message is displayed in a dialog box.
- The error message is logged

The parameter must be changed. When the program is restarted the current instruction is restarted from the beginning.

Supervision of valid program selection

Supervision of valid program selection is done if it is configured (Not default).

**Note**

For more information, see *Spot Weld timer configuration options on page 54*

If an error occurs then:

- The signal `process_fault` for the current equipment is set. The program stops.

*Continues on next page*

- An error message is displayed in a dialog box with retry possibilities.
- The error message is logged.

Supervision in the beginning of the movement

The default supervision checks are executed if configured, and the `SwPrepare` routine is run. See *Process hooks on page 160*.

If an error occurs then:

- The signal `process_fault` for the current equipment is set. The program stops.
- An error message is displayed in a dialog box with retry possibilities.
- The error message is logged.

See *The type Spot Media Equipment on page 36*

Gun closure supervision

The default gun closing sequence are executed if configured, and the `SwCloseGun` routine is run. See *Process hooks on page 160*.

For a pneumatic gun an error occurs if the `gun_open` signal is not set within a certain time.

- The signal `process_fault` the current equipment is set. The program stops.
- An error message is displayed in a dialog box with retry possibilities.
- The error message is logged

See *The type Spot Gun Equipment on page 31*

Detection of missing or improper plates (Servo guns only)

An error will be detected by the process kernel if the plate thickness differ more than the allowed limit, defined by the tolerance, from the programmed thickness.

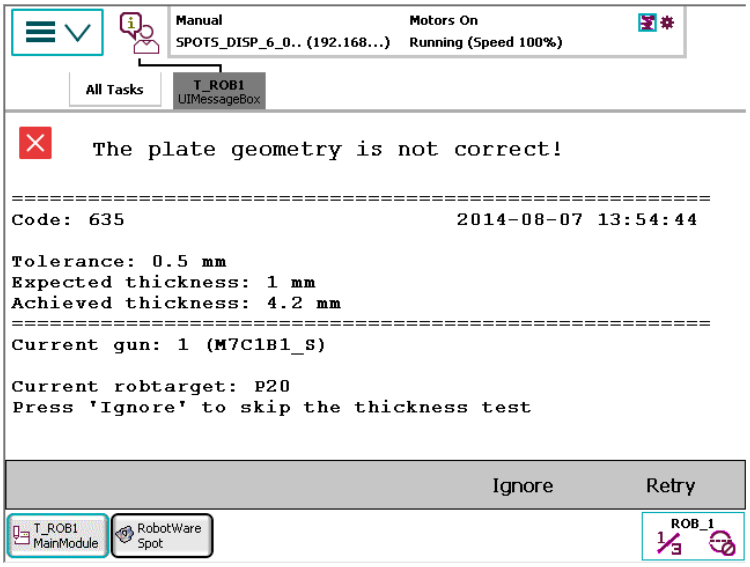
There are three types of errors:

- Negative gun position, one of the tips are missing on the gun, or a tip_wear calibration is needed.
- Missing plates, the plate thickness is smaller than the thickness defined in spotdata.
- Improper geometry, the plate thickness exceeds the tolerance defined in spotdata.

The gun opens.

- The signal `process_fault` for the current equipment is set. The program stops.
- An error message is displayed in a dialog box with retry possibilities.
- The error message is logged.

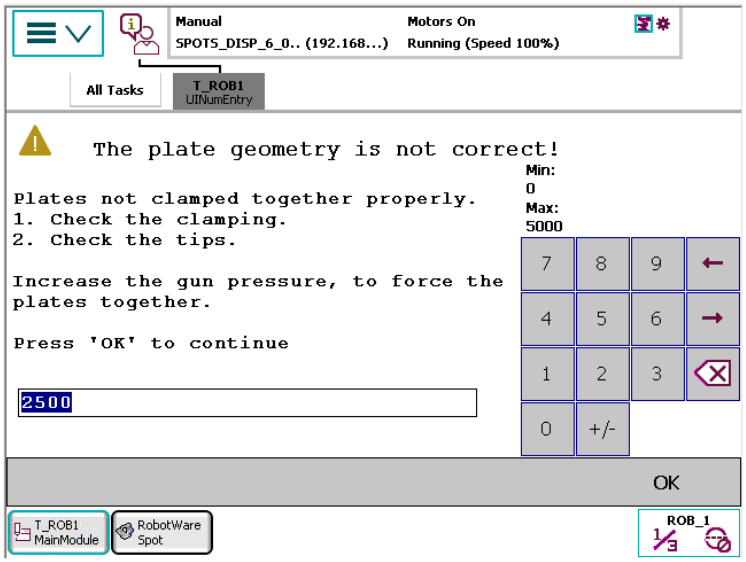
*Continues on next page*




xx1200000217

| | |
|---|---|
| Ignore | Close the gun again but without thickness detection and continue the execution. |
| Retry | Start the interrupted process from the beginning. |

If the error is of the type improper geometry there is a possibility to do a retry with a higher force on the gun and complete the current weld, that is. when the plates are not properly fixed together.




xx1200000216

**Note**

The accuracy of the thickness supervision is highly dependent of good gun tuning and correct mechanical data, e.g. Transmission Gear Ratio.

*Continues on next page*

### Supervision before the weld is started

The default preweld supervisions are executed if configured, and the `SwPreWeld` routine is run. See *Process hooks on page 160*.

If an error occurs then:

- The signal `process_fault` for the current equipment is set. The program stops.
- An error message is displayed in a dialog box with retry possibilities.
- The error message is logged

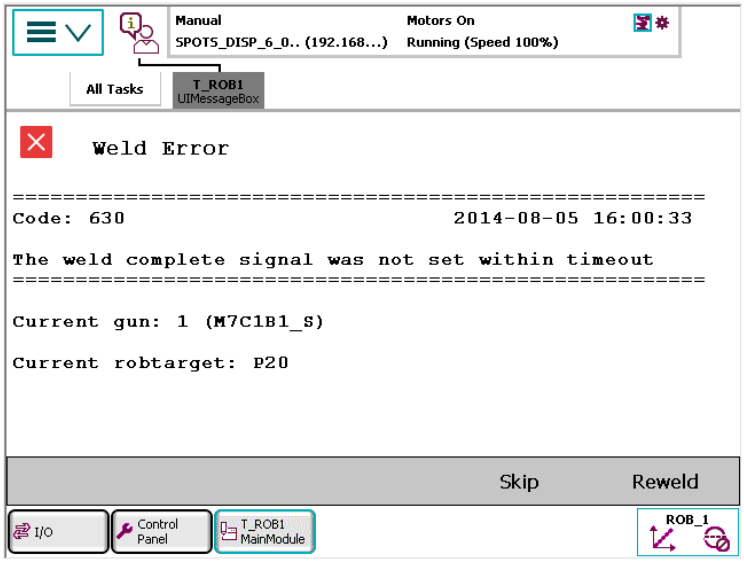
See *The type Spot Weld Equipment on page 27*

### Weld error

A weld error occurs either if the `weld_fault` signal is set during the weld process or if the weld complete signal from the weld timer has not been set in a certain time, `Weld timeout` in the process configuration. See *The type Spot Weld Equipment on page 27*.

`SpotL/J` and `SpotML/MJ` can be configured to automatically reweld a certain number of times before the error is displayed and the execution stops, waiting for a manual action.

- The gun opens.
- The signal process_error for current gun is set. The program stops.
- An error message is displayed in a dialog box with retry possibilities.
- The error message and the current robtarget name is logged.




xx1200000215

| | |
|---|---|
| **Skip** (only available in manual mode) | • The `reset_fault` signal is pulsed.<br>• The corresponding process error signal is reset.<br>• The current robtarget name will be stored in the log. The program execution is resumed but omitting the faulty weld. |

*Continues on next page*

| | |
|---|---|
| **Reweld** | • The `reset_fault` signal is pulsed.<br>• The corresponding process error signal is reset.<br>• The gun closes.<br>• The start signal is set after a short time delay and the program execution is resumed.<br><br>**Note**<br><br>If the optional signal `Reset fault with reweld [DO]` is configured, the start weld signal will be set to 1 during the reweld sequence. This will enable the KSR mode in a Bosch weld timer when performing a re-weld in adaptive weld mode. See *The type Spot Weld Equipment on page 27* |

**Skip** and **Reweld** error recovery can also be activated by using the digital inputs `skip_proc` and `reweld_proc`, see *Other signals on page 48*.

**Note**

If the spot system is built with an additional Bosch weld timer option there will be more information in the operator dialog about the reason for the error, for example, hardware fault in the weld timer etc.

**Tip**

The setup parameter `Number of automatic rewelds` in `Spot Error Handling` can be set to the number of welds required. See *The type Spot Error Handling on page 25*

Supervision after welding

The default gun opening sequence are executed if configured, and the `SwOpenGun` routine is run. See *Process hooks on page 160*.

For a pneumatic gun an error occurs if the `gun_open` signal is not set within a certain time.

- The signal `process_fault` for *current gun* is set.The program stops.
- An error message is displayed in a dialog box with retry possibilities.
- The error message is logged

See *The type Spot Gun Equipment on page 31*

Gun opening supervision (Servo guns only)

Any errors during gun opening will be detected by internal motion software. An error results in an error message on the FlexPendant and a program stop.

**User defined error handling**

All error situations described above can also be handled in a predefined user routine `SwErrorRecover` as an option to the built in error handling if needed. See *The type Spot Error Handling on page 25*.

If the "user defined" error handling is activated, a dedicated routine `SwErrorRecover` in `SWUSER` will be executed if any of the error cases described

*Continues on next page*

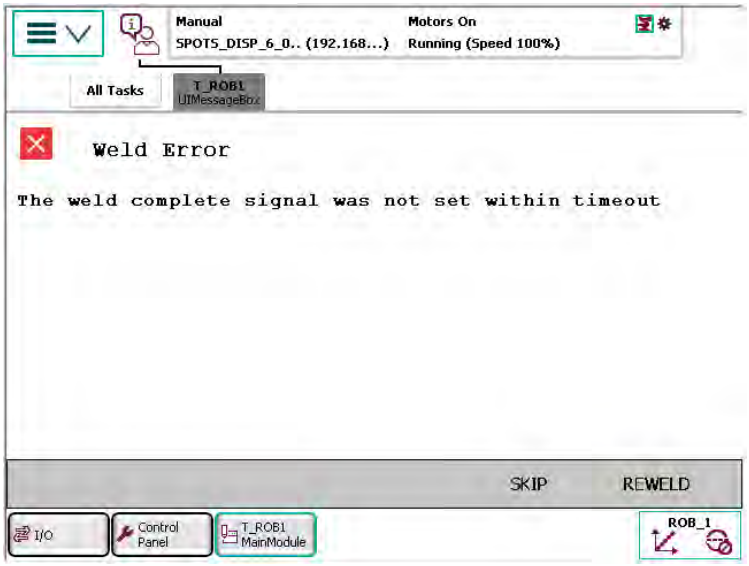
in section *Error handling on page 81* occur, except for parameter errors. `SwErrorRecover` is always executed from the robot task.

The input parameters to the `SwErrorRecover` routine carry information about the error reason and the chosen error text.

This routine allows customizing of the error handling response, that is. the FlexPendant layout and how to resume. For more information. See `SwErrorRecover` in *SWUSER on page 158*.

### Example

Default example if weld error occurs.




xx1200000214

## Software Equalizing

When the software equalizing functions are activated the execution of the `SpotL/J` instructions is influenced in different ways:

- The movement to the programmed position will be different.
- The gun preclose function is handled automatically.
- The \Inpos switch will not affect the program execution.

**Note**

The software equalizing functions are not implemented for the `SpotML/MJ` instructions.

For more information, see *Software Equalizing on page 169*.

## Multiple gun forces during welding

During the welding phase when a `SpotL/J` or a `SpotML/MJ` instruction is executed there is a possiblity to use multiple gun forces if needed.

The servo gun force can be controlled from the welding controller via group inputs.

*Continues on next page*

Internally in the Spot software an input group will monitored during the weld, and if the value on the input group changes, the gun force will change immediately to a lower or higher force.

For more information, see *How to use spot data programmed in the weld timer on page 258* and *Servo gun force calibration on page 203*.

> **Note**
>
> Note that the force calibration procedure is very important if multiple forces has to be used. The gun position at each force will be stored in the motion parameters when running this routine. For more information see *Servo gun force calibration on page 203*.

## Customizing

The Spot package gives the user plenty of scope for customizing the Spot functionality, see *Customizing RobotWare-Spot on page 247*.

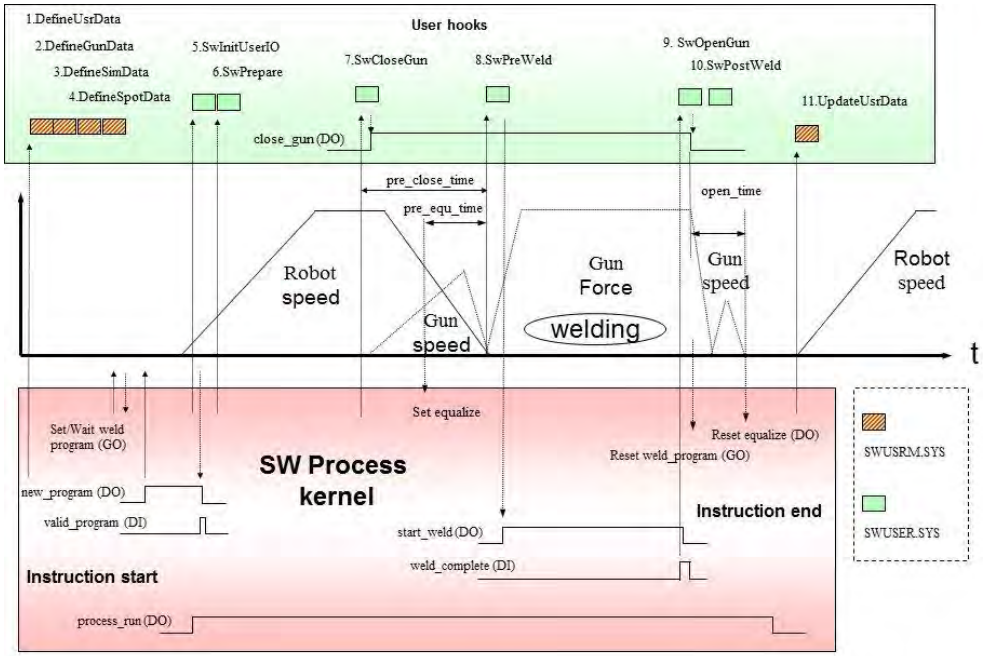
However the main subject of the `SpotL/J` and `SpotML/MJ` instructions description is the default setup.

## 3.7 Weld process timing

### Weld process timing for pneumatic guns

The following graphic shows the weld process timing for a pneumatic gun and where in the sequence the user hooks will be executed and affect the internal behavior.

If welding is done with several guns at the same time then each process is handled in separate tasks independent of each other.




en1200000238

### Weld process timing for servo guns

The following figure shows the weld process timing for a servo gun and where in the sequence the user hooks will be executed and affect the internal behavior.

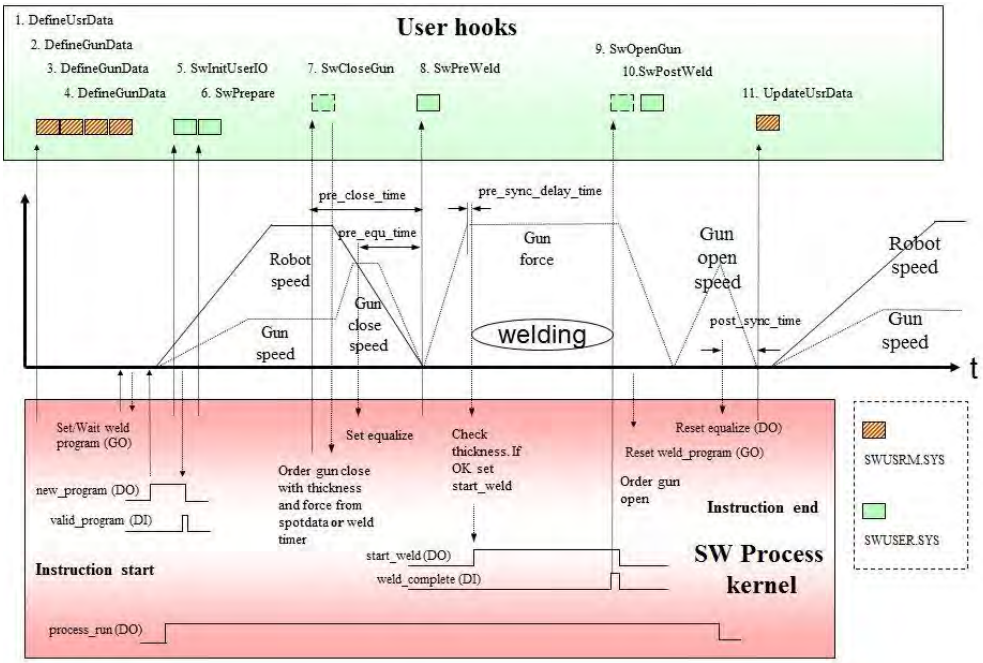
If welding is done with several guns at the same time then each process is handled in separate tasks independent of each other.

The system parameter *Post_sync_time* (*Post-synchronization Time*) in the topic *Motion*, type *SG Process*, defines the predicted release time of the next robot movement after a weld. Can be used to shorten the cycle time, the robot will start to move before the gun is completely opened. Default value is 0.

**Note**

The value of this parameter (*Post-synchronization Time*) can affect the cycle time of the program negatively if for example two welding points are programmed at the same position. To minimize this risk the value can be increased. See *Application manual - Additional axes and stand alone controller*.

*Continues on next page*




en1200000239

## Weld process timing for software equalizing with servo guns

The following graphic shows the weld process timing for a servo gun when software equalizing is activated, and where in the sequence the user hooks will be executed and affect the internal behavior.

If welding is done with several guns at the same times then each process is handled in separate tasks independent of each other.

The system parameter `Post_sync_time` (*Post-synchronization Time*) in the topic *Motion*, type *SG Process*, defines the predicted release time of the next robot movement after a weld. Can be used to shorten the cycle time, the robot will start to move before the gun is completely opened. Default value is 0.

**Note**

If *Soft Equalizing* is activated the `Spot Gun Equipment` process data parameters `Gun pre closing time` and `Gun pre equalizing time` are not used. The preclosing of the gun is in this case handled automatically, see *Software Equalizing on page 169*.

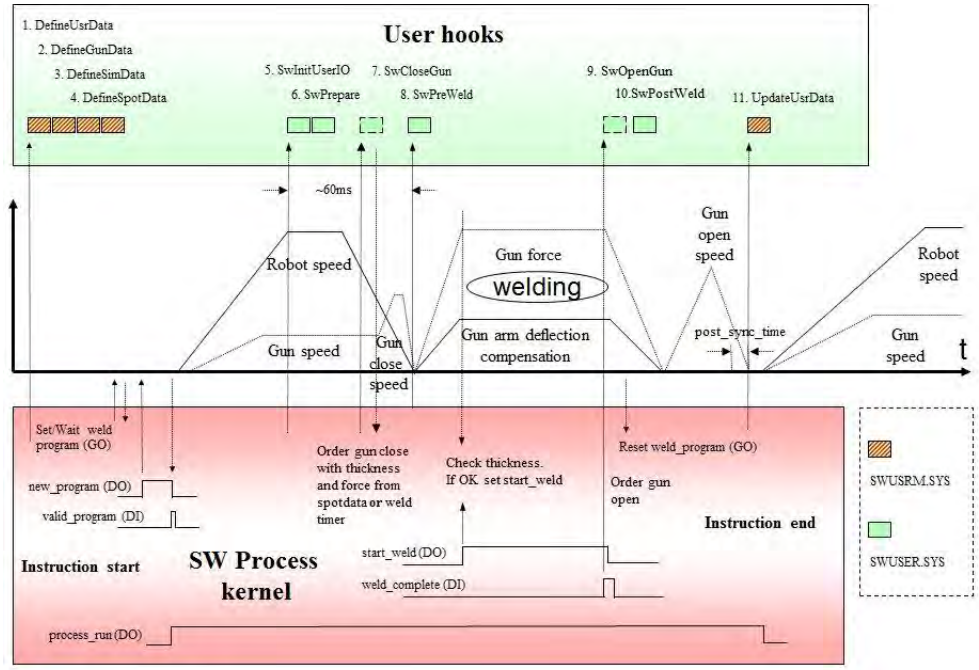
**Note**

The value of this parameter (*Post-synchronization Time*) can affect the cycle time of the program negatively if for example two welding points are programmed at the same position. To minimize this risk the value can be increased. See *Application manual - Additional axes and stand alone controller*.

*Continues on next page*

## 3.7 Weld process timing

*Continued*




en1200000237

## 3.8 Miscellaneous information

### 3.8.1 Jogging the robot after unintentional servo gun disconnection

**Servo gun disconnection**

If the motor cables are unintentional disconnected when the servo gun is activated, the servo gun must be deactivated in order to jog the robot to a service position. Deactivation is done in the **Jogging** window by selecting axis and tapping **Deactivate**. After service or repair the revolution counter must be updated since the position has been lost.

For more information, see *Recover from accidental servo gun disconnection on page 208*.

## 3.8.2 Tip dressing for servo guns

**Tip dressing for servo guns**

The `gundata` contains counters and tip wear information for each used gun. The counters will be automatically incremented for each spot and the tip wear information is updated after each gun calibration. This information can be used to decide when to do next tip dressing or tip exchange.

For more information see *gundata - Equipment specific weld data on page 135* and *Tip management on page 200*

## 3.8.3 Pneumatic spot welding gun and gripper

### Pneumatic spot welding gun and gripper

When the robot has a pneumatic spot welding gun and a gripper, with or without a tool changer, it takes some special arrangements to control the clamps on the gripper. The reason is that the air pressure valve on the Media Panel is controlled by the weld timer, which uses the valve to obtain different gun forces. The weld timer is in control of the air pressure valve, even when the robot is holding the gripper.

### Preparing control of the clamps

Use this procedure to prepare control of the clamps on the gripper:

1. In the weld timer, create weld programs for the desired pressures for gripper control.

   **Note**

   The weld current MUST be deactivated in the programs.

2. In the RAPID code, create the necessary control routines, and include `SetGO` instructions that sets the group output to the program number to the corresponding program in the weld timer.

   **Note**

   The *gX_new_prog* signal must be on at all times for the air pressure valve to follow immediately a new program number.

## 3.9 Supervision task SW_SUP

### Description

In spot options, there is a separate semistatic monitoring task that runs in the background, SW_SUP. This task is selected by default when building a spot system in *Installation Manager* and can be deselected if a supervision task is not needed, e.g. if the supervision is handled by external equipment.

The SW_SUP task is handling the built in water supervision. There are some different configuration possibilities regarding the behaviour of the water supervision, see *The type Spot Media Equipment on page 36*.

In the SWUSER module there is a routine that is called from SW_SUP task, SupervisionInit, here it is possible to add custom functionality / monitoring to be run independently of program execution in motion task. No default functionality, *Supervision task hook on page 163*.

# 4 RAPID references

## 4.1 Instructions

### 4.1.1 SpotL/SpotJ - The basic spot welding instructions

**Descriptions**

`SpotL` and `SpotJ` are used in spot welding when welding with one gun or several guns in sequence. The instructions are used to control the complete welding sequences, that is, the motion, gun closure/opening, and the welding process. `SpotL` moves the TCP linearly to the weld position and then activates the weld process. `SpotJ` moves the TCP non-linearly to the weld position before the weld process is activated.

This instruction can only be used in the Main task or, if in a MultiMove system, in Motion tasks.

**Example**

```
SpotL p100, vmax, gun1, spot10, tool1;
```

This is the only instruction needed to implement a complete welding operation with one gun equipment.

- The TCP for `tool1` is moved on a linear path to the position `p100` with the speed given in `vmax`.
- The weld position is always a stop position since the welding is always performed while the robot is standing still.
- The gun is closed in advance when the robot is moved[I].
- The weld process is started and supervised until finished and the gun is reopened.
- The parameter `spot10` is a data of type *spotdata* containing spot weld specific parameters for the spot in *p100*, for example desired weld timer program number and gun pressure.
- The parameter `gun1` is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the *gundata* array `curr_gundata` located in SWUSER.SYS module, see *SWUSER on page 158*.

[I] May differ depending on configuration.

**Arguments**

SpotL ToPoint Speed GunNo [\GunD] Spot [\InPos] [\OpenHLift] [\CloseHLift] [\QuickRelease] [\SMEQ] Tool [\WObj] [\TLoad]

SpotJ ToPoint Speed GunNo [\GunD] Spot [\InPos] [\OpenHLift] [\CloseHLift] [\QuickRelease] [\SMEQ] Tool [\WObj] [\TLoad]

`ToPoint`

Data type: robtarget

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

*Continues on next page*

4.1.1 SpotL/SpotJ - The basic spot welding instructions
*Continued*

`Speed`

Data type: speeddata

The speed data that applies to movements. Speed data defines the velocity for the tool center point, the tool reorientation and additional axes.

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the gundata array curr_gundata located in the SWUSER.SYS module, see *SWUSER on page 158*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`Spot`

Data type: spotdata

Spot specific data for the weld process, weld program number, gun force etc, see *spotdata - Spot weld data on page 140*

`[\InPos]`

Data type: switch

The optional argument \InPos inhibits the preclosing of the gun. The gun is closed first when the robot has reached the end position. This argument will increase the execution time but is useful in narrow situations.This switch will not affect the execution when software equalizing is active.

`[\OpenHLift]`

Data type: switch

The optional argument \OpenHLift will set the gun to its large gap after the weld. If the argument is omitted the gun opens to its small gap (work stroke). If the instruction is executed backwards the gun opens to the large position before the motion. (Only valid for pneumatic guns).

`[\CloseHLift]`

Data type: switch

The optional argument \CloseHLift will set the gun to its small gap (work stroke) before closing the gun. If the instruction is executed backwards the gun opens to the large position after the motion. (Only valid for pneumatic guns).

`[\QuickRelease]`

Data type: switch

The optional argument \QuickRelease will skip the release movement after the weld if software equalizing is activated. Can be used to save cycle time.

*Continues on next page*

`[\SMEQ]`

Data type: smeqdata (SoftMove Equalizing data)

If the optional data \SMEQ is used the robot will be set into a soft state in the tool z direction during the approach movement to the position.

This method can be used as a complement to the standard software equalizing method if the tolerances of the parts to be welded are less exact.

For more information see *SoftMove Equalizing on page 185*

**Note**

The SoftMove functionality is only avalible if the options *Spot Servo Equalizing (635-6)* and *SoftMove (885-1)* are installed together.

`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, and should be the position for the electrode tips when the gun is closed.

`[\WObj]`

Data type: wobjdata

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 TCP or coordinated additional 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`.

**Communication**

`SpotL/J` instructions communicates with the surrounding weld equipment using digital signals.

The default I/O setup is located on local I/O devices unless a specific SpotPack option is selected.

For a complete description of the I/O configuration, see *Spot I/O configuration on page 41* or *Spot Weld timer configuration options on page 54*.

*Continues on next page*

4.1.1 SpotL/SpotJ - The basic spot welding instructions
*Continued*

### Program execution

For a complete description of the program execution sequence and error handling in the `SpotL/J` instruction, see *Process sequence and error handling on page 76*.

### Limitations

**Note**

It is not possible use independent gun mode when Software Equalizing is active. This will cause an error message. For more information, see *IndGunMove - Activates independent mode for a servo gun on page 130*.

**Note**

It is only possible to run this instruction in semi coordinated mode.

**Note**

The `\QuickRelease` function is suitable to use if weld positions are located close to each other, not when there is a large distance between weld positions.

### Syntax

```
SpotL or SpotJ
  [ ToPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ GunNo ':='] < expression (IN) of num >
  [ '\' GunD ':='] < persistent(PERS) of gundata > ] ','
  [ Spot ':='] < persistent (PERS) of spotdata >
  [ '\' InPos ]
  [ '\' OpenHLift ]
  [ '\' CloseHLift ]
  [ '\' QuickRelease ]
  [ '\' SMEQ ':='] < persistent(PERS) of smeqdata > ] ','
  [ Tool ':=' ] < persistent (PERS) of tooldata > ]
  [ '\' WObj ':=' ] < persistent (PERS) of wobjdata > ]
  [ '\' TLoad ':=' ] < persistent (PERS) of loaddata > ] ';'
```

### Related information

| | Described in: |
|---|---|
| Definition of velocity, `speeddata` | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of zone data, `zonedata` | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of tool, `tooldata` | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of work objects, `wobjdata` | *Technical reference manual - RAPID Instructions, Functions and Data types* |

*Continues on next page*

| | Described in: |
|---|---|
| MoveL | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of load data, `loaddata` | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of spot data, `spotdata` | *spotdata - Spot weld data on page 140* |
| Definition of gun data, `gundata` | *gundata - Equipment specific weld data on page 135* |
| SpotML/MJ | *SpotML/SpotMJ - Spot welding with multiple guns on page 100* |
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| Customizing possibilities | *Customizing RobotWare-Spot on page 247* |
| I/O configuration | *Spot I/O configuration on page 41* |
| Servo gun introduction | *Servo gun motion control on page 195* |
| Servo gun motion parameters | *Application manual - Additional axes and stand alone controller* |
| Motion in general | *Technical reference manual - RAPID overview* |
| Software Equalizing | *Software Equalizing on page 169* |
| SoftMove Equalizing | *SoftMove Equalizing on page 185* |

## 4.1.2 SpotML/SpotMJ - Spot welding with multiple guns

### Description

SpotML and SpotMJ can be used in spot welding if welding with several guns at the same time is desired. For servo guns it is possible to use two guns simultaneously and for pneumatic guns it is possible to use four guns at the same time. The instructions are used to control the complete welding sequences that is. the motion, gun closure/opening and the welding processes.

- SpotML moves the TCP linearly to the weld position and then activates the gun equipments.
- SpotMJ moves the TCP non-linearly to the weld position before the gun equipment are activated.

These instructions can only be used in the Main task or, if in a MultiMove system, in Motion tasks.

### Example

```
SpotML p100, vmax \G1:=spot10 \G2:=spot20, tool1;
```

This is the only instruction needed to implement a complete welding operation with two gun equipment.

- The TCP for tool1 is moved on a linear path to the position p100 with the speed given in vmax. The weld position is always a stop position since the welding is always performed while the robot is standing still. The guns are closed in advance when the robot is moved. The weld processes are started and supervised until finished and the guns are reopened.
- The optional arguments G1 and G2 will activate gun equipment 1 and gun equipment 2. The parameter spot10 is a spotdata containing weld parameters for the welding with gun equipment 1, for example desired weld timer program number and gun pressure. The parameter spot20 contains weld parameters for the welding with gun equipment 2.

All gun equipment used are defined in the gundata array curr_gundata in SWUSER.SYS module, see *SWUSER on page 158*.

### Arguments

SpotML ToPoint Speed [\G1] [\G2] [\G3] [\G4] [\Gun1] [\Gun2] [\Gun3] [\Gun4] [\InPos] [\OpenHLift] [\CloseHLift] Tool [\WObj] [\TLoad]

SpotMJ ToPoint Speed [\G1] [\G2] [\G3] [\G4] [\Gun1] [\Gun2] [\Gun3] [\Gun4] [\InPos] [\OpenHLift] [\CloseHLift] Tool [\WObj] [\TLoad]

ToPoint

Data type: robtarget

The destination point of the robot and additional axes. It is defined as a named position or stored directly in the instruction (marked with an * in the instruction). This name will be stored in the log if a error occurs during the welding.

Speed

Data type: speeddata

*Continues on next page*

The speed data that applies to movements. Speed data defines the velocity for the tool center point, the tool reorientation and additional axes.

`[G1] - [G4]`

Data type: spotdata for gun equipment 1 - 4

Spot data with the spot specific data associated with the weld with gun equipment 1 - 4, see *spotdata - Spot weld data on page 140* and *gundata - Equipment specific weld data on page 135*.

`[\Gun1] - [\Gun4]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`[\InPos]`

Data type: switch

The optional argument \InPos inhibits the preclosing of the guns. The guns are closed first when the robot has reached the end position. This argument will increase the execution time but is useful in narrow situations.

`[\OpenHLift]`

Data type: switch

The optional argument \OpenHLift will set the guns to its large gap after the weld. If the argument is omitted the guns opens to its small gap (work stroke). If the instruction is executed backwards the guns opens to the large position before the motion. (Only valid for pneumatic guns).

`[\CloseHLift]`

Data type: switch

The optional argument \CloseHLift will set the guns to its small gap (work stroke) before closing the guns. If the instruction is executed backwards the guns opens to the large position after the motion. (Only valid for pneumatic guns).

`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, and should be the position for the electrode tips when the gun is closed.

`[\WObj]`

Data type: wobjdata

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 TCP or coordinated additional

*Continues on next page*

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`.

### Communication

`SpotML/MJ` instructions communicates with the weld equipment using digital signals. The default I/O setup is located on local I/O devices.

For a complete description of the I/O configuration, see *Spot I/O configuration on page 41*

### Program execution

For a complete description of the program execution sequence and error handling in the `SpotML/MJ` instruction, see *Process sequence and error handling on page 76*.

### Limitations

This instruction will not be installed if a *Weld Timer Configuration* option is selected when building a spot system in RobotStudio. In this case only a minimal installation of spot will be done with predefined signals for one weld equipment only.

**Note**

It is not possible to use Software Equalizing mode for this instruction, `SpotML/SpotMJ`.

For more information, see *Software Equalizing on page 169*.

**Note**

It is only possible to run this instruction in semi coordinated mode.

### Syntax

```
SpotML or SpotMJ
  [ ToPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ '\' G1 ':=' < persistent (PERS) of spotdata > ]
  [ '\' G2 ':=' < persistent (PERS) of spotdata > ]
  [ '\' G3 ':=' < persistent (PERS) of spotdata > ]
  [ '\' G4 ':=' < persistent (PERS) of spotdata > ]
  [ '\' Gun1 ':=' < persistent (PERS) of gundata > ]
  [ '\' Gun2 ':=' < persistent (PERS) of gundata > ]
  [ '\' Gun3 ':=' < persistent (PERS) of gundata > ]
```

*Continues on next page*

```
[ '\' Gun4 ':=' < persistent (PERS) of gundata > ]
[ '\' InPos ]
[ '\' OpenHLift ]
[ '\' CloseHLift ]','
[ Tool ':=' ] < persistent (PERS) of tooldata > ]
[ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
[ '\' TLoad':=' ] < persistent (PERS) of loaddata > ] ';'
```

## Related information

| | Described in: |
|---|---|
| Definition of velocity, speeddata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of zone data, zonedata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of tool, tooldata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of work objects, wobjdata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| MoveL | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of load data, loaddata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of spot data, spotdata | *spotdata - Spot weld data on page 140* |
| Definition of gun data, gundata | *gundata - Equipment specific weld data on page 135* |
| SpotL/J | *SpotL/SpotJ - The basic spot welding instructions on page 95* |
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| Customizing possibilities | *Customizing RobotWare-Spot on page 247* |
| I/O configuration | *Spot I/O configuration on page 41* |
| Servo gun introduction | *Servo gun motion control on page 195* |
| Servo gun motion parameters | *Application manual - Additional axes and stand alone controller* |
| Motion in general | *Technical reference manual - RAPID overview* |

## 4.1.3 SetForce - Close and Open a gun with desired force and time

**Description**

`SetForce` is used in spot welding to close the gun and apply a predefined force during a desired time without activating a weld process. The gun will open again after the elapsed time or when a digital input signal is set. This instruction can for example be used for tip dressing.

**Example**

```
SetForce gun1, force10;
```

Forcedata `force10` contains the parameters for the `SetForce` action, for example desired tip force and force time.

The parameter `gun1` is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the gundata array `curr_gundata` located in `SWUSER.SYS` module. See *SWUSER on page 158*.

**Arguments**

SetForce GunNo [\GunD] Force [\RetThickness] [\PrePos] [\CloseSpeed] [\OpenHLift] [\CloseHLift]

`GunNo`

Data type: num

Used gun number. Corresponding to the element number in the gundata array `curr_gundata` in the `SWUSER.SYS` module. See *SWUSER on page 158*.

`[\GunD]`

Data type: gundata

Optional parameter. Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`Force`

Data type: forcedata

The forcedata with the force parameters. See *forcedata - Spot gun force data on page 143*.

`[\RetThickness]`

(returned thickness)

Data type: num

Optional parameter. The achieved thickness [mm] (servo guns only).

`[\IndPos]`

(independent pre-position)

Data type: num

Optional parameter. The desired independent pre-position when the specified gun speed should be used [mm]. (servo guns only).

*Continues on next page*

`[\GunSpeed]`

(gun speed)

Data type: num

Optional parameter. The desired gun speed that shall be used from the specified independent pre-position [%]. (servo guns only). This parameter can be used to get a better performance when e.g tip dressing by reducing the gun speed.

If an independent pre-position is **not** used the gun speed will be reduced from the actual start position.

`[\OpenHLift]`

Data type: switch

The optional argument \OpenHLift will set the gun to its large gap after the instuction. If the argument is omitted the gun opens to its small gap (work stroke). If the instruction is executed backwards the gun opens to the large position before the motion. (Only valid for pneumatic guns).

`[\CloseHLift]`

Data type: switch

The optional argument \CloseHLift will set the gun to its small gap (work stroke) before closing the gun. If the instruction is executed backwards the gun opens to the large position. (Only valid for pneumatic guns).

**Program execution**

Internal sequence when a SetForce instruction is executed:

1. The gun is closed to the specified thickness in the used `forcedata`. If **pre-position** is used `\IndPos`, gun will be set to independent mode internally, and the closing speed will be reduced from the independent position according to the specified value in `\GunSpeed`.
2. The plate thickness is checked (servo guns only).
3. The requested gun force is established.
4. Wait until the desired force time elapsed or the force complete signal is activated.
5. If configured, the second gun force in `curr_forcedata` is established, see *forcedata - Spot gun force data on page 143*.
6. If configured, wait until the second force time has elapsed or the force complete signal is activated.
7. The gun is opened to the previous position. If an independent **pre-position** is used, the opening speed will be reduced to the independent position according to the specified value in `\GunSpeed` and the independent mode will be reset.

The force complete signal for each used gun is predefined in the I/O configuration.

For a complete description of the I/O configuration, see *Spot I/O configuration on page 41*.

*Continues on next page*

4.1.3 SetForce - Close and Open a gun with desired force and time
*Continued*

### Error handling

Instruction parameter supervision

The error occurs when `SetForce` is called with faulty parameters. The program stops.

The parameter must be changed. When the program is restarted the current instruction is restarted from the beginning.

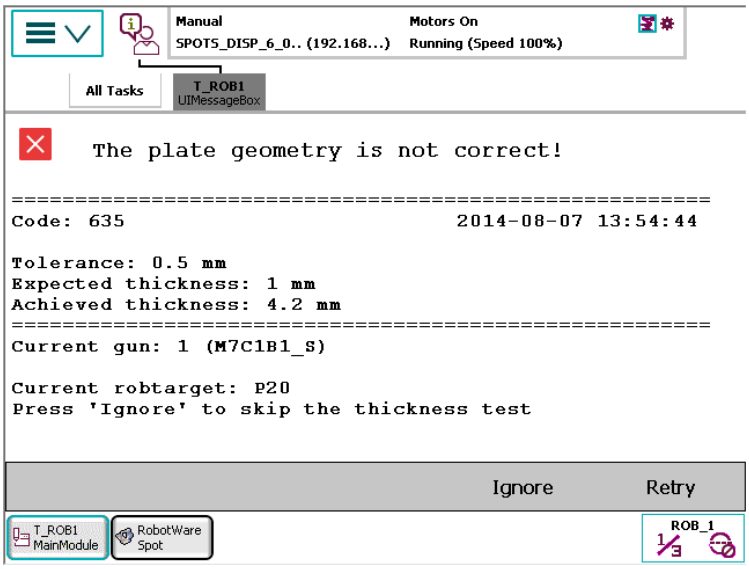
Detection of missing or improper plates (Only for servo guns)

An error will be detected by the process kernel if the plate thickness differ more than the allowed limit defined by the tolerance from the programmed thickness.

There are three different types of errors:

- Negative gun position, one of the tips are missing on the gun, or a `tip_wear` calibration is needed.
- Missing plates, the plate thickness is smaller than the thickness defined in `forcedata`.
- Improper geometry, the plate thickness exceeds the tolerance defined in `forcedata`.

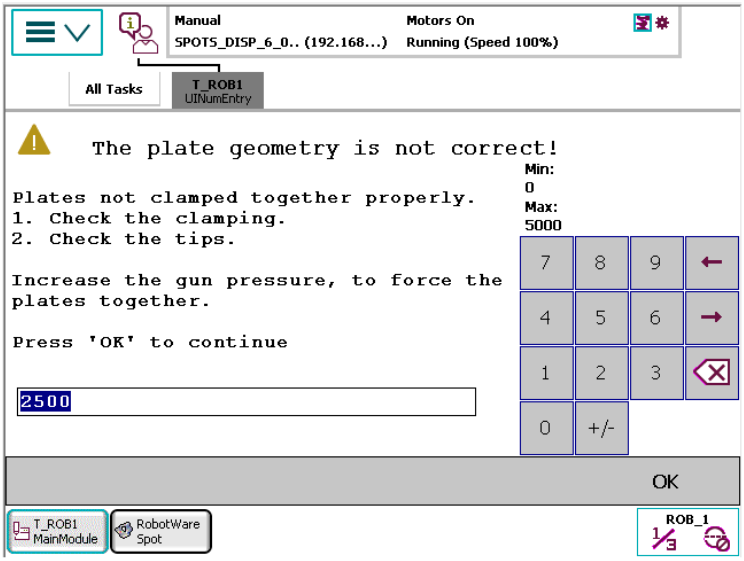
1. The gun opens.
2. The process error signal for the current gun equipment is set. The program stops.
3. An error message is displayed in a dialog box with retry possibilities.
4. The error message is logged.




xx1200000217

| | |
|---|---|
| Ignore | Close the gun again but without thickness detection and continue the execution. |
| Retry | Retry the `SetForce` instruction. |
| Skip | If the error is of the type improper geometry there is a possibility to do a retry with a higher force on the gun and complete the current weld, that is. when the plates are not properly fixed together.<br>Only available in manual mode. |

*Continues on next page*




xx1200000216

**Note**

The accuracy of the thickness supervision is highly dependent of good gun tuning and correct mechanical data, e.g. Transmission Gear Ratio.

## Limitations

If the `\IndPos` argument is used the gun will be set to an independent position. If the instruction is aborted and the program pointer is moved, or an error occur while the independent mode is active, the independent mode will be cleared depending on if the system is in motors on state or not. If motors off state independent mode will be cleared at the next start or restart.

Independent mode will cleared in the following situations:

Stop / QStop / Start / ReStart or program pointer moved:

- If motors on state: Independent mode will be cleared.
- If motors off state: Independent mode will **not** be cleared.

For more information about independent gun mode, see *IndGunMove - Activates independent mode for a servo gun on page 130* and *IndGunMoveReset - Resets servo gun from independent mode on page 132*.

## Syntax

```
SetForce
  [ GunNo ':='] < expression (IN) of num >
  [ '\' GunD ':='] < persistent(PERS) of gundata > ] ','
  [ Force ':='] < persistent (PERS) of forcedata >
  [ '\' RetThickness ':=' < variable or persistent(INOUT) of num
       > ]
  [ '\' IndPos ':=' < expression (IN) of num > ]
  [ '\' GunSpeed ':=' < expression (IN) of num > ]
  [ '\' OpenHLift ]
  [ '\' CloseHLift ] ';'
```

## 4.1.4 CalibL/CalibJ - Calibrate a servo gun during robot movement

### Description

`CalibL/J` is used in spot welding to calibrate the distance between the gun tips for servo guns. This is necessary after tip change or tool change and it is recommended after welding of a number of spots or performing a tip dress. Calibrate will also update the tip wear data in the used gundata. The calibration is done during a robot movement to a programmed position.

NB: The gun performs two non-synchronized close/open movements during the calibration.

If the option `Spot Servo Equalizing` is installed there are additional error handling included for lost tips when a TipChg calibration is done and supervision of the tip wear when a Tip-Wear calibration is done.

### Example

```
CalibL p400, v500, gun1\ TipWear, fine, tool1;
```

- The gun `gun1` is calibrated for TipWear during the linear movement to p400.
- The parameter `gun1` is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the gundata array `curr_gundata` in SWUSER. See *SWUSER on page 158*.
- The data `curr_tip_wear` in `curr_gundata` will be automatically updated.

For more information about tip management, see *Tip management on page 200*.

### Arguments

CalibL ToPoint Speed GunNo [\GunD] [\TipChg] | [\ToolChg] | [\TipWear] [\RetTipWear] [\RetPosAdj] [\PrePos] [\TWeld], Zone Tool [\WObj] [\TLoad]

CalibJ ToPoint Speed GunNo [\GunD] [\TipChg] | [\ToolChg] | [\TipWear] [\RetTipWear] [\RetPosAdj] [\PrePos] [\TWeld], Zone Tool [\WObj] [\TLoad]

`ToPoint`

Data type: robtarget

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

A movement of the gun tip position can not be programmed. This will cause an error message.

`Speed`

Data type: speeddata

The speed data that applies to movements. Speed data defines the velocity for the tool center point, the tool reorientation and additional axes.

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the gundata array `curr_gundata` located in the `SWUSER.SYS` module. See *SWUSER on page 158*.

*Continues on next page*

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`[\TipChg]`

(tip change calibration)

Data type: switch

Calibration type. This calibration type is used after tip change.

The gun will close and open two times. The first close movement will be slow to find the unknown contact position. The total tip wear is reset to zero.

If the option `Spot Servo Equalizing` is selected the difference since last calibration will be supervised. If the difference since the last calibration exceeds the supervision value in the `Tip change supervision value` an error will be raised. See *The type Spot Gun Equipment on page 31*.

For more information about tip management, see *Tip management on page 200*, *Tip wear compensation on page 180*, *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 126*.

`[\ToolChg]`

(tool change calibration)

Data type: switch

Calibration type. This calibration type is used after tool change.

The gun will close and open two times. The first close movement will be slow to find the unknown contact position. The total tip wear will remain unchanged.

`[\TipWear]`

(tip wear calibration)

Data type: switch

Calibration type. This calibration type is used to update the tip wear and adjust the contact position after tip dress or after welding a number of spots.

The gun will close and open fast two times. The total tip wear is updated.

If the option `Spot Servo Equalizing` is selected the difference since last calibration will be supervised. If the difference since the last calibration exceeds the supervision value in the `Tip wear supervision value` an error will be raised. See *The type Spot Gun Equipment on page 31*.

For more information about tip management, see *Tip management on page 200*, *Tip wear compensation on page 180*, *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 126*.

*Continues on next page*

4.1.4 CalibL/CalibJ - Calibrate a servo gun during robot movement
*Continued*

`[\RetTipWear]`

Data type: num

The achieved tip wear [mm].

`[\RetPosAdj]`

Data type: num

The positional adjustment since the last calibration [mm].

`[\PrePos]`

(pre position)

Data type: num

The position to move with high speed to before search for contact position with slower speed is started [mm].

`[\TWeld]`

(test weld)

Data type: spotdata

If selected, a weld with the specified parameters will be performed after the calibration and during the robot movement.

A weld can be done after tip dressing to check the tips, and to save cycle time the test weld argument can be used instead of adding an extra `SpotL` instruction after the calibration.

`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, and should be the position for the electrode tips when the gun is closed.

`[\WObj]`

Data type: wobjdata

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 TCP or coordinated additional axes are used, this argument must be specified in order to perform a linear movement relative to the work object.

`[\TLoad]`

Data type: loaddata

*Continues on next page*

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.

**Program execution**

Internal sequence when a CalibL/J instruction is executed:

- The robot starts the movement to the destination position.
- The gun will close and open two times during the robot movement. Different tip speeds depending on selected calibration type.
- If the \TWeld is selected a test weld with the specified data will be done.
- The gun is opened to the previous position.
- For certain calibration types: curr_tip_wear in the array curr_gundata in SWUSER is updated and saved.

Positional adjustment

The optional argument RetPosAdj can be used to detect if for example the tips are lost after a tip change. The parameter will hold the value of the positional adjustment since the last calibration. The value can be negative or positive.

If the option Spot Servo Equalizing is selected this value will be used to calculate the difference since last calibration and supervise the tips when calibrating.

Using a pre position

In order to speed up the calibration, it is possible to define a pre position.When the calibration starts, the gun arm will be run fast to the pre position, stop and then continue slowly forward in order to detect the tip contact position. A pre position will be ignored if it is larger than the current gun position (in order not to slow down the calibration).

**Instruction by instruction execution**

| | |
|---|---|
| Forward | As during continuous execution. |
| Backward | The motion is performed backwards to the programmed position, but no calibration is activated. NB, the tip distance in this case is the programmed value in the instruction. |
| Positional adjustment | The optional argument RetPosAdj can be used to detect if for example the tips are lost after a tip change. The parameter will hold the value of the positional adjustment since the last calibration. The value can be negative or positive.<br>If the option *Spot Servo Equalizing* is selected this value will be used to calculate the difference since last calibration and supervise the tips when calibrating. |
| Using a pre position | In order to speed up the calibration, it is possible to define a pre position. When the calibration starts, the gun arm will be run fast to the pre position, stop and then continue slowly forward in order to detect the tip contact position. A pre position will be ignored if it is larger than the current gun position (in order not to slow down the calibration). |

*Continues on next page*

4.1.4 CalibL/CalibJ - Calibrate a servo gun during robot movement
*Continued*

### Error handling

Instruction parameter supervision

The error occurs when `CalibL/J` is called with faulty parameters or if no calibration type switch is programmed. The program stops with error text.

The parameter must be changed. When the program is restarted the current instruction is restarted from the beginning.

Tip change supervision

If the calculated difference to the last calibration of the gun exceeds the supervision value defined in the configuration data `Tip change supervision value` an error will be raised and the program execution will be stopped. This error can occur for example after tip change and when `CalibL/J ... \TipChg` is called with wrong (too large or too small tips) tips. The program stops with error message. See *The type Spot Gun Equipment on page 31*.

This error handling only exists for the `Spot Servo Equalizing` option. See *Additional components for Spot Servo Equalizing on page 136*.

For more information about tip management, see *Tip management on page 200*.

Tip wear supervision

If the calculated difference to the last calibration of the gun exceeds the supervision value defined in the configuration data `Tip wear supervision value` an error will be raised and the program execution will be stopped. This error can occur for example after tip dressing when `CalibL/J .. \TipWear` is called with badly dressed tips. The program stops with error message. See *The type Spot Gun Equipment on page 31*.

This error handling only exists for the `Spot Servo Equalizing` option. See *Additional components for Spot Servo Equalizing on page 136*.

For more information about tip management, see *Tip management on page 200*.

Test weld error

If a weld error occur during the robot movement it will be handled in the same way as a normal weld error, see *Weld error on page 84*

It is also possible to handle a weld error in the user defined error handling if needed, see *User defined error handling on page 85*.

### Limitations

**Note**

It is only possible to run this instruction from a motion task.

**Note**

It is only possible to run this instruction in semi coordinated mode.

*Continues on next page*

## Syntax

```
CalibL or CalibJ
  [ ToPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ GunNo ':='] < expression (IN) of num >
  [ '\' GunD ':='] < persistent(PERS) of gundata > ]
  [ \TipChg] | [\ToolChg] | [\TipWear]
  [ '\' RetTipWear ':=' < variable or persistent(INOUT) of num >
      ]
  [ '\' RetPosAdj ':=' < variable or persistent(INOUT) of num > ]
  [ '\' PrePos ':=' < variable or persistent(IN) of num > ]
  [ '\' TWeld ':=' < persistent(IN) of spotdata > ] ','
  [ Zone ':=' ] < expression (IN) of zoneddata > ] ','
  [ Tool ':=' ] < persistent (PERS) of tooldata > ]
  [ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
  [ '\' TLoad':=' ] < persistent (PERS) of loaddata > ] ';'
```

## Related information

| | Described in: |
|---|---|
| Definition of velocity, speeddata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of zone data, zonedata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of tool, tooldata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of work objects, wobjdata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| MoveL | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of load data, loaddata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| Servo gun introduction | *Servo gun motion control on page 195* |
| Calibration without movement | *Servo gun motion control on page 195* |
| Software Equalizing | *Software Equalizing on page 169* |
| Setup data for Software Equalizing | *The type Spot SoftWare Equalizing on page 38* |

## 4.1.5 Calibrate - Calibrate a servo gun

### Description

`Calibrate` is used in spot welding to calibrate the distance between the gun tips for servo guns. This is necessary after tip change or tool change and it is recommended after welding of a number of spots or performing a tip dress. `Calibrate` will also update the tip wear data in the used gundata. NB The gun performs two non-synchronized close/open movements during the calibration. The open distance after the calibration is finish will be the same as before the calibration started.

If the option `Spot Servo Equalizing` is installed there are additional error handling included for lost tips when a TipChg calibration is done and supervision of the tip wear when a Tip-Wear calibration is done.

### Example

```
Calibrate gun1\ TipChange;
```

- The gun gun1 is calibrated after TipChange.
- The parameter gun1 is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the gundata array `curr_gundata` in SWUSER.
- The data `curr_tip_wear` in `curr_gundata` will be automatically set to zero.

For more information about tip management, see *Tip management on page 200*.

### Arguments

Calibrate GunNo [\GunD] [\TipChg] | [\ToolChg] | [\TipWear] [\RetTipWear] | [\RetPosAdj] | [\PrePos]

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the gundata array `curr_gundata` located in the `SWUSER.SYS` module . See *SWUSER on page 158*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`[\TipChg]`

Data type: switch

Calibration type. This calibration type is used after tip change.

The gun will close and open two times. The first close movement will be slow to find the unknown contact position. The total tip wear is reset to zero.

*Continues on next page*

If the option `Spot Servo Equalizing` is selected the difference since last calibration will be supervised. If the difference since the last calibration exceeds the supervision value in the `Tip change supervision value` an error will be raised. See *The type Spot Gun Equipment on page 31*.

For more information about tip management, see *Tip management on page 200*, *Tip wear compensation on page 180*, *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 126*.

`[\ToolChg]`

Data type: switch

Calibration type. This calibration type is used after tool change.

The gun will close and open two times. The first close movement will be slow to find the unknown contact position. The total tip wear will remain unchanged.

`[\TipWear]`

Data type: switch

Calibration type. This calibration type is used to update the tip wear and adjust the contact position after tip dress or after welding a number of spots.

The gun will close and open fast two times. The total tip wear is updated.

If the option `Spot Servo Equalizing` is selected the difference since last calibration will be supervised. If the difference since the last calibration exceeds the supervision value in the `Tip wear supervision value` an error will be raised. See *The type Spot Gun Equipment on page 31*.

For more information about tip management, see *Tip management on page 200*, *Tip wear compensation on page 180*, *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 126*.

`[\RetTipWear]`

Data type: num

The achieved tip wear [mm].

`[\RetPosAdj]`

Data type: num

The positional adjustment since the last calibration [mm].

`[\PrePos]`

Data type: num

The position to move with high speed to before search for contact position with slower speed is started [mm].

**Program execution**

Internal sequence when a Calibrate instruction is executed:

- The gun will close and open two times during the robot movement. Different tip speeds depending on selected calibration type.

*Continues on next page*

4.1.5 Calibrate - Calibrate a servo gun
*Continued*

- The gun is opened to the previous position.
- For certain calibration types: `curr_tip_wear` in the array `curr_gundata` in SWUSER is updated and saved.

Positional adjustment

The optional argument `RetPosAdj` can be used to detect if for example the tips are lost after a tip change. The parameter will hold the value of the positional adjustment since the last calibration. The value can be negative or positive.

If the option `Spot Servo Equalizing` is selected this value will be used to calculate the difference since last calibration and supervise the tips when calibrating.

Using a pre position

In order to speed up the calibration, it is possible to define a pre position.When the calibration starts, the gun arm will be run fast to the pre position, stop and then continue slowly forward in order to detect the tip contact position. A pre position will be ignored if it is larger than the current gun position (in order not to slow down the calibration).

**Error handling**

Instruction parameter supervision

The error occurs when `Calibrate` is called with faulty parameters or if no calibration type switch is programmed. The program stops with error text.

The parameter must be changed. When the program is restarted the current instruction is restarted from the beginning.

Tip change supervision

If the calculated difference to the last calibration of the gun exceeds the supervision value defined in the configuration data `Tip change supervision value` an error will be raised and the program execution will be stopped. This error can occur for example after tip change and when `Calibrate ... \TipChg` is called with wrong (too large or too small) tips. The program stops with error message. See *The type Spot Gun Equipment on page 31*.

This error handling only exists for the `Spot Servo Equalizing` option.

For more information about tip management, see *Tip management on page 200*.

Tip wear supervision

If the calculated difference to the last calibration of the gun exceeds the supervision value defined in the configuration data `Tip wear supervision value` an error will be raised and the program execution will be stopped. This error can occur for example after tip dressing when Calibrate ... \Tip- Wear is called with badly dressed tips. The program stops with error message. See *The type Spot Gun Equipment on page 31*.

This error handling only exists for the Spot Servo Equalizing option.

For more information about tip management, see *Tip management on page 200*.

*Continues on next page*

## Syntax

```
Calibrate
  [ GunNo ':='] < expression (IN) of num >
  [ '\' GunD ':='] < persistent(PERS) of gundata > ]
  [ \TipChg]
  | [\ToolChg]
  | [\TipWear]
  [ '\' RetTipWear ':=' < variable or persistent(INOUT) of num >
       ]
  [ '\' RetPosAdj ':=' < variable or persistent(INOUT) of num > ]
  [ '\' PrePos ':=' < variable or persistent(IN) of num > ] ';'
```

## Related information

| | Described in: |
|---|---|
| Overview *Spot Servo* | *Introduction to RobotWare Spot on page 15* |
| Servo gun introduction | *Servo gun motion control on page 195* |
| Calibration with movement | *Servo gun motion control on page 195* |
| Software Equalizing | *Software Equalizing on page 169* |
| Setup data for Software Equalizing | *The type Spot SoftWare Equalizing on page 38* |

## 4.1.6 MeasureWearL - Measure current electrode wear and recalculate the TCP

### Description

`MeasureWearL` is used in spot welding to measure current electrode wear for the tip on the fixed electrode. This can be done with or without external measurement equipment, and without manual interaction. The TCP is automatically recalculated after the measurement. The instruction also updates tip wear data in the used `gundata`.

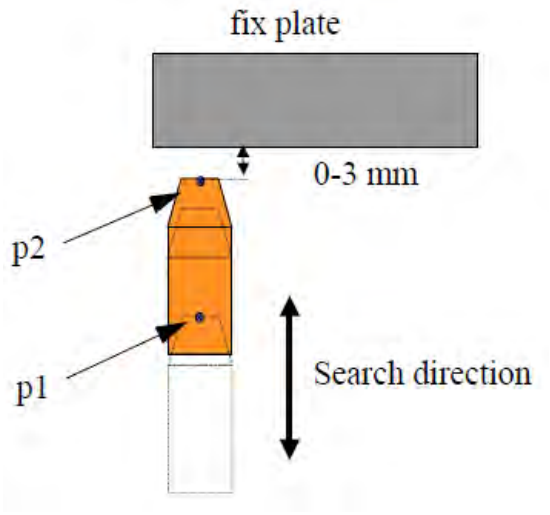
When the gun is held by the robot the gun performs a search movement during the measurement. The gun is moving in the z-direction, in the tool coordinate system, until the fixed electrode touches a fixed reference plate or a sensor of some sort, e.g. *BullsEye.*

This instruction can be used also for stationary guns. In this case the robot moves the gripper and the work object until a reference position on the gripper is touching the tip on the fixed electrode.

This instruction is only available if the `Spot Servo Equalizing` configuration is installed.

### Example

In this example the gun is held by the robot. The principles are the same also when stationary guns are used.




xx1200000206

#### Measurement preparation

The measurement instruction is executed with a reference tip with an accurate TCP (tooldata in this example: `ref_tool1`). This reference measurement has to be done before the tip wear measuring is performed the first time. Also each time when the TCP for this gun is changed for some reason or if the reference plate or sensor is dislocated for some reason.

Running the instruction `Calibrate\TipChg` after the `MeasureWearL` will also reset the total wear of the tips `curr_tip_wear` in `curr_gundata` and check the difference since the last calibration.

*Continues on next page*

Program example for reference measurement:

```
MoveJ p1,v1000,z50,ref_tool1;
MeasureWearL p2,v1000,gun1\Reference,ref_tool1;
tool1 := ref_tool1;
! tool1 is then used during the production
MoveL p1,v1000,z50,tool1;
Calibrate gun1\TipChg;
```

When the `MeasureWearL` instruction with the optional argument `\Reference` is executed, first a linear movement to a position about 10 mm outside *p2* is done. Then the gun is moved in the z direction in the tool coordinate system until the fixed tip touches the reference plate. During this reference measurement the reference plate is touched twice. When the measurement is ready some reference data is stored, `tw_ref_tool` and `tw_ref_dist` in SWUSER.

The parameter `gun1` is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the `gundata` array `curr_gundata` in SWUSER.

**Tip**

To verify if the selected measuring position or gun orientation is good enough a service routine is available; `ManualCheckMeasPos`. Run this routine is run in the selected position and status information will be presented on the FlexPendant whether the position is suitable or not. See *Manual actions on page 74*

Measurement after tip wear

When it is time to compensate for current tip wear, probably after each tip dressing, the following instruction sequence should be executed:

Program example for tipwear measurement:

```
MoveJ p1,v1000,z50,tool1;
MeasureWearL p2,v1000,gun1\TipWear,tool1;
MoveL p1,v1000,z50,tool1;
Calibrate gun1\TipWear;
```

When the instruction with the optional argument `TipWear` is executed, a search movement to the reference plate or sensor is performed and the tip wear of the fixed electrode is measured. The TCP in the used tooldata *tool1* is then recalculated and the data `curr_wear_fix` in `curr_gundata` is automatically updated.

Running the instruction `Calibrate\TipWear` after the `MeasureWearL` will also update the total wear of the tips `curr_tip_wear` in `curr_gundata` and check the difference since the last calibration.

Measurement after tip change (with or without tip dressing)

In the first measurement after tip change a similar sequence can be used as after tip wear. In this case the optional argument `\TipChange` has to be used.

Program example for tip change measurement:

```
MoveJ p1,v1000,z50,tool1;
MeasureWearL p2,v1000,gun1\TipChange,tool1;
MoveL p1,v1000,z5,tool1;
Calibrate gun1\TipChg;
```

*Continues on next page*

4.1.6 MeasureWearL - Measure current electrode wear and recalculate the TCP
*Continued*

When the instruction with the optional argument `TipChange` is executed, similar movements as above are performed and the tip wear of the fixed electrode is measured. The TCP in the used tooldata *tool1* is then recalculated and the data `curr_wear_fix` in `curr_gundata` is automatically updated. This is the same functionality as after tip wear above. Only some extra error handling is done internally.

Running the instruction `Calibrate\TipChg` after the `MeasureWearL` will also reset the total wear of the tips `curr_tip_wear` in `curr_gundata` and check the difference since the last calibration. See *SWUSER on page 158*

**Note**

It is important that p2 is the same position in all cases above. If this position is modified a new reference or reference changed measurement has to be done.

Measurement after reference plate/sensor changed

This mode can be used when the TCP for this gun is changed for some reason or if the reference plate or sensor is dislocated for some reason.

Program example for reference changed measurement:

```
MoveJ p1,v1000,z50,ref_tool1;
MeasureWearL p2,v1000,gun1\RefChange,ref_tool1;
MoveL p1,v1000,z50,tool1;
```

When the `MeasureWearL` instruction with the optional argument `\RefChange` is executed, first a linear movement to a position about 10 mm outside *p2* is done. Then the gun is moved in the z direction in the tool coordinate system until the fixed tip touches the reference plate. During this reference measurement the reference plate is touched twice. When the measurement is ready some reference data is stored, `tw_ref_dist` in the `SWUSER.SYS` module.

The parameter `gun1` is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the `gundata` array `curr_gundata` in the `SWUSER.SYS` module. See *SWUSER on page 158*.

**Tip**

To verify if the selected measuring position or gun orientation is good enough a service routine is available; `ManualCheckMeasPos`. Run this routine is run in the selected position and status information will be presented on the FlexPendant whether the position is suitable or not. See *Manual actions on page 74*

**Arguments**

MeasureWearL ToPoint Speed GunNo [\GunD] [\Reference] | [\TipWear] | [\TipChange] | [\RefChange] [\SSearch], Tool [\WObj] [\TLoad]

`ToPoint`

Data type: robtarget

*Continues on next page*

The destination point for the robot and additional axes. This position should be a point close to the reference position, see figure in the example above. If this position is modified a new reference measurement has to be done.

`Speed`

Data type: speeddata

The speed data that applies to movements. Speed data defines the velocity for the tool center point, the tool reorientation and additional axes.

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the `gundata` array `curr_gundata` located in the `SWUSER.SYS` module. See *SWUSER on page 158*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`[\Reference]`

(reference measurement)

Data type: switch

Measurement type. This calibration type is used for the reference measurement with a reference tip with a well known TCP.

This measurement has to be done before the tip wear measuring is done the first time and each time when the TCP for this gun (with the reference tip mounted) is changed. It has also to be done if the reference plate (or reference position when a stationary gun is used) is dislocated of any reason.

If the reference plate is moved the switch `\RefChange` can be used instead.

Fore more information abot tip management, see *Tip wear compensation on page 180*.

`[\TipWear]`

(tip wear measurement)

Data type: switch

Measurement type. This measurement type is used when it is time to compensate for current tip wear, probably after each tip dressing. The data `curr_wear_fix` in `curr_gundata` will be automatically updated and the TCP in the used tooldata is recalculated.

For more information about tip management, see *Tip management on page 200* and *Tip wear compensation on page 180*.

`[\TipChange]`

(tip change measurement)

Data type: switch

*Continues on next page*

4.1.6 MeasureWearL - Measure current electrode wear and recalculate the TCP
*Continued*

Measurement type. This measurement type is used in the first measurement after tip change. The data `curr_wear_fix` in `curr_gundata` will be automatically updated and the TCP in the used tooldata is recalculated.

For more information about tip management, see *Tip management on page 200* and *Tip wear compensation on page 180*.

`[\RefChange]`

(reference changed measurement)

Data type: switch

Measurement type. This calibration type is used if the reference plate (or reference position when a stationary gun is used) is dislocated of any reason.

The reference tool `tw_ref_tool` in `SWUSER` module will not be updated if this calibration type is used.

`[\SSearch]`

(signal/sensor search)

Data type: switch

Measurement method. If this switch is used, the search will be done against a sensor signal instead of a fixed reference surface.

The required I/O signal that should be used is defined in the process configuration, see *The type Spot Gun Equipment on page 31*.

`Tool`

Data type: tooldata

The tool in use when the robot moves. The tool center point (TCP) is the point moved to the specified destination position, and should for a spot weld gun be the position on the tip of the fixed electrode.

**Note**

The TCP is automatically recalculated and changed when the optional argument `\TipWear` or `\TipChange` is used.

`[\WObj]`

Data type: wobjdata

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 gun is 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.

*Continues on next page*

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.

### Program execution

Internal sequence when a MeasureWearL instruction is executed:

1. The robot starts the movement to the destination position.
2. When the destination position is reached the search movements to the reference position or sensor is started.
3. If using the reference position search method the fixed tip will touch the reference position with a predefined pressure, this force can be modified by changing the setup data MeasureWearL TouchUp force in the process configuration. See *The type Spot SoftWare Equalizing on page 38*.
4. If the optional argument \Reference is used: Some reference data is stored in the user module swuser.sys, tw_ref_tool and tw_ref_dist.
5. If the optional argument \TipWear or \TipChange is used: The TCP in the used tooldata is recalculated and the data curr_wear_fix in curr_gundata is updated.

### Instruction by instruction execution

| | |
|---|---|
| Forward | As during continuous execution. |
| Backward | The motion is performed backwards to the destination position, but no measurement is activated. |

### Error handling

Following error situations are handled:

- If the search distance after tip wear measurement or measurement after tip change differs a lot from expected (for example missed tip). It is possible to change the tip change and tip wear supervision limit values, see *The type Spot Gun Equipment on page 31*.
- If the search sequence is interrupted by for example a Stop or Emergency Stop then the search sequence is automatically restarted from the beginning at program restart.

### Limitations

About how to place the fixed reference plate:

The reference plate can be mounted in an optional position in the work range, but it is necessary to orient the tool in the measuring position in that way that an

*Continues on next page*

4.1.6 MeasureWearL - Measure current electrode wear and recalculate the TCP
*Continued*

**additional torque** is generated on at least one of the robot motors when the robot is touching the reference position, preferably axis **4** to **6**.

> **Note**
>
> When using the reference plate search method there are occasions when the `MeasureWearL` is less suitable, for example very large guns and/or when an acceptable touchup position is not possible to reach for some reason (poor position). Then the `ReCalcTCP` method should be used instead.

> **Note**
>
> When using the sensor search method (\SSearch) a fast I/O response is critical for a good performance. A slow or inconsistent I/O response can give poor accuracy.

> **Note**
>
> It is only possible to run this instruction in semi coordinated mode.

## Syntax

```
MeasureWearL
  [ ToPoint ':=' ] < expression (IN) of robtarget > ','
  [ Speed ':=' ] < expression (IN) of speeddata > ','
  [ GunNo ':=' ] < expression (IN) of num >
  [ '\' GunD ':='] < persistent(PERS) of gundata > ]
  [ \Reference] | [\TipWear] | [\TipChange] | [\RefChange]
  [ '\' SSearch ] ','
  [ Tool ':=' ] < persistent (PERS) of tooldata > ]
  [ '\' WObj ':=' < persistent (PERS) of wobjdata > ]
  [ '\' TLoad':=' ] < persistent (PERS) of loaddata > ] ';'
```

## Related information

| | Described in: |
|---|---|
| Definition of velocity, speeddata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of tool, tooldata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of work objects, wobjdata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| MoveL | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Definition of load data, loaddata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| System module SWUSER | *SWUSER on page 158* |
| Definition of gundata | *gundata - Equipment specific weld data on page 135* |

*Continues on next page*

| | Described in: |
|---|---|
| Software Equalizing | *The type Spot SoftWare Equalizing on page 38* |

## 4.1.7 ReCalcTCP - Calculate current electrode wear and recalculate the TCP

### Description

`ReCalcTCP` is used in spot welding to calculate current electrode wear for the tip on the fixed electrode and then recalculate the used TCP to compensate for current tip wear. The calculation is based on stored information about the **total tip wear** and about the expected **tip wear ratio**, the wear of the fixed tip related to the total tip wear. The instruction also updates tip wear data in the used gundata.

This instruction can be used also for stationary guns.

This instruction is only available if the `Spot Servo Equalizing` option is installed.

### Example

In this example the gun can be hold by the robot or stationary. The principles are the same also when stationary guns are used.

#### Preparation

First the expected relation between the tip wear of the fixed tip and the total tip wear must be established, the data `Tip wear ratio, fixed vs total wear` in the process configuration must be set to a relevant value. For example 50, the wear of the fixed tip is 50% of the total wear. See *The type Spot Gun Equipment on page 31*.

This instruction has to be executed with the `\Reference` switch activated before it is used for tip wear compensation the first time. This also has to be done when the TCP for this gun, with new tips mounted, is changed for some reason. The TCP in the `tooldata` parameter, `ref_tool1` in this example, has to be valid for gun1 with new tips with the same size mounted.

```
ReCalcTCP gun1\Reference,ref_tool1;
tool1 := ref_tool1;
! tool1 is then used during the production.
```

When the `ReCalcTCP` instruction with the optional argument `\Reference` is executed some reference data (the `tooldata ref_tool1`) is stored internally in the user module swuser.sys. See *Data on page 159*.

The parameter `gun1` is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the `gundata` array `curr_gundata` located in the SWUSER module.

#### Compensation after tip wear

When it is time to compensate for tip wear, after each tip dressing, the `ReCalcTCP` instruction should be executed with the `\TipWear` switch activated. This has to be done after the gun calibration, since the total tip wear is updated during the calibration and used when executing `ReCalcTCP`.

```
Calibrate gun1\TipWear;(CalibL/J can also be used)
ReCalcTCP gun1\TipWear,tool1;
```

When the `ReCalcTCP` instruction with the optional argument `\TipWear` is executed, the TCP in the used tooldata (*tool1* in this example) is recalculated and the data `curr_wear_fix` in `curr_gundata` is automatically updated. The data

*Continues on next page*

`curr_tip_wear` and in gundata and the `Tip wear ratio, fixed vs total wear` in the process configuration is used for the calculations. See *The type Spot Gun Equipment on page 31*.

Reset the TCP after tip change

After the tips has been replaced with new ones, the instruction has to be executed, with the `\TipChange` switch activated.

```
Calibrate gun1\TipChange;
ReCalcTCP gun1\TipChange,tool1;
```

When the `ReCalcTCP` instruction with the optional argument `\TipChange` is executed, the TCP in the used tooldata, *tool1*, is set to the value used for new tips and the data `curr_wear_fix` in `curr_gundata` is cleared.

**Arguments**

ReCalcTCP GunNo [\GunD] [\Reference] | [\TipWear] | [\TipChange] Tool

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the `gundata` array `curr_gundata` located in the `SWUSER.SYS` module. See *SWUSER on page 158*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`[\Reference]`

Data type: switch

This switch is used for preparation of the calculations.This preparation has to be done before the tip wear compensation is done the first time and also each time the TCP for this gun (with new tips mounted) is changed.

The TCP in tooldata has to be valid for a gun with new tips mounted.

Fore more information abot tip managemant, see *Tip wear compensation on page 180*.

`[\TipWear]`

Data type: switch

This switch is used when it is time to compensate for current tip wear, probably after the gun calibration after each tip dressing. The data `curr_wear_fix` in `curr_gundata` will be automatically updated and the TCP in the used tooldata is recalculated. See *Additional components for Spot Servo Equalizing on page 136*.

For more information about tip management, see *Tip management on page 200* and *Tip wear compensation on page 180*.

*Continues on next page*

4.1.7 ReCalcTCP - Calculate current electrode wear and recalculate the TCP
*Continued*

`[\TipChange]`

Data type: switch

This switch is used when the instruction is executed after tip change. The data `curr_wear_fix` in `curr_gundata` is cleared and the TCP in the used tooldata is set to the value valid for new tips. See *Additional components for Spot Servo Equalizing on page 136*.

For more information about tip management, see *Tip management on page 200* and *Tip wear compensation on page 180*.

`Tool`

Data type: tooldata

Tooldata for the used gun. The tool center point (TCP) should for a spot weld gun be the tip position for the fixed electrode tip.

**Note**

The TCP in current `tooldata` is automatically recalculated and changed when the optional argument `\TipWear` or `\TipChange` is used.

## Program execution

Internal sequence when a `ReCalcTCP` instruction is executed:

- If the optional argument `\Reference` is used: Some reference data is stored internally.
- If the optional argument `\TipWear` is used: The TCP in the used tooldata is recalculated and the data `curr_wear_fix` in `curr_gundata` is updated.
- If the optional argument `\TipChange` is used: The TCP in the used tooldata is set to a value valid for new tips and the data `curr_wear_fix` in `curr_gundata` is cleared.

## Error handling

Following error situations are handled:

- If the calculated tip wear differ a lot from expected (for example missed tip or wrong sized tip). It is possible to change the tip wear supervision limit value if needed, see *The type Spot Gun Equipment on page 31*.

## Syntax

```
ReCalcTCP
  [ GunNo ':='] < expression (IN) of num >
  [ '\' GunD ':='] < persistent(PERS) of gundata > ]
  [ \Reference] | [\TipWear] | [\TipChange]
  [ Tool ':=' (PERS) of tooldata > ';'
```

## Related information

| | Described in: |
|---|---|
| Definition of gun data, gundata | *gundata - Equipment specific weld data on page 135* |

*Continues on next page*

| | Described in: |
|---|---|
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| System module SWUSER | *SWUSER on page 158* |
| Software Equalizing | *The type Spot SoftWare Equalizing on page 38* |

## 4.1.8 IndGunMove - Activates independent mode for a servo gun

### Description

`IndGunMove` (Independent Gun Movement) is used to set the gun in independent mode and thereafter move the gun to a specified independent position. The gun will stay in independent mode until the instruction `IndGunMoveReset` is executed.

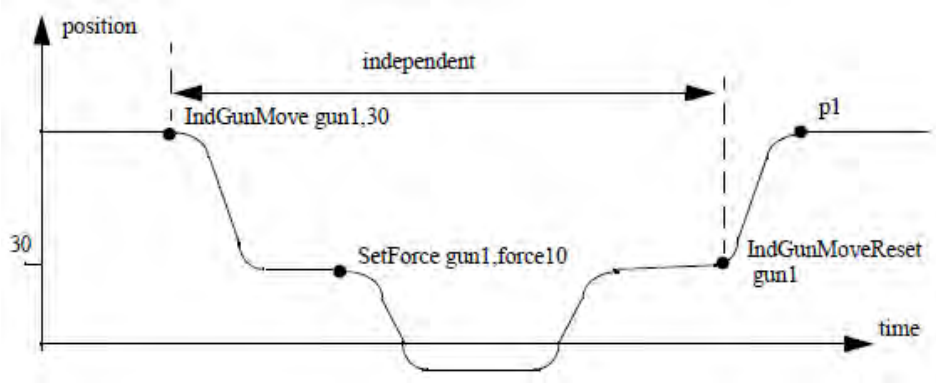
During independent mode, the control of the servo gun is separated from the robot. The gun can be closed, opened, calibrated or moved to a new independent position, but it will not follow coordinated robot movements.

It is also possible to set the gun in independent mode from a background task while the robot in the main task can continue with for example move instructions. For more information of how to set the gun in independent mode, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

### Example

```
PROC tipdress()
! Note that the gun will move to current robtarget position, if
     already in independent mode.
IndGunMoveReset gun1;
.......
.......
.......
IndGunMove gun1, 30;
......
SetForce gun1, force10;
......
IndGunMoveReset gun1;
ENDPROC
```

Independent mode is activated and the gun is moved to an independent position (30 mm). During independent mode the instruction `SetForce` is executed, without interfering with robot motion. The instruction `IndGunMoveReset` will take the gun out of independent mode and move the gun to current robtarget position.




xx1200000207

The position p1 depends on the position of the gun given in the robtarget just performed by the robot.

*Continues on next page*

## Arguments

IndGunMove GunNo [\GunD] GunPos

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the gundata array `curr_gundata` located in the `SWUSER.SYS` module. See *SWUSER on page 158*

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

`GunPos`

Data type: num

The position (stroke) of the servo gun in mm.

## Program execution

The instruction activates independent mode and moves the gun from the coordinated position to a specified independent position. During the independent mode the gun may be closed, opened, calibrated or moved to a new independent position without interfering with robot motion.

Program restart during independent mode will always start with a regain movement to the current independent position.

The gun will recover independent mode after a system restart. Moving the program pointer will NOT reset independent mode. When the program is started, no regain movement will occur but the gun will return to independent mode after the first gun closing or calibration.

## Limitations

It is not possible to use this instruction is used in combination with spot instructions when `Spot Servo Equalizing` is activated.

## Syntax

```
IndGunMove
  [ GunNo ':=' <expression (IN) of num> ]
  [ '\' GunD ':='] < persistent(PERS) of gundata > ] ','
  [ GunPos ':=' <expression (IN) of num> ] ';'
```

## Related information

| | Described in: |
|---|---|
| `SetForce` | *SetForce - Close and Open a gun with desired force and time on page 104* |
| `STIndGun` | *Technical reference manual - RAPID Instructions, Functions and Data types* |

## 4.1.9 IndGunMoveReset - Resets servo gun from independent mode

### Description

`IndGunMoveReset` (Independent Gun Movement Reset) is used to reset the gun from independent mode and thereafter move the gun to current robtarget position.

### Example

```
IndGunMoveReset gun1;
```

### Arguments

IndGunMoveReset GunNo [\GunD]

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the gundata array `curr_gundata` located in the `SWUSER.SYS` module. The gun was previously set independent with the instruction `IndGunMove`.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

### Program execution

The instruction will reset the gun from independent mode and move the gun to current robtarget position. During this movement the coordinated speed of the gun must be zero, otherwise an error will occur. The coordinated speed will be zero if the robot is standing still or if the current robot movement includes a "zero movement" of the gun.

### Syntax

```
IndGunMove
  [ GunNo ':=' <expression (IN) of num> ]
  [ '\' GunD ':='] < persistent(PERS) of gundata > ] ';'
```

### Related information

| | Described in: |
|---|---|
| Definition of gundata | *gundata - Equipment specific weld data on page 135* |

## 4.1.10 OpenHighLift/CloseHighLift - Control the position of a pneumatic gun

### Description

`OpenHighLift` is used in spot welding to open a pneumatic gun to the highlift position (large gap).

`CloseHighLift` is used in spot welding to close a pneumatic gun to the work stroke position (small gap).

### Example

```
OpenHighLift, gun1;
```

- The gun `gun1` is opened to the highlift position.

```
CloseHighLift, gun1;
```

- The gun `gun1` is closed to the work stroke position.

The parameter `gun1` is a `num` corresponding to the used gun equipment. All gun equipment used are defined in the gundata array `curr_gundata` in SWUSER.

### Arguments

OpenHighLift GunNo [\GunD]

CloseHighLift GunNo [\GunD]

`GunNo`

Data type: num

Used gun equipment number. Corresponding to the element number in the gundata array `curr_gundata` in SWUSER.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 135*.

Can be used if external gundata is required. If used the external gundata will be temporary stored in the curr_gundata array during the process.

### Program execution

Internal sequence when a `OpenHighLift` instruction is executed:

- The `inhibit_close` is simulated.
- The user routine `SwInitUserIO` is executed.
- The user routine `SwOpenGun` is executed and the gun is opened to the highlift position.

Internal sequence when a `CloseHighLift` instruction is executed:

- The `inhibit_close` is simulated.
- The user routine `SwInitUserIO` is executed.
- The user routine `SwCloseGun` is executed and the gun is closed to the work stroke position.

*Continues on next page*

4.1.10 OpenHighLift/CloseHighLift - Control the position of a pneumatic gun
*Continued*

### Instruction by instruction execution

| | |
|---|---|
| Forward | As during continuous execution. |
| Backward | As during continuous execution. |

### Error handling

No error handling.

### Syntax

```
OpenHighLift or CloseHighLift
  [ GunNo ':='] < expression (IN) of num >
  [ '\' GunD ':='] < persistent(PERS) of gundata > ] ';'
```

### Related information

| | Described in: |
|---|---|
| Overview *Spot Servo* | *Introduction to RobotWare Spot on page 15* |

## 4.2 Data types

### 4.2.1 gundata - Equipment specific weld data

**Description**

`Gundata` is used to define spot weld equipment specific data, to control the gun in an optimal way in the weld process when the spot instructions are used. Each gundata defines one gun equipment.

> **Note**
>
> The `gundata` structure and order of parameters differs between different spot options.

`Gundata` has the following default structure when servo guns are used:

- Gun name
- Weld counter and a max value.
- Current tip wear and a max value.
- Specific parameters for the Software Equalizing functions. (Only if Spot Servo Equalizing is installed.)

`Gundata` has the following default structure when pneumatic guns are used:

- Gun name
- Weld counter and a max value.

**Components**

`gun_name`

(gun name)

Data type: string

The name of the mechanical unit used for the servo gun. This name must be identical with the name of the mechanical unit defined in the motion servo gun parameters.

Normally the gun name will be updated automatically at startup. A service routine is available to search the system for servo guns and update the gun name, ManualGunSearch, see *Manual actions on page 74*.

`weld_counter`

(weld counter)

Data type: num

Counter for the number of welds done with this gun. The counter is automatically incremented after process is ready. Use of this data is optional. Zero set shall be handled by the user program.

`max_nof_welds`

(max number of welds)

Data type: num

*Continues on next page*

4.2.1 gundata - Equipment specific weld data
*Continued*

Max number of performed welds. Use of this data is optional.

`curr_tip_wear`

(current tip wear)

Data type: num

Current tip wear [mm]. This data is automatically updated after each gun calibration. Use of this data is optional. (Servo guns only).

`max_tip_wear`

(max tip wear)

Data type: num

Max allowed tip wear before tip exchange [mm]. Use of this data is optional. (Servo guns only).

**Additional components for Spot Servo Equalizing**

`curr_wear_fix`

(current tip wear for the fixed tip)

Data type: num

Current tip wear for the fixed gun electrode tip [mm]. This data is automatically updated when `MeasureWearL` or `ReCalcTCP` is used.

`wear_moveable`

(current tip wear for the moveable tip)

Data type: num

Current tip wear for the moveable gun electrode tip [mm]. This data is automatically updated when `CalibL/J` and `Calibrate` is used.

`release_dist`

(release distance)

Data type: num

The release distance [mm] when the robot is moving between weld positions during normal program execution and during Weld position Touch Up.

`deflection_dist_z`

(deflection distance in z-direction)

Data type: num

TCP deviation [mm] in z-direction caused of gun arm deflection when the gun is closed with the force specified in `deflection_force`. This data is used for the deflection compensation movement of the robot. Default value 0 mm.

Only positive values are allowed, the `Opposite z-direction` parameter determines the direction for the deflection compensation. See *The type Spot Gun Equipment on page 31*.

`deflection_dist_x`

(deflection distance in x-direction)

Data type: num

*Continues on next page*

TCP deviation [mm] in x-direction caused of gun arm deflection when the gun is closed with the force specified in `deflection_force`. This data is used for the deflection compensation of the robot. Default value 0mm.

This value can be both positive and negative depending on which direction the deflection compensation shall be performed.

Example: If the gun bends outwards 2mm (positive x), this value should be set to 2mm.

`deflection_force`

(deflection force)

Data type: num

Applied force [N] corresponding to the TCP deviation `deflection_dist_` parameters caused of gun arm deflection. This data is used for the deflection compensation.

`deflection_time`

(deflection time)

Data type: num

The time for the gun to build up the gun force [s]. This data is used for the deflection compensation. If no data information exists, use the default value (0.1 s).

**Default structure**

For servo guns if Spot Servo is installed:

```
<dataobject of gundata>
  <gun_name of num>
  <weld_counter of num>
  <max_nof_welds of num>
  <curr_tip_wear of num>
  <max_tip_wear of num>
```

For servo guns if Spot Servo Equalizing is installed:

```
<dataobject of gundata>
  <gun_name of num>
  <weld_counter of num>
  <max_nof_welds of num>
  <curr_tip_wear of num>
  <max_tip_wear of num>
  <curr_wear_fix of num>
  <wear_moveable of num>
  <release_dist of num>
  <deflection_dist_z of num>
  <deflection_dist_x of num>
  <deflection_force of num>
  <deflection_time of num>
```

For pneumatic guns:

```
<dataobject of gundata>
  <gun_name of num>
  <weld_counter of num>
  <max_nof_welds of num>
```

*Continues on next page*

4.2.1 gundata - Equipment specific weld data
*Continued*

### Predefined data

For servo guns if Spot Servo is installed:

```
PERS gundata curr_gundata{4} :=
  [["SGUN_1", 0, 1000, 0, 10],
  ["NOT USED", 0, 1000, 0, 10],
  ["NOT USED", 0, 1000, 0, 10],
  ["NOT USED", 0, 1000, 0, 10]];
```

For servo guns if Spot Servo Equalizing is installed:

```
PERS gundata curr_gundata{4} :=
  [["SGUN_1",0,1000,0,10,0,0,5,0,0,5000,0.1],
  ["NOT USED",0,1000,0,10,0,0,5,0,0,5000,0.1],
  ["NOT USED",0,1000,0,10,0,0,5,0,0,5000,0.1],
  ["NOT USED",0,1000,0,10,0,0,5,0,0,5000,0.1]];
```

For pneumatic guns:

```
PERS gundata curr_gundata{4} :=
  [["PGUN_1", 0, 1000],
  ["PGUN_2", 0, 1000],
  ["PGUN_3", 0, 1000],
  ["PGUN_4", 0, 1000]];
```

`curr_gundata` is an array with active gundata parameters for each used gun. These parameters have to be changed by the user during the installation and programming phase to be in agreement with the weld equipment in use. In the default package, `curr_gundata` is defined in module SWUSER.

It is also possible to use external gundata in the spot instructions, *Arguments on page 95*.

**Note**

The size of the gundata array may depend on the selected spot configuration, for a single gun configuration the size of the array will be only one instance.

### Customizing

The Spot package provides opportunities for the user to customize the functionality to adapt to different types of spot weld 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.

See *Customizing RobotWare-Spot on page 247*.

### Related information

| | Described in: |
|---|---|
| `SpotL/SpotJ` | *SpotL/SpotJ - The basic spot welding instructions on page 95* |
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| Customizing possibilities | *Customizing RobotWare-Spot on page 247* |

*Continues on next page*

| | Described in: |
|---|---|
| Definition of spot data, spotdata | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| System module SWUSER | *SWUSER on page 158* |

## 4.2.2 spotdata - Spot weld data

### Description

`Spotdata` is used to define the parameters that control the weld equipment when welding a certain spot.

`Spotdata` is used by the `SpotL/J` and `SpotML/J` instructions and contains data which controls the welding of one spot.

`Spotdata` has the following default structure when servo guns are used:

- Program number for the program in the weld timer to be used.
- Desired gun tip force.
- Expected total plate thickness.

  Only valid for servo guns.
- Allowed variation when checking the plate thickness.

  Only valid for servo guns.

> **Tip**
>
> It is possible to use spot data parameters programmed in the weld timer if needed instead of the spotdata parameters, see
>
> *How to use spot data programmed in the weld timer on page 258*

### Components

`prog_no`

(program number)

Data type: dnum

Defines the internal program in the weld timer to be used for the welding. This data will set an output group signal when a spot instruction is run.

Permitted values: 0 - to the size of the I/O group for the equipment.

Absolute max is the size of dnum, 4294967295 - 32bit.

`tip_force`

(gun tip force)

Data type: num

Defines the desired gun tip force. [N]

(If pneumatic gun, this value controls a group output)

Permitted values: -1, to the defined max gun force, see *The type Spot Gun Equipment on page 31*

-1 will disable this parameter, and external timer data will be used instead, see *How to use spot data programmed in the weld timer on page 258*.

`plate_thickness`

(plate thickness) (Only valid for servo guns)

*Continues on next page*

Data type: num

Defines the expected total plate thickness. [mm].

Permitted values: -1, to the defined max plate thickness, see *The type Spot System on page 23*.

-1 will disable this parameter, and external timer data will be used instead, see *How to use spot data programmed in the weld timer on page 258*.

`plate_tolerance`

(plate tolerance) (Only valid for servo guns)

Data type: num

Defines the allowed variation when checking the plate thickness [mm]

If the value is 0 the thickness check is deactivated.

Permitted values: -1, to the defined max plate tolerance, see *The type Spot System on page 23*.

-1 will disable this parameter, and external timer data will be used instead, see *How to use spot data programmed in the weld timer on page 258*.

**Predefined data**

Servo guns:

```
PERS spotdata spot1 := [1, 1000, 0, 0];
```

Pneumatic guns:

```
PERS spotdata spot1 := [1, 1];
```

Defined in module SWUSRM.

`Spot1` is used as default in the first programmed Spot instruction and has following default data:

- The program number 1 in the weld controller shall be used.
- Desired gun tip force = 1000 N. (Gun pressure level 1 for pneumatic guns).
- Expected total plate thickness = 0 mm. (Servo guns).
- Allowed variation in the thickness = 0 (thickness check is deactivated) (Servo guns).

Servo guns:

```
PERS spotdata
curr_spotdata{4} := [[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0 ]];
```

Pneumatic guns:

```
PERS spotdata curr_spotdata{4} := [[0,0],[0,0],[0,0],[0,0]];
```

Defined in module SWUSER.

`curr_spotdata` is an array with active or latest used `spotdata` parameters for each defined gun. This parameters are automatically updated by the kernel when spot instructions are executed. This `spotdata` are used for reweld situations and if welding is manually activated (see *Manual actions on page 74*)

*Continues on next page*

4.2.2 spotdata - Spot weld data
*Continued*

### Customizing

The Spot package provides opportunities for the user to customize the functionality to adapt to different types of spot weld 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.

See *How to change the Spot data types on page 255*.

### Default structure

For servo guns:

```
<dataobject of spotdata>
  <prog_no of dnum>
  <tip_force of num>
  <plate_thickness of num>
  <plate_tolerance of num>
```

For pneumatic guns:

```
<dataobject of spotdata>
  <prog_no of dnum>
  <tip_force of num>
```

### Related information

| | Described in: |
|---|---|
| SpotL/J | *SpotL/SpotJ - The basic spot welding instructions on page 95* |
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| Customizing possibilities | *Customizing RobotWare-Spot on page 247* |
| Definition of gun data, gundata | *gundata - Equipment specific weld data on page 135* |
| System module SWUSER | *SWUSER on page 158* |

## 4.2.3 forcedata - Spot gun force data

**Description**

`Forcedata` is used to define the parameters for control of the spot weld gun when it is closed without welding, e.g. when tip dressing.

`Forcedata` is used when a `SetForce` instruction is run, or from certain manual actions.

It has the following **default** structure when servo guns are used:

- Desired gun tip force.
- Desired force time.
- Expected total plate thickness. (Only valid for servo guns.)

Allowed variation when checking the plate thickness. (Only valid for servo guns.)

**Components**

`tip_force`

(gun tip force)

Data type: num

Defines the desired gun tip force [N]. If pneumatic gun, this value controls a group output.

`force_time`

(gun force time)

Data type: num

Defines the desired gun force time [s].

`plate_thickness`

Data type: num

Defines the expected total plate thickness. [mm] (Only valid for servo guns).

`plate_tolerance`

(plate tolerance)

Data type: num

Defines the allowed variation when checking the plate thickness [mm]

If the value is 0 the thickness check is deactivated. (Only valid for servo guns).

`\tip_force2`

(optional second gun tip force)

Data type: num

Defines the second gun tip force [N]. This data can be used when a second gun force is required. See *SwSetIntForceData - Set the internal forcedata on page 153*.

`\force_time2`

(optional second gun force time)

Data type: num

*Continues on next page*

4.2.3 forcedata - Spot gun force data
*Continued*

Defines the second gun force time [s]. This data can be used when a second force time is required. See *SwSetIntForceData - Set the internal forcedata on page 153*

**Predefined data**

Servo guns:

```
PERS forcedata force1 := [1000, 1, 0, 0];
```

Pneumatic guns:

```
PERS forcedata force1 := [1, 1];
```

Defined in module SWUSRM.

force1 is used as default in the first programmed `SetForce` instruction and has following default data:

- Desired gun tip force = 1000 N.

  (Gun pressure 1 for pneumatic gun)
- Desired force time = 1 s.
- Expected total plate thickness = 0 mm.

  (Servo gun)
- Allowed variation in the thickness = 0 (thickness check is deactivated)

  (Servo gun)

Servo guns:

```
PERS forcedata curr_forcedata{2} := [[0,0,0,0],[0,0,0,0]];
```

Pneumatic guns:

```
PERS forcedata curr_forcedata{2} := [[0,0],[0,0]];
```

Defined in module SWUSER.

`curr_forcedata` is an array with active or latest used forcedata parameters for each defined gun. This parameters are automatically updated by the kernel when a `SetForce` instruction is executed. The parameters are used when gun closure is manually activated, see *Manual actions on page 74*.

**Customizing**

The Spot package provides opportunities for the user to customize the functionality to adapt to different types of spot weld 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. See *Customizing RobotWare-Spot on page 247*.

**Default structure**

For servo guns:

```
<dataobject of forcedata>
  <tip_force of num>
  <force_time of num>
  <plate_thickness of num>
  <plate_tolerance of num>
```

*Continues on next page*

For pneumatic guns:

```
<dataobject of forcedata>
  <tip_force of num>
  <force_time of num>
```

## Related information

| | Described in: |
|---|---|
| `SetForce` | *SetForce - Close and Open a gun with desired force and time on page 104* |
| Overview Spot options | *Introduction to RobotWare Spot on page 15* |
| Customizing possibilities | *Customizing RobotWare-Spot on page 247* |
| Definition of spot data, spotdata | *spotdata - Spot weld data on page 140* |
| System module SWUSER | *SWUSER on page 158* |

## 4.2.4 simdata - Simulation data

### Description

`Simdata` is used to define the parameters that control the different simulation modes used when testing spot weld programs.

`Simdata` has the following default structure when servo guns are used:

- Desired simulation type.
- Desired simulation time.
- If testing is performed with/without gun closure.
- If testing is performed with/without plates. (Only valid for servo guns).

### Components

`sim_type`

(simulation type)

Data type: num

Desired simulation type. Permitted values:

| | |
|---|---|
| 0 | All simulations are deactivated. (Weld mode) |
| 1 | Simulation of the weld is performed in the robot controller. (No start signal to weld controller). Program valid check is done and timer input groups are checked if configured. |
| 2 | Simulation of the weld is performed in the weld controller with no current. The signal `enable_current` is reset. |
| 3 | Weld position Touch Up mode. Only available if Spot Servo Equalizing is installed. |

`sim_time`

(simulation time)

Data type: num

Defines the desired simulation time [s] when simulation of the weld is performed in the robot controller (`sim_type = 1`).

`inhib_close`

(inhib close)

Data type: bool

Testing without closing the guns. Only relevant if `sim_type = 1 or 2`.

`no_plates`

(no plates)

Data type: bool

Testing without plates. Only relevant if `sim_type = 1 or 2`. If this mode is set, the robot and gun will move to the nominal positions. The plate thickness supervision, SoftMove Equalizing will be disabled to be able to run a cycle without parts. (Only valid for servo guns).

*Continues on next page*

### Predefined data

Servo guns:

```
PERS simdata data curr_simdata := [0, 0.5, FALSE, FALSE];
```

Pneumatic guns:

```
PERS simdata data curr_simdata := [0, 0.5, FALSE];
```

Defined in module SWUSER.

`curr_simdata` is holding all active simulation data. This data influences all used weld equipment when `SpotL/J` or `SpotML/J` instructions are executed. The user has to change this data to activate a simulation mode. All simulations are deactivated if `sim_type = 0` (default).

### Customizing

The Spot package provides opportunities for the user to customize the functionality to adapt to different types of spot weld 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.

See *Customizing RobotWare-Spot on page 247*.

### Default structure

For servo guns:

```
<dataobject of simdata>
  <sim_type of num>
  <sim_time of num>
  <inhib_close of bool>
  <no_plates of bool>
```

For pneumatic guns:

```
<dataobject of simdata>
  <sim_type of num>
  <sim_time of num>
  <inhib_close of bool>
```

### Related information

| | Described in: |
|---|---|
| `SpotL` | *SpotL/SpotJ - The basic spot welding instructions on page 95* |
| `SpotML` | *SpotML/SpotMJ - Spot welding with multiple guns on page 100* |
| Overview *Spot* & *Spot Servo* | *Introduction to RobotWare Spot on page 15* |
| Customizing possibilities | *Customizing RobotWare-Spot on page 247* |
| System module SWUSER | *SWUSER on page 158* |

## 4.2.5 smeqdata - SoftMove Equalizing data

### Description

`smeqdata` is used to define the parameters used to control the SoftMove function that can be used when welding a certain spot. This data has to be used by the `SpotL/J` instructions if SoftMove equalizing is required.

`smeqdata` has the following structure:

- Desired `smeq_type` of the robot for the specific position.
- Desired `force_offset` friction compensation for the specific position.

### Components

`smeq_type`

(SoftMove type of the robot)

Data type: smeqtype (alias num)

Defines what SoftMove type that shall be used in the specific position, there are 2 types possible, no gun deflection mode(0), standard mode(1).

**0 - No gun deflection mode:** During the final robot movement to the position and closing of the gun the robot will be set into soft state using the specified `force_offset` value during the complete weld cycle.

**1 - Standard mode:** During the final robot movement to the position and closing of the gun the robot will be set into soft state using the specified `force_offset` value, during the weld cycle the robot will be stiff and gun arm deflection will be used.

`force_offset`

(friction compensation)

Data type: num

Defines the desired force in Newton, corresponding to the static friction of the robot in the soft z direction and the force needed to achieve a small movement to the plate. This value must be set for **each** position the first time the program is executed.

If the force value is too low it will be difficult for the robot to reach the position, and if the force value is too high the robot may damage the plate. Possible values for this parameter are in between 0.1 and 2000 N. The friction compensation needed will be measured for each `SpotL/J` instruction if the initial value is zero, see *Friction compensation procedure on page 191*.

> **Note**
>
> The `force_offset` should be seen as friction force and force applied to the plates.
>
> Example: 100N is identified in the friction compensation procedure.
>
> To be sure that robot will move to the plate the `force_offset` can be set to 130N. If the **friction is constant** during the movement it will be 30N applied on the plate.

*Continues on next page*

## Example

Program example with and without SoftMove Equalizing functionality:

```
PERS smeqdata smeq10:=[1,150];
PERS smeqdata smeq11:=[1,170];

PROC main()
  MoveJ P10, v1000, z50, tool1;
  SpotL P20, vmax, gun1, spot11, tool1;
  SpotL P30, vmax, gun1, spot12 \SMEQ:=smeq10, tool1;
  SpotL P40, vmax, gun1, spot13 \SMEQ:=smeq11, tool1;
  SpotL P50, vmax, gun1, spot14, tool1;
ENDPROC
```

## Default structure

```
<dataobject of smeqdata>
  <smeq_type of smeqtype>
  <force_offset of num>
```

## Related information

| | Described in: |
|---|---|
| SpotL/J | *SpotL/SpotJ - The basic spot welding instructions on page 95* |
| SoftMove Equalizing | *SoftMove Equalizing on page 185* |
| SoftMove | *Application manual - SoftMove* |

## 4.3 Global instructions and functions

### 4.3.1 SwGetCurrTargetName - Get the current robtarget name

**Description**

`SwGetCurrTargetName` can be used in the user routines to retrieve the current robtarget name for each spot instruction.

**Example**

A basic example of the function `SwGetCurrTargetName` used in the `SwPrepare` hook is illustrated below.

```
PROC main()
  SpotL p100, vmax, gun1, spot10, tool1;
ENDPROC
PROC SwPrepare(num GunNum, string ErrText)
  VAR string tmp_str;
  tmp_str := SwGetCurrTargetName();
  TPWrite "Current robtarget name - "+tmp_str;
ENDPROC
```

1. The robot executes a `SpotL` instruction to the position `p100`.
2. In the *SwPrepare* user hook the function *SwGetCurrTargetName* is called.
3. The `TPWrite` instruction will write "`p100`" on the FlexPendant.

**Return value**

Data type: string

If a spot instruction has been executed prior to this, the robtarget name will be returned, otherwise an empty string will be returned.

**Syntax**

```
SwGetCurrTargetName '(' ')'
```

A function with a return value of the data type string.

## 4.3.2 SwGetCurrSpotName - Get the current spotdata name

### Description

`SwGetCurrSpotName` can be used in the user routines to retrieve the current `spotdata` name for each spot instruction.

### Example

A basic example of the function `SwGetCurrSpotName` used in the `SwPrepare` hook is illustrated below.

```
PROC main()
  SpotL p100, vmax, gun1, spot10, tool1;
ENDPROC
PROC SwPrepare(num GunNum, string ErrText)
  VAR string tmp_str;
  tmp_str := SwGetCurrSpotName();
  TPWrite "Current spotdata name - "+tmp_str;
ENDPROC
```

1. The robot executes a `SpotL` instruction to the position `p100`.
2. In the *SwPrepare* user hook the function *SwGetCurrSpotName* are called.
3. The `TPWrite` instruction will write "spot10" on the FlexPendant.

### Return value

Data type: string

If a spot instruction has been executed prior to this, the `spotdata` name will be returned, otherwise an empty string will be returned.

### Syntax

```
SwGetCurrSpotName '(' ')'
```

A function with a return value of the data type string.

## 4.3.3 SwSetIntSpotData - Set the internal spotdata

### Description

The `SwSetIntSpotData` routine is used to transfer user `spotdata` components to internally used `spotdata` components.

### Example

An example of the function `SwSetIntSpotData` used in the `DefineSpotData` routine is illustrated below.

```
PROC DefineSpotData(spotdata Spot, num GunNum)
  SwSetIntSpotData GunNum \ProgNo:=Spot.prog_no
       \TipForce:=Spot.tip_force;
ENDPROC
```

1. The robot executes a `SpotL` instruction and calls the `SwSetIntSpotData` routine.
2. The internally used `spotdata` parameters are updated and used during the process.

### Syntax

```
[ GunNum ':=' ] < expression (IN) of num >
[ '\' ProgNo ':=' ] < expression (IN) of dnum >
[ '\' TipForce ':='] < expression (IN) of num > ]
[ '\' PlateThickness ':='] < expression (IN) of num > ]
[ '\' PlateTolerance ':=' ] < expression (IN) of num > ]';'
```

## 4.3.4 SwSetIntForceData - Set the internal forcedata

### Description

The `SwSetIntForceData` routine is used to transfer user `forcedata` components to internally used `forcedata` components.

### Example

An example of the function `SwSetIntForceData` used in the `DefineForceData` routine is illustrated below.

```
PROC DefineForceData(forcedata Force, num GunNum)
  SwSetIntForceData GunNum \TipForce:=Force.tip_force
    \ForceTime:=Force.force_time
    \PlateThickness:=Force.plate_thickness
    \PlateTolerance:=Force.plate_tolerance;
ENDPROC
```

Example of the function `SwSetIntForceData` routine with optional force data parameters is illustrated below.

```
PROC DefineForceData(forcedata Force, num GunNum)
  SwSetIntForceData GunNum \TipForce:=Force.tip_force
    \ForceTime:=Force.force_time
    \PlateThickness:=Force.plate_thickness
    \PlateTolerance:=Force.plate_tolerance
    \TipForce2:=Force.tip_force2
    \ForceTime2:=Force.force_time2;
ENDPROC
```

1. The robot executes a `SetForce` instruction and calls the `SwSetIntForceData` routine.
2. The internally used `forcedata` parameters are updated and used during the process.

### Syntax

```
[ GunNum ':=' ] < expression (IN) of num >
[ '\' TipForce ':='] < expression (IN) of num > ]
[ '\' ForceTime ':=' ] < expression (IN) of num > ]
[ '\' PlateThickness ':='] < expression (IN) of num > ]
[ '\' PlateTolerance ':=' ] < expression (IN) of num > ]
[ '\' TipForce2 ':='] < expression (IN) of num > ]
[ '\' ForceTime2 ':=' ] < expression (IN) of num > ]';'
```

## 4.3.5 SwSetIntGunData - Set the internal gundata

### Description

The `SwSetIntGunData` routine is used to transfer user `gundata` components to internally used `gundata` components.

### Example

An example of the function `SwSetIntGunData` used in the `DefineGunData` routine is illustrated below.

```
PROC DefineGunData()
  SwSetIntGunData GunNum \GunName:=curr_gundata{GunNum}.gun_name
       \TotalTipWear:=curr_gundata{GunNum}.curr_tip_wear;
ENDPROC
```

1. The robot executes a `SpotL` instruction and calls the `SwSetIntGunData` routine.
2. The internally used `gundata` parameters are updated and used during the process.

### Syntax

```
[ GunNum ':=' ] < expression (IN) of num >
[ '\' GunName ':=' ] < expression (IN) of string >
[ '\' CurrWearFix ':='] < expression (IN) of num > ]
[ '\' CurrWearMov ':='] < expression (IN) of num > ]
[ '\' TotalTipWear ':=' ] < expression (IN) of num > ]
[ '\' MaxTipWear ':=' ] < expression (IN) of num > ]
[ '\' ReleaseDist ':=' ] < expression (IN) of num > ]
[ '\' DeflectionDistZ ':=' ] < expression (IN) of num > ]
[ '\' DeflectionDistX ':=' ] < expression (IN) of num > ]
[ '\' DeflectionForce ':=' ] < expression (IN) of num > ]
[ '\' DeflectionTime ':=' ] < expression (IN) of num > ] ';'
```

## 4.3.6 SwSetIntSimData - Set the internal simdata

### Description

The `SwSetIntSimData` routine is used to transfer user `simdata` components to internally used `simdata` components.

### Example

An example of the function `SwSetIntSimData` used in the `DefineSimData` routine is illustrated below.

```
PROC DefineSimData()
  SwSetIntSimData \SimType:=curr_simdata.sim_type
       \SimTime:=curr_simdata.sim_time
       \InhibGunClose:=curr_simdata.inhib_close;
ENDPROC
```

1. The robot executes a `SpotL` instruction and calls the `SwSetIntSimData` routine.
2. The internally used `simdata` parameters are updated and used during the process.

### Syntax

```
[ '\' SimType ':='] < expression (IN) of num > ]
[ '\' SimTime ':=' ] < expression (IN) of num >
[ '\' InhibGunClose ':='] < expression (IN) of bool > ]
[ '\' PlatesCheck ':=' ] < expression (IN) of bool > ]';'
```

## 4.3.7 SwGetCalibData - Get the latest total tip wear and position adjustment

### Description

SwGetCalibData can be used to retrieve the current total tip wear and positional adjustment for the specified gun after a CalibL/J or Calibrate instruction has been run.

### Example

A basic example of the routine SwGetCalibData is illustrated below.

```
PROC MyProc()
    VAR num curr_tip_wear;
    VAR num curr_pos_adj;
    CalibL p10, vmax, gun1\TipChg, z50, tool1;
    SwGetCalibData \CurrTipWear:=curr_tip_wear
         \CurrPosAdj:=curr_pos_adj;
ENDPROC
```

- The robot executes a CalibL instruction to the position p10.
- The instruction SwGetCalibData are called.
- The curr_tip_wear and curr_pos_adj variables are assigned with the return values from the CalibL instruction.

### Syntax

```
[ GunNum ':=' ] < expression (IN) of num > ]
[ '\' CurrTipWear ':='] < expression (INOUT) of num > ]
[ '\' CurrPosAdj ':=' ] < expression (INOUT) of num > ]';'
```

## 4.3.8 SwGetFixTipData - Get the latest fixed tip wear and position adjustment

### Description

SwGetFixTipData can be used to retrieve the current fixed tip wear and positional adjustment for the specified gun after a MeasureWearL or ReCalcTCP instruction has been run..

### Example

A basic example of the routine SwGetFixTipData is illustrated below.

```
PROC MyProc()
    VAR num curr_tip_wear;
    VAR num curr_pos_adj;
    ReCalcTCP gun1\TipWear, tool1;
    SwGetFixTipData \CurrTipWear:=curr_tip_wear
         \CurrPosAdj:=curr_pos_adj;
ENDPROC
```

- The robot executes a ReCalTCP instruction.
- The instruction SwGetFixTipData is called.
- The curr_tip_wear and curr_pos_adj variables are assigned with the return values from the ReCalTCP instruction.

### Syntax

```
[ GunNum ':=' ] < expression (IN) of num > ]
[ '\' CurrTipWear ':='] < expression (INOUT) of num > ]
[ '\' CurrPosAdj ':=' ] < expression (INOUT) of num > ]';'
```

## 4.4 System modules

### 4.4.1 SWUSER

**Description**

The SWUSER user module is configured to run in **all tasks** in the system, and contains the spot data definitions, and routines that can be used to shape the behavior of the process, e.g add additional supervisions in the process sequence if needed.

In normal cases there is no need to change this module. he default functionality should be good enough in most cases. But if the default data types has to be modified and/or additional logic has to be added in the process sequence this module needs to be changed.

It contains process routines (hooks) (for example SwPreWeld and SwPostWeld), and it also contains a supervision task routine where custom functionality/supervision can be added if needed.

The process routines has no default functionality, but can easily be changed to fit different environment/equipment in case the default process behavior is not suitable, for example, add supervision in the process sequence.

**Note**

Default content depends on the spot configuration.

**Note**

After changing any routines in SWUSER, the following steps must be taken before there is an effect on the application:

- Save SWUSER. The old one is overwritten.
- Generate a Reset Rapid restart to affect all tasks.

**Data definitions**

The following global data records are predefined.

| Record data | Description | Default value |
|---|---|---|
| forcedata | Definition of force data | num tip_force;<br>num force_time;<br>num plate_thickness;<br>num plate_tolerance; |
| simdata | Definition of simulation data | num sim_type;<br>num sim_time;<br>bool inhib_close;<br>bool no_plates; |

*Continues on next page*

| Record data | Description | Default value |
|---|---|---|
| spotdata | Definition of process spot data | dnum prog_no;<br>num tip_force;<br>num plate_thickness;<br>num plate_tolerance; |
| gundata | Definition of process gun data | string gun_name;<br>num weld_counter;<br>num max_nof_welds;<br>num curr_tip_wear;<br>num max_tip_wear;<br>num curr_wear_fix;<br>num curr_wear_mov;<br>num release_dist;<br>num deflection_dist_z;<br>num deflection_dist_x;<br>num deflection_force;<br>num deflection_time; |

**Note**

Some of the parameters in gundata only concerns servo guns and Software Equalizing and depends on the selected configuration.

## Data

The names are predefined and used internally when Spot instructions are used. They must therefore not be deleted or renamed.

### Global data

The following global data are predefined:

| Name | Declaration | Description |
|---|---|---|
| curr_gundata{4} | PERS gundata | Current gun specific data for gun equipment 1 to 4. |
| curr_spotdata{4} | PERS spotdata | Current or latest used spot data for gun equipment 1 to 4. Is automatically updated from the instruction before the first process hook is called. This data is used when the manual action `ManualSpot` is activated. |
| curr_forcedata{4} | PERS forcedata | Current or latest used forcedata for gun equipment 1 to 4. Is automatically updated when the `SetForce` instruction are run. Is also used when manual actions are activated (`ManualGunControl` and (`ManualSetForce`. |
| curr_simdata | PERS simdata | Current parameters for simulation. These parameters have influence on all used equipment. |
| tw_ref_dist{4, 2} | PERS num | Distance to the reference surface for the reference fixed tip. See *Tip measurement sequence on page 181*.<br>(This only concerns Spot servo equalizing) |

*Continues on next page*

4.4.1 SWUSER
*Continued*

| Name | Declaration | Description |
|---|---|---|
| tw_ref_tool{4} | PERS tooldata | Reference tooldata for each gun, used in software equalizing. See *Tip measurement sequence on page 181*.<br>(This only concerns Spot servo equalizing) |
| reference_done{4} | PERS bool | Boolean used in routine ReCalcTCP to check if a reference measurement has been done.<br>(This only concerns Spot servo equalizing) |

**Process hooks**

The following predefined routines are installed with the application. They are called from the kernel during the process. These routines has no default functionality but can easily be modified to fit specific equipment's.

Parameters description for the process hooks:

- `num GunNum`: Gun equipment number.
- `INOUT ErrText`: Error message. If an error text is returned in this parameter it will generate an error dialog with possibilities for the operator to decide what to do. If ErrText = "Retry" is returned from some of the hooks then no interaction with the operator will be performed. The process is restarted from the beginning.

PROC SwInitUserIO(num GunNum)

This routine is the first called process hook, called in the beginning of the motion towards the position.

There is no default functionality.

PROC SwPrepare(num GunNum, INOUT string ErrText)

This routine is called in the beginning of the motion part but after `SwInitUserIO`. See *The type Spot Media Equipment on page 36*

No default functionality.

PROC SwCloseGun(num GunNum, INOUT string ErrText)

This routine is called a predefined time, pre closing time before the robot TCP reaches the weld position. See *The type Spot Gun Equipment on page 31*

No default functionality.

**Note**

The presence of this routine depends on the selected spot configuration.

PROC SwPreWeld(num GunNum, INOUT string ErrText)

This routine is called in the weld position and is the last routine to be called before the start signal to the timer is activated. See *The type Spot Weld Equipment on page 27*

No default functionality.

*Continues on next page*

PROC SwOpenGun(num GunNum, INOUT string ErrText)

This routine is called just after receiving the weld complete signal from the timer, before the open gun order is activated. See *The type Spot Gun Equipment on page 31*

No default functionality.

> **Note**
>
> The presence of this routine depends on the selected spot configuration.

PROC SwPostWeld(num GunNum, INOUT string ErrText)

This routine is called when the process is ready, after the `SwOpenGun` is executed.

- If no simulations are active then the weld counter in `curr_gundata` is updated.

PROC SwWeldFault(num GunNum, INOUT string ErrText)

This routine is called when the configured weld timeout time has elapsed without receiving the weld complete signal frtom the timer, or when receiving the fault signal from the weld timer during the weld sequence. The gun has been ordered to open just before this hook is called. See *The type Spot Weld Equipment on page 27*

No default functionality.

PROC SwErrorRecover(num GunNum, string ErrType, string ErrText, \num CurrThickness INOUT num Status)

This routine will be called instead of the built-in error handling from the motion task if the process configuration data `User defined error handling` is set to Yes. When using this routine it is possible to customize the error dialogs on the FlexPendant when an error has occurred. No means that standard built-in error recovery is used. See Spot Error Handling data*The type Spot Error Handling on page 25*.

| Parameter | Description |
|---|---|
| GunNum | Current gun number |

*Continues on next page*

4.4.1 SWUSER
*Continued*

| Parameter | Description |
|---|---|
| ErrType | Type of error that occurred. Possible cases are:<br>• SW_PREPARE_ERR: prepare error. Error reported in the prepare sequence or by the `SwPrepare` routine.<br>• SW_CLOSE_GUN_ERR: close gun error. Error reported in the close gun sequence or by the `SwCloseGun` routine.<br>• SW_PRE_WELD_ERR: preweld supervision error. Error reported in the preweld sequence or by the `SwPreWeld` routine.<br>• SW_WELD_ERR: weld error timeout.<br>• SW_WELD_TEST_ERR: weld test error timeout. Error if test weld error in the `CalibL` or `CalibJ` instructions.<br>• SW_OPEN_GUN_ERR: open gun error. Error reported in the open gun sequence or by the `SwOpenGun` routine.<br>• SW_POST_WELD_ERR: postweld supervision error. Error reported in the post weld sequence or by the `SwPostWeld` routine.<br>• SW_TIP_POS_ERR: tip position error. Only for servo guns.<br>• SW_PROG_VALID_ERR: program valid timeout.<br>• SW_WATER_SUP_ERR: water flow timeout. Water flow error reported by the `SW_SUP` task if the continuous water supervision is activated. See *The type Spot Media Equipment on page 36*<br>• SW_MEAS_TIP_CHANGE_ERR: Tip change supervision error in the `MeasureWearL` instruction.<br>• SW_MEAS_TIP_WEAR_ERR: Tip wear supervision error in the `MeasureWearL` instruction.<br>• SW_RECAL_TIP_WEAR_ERR: Tip wear supervision error in the `ReCalcTCP` instruction.<br>**Note**<br>To retrieve the latest measured data when a tip management error has occurred, the instructions *SwGetFixTipData - Get the latest fixed tip wear and position adjustment on page 157* and *SwGetCalibData - Get the latest total tip wear and position adjustment on page 156* can be used. |
| ErrText | Text string that was returned by the function that reported the error. |

*Continues on next page*

The return values of this function defines how the Spot options shall resume after this error. There are three possible return values:

| Return value | Description |
|---|---|
| SW_RETRY | The weld process is started from the beginning after weld error and after errors reported by:<br>• `SwPrepare`<br>• `SwCloseGun`<br>• `SwPreWeld` |
| SW_SKIP | The current spot weld process is abandoned and cleaned up. |
| SW_IGNORE | The current tip position error is ignored and the weld is executed again without plate thickness supervision. (Only for servo guns.) |

**Note**

The number of available user hooks depends on the selected configuration.

### Supervision task hook

The SupervisionInit routine is called from the main routine in the SW_SUP task at power on.

- `SupervisionInit()`

PROC SupervisionInit()

There is no default functionality.

### Related information

| | Described in: |
|---|---|
| Customizing possibilities | *Customizing RobotWare-Spot on page 247* |

## 4.4.2 SWUSRM

### Description

The SWUSRM user module is configured to run in **all motion tasks** in the system, and contains some default Spot related data. It also contains routines for data transfer (for example DefineSpotData, used to copy user defined spotdata to internally used spotdata.

In normal cases there is no need to change this module. The default functionality should be good enough in most cases. But if the default data types are changed there may be a need to modify this module also.

**Note**

Default functionality depends on the spot configuration.

**Note**

After changing any routines in SWUSRM, the following steps must be taken before there is an affect on the application:

- Save SWUSRM. The old one is overwritten.
- Generate a Reset Rapid restart to affect all tasks

**Note**

If data is moved from this module the Spot MMI application might not work properly!

### Default data

The following default data are predefined.

| Name | Description | Default value |
|---|---|---|
| gun1, 4 | Gun number used in the spot instruction, gun index number in curr_gundata. | 1 to 4 |
| spot1 | Default spotdata when programming the spot instructions on the FlexPendant.<br>**Tip**<br>It is possible to use spot data parameters programmed in the weld timer if needed instead of the spotdata parameters, see *How to use spot data programmed in the weld timer on page 258* | prog_no - 1<br>tip_force - 1000 N<br>plate_thickness 0 mm<br>plate_tolerance - 0 mm |

*Continues on next page*

| Name | Description | Default value |
|---|---|---|
| force1 | Default forcedata when programming the SetForce instruction on the FlexPendant | tip_force - 1000 N<br>force_time 1 s<br>plate_thickness 0 mm<br>plate_tolerance - 0 mm |

### Process data routines

These routines can be used to perform actions inside the Spot routines. The following process routines are installed with the application.

PROC DefineUsrData(num GunNum \INOUT gundata UserGunData)

This routine is called in the beginning of all Spot shell routines. Here user gundata can be transferred into the Spot instruction, using this data instead of the default curr_gundata.

**Note**

The optional \UserGunData parameter will be used if the optional \GunD argument is used in the spot instructions. See *SpotL/SpotJ - The basic spot welding instructions on page 95*

```
IF Present (UserGunData) THEN
  curr_gundata{GunNum} := UserGunData;
ENDIF
```

PROC UpdateUsrData(num GunNum \INOUT gundata UserGunData)

This routine is called at the end of all Spot shell routines. Here gundata can be transferred back to the user gundata.

**Note**

The optional \UserGunData parameter will be used if the optional \GunD argument is used in the spot instructions. See *SpotL/SpotJ - The basic spot welding instructions on page 95*

```
IF Present (UserGunData) THEN
  UserGunData := curr_gundata{GunNum};
ENDIF
```

### Data definition routines

The following predefined routines are used to transfer user defined data to internally used data. They are used by the spot welding instructions and are called from the kernel during the process. Some of them are also called during system events such like poweron, program start, program stop etc.

*Continues on next page*

4.4.2 SWUSRM
*Continued*

These routines have a default functionality but can easily be changed. The routines cannot be deleted since they are called from internal modules.

PROC DefineSpotData(spotdata Spot, num GunNum)

This routine is executed in the beginning of each `Spot` instruction. Transfer user `spotdata` to internal spot data, that is `spotdata` in the `SpotL` instruction.
The weld program group output signal will be set just after leaving this routine.
See *SwSetIntSpotData - Set the internal spotdata on page 152*.

PROC DefineGunData()

This routine is executed in the beginning of all Spot shell routines, `SwStart` or `SwReStart`. Transfer user gun data to internal gun data, that is `curr_gundata`.
See *SwSetIntGunData - Set the internal gundata on page 154*.

PROC DefineForceData(forcedata Force, num GunNum)

This routine is executed in the beginning of each `SetForce` instruction. Transfer user `forcedata` to internal force data, that is `forcedata` in the `SetForce` instruction. See *SwSetIntForceData - Set the internal forcedata on page 153*.

PROC DefineSimData()

This routine is executed in the beginning of all Spot shell routines. Transfer user simdata to internal `simdata`, that is `curr_simdata`. See *SwSetIntSimData - Set the internal simdata on page 155*.

PROC UpdateCalibData(num TotalTipWear, num WearMoveable, num GunNum \switch ToolChg | switch TipChg | switch TipWear | switch FineCalib)

This routine updates the current tipwear parameters in `curr_gundata`. It is executed at the end of each `CalibL/J` and `Calibrate` instruction.

PROC UpdateFixTipData(num CurrWearFixed, num DiffDistance, num GunNum \switch Reference | switch TipChange | switch TipWear | switch RefChange)

This routine updates the current fixed tipwear parameters in `curr_gundata` It is executed at the end of each `MeasureWearL` and `ReCalcTcp` instruction.

**Note**

This routine is only present if the SoftWare Equalizing option is selected.

PROC UpdateSpotData(z_int_spotdata Spot, num GunNum)

This routine updates the `curr_spotdata` with the latest used `spotdata`. It is executed at the start of motion, that is when the robot starts to move.

**Note**

Default functionality depends on the spot configuration.

*Continues on next page*

**Event routines**

The following predefined event routines are installed with the application. These routines have no default functionality but can easily be changed. If not needed they can be removed.

PROC SwPowerOn()

This routine is called when the robot is restarted (warm started) or by power on.

There is no default functionality.

PROC SwStart()

This routine is called when execution is started from the beginning of the program.

There is no default functionality.

PROC SwReStart()

This routine is called when execution is started from the position where it was stopped.

There is no default functionality.

PROC SwStop()

This routine is called when the program is stopped.

There is no default functionality.

PROC SwQStop()

This routine is called when the robot is quick stopped (E-stop).

There is no default functionality.

## 4.4.3

# 5 Software Equalizing

## 5.1 Introduction to Software Equalizing

### Introduction

This chapter describes the Software Equalizing functions. These functions makes it possible to use spot welding guns without mechanical equalizing systems. The functions are available if the option *Spot Servo Equalizing* is installed.

### Available functions

The Software Equalizing functions are a number of functions intended to handle these issues for the user. However, it is not always necessary to use all functions. It depends on desired accuracy, sheet stiffness, gun properties as type, size and stiffness and so on.

The following Equalizing functions are available:

- Weld position Touch Up.
- Release of the fixed gun arm.
- Gun arm Deflection Compensation.
- Tip Wear Measurement and Compensation.
- SoftMove Equalizing

It is possible to use guns with mechanical equalizing and guns using Software Equalizing in the same user program. The equalizing type is determined in the process data `Use SoftWare equalizing` located in `Spot Gun Equipment`. For more information, see *gundata - Equipment specific weld data on page 135*.

### Recommendations

When guns without mechanical equalizing systems are used it is very important to have good accuracy when the TCP is defined and when the weld positions are taught.

It is also important to handle the tip wear and recalculate the TCP regularly and also to release the fixed gun arm from the sheet when the gun is moved between weld positions. For most guns it is also necessary to handle the gun arm deflection during the weld.

**Note**

When using Software equalizing it is important to have good control of the part tolerances and the tip wear of the electrodes. This functionality can only tolerate variations up to approximately 1-2 mm.

**Note**

When using SoftMove equalizing functionality the tolerances can be a bit bigger than with Software equalizing only, up to 4-5 mm.

*Continues on next page*

**Limitations**

- The functions Gun Arm Release and Deflection Compensation are only availible when `SpotL`/`SpotJ` instructions are used, not for the `SpotML`/`SpotMJ` instructions.
- The gun pre closing time is **not** used when software equalizing is active. In this case the pre closing is handled automatically during the movement from the release distance to the weld position.
- For some special configurations an acceptable touch up position can be hard to reach with the `MeasureWearL` instruction. If that is the case, `ReCalcTcp` instruction can be used as an alternative method.
- It is not possible to run `SpotL`/`SpotJ` instructions with software equalizing active if independent mode is activated.
- It is only possible to run spot instructions in semi coordinated mode.
- When using only gun arm deflection compensation it is important to have good control of the tolerances of the parts, and the wear of the electrodes. This functionality can only tolerate variations up to approximately 1-2 mm.

  (If the tolerances of the parts are inferior, then the Soft Move Equalizing method can be used at the locations where it is needed.)

**Note**

Software Equalizing does not work for the `SpotML` and `SpotMJ` instructions.

## 5.2 Some basic definitions

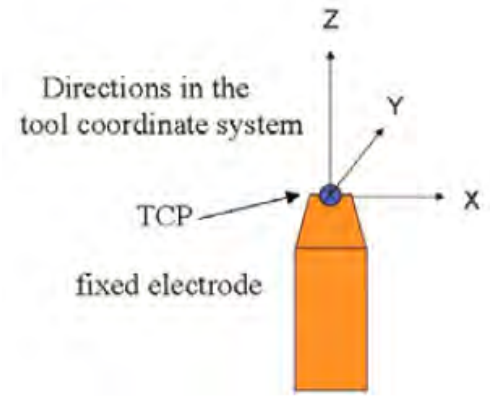
### How to define the TCP

The TCP has to be defined, as normally, on the tip of the fixed gun arm.

It is important to define also the z-direction in the tool coordinate system since all automatically search movements and compensations will be done in the z-direction. See the graphics below.

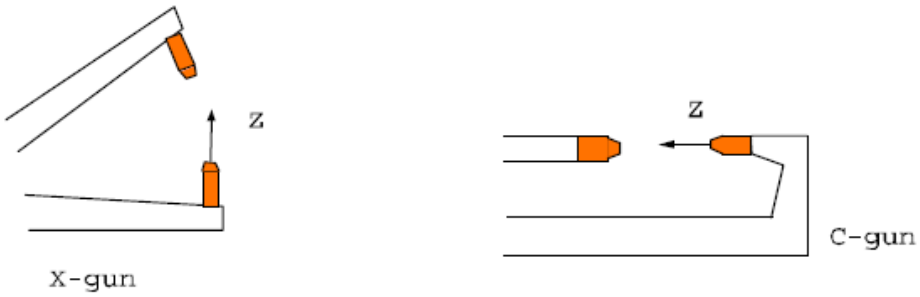
Normally the z-direction should point out from the fixed tip, and the tip of the gun should move towards the work piece when jogging the robot in the positive z-direction.

To achieve the same behavior on a stationary tool the z-direction needs to be reversed into the fixed tip, and in order to get the correct search and compensation movements, the setup data `Opposite z-direction` has to be set to `Yes` in `Spot Gun Equipment` process data, see *The type Spot Gun Equipment on page 31*.




en1200000236

The z-direction when different gun types are used:




en1200000235

**Note**

It is very important to define the z-direction correctly in the tool coordinate system since all automatically search movements and compensations will be done in the z-direction.

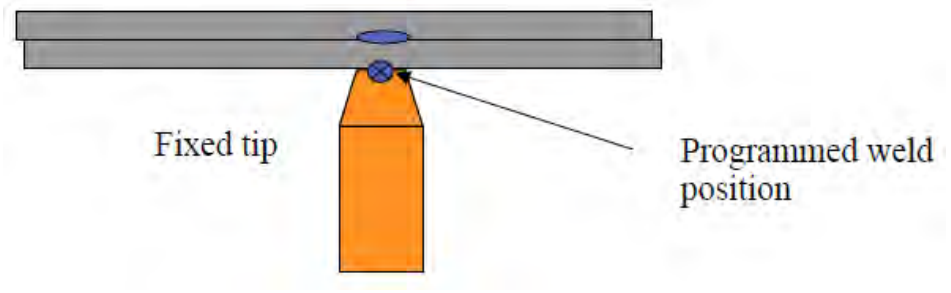
*Continues on next page*

### How to setup the tool TCP

1. Define the TCP for the used gun with a new tip mounted. Use the 5 point method.

   See *Operating manual - IRC5 with FlexPendant*.
2. Store the result in a `tooldata` for example `ref_tool1`.
3. Save the tip (tool) as a reference tip (the physical tip).

### How to program the weld positions




en1200000234

The weld position should be teached in the position where the fixed electrode is touching the sheet during the weld process. See the graphic above.

**Note**

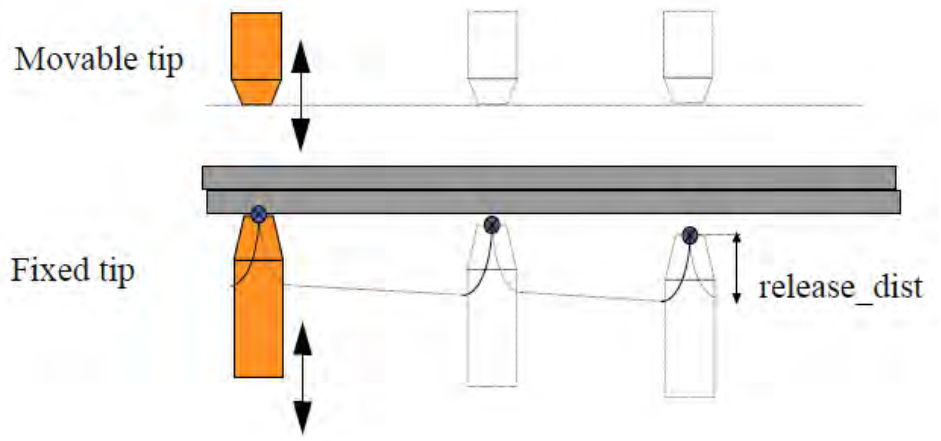
Before touching up the weld positions a `MeasureWearL` with the `Reference` switch has to be done. See *Tip measurement sequence on page 181*.

## 5.3 Weld position Touch Up

### Introduction

This is a support function used in manual mode to get a faster and easier way to adjust the programmed weld positions. During the touch up it is possible to change the fixed gun tip in the z-direction and it is also possible to change the position for the movable gun arm.

Normally a touch up has to be done at least once in the beginning after manual or offline programming of the weld positions.




en1200000232

### How to use the weld position touch up function

1. Make sure that current TCP is relevant for the used tip, define the tool data correctly.
2. Set the system in weld position touch up mode (`simtype = 3` in *curr_simdata*).

   See *simdata - Simulation data on page 146*.
3. Start the program. The robot is running the program as normal, but all `SpotL` and `SpotJ` instructions are executed as move instructions.
4. The robot stops in each programmed weld position and a user interaction is started which gives possibilities to confirm and directly go to next spot or adjust current position.
5. During adjustment the fixed gun tip is moved in small steps to the sheet or to desired distance from the sheet.

   It is also possible to adjust the position of the movable gun arm in a similar way.
6. When both tips are in desired position it is possible to do a **Modpos**, with confirmation, to definitely reprogram the position.
7. When the gun is moved between programmed weld positions the fixed gun tip is automatically released from the sheet.

   Desired distance to the sheet is a user defined data predefined for each used gun, `release_dist`.

*Continues on next page*

This release movement can be skipped to save cycle time if the optional switch `\QuickRelease` is selected in the `SpotL` or `SpotJ` instruction.

See *gundata - Equipment specific weld data on page 135* and *SpotL/SpotJ - The basic spot welding instructions on page 95*.

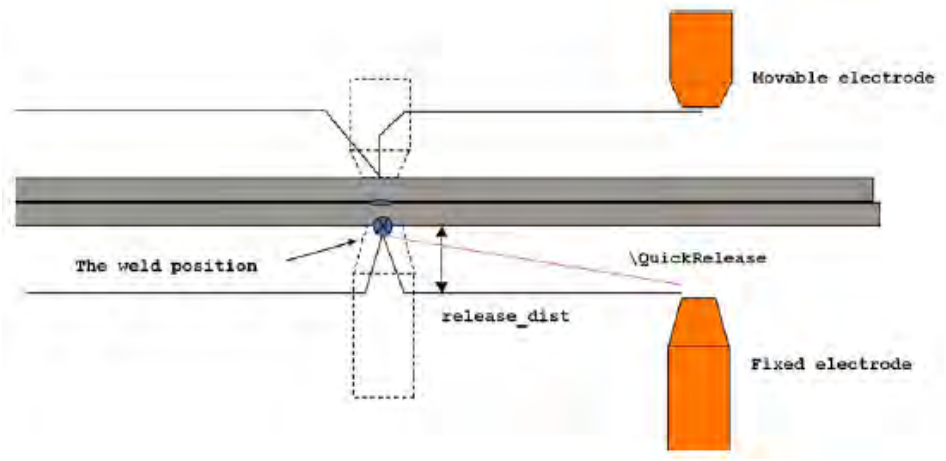
8 When all weld positions are checked, set `simtype = 0` to disconnect the Touch Up mode.

See *gundata - Equipment specific weld data on page 135*.

## 5.4 Releasing of the fixed gun arm and gun opening

### Description

This function is used to get an automatic release of the fixed gun arm from the sheet, when the gun is moved between the weld positions during normal program execution.




en1200000231

### Releasing the fixed gun arm

During execution of `SpotL`/`SpotJ` instructions, the robot moves the gun to the weld position, via a position a release distance from the sheet.

After the weld when the gun is opened, an extra movement is performed to release the fixed gun arm from the sheet except if the `QuickRelease` functionality is activated in the `SpotL`/`SpotJ` instruction.

If the `\QuickRelease` switch is selected in the `SpotL`/`SpotJ` instruction the release distance movement after the weld will be skipped, this may save some cycle time and can be used when the spots are located close together

**Note**

The `\QuickRelease` function is suitable when programming close weld positions, not when there are large distances between the weld positions.

The release distance, `release_dist`, is a user defined data predefined for each used gun. see *gundata - Equipment specific weld data on page 135*.

*Continues on next page*

This function will be disabled if `release_dist` is set to zero or if `softw_equ` is set to `FALSE` in current `gundata`.

**Note**

To get a good synchronization between the release movement and the gun opening when software equalizing is used, the gun is always held in the closed state 40 ms extra after weld complete. To save cycle time, the programmed cool time after weld in the weld controller can be reduced this amount of time (2 periods).

### Gun opening

The gun opening gap must be large enough that the tips are free from the plates when welding.

So therefore, the software will compensate for the release distance that is used, and the plate thickness, as the opening position is the same as the tips closed with plates + release_dist.

Example:

If release_dist is 10mm, the moving tip will open to 10mm even if you modify the position with the gun closed on the plate surface.

A simple recommendation is to have approximately the same distance from the plate to the movable electrode as the release_dist that has been configured.

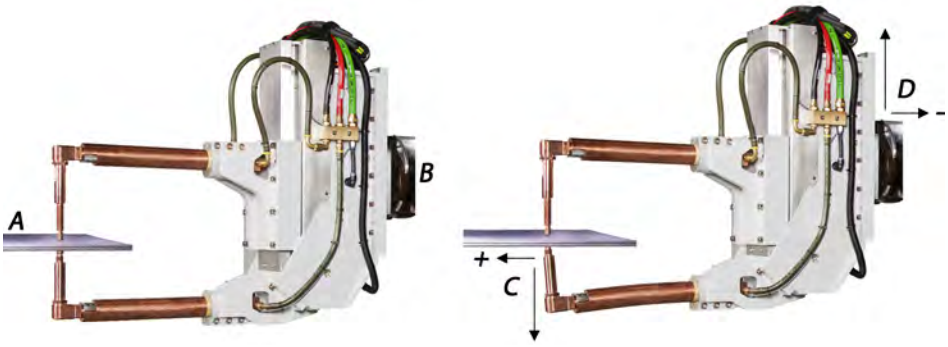
For more information about the release_dist parameter, see *Additional components for Spot Servo Equalizing on page 136*.

## 5.5 Gun arm deflection compensation

### Introduction

Most weld guns deflects more or less depending on the stiffness of the gun arms when the gun is closed with force. In these cases there is a need to compensate for the fixed gun arm bending with an extra **robot movement** to minimize the risk of deformation on the sheets. The deflection compensation can be done in the tool z and x-direction.

With this function the gun arm deflection is automatically compensated with an extra **robot movement** when the gun is closed and the gun force is applied. See graphic below.

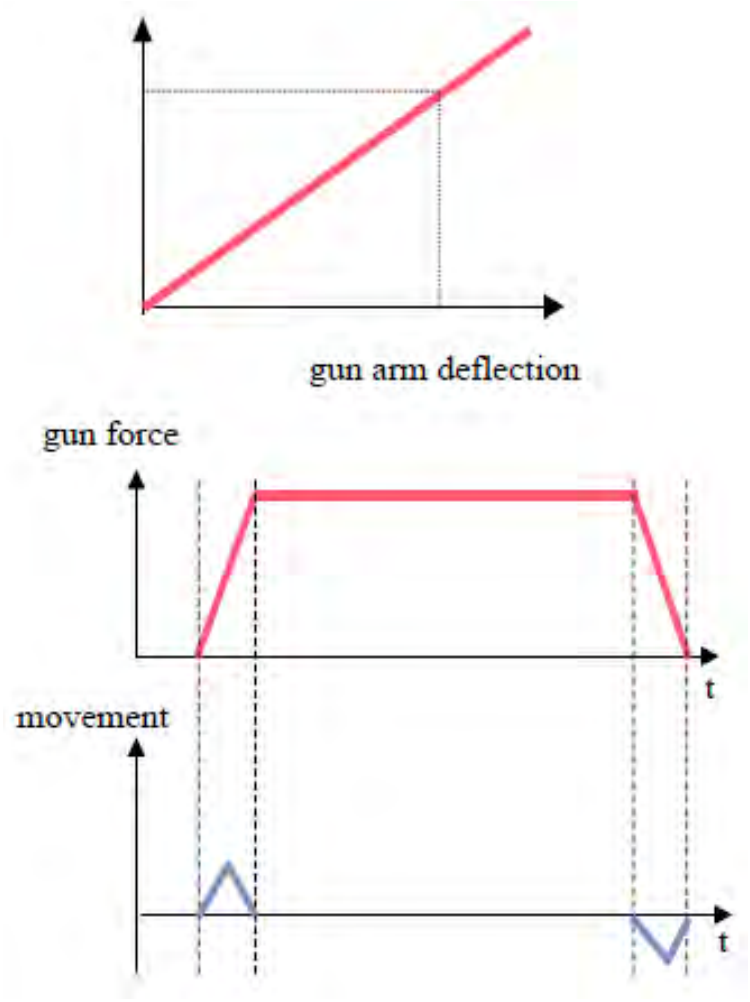



xx1200000230

| | |
|---|---|
| A | Sheets |
| B | Robot flange |
| C | Gun arm deflection |
| D | Robot compensation movement |

The graphic shows gun arm deflection in both z and x-directions, and robot compensation movement in the opposite directions. If the gun bends outwards (positive x), the robot will move in the opposite direction.

*Continues on next page*




en1200000229

## Data

Data for the correlation between the gun force and the arm deflection is user defined data, predefined for each used gun: `deflection_dist_z`, `deflection_dist_x`, `deflection_force` and `deflection_time`, see *gundata - Equipment specific weld data on page 135*. This data is normally found in the data sheet for the used gun.

Then, during program execution of `SpotL/J` instructions, there is an added robot movement, activated at the same time as the gun force is established, to compensate for the gun arm deflection, see graphics above.

The actual gun arm deflection is calculated from the force value (`tip_force`) in current `spotdata`. Deflection calculation in `SpotL/SpotJ` instruction.

```
deflection := spotdata.tip_force * gundata.deflection_dist /
     gundata.deflection_force;
```

A movement in the opposite direction is performed after the weld, when the gun is opened. This movement is combined with the release movement.

*Continues on next page*

This function is disabled if the deflection distance parameters are set to zero, or if the `Spot Gun Equipment`, `Use SoftWare equalizing` setup data is not activated.

**How to setup the data for gun arm deflection**

1. Find out how much gun arm deflection there is at a specific force when the gun arms are closed, this data is normally found in the data sheet for the used gun.

   **Tip**

   If this information is missing the gun arm deflection can be measured manually by closing the gun at a specific force, for example 4000 N and measure how much the gun arm deflects related to a fixed reference position on the tip dresser stand for instance.

2. Enter the measured values in `gundata`, `deflection_dist_z` and `deflection_force`, for example 5 mm at 4000 N.

   See *gundata - Equipment specific weld data on page 135*.

3. Save the user module swuser.sys since the current `gundata` is located there.

## 5.6 Tip wear compensation

### Introduction

When guns without mechanical equalizing systems are used, it is important to handle the tip wear, especially the tip wear on the fixed tip, since this tip is controlling the weld position. Therefore the tip wear has to be regularly compensated during production. There are two methods available for the compensation.

Method 1

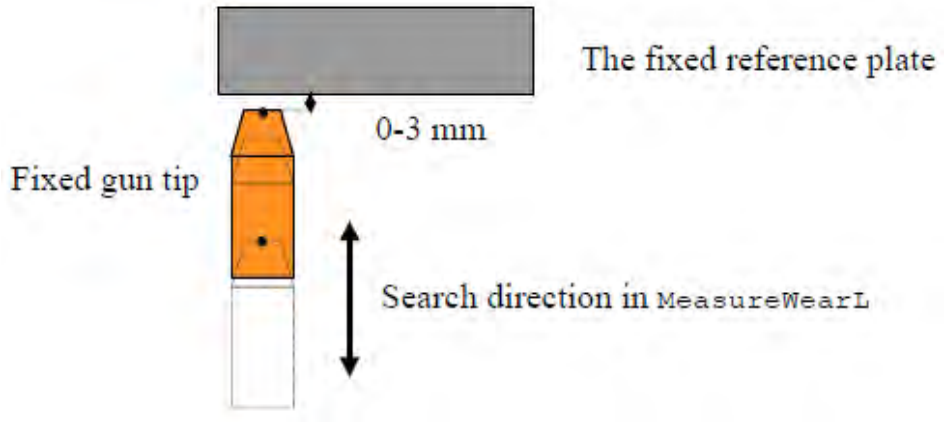
`MeasureWearL`, the tip wear for the fixed tip is measured and the tip wear is then compensated in current used `tooldata`.

Method 2

`ReCalcTcp`, the tip wear of the fixed tip is calculated based on stored information about the total tip wear and the expected relation between the tip wear of the fixed tip and the total tip wear. The tip wear is then compensated in current used `tooldata`. This method can be used as an alternative if method 1 is not suitable.

### Method 1: Tip wear measurement and compensation with `MeasureWearL`

A RAPID instruction is available for tip wear measuring and TCP adjustment: `MeasureWearL`. This instruction is used one time for a reference measurement with new tips and then one time after each tip dressing and after the tips have been exchanged.




en1200000228

A code sequence with the measuring instruction included has to be prepared in the user program. The position in the `MeasureWearL` instruction has to be programmed close to a fix reference plate, see figure above. Also see *Measurement preparation on page 118*.

When the instruction is executed the robot first moves the gun to a start position for the search movement about 10 mm from the reference plate. Then the gun is moved until the fixed gun tip touches the reference plate. The tip wear of the fixed

*Continues on next page*

gun tip is calculated. Currently used TCP value is automatically recalculated and tip wear data for the fixed tip in `gundata` is automatically updated.

> **Note**
>
> This movement of 10mm has to take care of the wear of the tips that could be for example 20mm, it means that the total gun opening should take care of the 10mm position, the plate thickness and the total wear of the tips.

When using this method the gun calibration instructions (`Calibrate` or `CalibL/J`) has to be executed **after** `MeasureWearL` since the moveable tip wear will be calculated based on the result of the measurement.

The reference plate can be mounted in an optional position in the work range, preferably on the tip dresser stand. But it is necessary to orient the tool in the measuring position in that way that an additional torque is generated on at least one of the robot motors when the robot is touching the fixed plate, preferably **axis 4 to 6**.

It is possible to verify if the selected measuring position or gun orientation is good enough by using a service routine; **ManualCheckMeasPos**. Just run the service routine when the robot is in the selected position and you will get status information on the display.

### How to find a good measuring position

1. Jog the robot to the position where the fixed plate is mounted, for example on the tip dresser stand.
2. Run the service routine **ManCheckMeasPos** to find out if the position is good or not, if not jog/reorient the robot to a new position.

   See *Manual actions on page 74*.
3. Create a program with a code sequence with the measuring instruction included.

   See *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118*.

### Tip measurement sequence

The reference measurement with `MeasureWearL \Reference` will calibrate the position of the reference plate for the tip wear measurement with the robot. The parameters of the tool used for the reference measurement (RefTool) and the calibration values are stored in the persistent variables `tw_ref_tool` and `tw_ref_dist` located in the user module swuser.sys. All following calls of `MeasureWearL` (`\TipWear`, `\TipChg`) measures only the difference to the reference position.

The tips (or the real TCP) and the tool (RefTool) used for the reference measurement must be the same as used for teaching of the weld positions (robtargets).

The reference measurement needs to be done again when the reference plate has been moved or the TCP has been changed (for example after a crash) or the

*Continues on next page*

`\RefChange` switch can be used instead, see *Measurement after reference plate/sensor changed on page 120*.

For the measurements the robot contacts the reference surface always with the same force, `MeasureWearL TouchUp force` in the `Spot Equalizing` process configuration. See *The type Spot SoftWare Equalizing on page 38*.

Since the calibration values are stored in swuser.sys the file should be saved after the reference measurement.

Input values of `CurrTipWear` and `RetPosAdj` in `UpdateCalibData` and `UpdateFixTipData`.

| | CurrTipWear | RetPosAdj |
|---|---|---|
| Calib* ... \TipWear | 0 | Difference to the last calibration of the gun.<br>Normally last call with `\TipWear`. Total difference between the new and the old (worn) tips. |
| Calib* ... \TipChg | Total wear of both tips | Difference to the last calibration of the gun. |
| Calib* ... \ToolChg | Total wear of both tips | Difference to the last calibration of the gun. |
| MeasureWearL .. \TipChg | 0 | Difference between the actual and the tip used for the reference measurement. |
| MeasureWearL .. \Reference | 0 | Measured reference distance. |

See *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118*.

**Note**

For some special configurations the `MeasureWearL` is less suitable, for example very large guns and/or when an acceptable touch up position is not possible to reach for some reason. Then the `ReCalcTcp` method should be used instead.

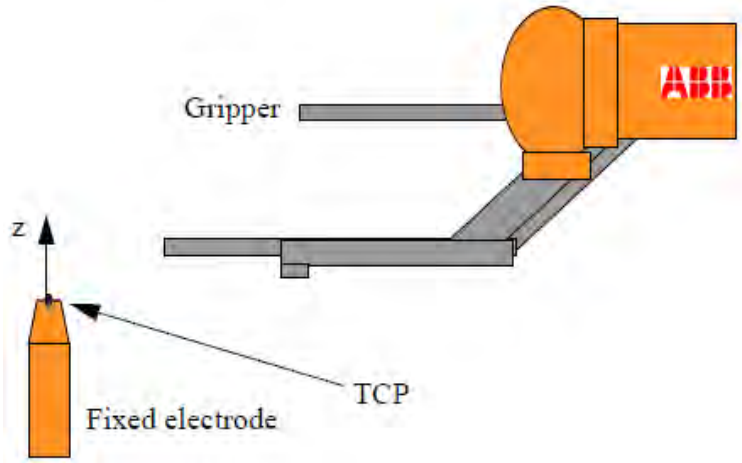
### Tip wear measurement and compensation for stationary guns

Generally when stationary tools are used, the work object is held by the robot and related to the wrist coordinate system and the TCP, still defined on the tool, is related to the world coordinate system. This is the case also when stationary guns are used. As before, the TCP has to be defined on the fixed tip and the z-direction has to be defined, see figure below and *How to define the TCP on page 171*

The parameter `robhold` in current `tooldata` and `wobjdata` defines whether the robot is holding the gun or not.

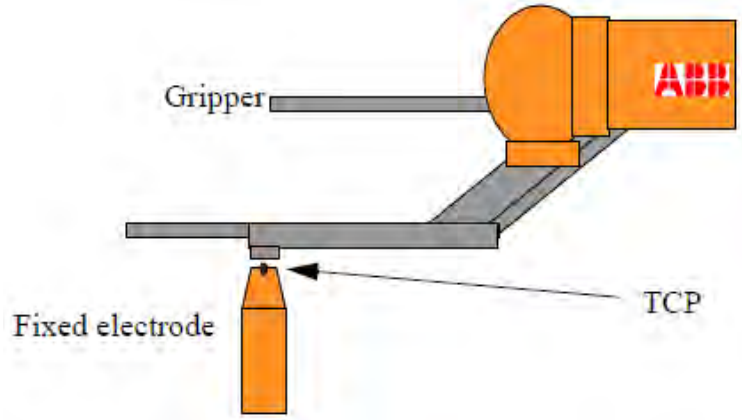
*Continues on next page*

For more information about how to define the stationary tool coordinate system, see *Operating manual - IRC5 with FlexPendant*.




en1200000227

All Software Equalizing functions are working in a similar way as when the robot is holding the gun. But the tip wear measurement has to be arranged a little different, since it is not possible to use a fix reference plate in this case. To be able to use the same principles for the measurement and the `MeasureWearL` instruction, it has to be done as described in following items:




en1200000226

- Select or create a relatively stable position on the robot held gripper. It shall be possible for the robot to move this point to the fixed gun tip. see an example in the figure above.
- As in the normal case, code sequences with the measuring instructions included have to be prepared in the user program to be used for the reference measurement and for the tip wear measurement.
- We recommend programming the gun calibration instructions in the same code sequences, directly after the tip wear measurement. See *Calibrate - Calibrate a servo gun on page 114* and *CalibL/CalibJ - Calibrate a servo gun during robot movement on page 108*.

*Continues on next page*

- The position in the `MeasureWearL` instruction has to be programmed close to the selected position on the gripper, see graphic above.
- When the instruction is executed the fixed gun tip is touched by the gripper. The tip wear of the fixed gun tip is calculated. Currently used TCP value is automatically recalculated and tip wear data in `gundata` is automatically updated. For more information, see *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118*.

**Method 2: Tip wear calculation and compensation with ReCalcTcp**

A RAPID instruction to be used for tip wear calculation and TCP adjustment with this method is available: `ReCalcTCP`.

As the measurement instruction used in method 1, `MeasureWearL` this instruction is used one time for a preparation and then one time after each tip dressing and after the tips have been exchanged. But in this case the instruction is a logical instruction without movements.

When the instruction is executed the tip wear of the fixed tip is calculated. The calculations are based on stored information about the total tip wear and the expected relation between the tip wear of the fixed tip and the total tip wear. The tip wear is then compensated in current `tooldata`. Current used TCP value is then automatically recalculated and tip wear data for the fixed tip in `gundata` is automatically updated.

In this case the gun calibration instructions (`Calibrate` or `CalibL/J`) has to be executed **before** `ReCalcTCP` is used since the total tip wear is used for the calculations.

For more information, see *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 126*.

The advantages with the calculation method is that it is faster since no extra measurement has to be done, and it is also easier to set up since no reference position has to be arranged for. The disadvantage is that the compensation will not be as accurate as with the measuring method in the cases when the tip wear ratio value not is set to a value in agreement with the reality, or after tip change if the new tips not have the same size every time.

## 5.7 SoftMove Equalizing

### Introduction

This section describes the SoftMove Equalizing method which can be used as a complement to the standard software equalizing method if there is a need to compensate for programming errors, for example if the plates are not located in the nominal position.

When a spot instruction with SoftMove activated is run, the robot will move to the programmed and nominal position, and set the robot into soft state using SoftMove in the tool z-direction during the last part of the movement into the position.

This method can be used together with the standard software equalizing functionality, that is, gun arm deflection compensation.

### Prerequisites

SoftMove Equalizing is available for the `SpotL` and `SpotJ` instructions, and it is only available if the RobotWare options *Spot Servo Equalizing* (635-6) and *SoftMove* (885-1) are installed together.

### Recommendations

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

Some general tips when using SoftMove Equalizing:

- Verify that the load definition of the tool is correct.
- Avoid singular robot orientations.
- The robot axis configuration will affect the softmove performance, there are configurations where performance can be poor.
- A gun configuration that allows axis 5 movement gives better performance in general (Avoid gun configurations when z-axis of the gun is in linearity with axis 6 of the robot).
- The clamping distances cannot be too far apart, as there must be some resistance for the robot to react on.

**Note**

When using SoftMove equalizing functionality the part tolerances can be higher than with Software equalizing only, up to 4-5 mm.

For more information about SoftMove, see *Application manual - SoftMove*.

### Limitations

The following limitations apply when using SoftMove Equalizing:

- The same limitations as for Software Equalizing also applies for SoftMove equalizing.

*Continues on next page*

- Cannot be used to compensate for too large variations in the objects to be welded, only up to a few mm.
- **Not all** positions or arm configurations are suitable for SoftMove Equalizing, if poor performance (for example not reaching the position or pressing too hard) is experienced, then try to reorient the robot axes or gun arm configuration if possible. Friction force may vary too much so that the force offset needed will be too high, this can deform the plate.
- SoftMove performance will in general depend on robot type, robot arm configuration, gun configuration, and load data definition. Larger robot types will have higher internal gear friction and inertia and that will affect the sensitivity, and it will be harder to find a friction value good enough.
- This function should in general not affect the plates more than 50N to 100N during the welding sequence, however there are positions in the work area where it will be difficult to get good results, and the force applied on plates can be higher.
- When tuning the friction compensation needed in a position, the robot can move in a unexpected direction because of high friction in axis 2 and/or 3, this may indicate a difficult or unsuitable position.
- Cycle times will be longer when using SoftMove Equalizing because of the the lower TCP speed and gun speed into position, compared to using standard SoftWare Equalizing.
- When tuning the friction compensation in a certain position the value is usually good enough, but the acheived value may need to be manually tuned to get better performance. If the measured value is higher than ~300 N there may also be a need to manually modify the measured value.
- When SafeMove is used together with SoftMove there is a risk for servo lag problems. The recommended action is to add a Contact Application Tolerance (CAP) in the area where SoftMove is active.

When the function is active, that is, during the welding phase in the `SpotL/J` instructions, the following functionality is not accessible:

- *Collision Detection* (option 613-1)

**Note**

For more information about SoftMove limitations, see *Application manual - SoftMove*.

**Note**

For safety reasons the position supervision limits in x, y, and z directions has been limited. These values can be changed in the motion parameter configuration if needed, for example type *Motion type CSS* and *Max pos error in z*. Default configuration is x = 10 mm, y = 10 mm, and z = 10 mm.

*Continues on next page*

**Note**

SoftMove Equalizing is only implemented for the `SpotL` and `SpotJ` instructions and does not work for the `SpotML` and `SpotMJ` instructions.

**Note**

This function allows the tolerances of the parts to vary a bit more compared to using only standard software equalizing, but it should be noted that the cycle time will be longer than without, 200-700 ms per spot depending on the current settings and the actual plate position.

**WARNING**

When using SoftMove Equalizing it is **very important** that the load definition of the tool is defined correctly.

## Function overview

SoftMove equalizing is activated by selecting an optional switch in the `SpotL` and `SpotJ` instructions, see *SpotL/SpotJ - The basic spot welding instructions on page 95*.

When a spot instruction with SoftMove activated is run, the robot will move to the programmed and nominal position, but during the last part of the movement into the position the robot will be set into soft state in the tool z-direction. When the gun has closed, the soft mode will be deactivated if SoftMove type is set to 1, and the deflection compensation will be activated as normal during the welding process if configured. If SoftMove type is set to 0, SoftMove will be activated during the compete weld cycle and no gun deflection compensation will be performed, see *smeqdata - SoftMove Equalizing data on page 148*.

The SoftMove equalizing method can be more forgiving regarding programming errors than only using the standard software equalizing method. For example, if the plates are not located exactly in the nominal position but some millimeters away, up to 4-5 mm.

**Note**

If the normal software equalizing is deactivated then the SoftMove equalizing will also be deactivated, and it is considered that mechanical equalizing system is being used.

## Programming

An optional parameter must be activated in the spot instructions to be able to use the SoftMove Equalizing functionality (`\SMEQ`). The parameter is an on/off switch for the SoftMove functionality, that is, if the parameter is not used, then the normal software equalizing functionality will be used for that position, see programming example below.

*Continues on next page*

The optional parameter `\SMEQ` of type `smeqdata` has data components for SoftMove, specific for each position. `smeqdata` has the following structure:

- Desired `smeq_type` (SoftMove type) used in the specific position.
- Measured `force_offset` (friction compensation and force applied on the plates) for the specific position.

The parameter `smeq_type` defines what SoftMove type that shall be used in the specific position. There are 2 types; no gun deflection mode (0) and standard mode (1).

The parameter `force_offset` is needed to compensate for the robot's static friction in a specific position and to achieve a small movement to the plates. The value must be set for each position for the first time the program is executed, see *Friction compensation procedure on page 191*.

**Note**

The `force_offset` should be seen as friction force and force applied to the plates.

Example: 100N is identified in the friction compensation procedure.

To be sure that robot will move to the plate the `force_offset` can be set to 130N. If the **friction is constant** during the movement it will be 30N applied on the plate.

**Note**

It may be possible to use the same `force_offset` value for several positions.

For more information about this data type, see *smeqdata - SoftMove Equalizing data on page 148*.

## Example

This program example uses `SpotL` instructions with and without SoftMove Equalizing functionality. Instructions at targets P20 and P40 will be executed with only software equalizing activated while instruction at P30 is executed with SoftMove Equalizing because of tolerance issues in that position.

```
PERS smeqdata smeq1:=[1,150]
PROC main()
  MoveJ P10, v1000, z50, tool1;
  SpotL P20, vmax, gun1, spot11, tool1;
  SpotL P30, vmax, gun1, spot12\SMEQ:=smeq1, tool1;
  SpotL P40, vmax, gun1, spot14, tool1;
ENDPROC
```

*Continues on next page*

### Execution

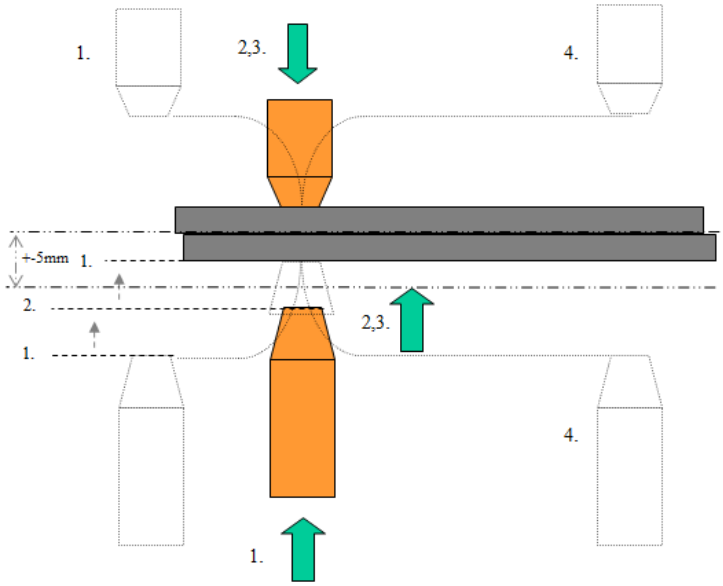
No gun deflection mode (SoftMove type 0)

When a spot instruction is executed with the `\SMEQ` data set, the robot will move to the programmed and nominal position, at the same time as the gun starts to close. During the movement and closing of the gun the robot will be set into soft state with the specified force offset activated, according to the used data to overcome the friction of the robot.

When the gun has closed the position is considered to be "found" and the weld will be performed from that new position. After the weld process is completed the gun will start to open and SoftMove will be deactivated, see figure below.

**Note**

This method is a bit quicker compared to the standard mode because no synchronization with the gun closing is done before the process is started, but it is only suitable for guns with no or very small arm deflection values.




en1200000225

1. Movement to the nominal position, via release release distance.
2. From the SoftMove offset distance SoftMove is activated with the specified `force_offset` (friction compensation+force applied on the plate) while the gun closes.
3. When the gun is closed the plate is considered to be "found" and the weld will be done from that new position.
4. After weld the gun and robot will move to the release distance at the same time as the gun opens and SoftMove is deactivated.

*Continues on next page*

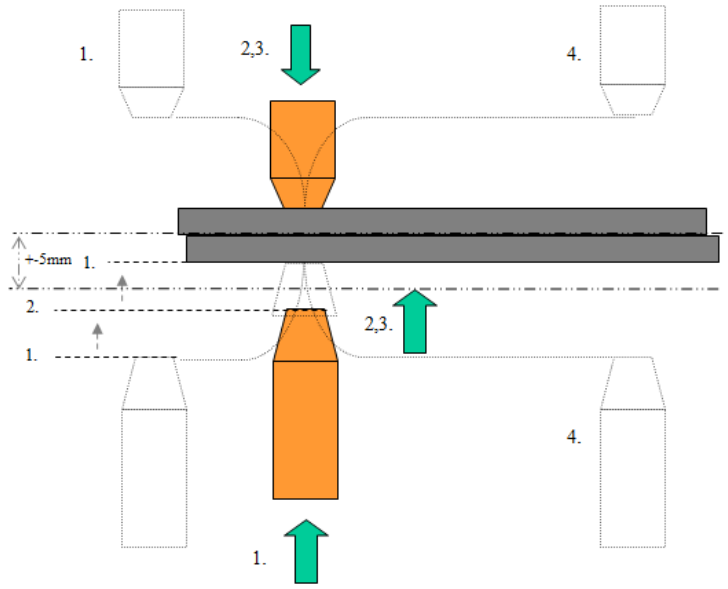
### Standard mode (SoftMove type 1)

When a spot instruction is executed with the `\SMEQ` data set, the robot will move to the programmed and nominal position, at the same time as the gun starts to close. During the movement and closing of the gun the robot will be set into soft state with the specified force offset activated, according to the used data to overcome the friction of the robot.

When the gun has closed the position is considered to be "found" and the soft state will be deactivated and the normal gun arm deflection compensation will be performed if configured. After the weld process is completed the gun will start to open and the gun arm deflection will be deactivated, see figure below.

**Note**

This method is a bit slower compared to the no gun deflection mode because the process waits for the gun closing before the process is started and performs arm deflection compensation during the weld. This method is suitable for guns with relatively big arm deflection values.




en1200000225

1 Movement to the nominal position, via release release distance.
2 From the SoftMove offset distance SoftMove is activated with the specified `force_offset` (friction compensation+force applied on the plate) while the gun closes.
3 When the gun is closed the plate is considered to be "found" and SoftMove will be deactivated and the weld will be done from that new position.
4 After weld the gun and robot will move to the release distance at the same time as the gun opens.

*Continues on next page*

### Friction compensation procedure

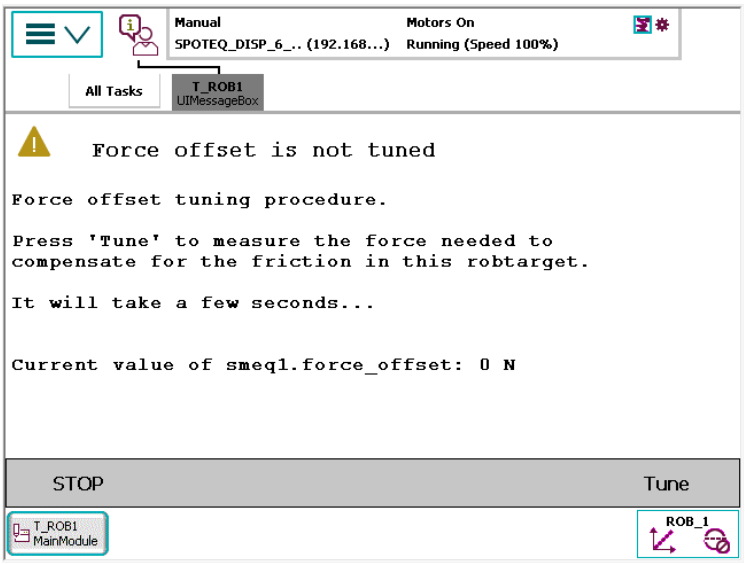
When a spot instruction is executed with SoftMove Equalizing activated for the first time there is a need to compensate for the robot's static friction in a specific position and to achieve a small movement towards the plate. For this there is a parameter that needs to be tuned to a certain value for each position that uses SoftMove in the program, `force_offset`

By default the tuning will be done automatically for each position, it is possible to configure manual tuning iff needed, see *The type Spot SoftWare Equalizing on page 38*

When a spot instruction is run the `force_offset` parameter in the current `smeqdata` will be checked, if the value is equal to zero the robot will move to the release distance outside the `robtarget` specified in the instruction and stop, or if configured, perform an auto tuning in that position. The operator can then tune the `force_offset` value needed for the current position, see figures below.

If the `force_offset` parameter is greater than zero the spot instruction will be executed normally with the current value to compensate for the friction. The `force_offset` should be seen as friction force and force applied to the plates.

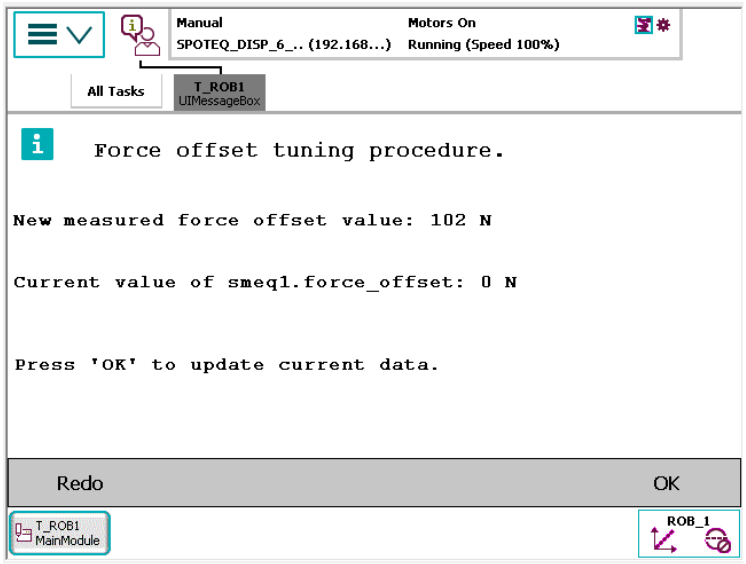
A typical force offset value for an IRB 6640 can be approximately 100-400 N depending on arm configuration.




en1200000223

| | |
|---|---|
| Stop | Stops the program execution. |
| Tune | The force offset will be tuned for the current position. The tuning will take a few seconds and after that a dialog appears with the possibility to accept the measured value or redo the tuning. |

*Continues on next page*




en1200000222

| | |
|---|---|
| Redo | Returns to the previous screen with the possibility to tune the force offset needed again. |
| OK | Updates the current `smeqdata` with the measured value. |

**Note**

The `force_offset` should be seen as friction force and force applied to the plates.

Example: 100N is identified in the friction compensation procedure.

To be sure that robot will move to the plate the `force_offset` can be set to 130N. If the **friction is constant** during the movement it will be 30N applied on the plate.

**Note**

The measured value of `force_offset` is an **approximate** value and it may be necessary to have a higher or lower value than measured for the current position since the friction value is measured while the robot is not moving.

The actual force needed to overcome the friction or overcome frictional force and accomplish a movement of the robot may vary slightly depending on cold/warm robot etc. And it may turn out that the value may seem a little too low when running the robot in manual mode, but when running in automatic mode, the value is right. If the robot does not reach the position in manual mode, increase the value manually by 5-10N and run again.

For more information about SoftMove limitations, see *Application manual - SoftMove*

*Continues on next page*

**Note**

When tuning the friction compensation in a certain position the value is usually good enough, but the acheived value may need to be manually tuned to get better performance. If the measured value is higher than ~300 N there may also be a need to manually modify the measured value.

**Note**

It is recommended that the friction compensation procedure is done without welding activated.

## Related information

| | Described in |
|---|---|
| Manual actions | *Manual actions on page 74* |
| Process configuration | *Spot process configuration on page 21* |
| MeasureWearL | *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118* |
| ReCalcTcp | *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 126* |
| Customizing | *Customizing RobotWare-Spot on page 247* |
| SpotL/SpotJ | *SpotL/SpotJ - The basic spot welding instructions on page 95* |
| SoftMove Equalizing data | *smeqdata - SoftMove Equalizing data on page 148* |
| SoftMove | *Application manual - SoftMove* |

This page is intentionally left blank

# 6 Servo gun motion control

## 6.1 Servo gun introduction

### Additional axes

The robot controller has functionality to control additional axes configured as servo guns (other types of supported additional axes are track motion, positioners, conveyors etc.). All servo guns are handled as separate mechanical units. This means that before a servo gun may be moved, the mechanical unit to which it belongs must be activated. Several servo guns may be active at the same time.

### Hardware overview

Servo gun axes are controlled by the drive module. Internal drive units are mounted inside a standard drive module (for example for an IRB 6700 with one servo gun or for an IRB 6700 with two stationary servo guns).

### Motion servo gun parameters

A set of motion servo gun parameter file should be installed in the controller for each servo gun. The parameter files are optimized designed concerning system behavior and motion/process performance.

It is possible to read and change most of the parameters from the RobotStudio application after installation, as well as from the FlexPendant. With the Spot Servo options some gun specific system parameters may be updated temporarily directly in the robot program using the instruction `STTune`. This function will make tuning of gun parameters easier.

Normally these parameters are supplied by the gun manufacturer.

### References

| Type of information | See |
|---|---|
| `CalibL`, `CalibJ`, `Calibrate`, `SpotL`, `SpotJ`, `SetForce`, `STTune`, `gundata`, `spotdata`, `forcedata` | *RAPID references on page 95* |
| General motion control and programming | *Operating manual - IRC5 with FlexPendant* |
| `ActUnit`, `DeactUnit`, `MoveL`, `MoveJ`, `robtarget`, `tooldata` | *Technical reference manual - RAPID Instructions, Functions and Data types* |
| How to tune a servo gun | *Application manual - Servo gun tuning* |
| Hardware: motors, resolvers, drives, servo gun parameters, tuning a servo gun | *Application manual - Additional axes and stand alone controller* |

## 6.2 General motion control for servo guns

### Introduction

The motion functionality described in this section is common for servo guns and most other types of additional axes. The description is however adapted for servo guns.

### Activation and deactivation

A servo gun may be activated when the robot and all additional axes have come to a standstill by using the `ActUnit` instruction. This means that the servo gun is controlled and monitored by the robot controller.

A servo gun is normally automatically activated directly after loading its parameters and starting up the system (activate at startup). It may be deactivated during program execution later.

If several guns are sharing one tool changer there will be no automatic activation at startup. When the connected gun is activated, it will not be possible to activate another gun until the first one is deactivated (mutual exclusion).

Deactivation of the gun is only needed if the gun has to be disconnected, for service or for a tool change. The deactivation will store the guns current position. This position will be restored when the gun is activated next time. Deactivation is performed with a `DeactUnit` instruction and this will also stop the control and monitoring of the axis.

### Jogging

The position of the gun arm can be jogged with the joystick (see *Operating manual - IRC5 with FlexPendant*). The distance between the two tips is displayed in the jogging window, expressed in mm. An out of range supervision will stop the movement if the gun is reaching max stroke or min stroke. Min stroke is normally zero or a small negative value (gun tips closed to contact with each other).

### Synchronous movements of robot and servo gun

Normally, as for other additional axes a servo gun axis is moved synchronous with the robot movements in such a way that both movements will be completed exactly at the same time. However, it can also be moved independent of the robot movements, for example when closing the gun tips with a force. But during normal movements (for example `MoveL`, `MoveJ`, `MoveC`) in program execution, the tool axis movement will be synchronized. The combined path of robot and servo gun(s) will be repeatable and independent of programmed speed. The robot TCP path, will be the same irrespective of the programmed movements of the servo gun's movable arm.

A robtarget includes position data for additional axes which also will be set when a **ModPos** is performed. Example:

- p10 is a robtarget RAPID data.
- p10.extax.eax_a is the position of the additional axis with logical axis 7.
- p10.extax.eax_b is the position of the additional axis with logical axis 8.

*Continues on next page*

- p10.extax.eax_f is the position of the additional axis with logical axis 12.

Logical axis is a system parameter defined for each axis (RobotStudio: Configuration Editor, Motion, Joint). The robot itself uses logical axes 1-6 and additional axes use 7-12. The user can change the logical axis number to fit the application. Only axes with unique logical axis numbers may be activated at the same time.

For a servo gun, the position is defined as the opening distance of the tips in mm. The value 9E+09 is defined for axes that are not activated.

### Independent gun movement

The gun is in independent mode and can be moved to a specified independent position. During independent mode, the control of the servo gun is separated from the robot. The gun can be closed, opened, calibrated or moved to a new independent position, but it will not follow coordinated robot movements.

The instruction `IndGunMove` is used to set the gun in independent mode and thereafter move the gun to a specific independent position, see *IndGunMove - Activates independent mode for a servo gun on page 130*. This mode can be reset by executing the instruction `IndGunMoveReset`. See *IndGunMoveReset - Resets servo gun from independent mode on page 132*.

### Supervision during general motion control

An out of range supervision will stop the movement if the gun is reaching max stroke or if it is closed to contact with the tips (reaching min stroke). Motion collision detection may be activated for the robot. There is also a separate motion supervision for each controlled axis, including the gun axis. This axis supervision will detect if the gun arm collides or get stuck. A motion error will occur and the motion will be stopped.

## 6.3 Asynchronous movements with force control

### Introduction

The motion functionality described in this section is only valid for servo gun axes.

### Opening and closing in general

The gun may be closed asynchronously (independent of current robot movement) to a predefined plate thickness and tip force. The closing will immediately start to run the gun arm to the expected contact position (thickness). The closing movement will interrupt an on-going synchronous movement of the gun. When the tips reaches the programmed plate thickness, the movement is stopped and there is an immediate switch from position control mode to force control mode. In the force control mode a motor torque will be applied to achieve the desired tip force.

The force remains constant until an opening is ordered unless support for multiple forces are configured. See *Multiple gun forces during welding on page 86*.

Opening of the gun will reduce the tip force to zero and move the gun arm back to the pre-close position, that is, the position of the axis specified in the robtarget. The gun opening may also take place while the robot is moving. But it is not possible if the robot movement includes a synchronized movement of the servo gun axis. In that case a motion error, tool opening could not be synchronized with robot movement, will occur.

### Welding

A gun closing is done when performing a weld. The applied force may be taken from the weld timer or from a RAPID data (spotdata). See *spotdata - Spot weld data on page 140*.

During force build up, the thickness of the plates will be measured. The welding is started when the force is reached but only if the measured plate thickness is approved. When the weld is ready, the gun is immediately opened to the pre-close to position.

In the Spot Servo options, the closing, opening, thickness measurement, weld start and opening is integrated in the `SpotL/J` and `SpotML/MJ` instructions. See *SpotL/SpotJ - The basic spot welding instructions on page 95* and *SpotML/SpotMJ - Spot welding with multiple guns on page 100*.

### Squeezing without welding

A gun closing is also typically done when doing tip dressing or when changing tips. The force will be held constant for a certain time, and then the gun is opened up again.

In the Spot Servo options, the `SetForce` instruction will squeeze the gun with a specified force, thickness and during a specified time. `SetForce` takes a `forcedata` as argument where these values are defined. A thickness test is integrated in the instruction. See *SetForce - Close and Open a gun with desired force and time on page 104*.

*Continues on next page*

### Supervision during asynchronous movements with force control

During the position control phase of the closing/opening, motion supervision is active for the servo gun to detect if the arm collides or gets stuck. There is a maximum motor torque defined in the motion parameters for the gun that never will be exceeded in order to protect the gun from damage.

If the force is programmed out of range according to the guns force-torque table, the output force will be limited to this maximum allowed motor torque and a motion warning will be logged.

During the force control phase, the motion supervision will supervise the gun position not to exceed a certain distance from the expected contact position. This distance, `Forced on Position Limit`, is defined in the motion gun parameters (topic `Motion`, type `Supervision`) and will typically depend on the flexibility of the gun arm. This supervision will protect the gun if for instance one tip is lost.

During the force control phase there is an active speed limitation which will limit the speed of the gun. The speed limit value is defined in the gun parameters (see the tuning chapter in *Application manual - Additional axes and stand alone controller*) or the servo gun tuning manual *Application manual - Servo gun tuning*.

The speed will be actively limited to increase further when the speed limit is reached. The speed limitation will give a controlled behavior of the gun when it is ordered to close to a position where the tips not are in contact, avoiding a hard impact when tip contact is established.

## 6.4 Tip management

### Introduction

The tip management functionality will find and calibrate the contact position of the gun tips automatically. It will also update and monitor the total tip wear of the gun tips. The total tip wear for each gun is stored in a RAPID data (see *gundata - Equipment specific weld data on page 135*). The tips are calibrated with special RAPID instructions. Typically, two gun closings will be performed during a calibration. The calibration may be done when the robot is standing still, see *Calibrate - Calibrate a servo gun on page 114*, or during a robot movement, see *CalibL/CalibJ - Calibrate a servo gun during robot movement on page 108*.

Three different types of calibrations are supported: tip wear, tip change and tool change. All three will calibrate the contact position of the tips. The total tip wear will however be updated differently by these methods.

**Note**

If software equalizing is used there are other methods available for the tip wear compensation. See *Software Equalizing on page 169*.

### Tip wear calibration

To be used after a tip dressing. The gun contact position is calibrated and the total tip wear of the gun is updated. The calibration movements are fast and the switch to force control mode will take place at the zero position.

**Note**

This method must only be used to make small positional adjustments (< 3 mm) caused by tip wear / tip dressing

### Tip change calibration

To be used after mounting a new pair of tips. The gun contact position is calibrated and the total tip wear of the gun is reset. The first calibration movement is slow in order to find the unknown tip collision position and switch to force control. The second calibration movement is fast. This calibration method will handle big positional adjustments of the gun.

This calibration may be followed by a gun closing in order to squeeze the tips in place (using the `SetForce` instruction). A new tip change calibration is then done to update possible positional differences after the tip squeeze.

### Tool change calibration

To be used after reconnecting and activating a servo gun. The gun contact position is calibrated and the total tip wear of the gun remains unchanged. The first calibration movement is slow in order to find the unknown tip collision position and switch to force control. The second calibration movement is fast. This calibration method will handle big positional adjustments of the gun.

*Continues on next page*

The method should always be used after reconnecting a gun since the activation will restore the latest known position of the gun, and that position may be different from the actual gun arm position; the gun arm may have been moved when disconnected. This calibration method will handle big positional adjustments of the gun.

**Tip change requirement**

The total tip wear of the gun (stored in RAPID `gundata`) may be supervised in order to detect when a tip change is needed. See *gundata - Equipment specific weld data on page 135*

**Tool center point adjustment**

Part of the total tip wear may be used to adjust / optimize the tool center point of the robot tool (RAPID tooldata). The instructions `MeasureWearL` or `RecalcTcp` should be used in combination with the `CalibL/J` or `Calibrate` instructions to update the fixed tip of the gun (tool center point). For more information see *Tip wear compensation on page 180* and *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 118* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 126*.

**Supervision during tip calibration**

The same supervision will be active during calibration as during asynchronous movements with force control.

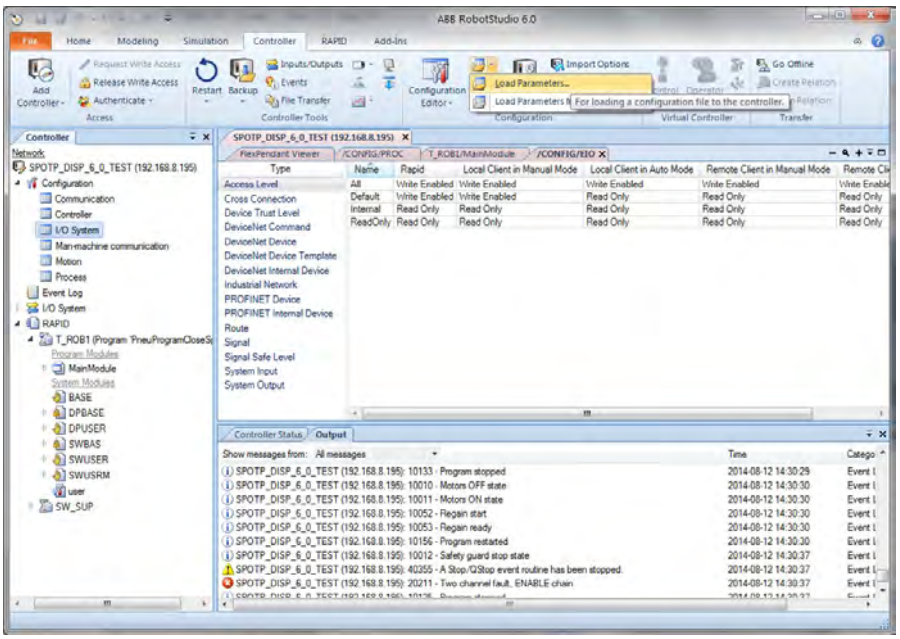
## 6.5 Installation and service

### Install servo gun parameters

If the system is cold started, the servo gun parameters are most likely not loaded.

1. Load the gun parameters from the FlexPendant via the configuration editor, tap **File** and then tap **Add new parameters**. If a complete `moc.cfg` file is loaded, then select **Delete existing parameters before loading** instead.
2. Restart the system.
3. Activate the gun in order to control and monitor the axis if it is not activated (normally the gun is setup activated at startup).

The servo gun parameters can also be loaded from RobotStudio, in the configuration editor.




en1200000249

**Tip**

If a backup is available, restore the backup instead, then the complete system will be ready for production after restart.

*Continues on next page*

### Set the servo gun name

After the gun parameters has been installed and the system has been restarted, the `gundata` needs to be updated with the servo gun name (mechanical unit name). For this a service routine is available to search for installed guns in the system, instead of manually enter the gun name in gundata.

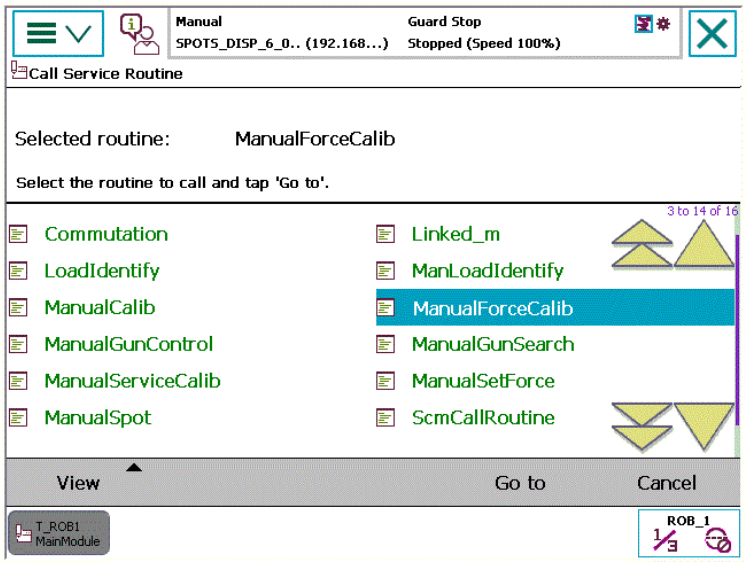
| | Action | Note |
|---|---|---|
| 1 | Run the service routine `ManualGunSearch` to find all configured servo guns in the system, and add their names to the `gundata` array (`curr_gundata`).<br>See *gundata - Equipment specific weld data on page 135*. | Follow the instructions in the routine. |
| 2 | Ready. | |

### Servo gun force calibration

There is a RAPID service routine to calibrate the motor torque versus tip force characteristics, `ManualForceCalib`. A separate sensor is needed to measure the tip force. An optional number of force recordings (2-10) can be made where measured tip force is inserted with corresponding motor torque.

A force calibration must be done in order to get a good force accuracy and also to protect the servo gun from too high forces.

In the Program Editor, Tap **Debug** and then tap **Call Service Routine**. This routine can also be run from the Spot FlexPendant interface.




en1200000585

| | Action | Note |
|---|---|---|
| 1 | Run the service routine `ManualForceCalib`, select 'Setup'. | Change the force calibration setup data, for example the number of calibrations, the sensor thickness and the max force to be used during the calibration. |
| 2 | Then select 'Run'. This will perform the calibrations. | Follow the instructions in the routine. |

*Continues on next page*

| | Action | Note |
|---|---|---|
| 3 | Ready. | |

**Note**

The first time this routine is run and if working from a servo gun template file, a default force table with 2 forces based on the entered max force will be created. Follow the instructions in the routine.

**Note**

For the force change functionality to work correctly, the gun positions are stored in the `SG_PROCESS` table when performing a force calibration, `squeeze_pos1 - 10`. It is really important for the gun force accuracy and performance that this procedure is done properly.

It is also very important that the gun forces used during production later on is in the range of the force calibration table in order to get a acceptable accuracy of the force. See *Multiple gun forces during welding on page 86*.

**Note**

If a gun service / repair has been made, the force calibration should be done again to ensure the performance is the same as before.

### Servo gun force gravity compensation

In the `ManualForceCalib` routine it is also possible to setup configuration data that can be used for gun force gravity compensation. See *The type Spot Gun Equipment on page 31*.

This function can be used if a servo gun loses force when the movable gun arm moves against gravity when closing. Normally there should be no need to compensate for loss of gun force, but for certain types of guns there may be a risk that gravity can influence the gun force negatively depending on on the moveable gun arm weight etc.

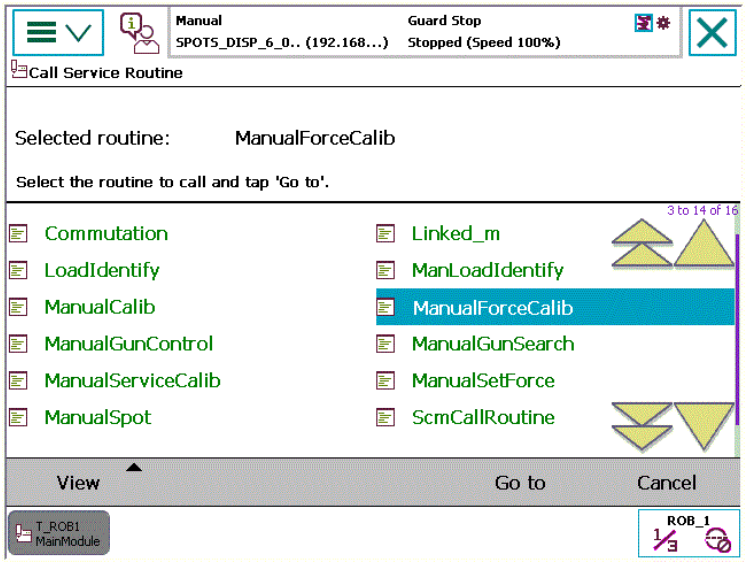
In those cases this functionality can be used to minimize the loss of gun force and maintain a stable force during welding.

There are two methods that can be used to setup the needed compensation data.

- A manual method that requires a hand held force sensor.
- An automatic method that will move the gun (axis 5) between 0 to 90° and 0 to -90° and calculate/estimate the gun force in the "worst" angle and update the compensation data.

*Continues on next page*

In the Program Editor, Tap **Debug** and then tap **Call Service Routine**. This routine can also be run from the Spot FlexPendant interface.




en1200000585

| | Action | Note |
|---|---|---|
| 1 | Run the service routine `ManualForceCalib`, select 'Setup'. | Change the force calibration setup data, i.e. the sensor thickness and the max force to be used during the setup. |
| 2 | Then select 'Gravity'. | Follow the instructions in the routine. |
| 3 | Ready. | |

**WARNING**

If using the automatic method the robot will first move to sync position, and then start move axis 5 in-between 0 and 90° and/or 0 to -90°. Make sure that the robot can move freely without crashing into objects around it.

**Note**

The actual force will not be compensated with 100% accuracy, but the deviation from ordered gun force will be less than without compensation.

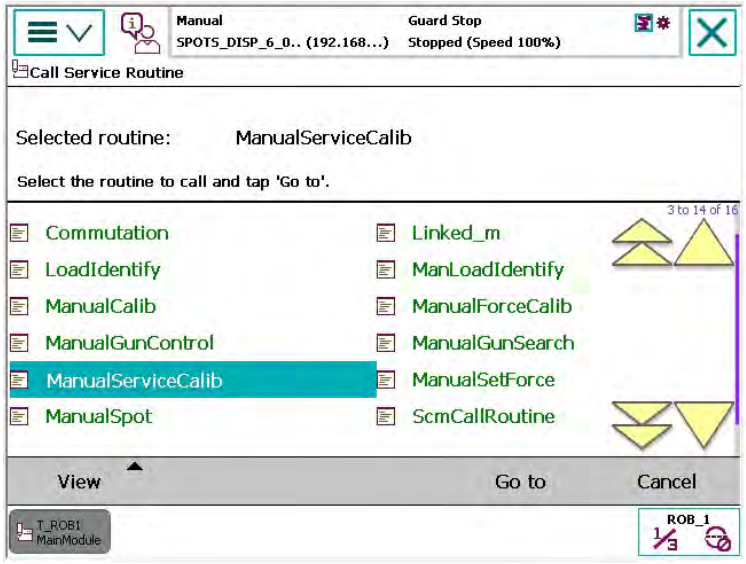
## Servo gun init calibration

After installing the gun parameters and restarting the system, the gun like any other additional axis must be calibrated by performing a fine calibration or a revolution counter update. Apart from other kinds of additional axes, it is also required to run a RAPID service routine, `ManualServiceCalib` to find the contact position or zero position of the gun.

There are two options in this routine that can be run depending on if the gun has been fine calibrated or if the revolution counters has been updated. See *Manual actions on page 74*.

*Continues on next page*

In the Program Editor, tap **Debug** and then tap **Call Service Routine**. This routine can also be run from the Spot FlexPendant interface.




en1200000247

## Servo gun init calibration after fine calibration

Fine calibration must be performed when installing a new servo gun or if the servo gun axis is in state Not Calibrated.

Use this procedure to fine calibrate.

| | Action | Note |
|---|---|---|
| 1 | On the FlexPendant go to the **Calibration** view, select the desired gun and then tap **Fine calibration**.<br>There is no need to jog the axis to any particular position. | **Note**<br>Make sure that new tips are used since tip wear data will be cleared. |
| 2 | Run the service routine *ManualServiceCalib* option 2.<br>Follow the instructions in the routine. | If the gun is considered to be force calibrated, the gun will move fast to the selected pre-position and then close slowly until tip contact is detected, since the zero position is unknown. Otherwise a warning dialog will be displayed with a question whether the gun has been force calibrated or not. Answering Yes will perform the service calibration anyway, No will end the routine.<br>**Note**<br>If the gun is not force calibrated and properly tuned, follow the tuning procedure described in *Application manual - Servo gun tuning*. |
| 3 | As a result, the gun position is updated to be zero in the position of contact and the tip wear value is reset. | Ready. |

*Continues on next page*

Servo gun init calibration after revolution counter update

An update of the revolution counter must be performed if the position of the axis is lost. If this happens, this is indicated by the calibration state Rev. Counter not updated. These steps are required to update the counter.

| | Action | Note |
|---|---|---|
| 1 | On the FlexPendant go to the **Calibration** view, select the desired gun and tap **Update revolution counters**.<br>There is no need to jog the axis to any particular position. | |
| 2 | Run the service routine `ManualServiceCalib` option 1.<br>Follow the instructions in the routine. | If the gun is considered to be force calibrated, the gun will move slowly until tip contact is detected, since the zero position is unknown. Otherwise a warning dialog will be displayed with a question whether the gun has been force calibrated or not. Answering Yes will perform the service calibration anyway, No will end the routine.<br>**Note**<br>If the gun is not force calibrated and properly tuned, follow the tuning procedure described in *Application manual - Servo gun tuning*. |
| 3 | As a result, the gun position is updated an integer number of revolutions to be zero in the position of contact. Tip wear of the gun remains unchanged. | |
| 4 | Ready. | |

**Note**

The first time this routine is run and if working from a servo gun template file, a default force table with 2 forces based on the entered max force will be created. Follow the instructions in the routine.

**Note**

It will not be possible to run any Spot instructions until a gun init calibration has been done.

**WARNING**

If the force calibration procedure has not been done properly, the servo gun can be damaged, please make sure that the servo gun is force calibrated and tuned and that the force calibration values are correct, see *Application manual - Servo gun tuning*.

*Continues on next page*

### Disconnect and reconnect a servo gun, tool changing

If the servo gun is deactivated, using the `DeactUnit` instruction, it may be disconnected and removed. The gun position at deactivation will be restored when the gun is connected and reactivated. Make a **tool change calibration** to make sure the tip position is OK in case the gun arm has moved while it was disconnected.

Simplified tool change procedure.

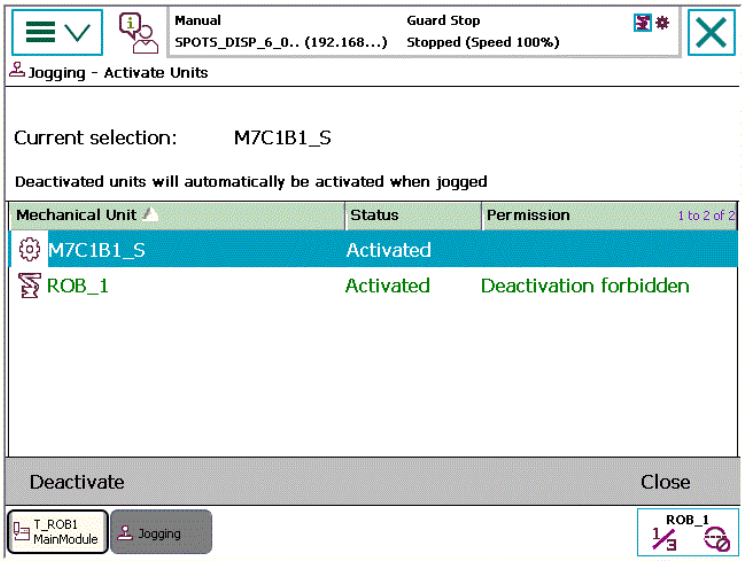
1. Run the routine `DeactUnit`.
2. Disconnect the gun.
3. Connect the second gun.
4. Run the routine `ActUnit`.
5. Perform a tool change calibration for the second gun.
6. Start using the second gun.

For more information about tool changing see *Servo tool change on page 211*.

### Recover from accidental servo gun disconnection

If the motor/resolver cables are disconnected by accident when the servo gun is activated, the servo gun must be deactivated in order to move the robot to a service position.

1. To deactivate the gun, tap **Deactivate** in the Jogging window and deactivate the mechanical unit.
2. Move the robot to a service position and repair the gun.
3. Perform a revolution counter update since the position has been lost.
4. Perform a gun position init calibration option 1, see *Servo gun init calibration on page 205*.




en1200000586

*Continues on next page*

### Replace a servo gun

Normally there is no need to replace the gun parameters if the the new gun is identical to the old one.

1. Connect the new gun.
2. Startup the system.
3. Perform a fine calibration of the gun.
4. Perform a gun position init calibration, see *Servo gun init calibration on page 205*.
5. Perform a force calibration if it is not done before for that gun, see *Servo gun force calibration on page 203*.

**Note**

The spare gun must have same parameter names as the original gun, otherwise the installation will just add the new gun, keeping the old gun in parallel. Eg `SGUN_1`.

## 6.6 Stationary gun

### Description

A stationary servo gun is mounted on the floor and the robot is holding the work piece. The only difference when using a stationary servo gun is that the robot tool (RAPID `tooldata`) should be defined as stationary (robhold = false), and the used work objects as robot held.

## 6.7 Servo tool change

### Description

It is possible to change servo gun during production. The functionality is realized as the option *Servo Tool Change*. There is no software limitation in how to combine different kinds of servo guns (for example brands, sizes or motors) with a tool changer.

The used servo guns share the same drive unit, and the same node on the measurement board. They are activated as different mechanical units, but of course never at the same time. They may use the same or different logical axis.

### Prerequisites

Changing gun requires a deactivation of the operating gun and then unplugging its motor cables. The motor cables are plugged in to the next gun, and this gun is activated and ready to run. The plug-in mechanism requires a mechanical tool changer interface to the guns. One individual set of gun parameters are installed for each gun.

### Limitations

Up to 8 additional axes (servo guns or other axes) can be installed simultaneously in one robot controller. All or some of them may be servo guns sharing a tool changer.

**Note**

Tool changing with servo guns requires the option *Servo Tool Change*.

### Changing *Motion* parameters

The system parameters in type *Mechanical Unit* and *Relay* and *Measurement Channel* (topic *Motion*) should be set like this when tool changing.

1. Set *Activate at StartUp* to No.
2. Set *Deactivate Ptc at Disconnect* to Yes.
3. Define *Use Connection Relay* with the same name as defined in *Name*, for example SGUN1.
4. Define an *Input Signal* in type *Relay* with a signal that is defined in the topic *I/O* (EIO.cfg),

   For example *diMecUnitName*, and this input signal should be connected to a sensor on the tool stand or the tool changer.
5. Set the parameter *Disconnect at Deactivate* in type *Measurement Channel* to Yes.

*Continues on next page*

If this setup is used a safe tool change functionality will be achieved.

**Note**

To be sure that the right servo gun is activated and have a safe way to tool change, it is recommended that the connection relay functionality is used when tool changing.

**Example, tool change procedure**

The procedure to switch between gun 7A and gun 7B must includes these minimal actions (excluded here is the needed communication with the tool changer, the tool stand and necessary robot movements):

1 Deactivate gun SGUN1.
   The position of gun SGUN1 is stored.
2 Disconnect gun SGUN1.
   Disconnect the servo gun motor cables.
3 Connect gun SGUN2.
   Connect the motor cables to motor SGUN2.
4 Activate gun SGUN2.
   The latest position of gun SGUN2 is restored.
5 Run a tool change calibration of gun SGUN2.
   Make sure the position is correct.

**WARNING**

If the servo gun axis has been moved during deactivation, the position of the axis might be wrong after activation, and this will not be detected by the controller. The position after activation will be correct if the axis not has been moved, or if the movement is less than 0.5 motor revolutions. Always use the tool change tip calibration after activation. The tool change calibration will adjust any positional error caused by gun movements during deactivation.

*Continues on next page*

**WARNING**

It is important that no other mechanical units that are used with a tool changer, are activated but only the one corresponding to the currently connected servo gun!

An activation of wrong mechanical unit may cause unexpected movements or errors. Some tool changers support I/O signals that specifies which gun is currently connected. That information may be used to make sure correct mechanical unit is activated. It is also possible to lock activation of not connected mechanical units by specifying a digital input (DI) in the connection relay which is a motion system parameter (type *Relay* in topic *Motion*) for each servo gun. This digital input, which also is setup in the EIO.cfg, is read when the mechanical unit is activated. If set to 1 the activation will take place normally, otherwise a recoverable motion error will occur and the activation will be denied.

This page is intentionally left blank

# 7 FlexPendant Interface

## 7.1 Application Overview

### Introduction

This chapter describes the Spot FlexPendant interface intended to simplify the use of the spot welding functionality. The operator have the most common information, data and signals collected together in one place, easy to use and understand. This is not a replacement for the standard FlexPendant functionality, but it can be seen as a complement. Spot related information are presented in an instructive way, enabling operators to easily and quickly get their every day tasks done.

To start *RobotWare Spot*, tap the ABB menu and then tap **RobotWare Spot**. The *RobotWare Spot* main view will be started. From here all spot related functions in the GUI can be accessed. For further information on using the FlexPendant, see *Operating manual - IRC5 with FlexPendant*.

en1200000603

### Topics

The Spot GUI covers following topics:

- **Main View**
- **Process Signals**
- **Process Data**
- **Manual Actions**

The Spot main view provides basic information about the current executing spot program with possibilities to reach other views and sub views. You can follow the welding process in every instruction by looking in the welding progress part of the window. It is also possible to quickly change or edit some of the data used in a spot instruction by tapping the blue colored text, `spotdata` and `gundata`. The equipment I/O status is shown in this window and the spot process status, also

*Continues on next page*

the latest errors and warnings are shown and can be accessed from here. The manual actions can be accessed and run from this view also.

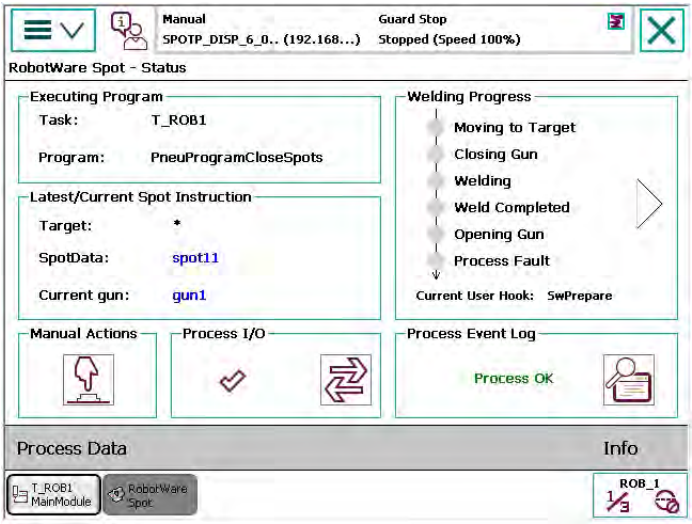
**Tip**

It is possible to configure an additional test view to be able to run RAPID program from the FlexPendant Interface. Go to, ABB/Control Panel/FlexPendant/Additional Test View.

## 7.2 Main View

### Basic functionality in the main view

1 Tap the **ABB menu** and then select **RobotWare Spot**. A window will appear containing functionality to follow the welding progress step by step, view the latest/current spot instruction, see the name and path to the executing weld program, access manual actions, access and edit the process data types and also see the overall status of the equipment signals connected to the welding equipment.




en1200000504

2 Tap the **Welding Progress** control to access and edit simulation data.

3 Tap **Process Data** to view and edit the process data.

4 Tap **Manual Actions** to access an run the spot service routines.

5 Tap **Process I/O** to view the connected equipment signals.

6 Tap the **Process Event Log** to view the latest spot error or warning messages (the latest 20 messages will be stored.)

7 Tap **Task** button to view another spot welding robot.

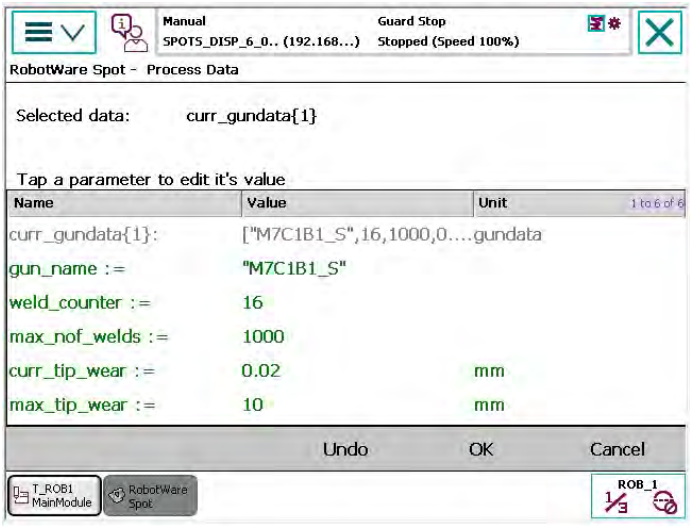
### Viewing or editing data

1 Tap the blue colored data or tap **Process Data**.

2 Select the data to be changed.

*Continues on next page*

3 View or edit the value.




en1200000505

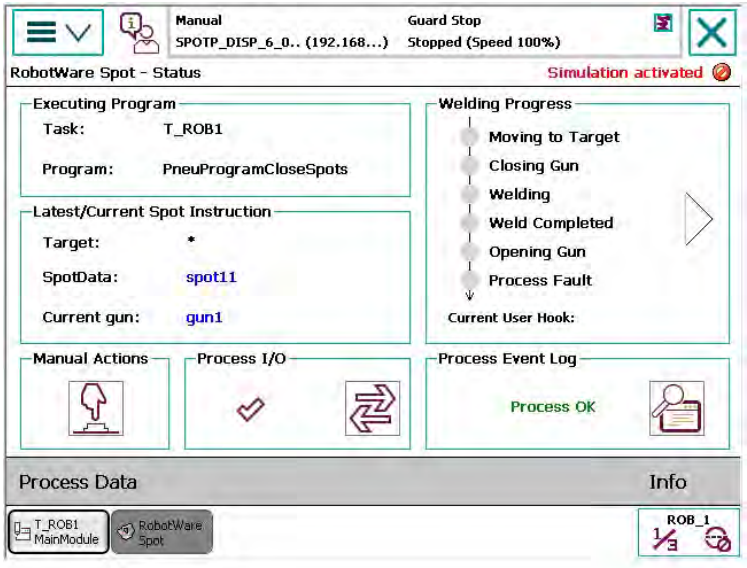
4 Tap **OK** to write the new value to the controller and to leave the data dialog and return to the Status window.

Tap **Undo** to reset the window from all changes.

Tap **Cancel** to return to the Status window without any changes.

## Simulation

If you change the sim_type in simdata, when the value is separated from 0 (0 means weld mode) a little stop sign with a status text is visible at the top right corner in every window. Tap the Welding Progress control to edit the simdata, see picture below. For more information about simulation see *Simulation modes on page 72*.

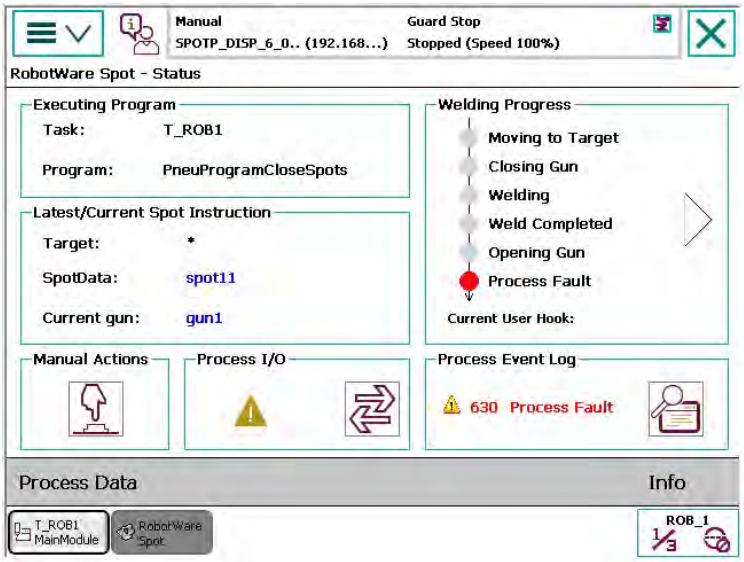



en1200000514

*Continues on next page*

### Process Event Log

Information in this area shows if there is a new Spot error. Touching the button will open a list with the latest Spot errors from the error log, with possibilities to get more information about the problems. The error information in the Main View will be cleared when a new Spot instruction is executed.




en1200000604

## 7.3 Process Signals

### Process Signals

Process Signals displays the input and output signals of the configured and connected spot process equipment, media panel, weld timer and gun.

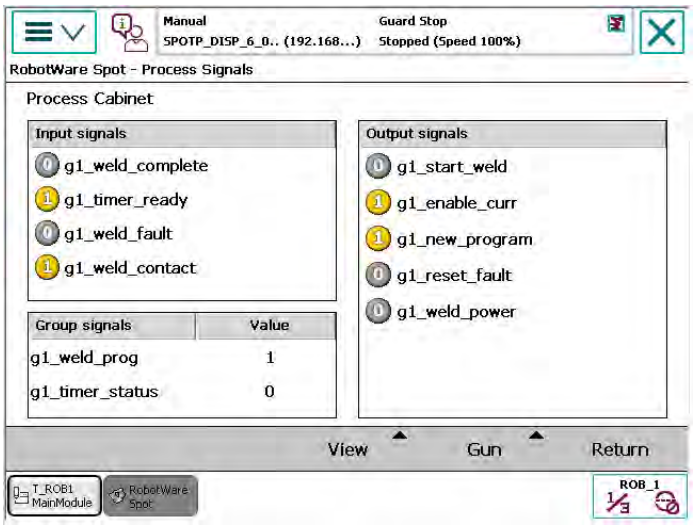
Digital signals are show in an illustrative way, reflecting the real equipment of the specific spot system in use. When running the system in manual mode, it is possible not only to view but also to set new values to process signals.

**Note**

In order to set signals, the right access is needed, *IO write access*. This gives the possibility to set I/O signal values. For more information about grants see *Operating manual - IRC5 with FlexPendant*.

### Basic functionality in the Process Signals window

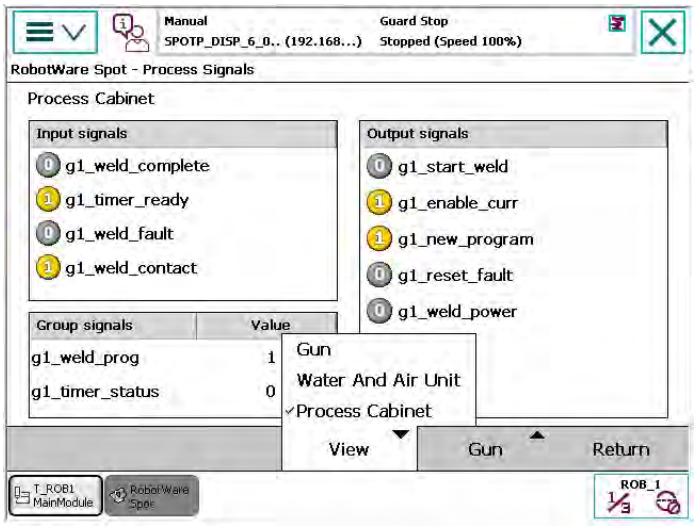
1 Tap the **Process I/O** button in the RobotWare Spot main view. A window appears containing functionality for viewing digital input signals connected to different parts of the equipment and for setting/resetting some output and group signals.




en1200000506

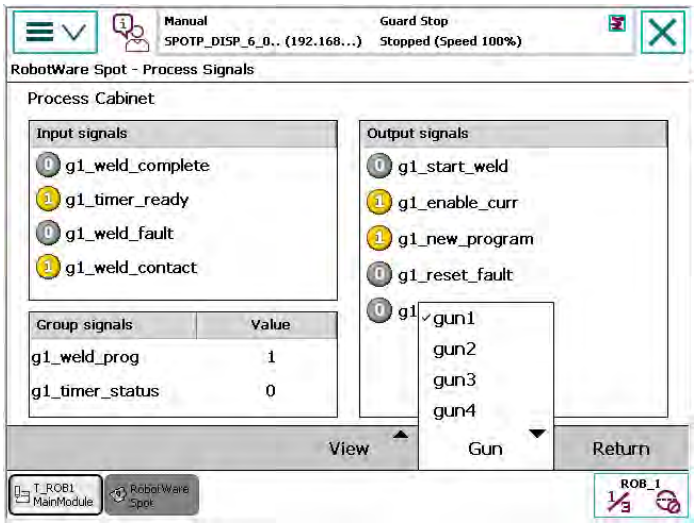
*Continues on next page*

2 Tap **View** to change to other parts of the equipment.




en1200000507

3 Tap **Gun** to switch between all available guns in the system.




en1200000508

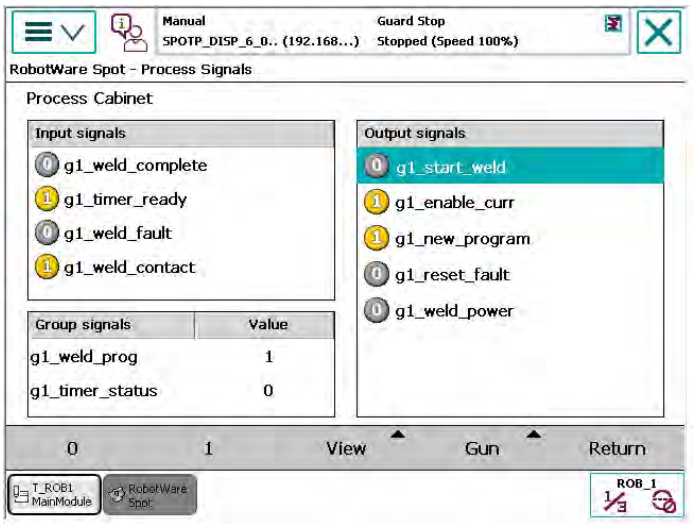
### Setting or resetting process signals

1 Select the signal to be changed.

Depending if it is an output or a group signal to change you get different behavior in the command bar.

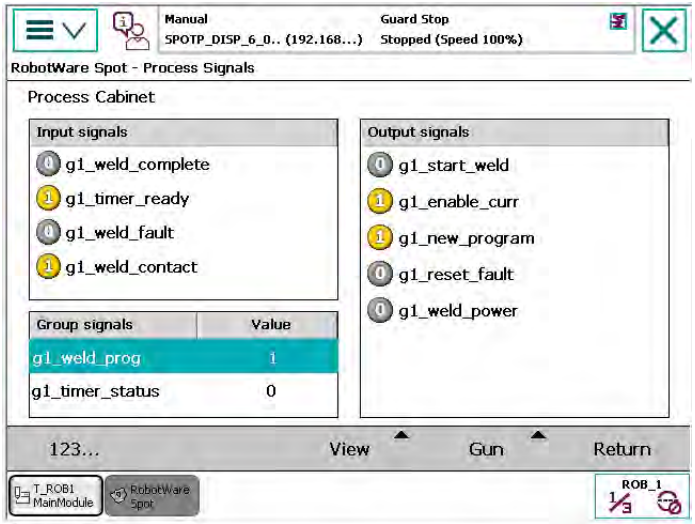
*Continues on next page*

2 To change an output signal toggle between 1 and 0 to set or reset the signal.




en1200000509

3 To change a group signal tap **123...** to open a numeric pad to written in a new value. Then tap **OK** to close the pad.




en1200000510

## 7.4 Process Data

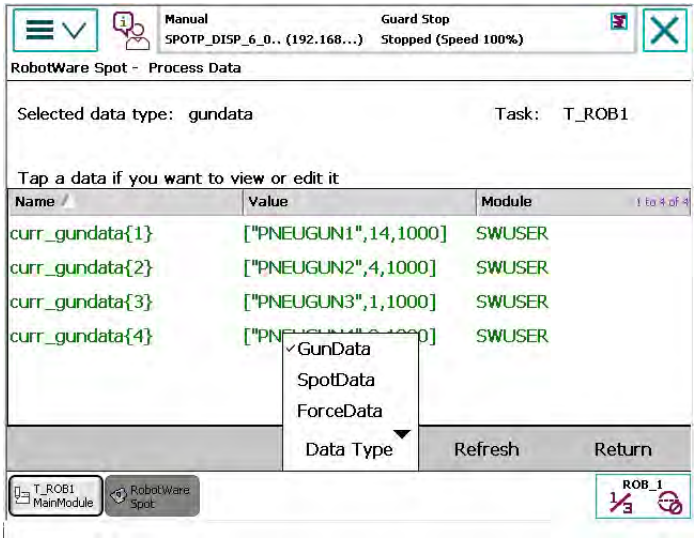
### Process Data

Process Data presents spot specific data types and lists all instances of a selected data type, thus offering a quick and easy way to edit or view `simdata`, `forcedata`, `gundata`, or `spotdata` in order to improved the system performance.

> **Note**
>
> In order to edit RAPID data, the right access is needed, *Modify current value*. This gives access to modify the value of RAPID variables. For more information about grants see *Operating manual - IRC5 with FlexPendant*.

### Basic functionality in the Process Data window

1 Tap **Process Data** in the command bar in `RobotWare Spot` main view. A window appears containing functionality for viewing and editing data types connected to the spot welding functionality.




en1200000511

2 To update the window and do a new search for the selected data type tap **Refresh**.

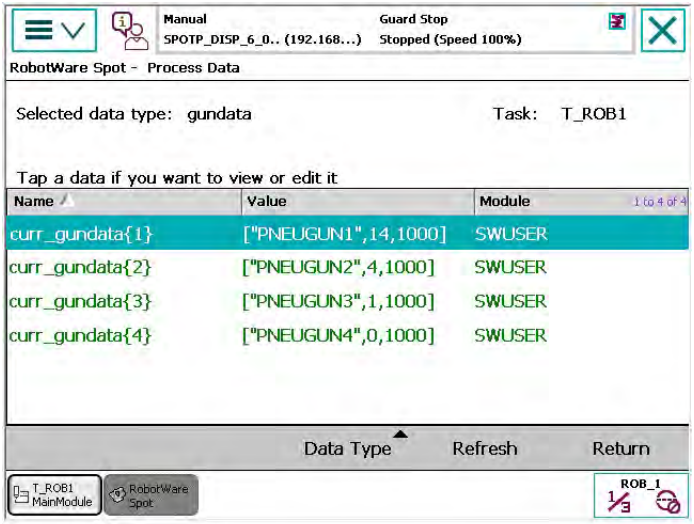
### Viewing data types

The data types available from the menu **Data Type** are `simdata`, `forcedata`, `gundata`, and `spotdata`. Each type is thoroughly described in *Programming on page 63*.

1 Tap **Data Type**. Select the data type you want to view or edit.

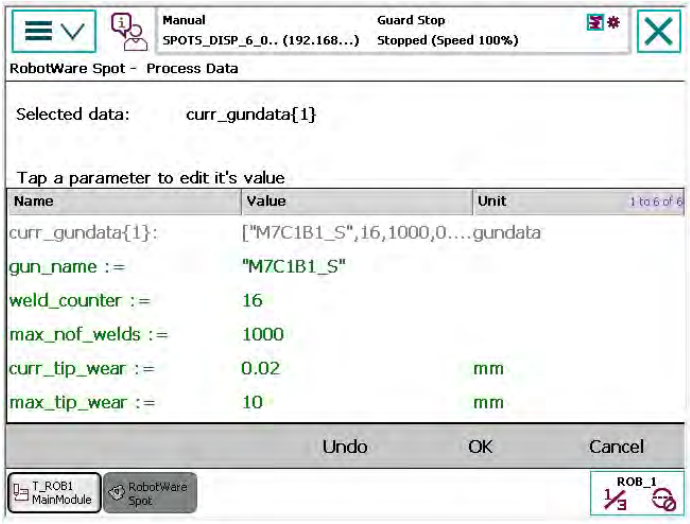
*Continues on next page*

2 To change the data of the listed data type, tap the data.




en1200000512

3 Tap the value. The declaration of the data is displayed.




en1200000513

## Editing data types

1 Select the data instance you wish to edit from the list.
2 Select the component you want to edit in the Data dialog.
3 Depending on the component data type, enter the new value in the numeric or alphabetic pad that will appear.
4 When ready tap **OK** button to close the pad.
5 Tap **Refresh** to cancel all changes and start from the beginning again.
6 Tap **OK** to write the new value to the controller and to leave the Data dialog.

*Continues on next page*

More information on the individual components can be found in *spotdata - Spot weld data on page 140*, *gundata - Equipment specific weld data on page 135*, *forcedata - Spot gun force data on page 143*, and *simdata - Simulation data on page 146*.

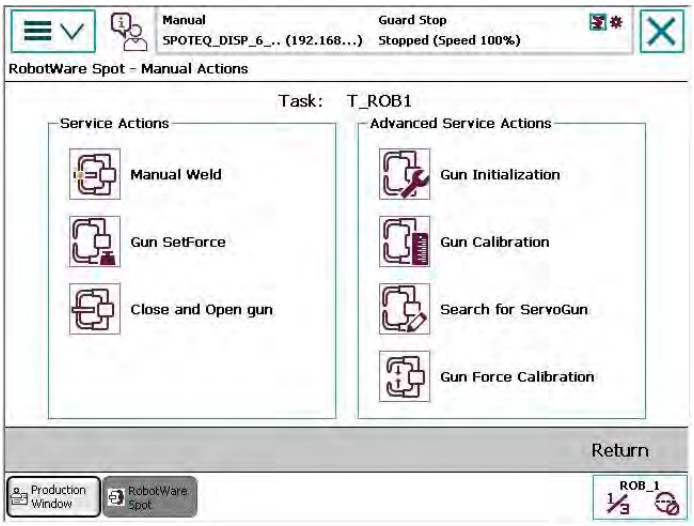
## 7.5 Manual Actions

### Manual Actions

Manual Actions contains all available spot related service routines in the system. The user can easily start any routine by tapping the button for the action he would like to run.

### Basic functionality in the Manual Actions window

1 Tap **Manual Actions** button in the `RobotWare Spot` main view. A window appears containing all available spot related service routines in the system.




en1200000515

### Starting a service routine

1 Make sure that a program is loaded without errors and set the system in MotorsOn state.

2 Tap the service routine you would like to run.

For a complete description of the available service routines, see *Manual actions on page 74*.

### Manual versus Automatic mode

If the system is in manual mode you can run service routines, view or edit data and set or reset output signals. When you switch to automatic mode you loose the possibility to edit data and set or reset output signals but you can still view the current data or output signals.

## 7.6 Customizing the GUI

### Configuration of the Process Signals view

The I/O signal content of the Process Signals view can be modified if needed. The signal names used are defined in the process configuration.

The configuration is separated for each view:

Process Cabinet: See *The type Spot Weld Equipment on page 27*

Gun: See *The type Spot Gun Equipment on page 31*.

Water And Air Unit: See *The type Spot Media Equipment on page 36*.

**Tip**

It is also possible to add custom signals in each view if needed in `User defined gui signal1-10` in the process configuration. These signals are not used in the spot process, only used to show additional signals in the spot gui.

### Configuration of visible Process Data

The simdata and forcedata can be hidden if needed in the Process Data view. See *The type Spot GUI on page 40*

### Configuration of the Manual Actions view

The Manual Actions view can be modified to fit the current configuration. Possible configurations are routines for a Servo, Pneu or Combined system. See *The type Spot GUI on page 40*

### Configuration of automatic startup of the Spot GUI

It is possible to configure automatic startup of the GUI via process configuration, `Show Spot GUI at startup`. See *The type Spot GUI on page 40*

This page is intentionally left blank

# 8 Bosch FlexPendant Interface

## 8.1 Application Overview

### Introduction

This chapter describes the user interface intend to simplify the use of the Bosch weld timer functionality. To access the Bosch FlexPendant interface you need the software option *Bosch Weld Timer Interface*. You also need recommended hardware (Bosch weld timer, cable for communication).

The operator has the most common data and weld timer errors and fault collected together in one place, easy to use and understand. To start Bosch Timer tap the **ABB menu** and then tap **Bosch Weld Timer**. The Bosch Timer desktop shows all common weld timer functions. For further information on using the FlexPendant, see *Operating manual - IRC5 with FlexPendant*. For information about weld parameters, see *Bosch operating and programming manual volume 2*.


en1200000605

Bosch Timer desktop has four function buttons:

- Pre Warning
- Weld Fault
- Last Weld
- Weld Parameters

### Pre Warning

Pre Warning present information about all electrodes configured for a certain weld timer.

### Weld Fault

Weld Fault lists all weld faults and warnings connected to the welding process and the weld timer.

*Continues on next page*

Last Weld

Last Weld present information about the last weld performed by the weld timer.

Weld Parameters

Weld Parameters offering a quick and easy way to view or edit ordinary weld parameters.

**General information before start using the Bosch interface**

The Bosch application cannot access all timer functions, so before using the Bosch application some necessary offline setup is needed, use the PC software BOS5000/BOS6000. Connect the PC to the X1 serial connection port on the timer.

- Transformer parameter setup.
- Gun force and weld current calibration of the electrodes.

**Note**

If or when an extra ordinary weld fault (hardware fault) occurs the user have to connect the BOS5000/BOS6000 pc-software to get the real cause of the problem.

**Note**

To do a backup or restore the weld timer the user have to connect the BOS5000/BOS6000 pc-software to take this action.

**Note**

Make sure you understand what happens to the welding sequence if you turn off the Ignition parameter under the Settings or General node on the FlexPendant, or if you change the simulation type in *RobotWare Spot*, see *Simulation modes on page 72*.

**Note**

Avoid to close the application during loading or saving weld parameters in the Weld Parameters window, this can lead to loss of data.

**Limitations**

- It is only possible to connect one weld timer to a robot controller.
- Avoid disconnecting the RS232 cable during loading or saving weld parameters in the **Weld Parameters** window, the Bosch MMI application should be closed before disconnecting the cable.
- It is not possible to access all timer functions and settings from the Bosch MMI application.
- It is not possible to backup or restore weld parameters from the Bosch MMI application.
- Not all possible timer faults will be visible in the **Weld Fault** view.
- Can not be combined with the option 634-1 MultiProcess.

*Continues on next page*

**Manual versus Automatic mode**

If the system is in manual mode you can view or edit weld parameters and see information about the warning and error that can occur in the system. When you switch to automatic mode or when executing a program you cannot open the Weld Parameters window but you can still open the other windows.

**General information**

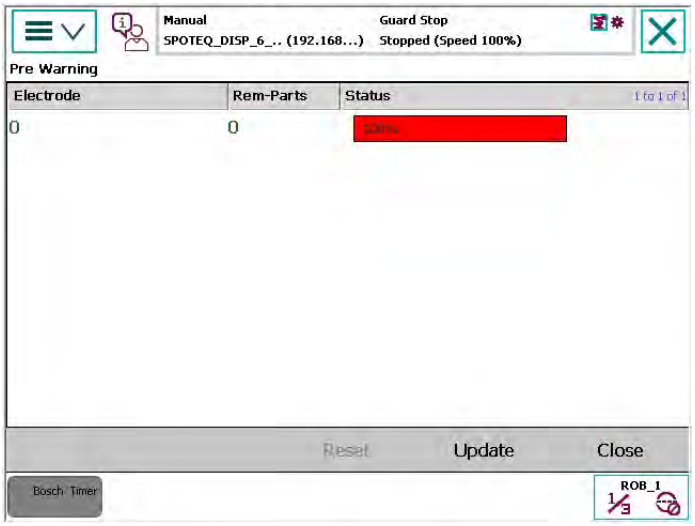
The information at the upper right side shows if the communication with the timer is ok or not (if problem see the system Event Log for further information). In the upper left side you have the information about the used device net protocol for the connected timer.

## 8.2 Pre Warning

### Basic functionality in the Pre Warning window

1 Tap **Pre Warning**.

A window will appear containing information of all configured electrodes in the weld timer.




en1200000588

2 Tap **Reset** to reset the value of the selected electrode (only possible if a row is selected in the list view).

3 Tap **Update** to search for new information about the electrodes.

4 Tap **Close** to return to the Bosch Timer desktop.

### Resetting the value of an electrode

1 Select the electrode in the list view to be changed.

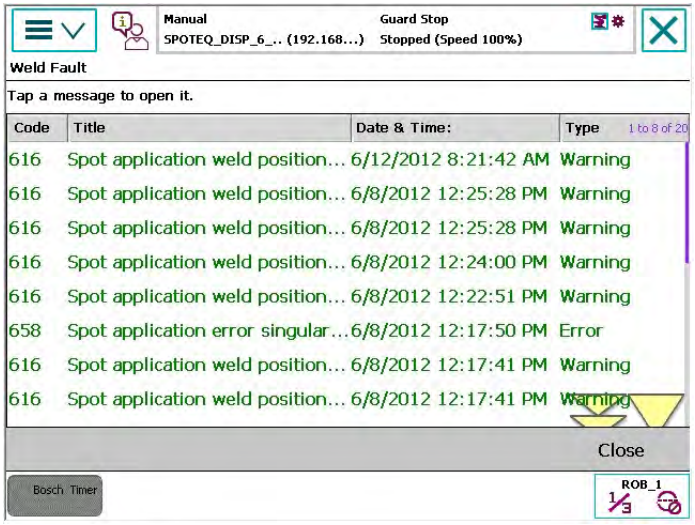
2 Tap **Reset**.

## 8.3 Weld Fault

### Basic functionality in the Weld Fault window

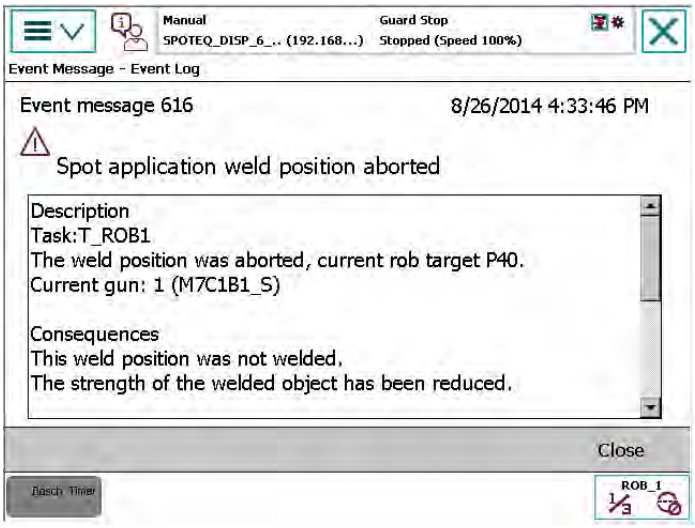
1 Tap **Weld Fault**.

A window will appear containing information about all errors and warnings connected to the welding process and the weld timer.




en1200000589

2 Tap twice on a row in the list view to open up a new window for more information about the weld error or warning.



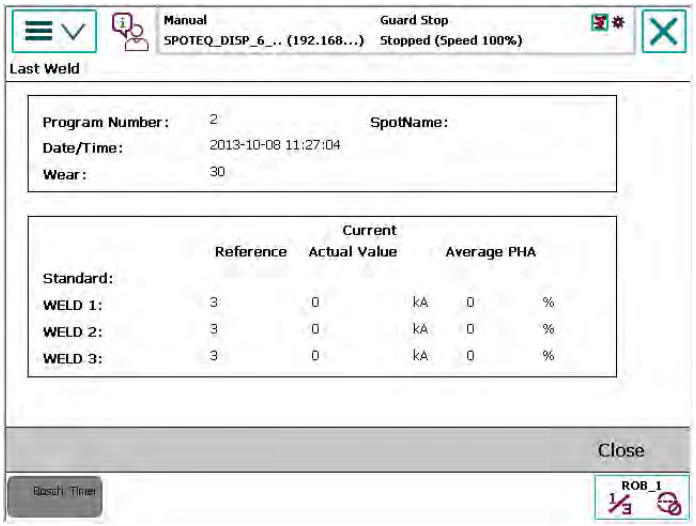

en1200000590

3 Tap **Close** to return to the Bosch Timer desktop or **OK** to return to the previous window.

## 8.4 Last Weld

**Basic functionality in the Last Weld window**

1 Tap **Last Weld**. A window will appear containing information about the last weld performed by the weld timer.




en1200000591

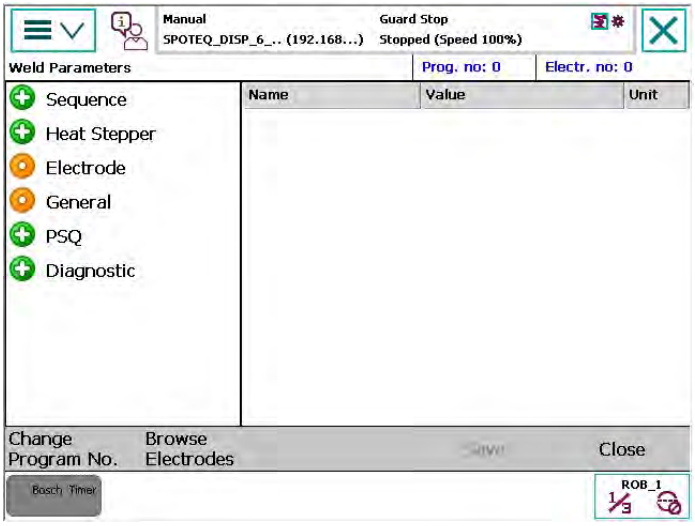
2 Tap **Close** to return to the Bosch Timer desktop.

## 8.5 Weld Parameters

### Basic functionality in the Weld Parameters window

1 Tap **Weld Parameters**.

A window will appear containing information about the weld parameters that is possible to view and edit in the weld timer (this view is not available in automatic mode).




en1200000592

2 Tap **Change Program No.** to view weld parameters for different programs in the weld timer.

3 Tap **Browse Electrodes** to change electrode number.

4 Tap **Close** to return to the Bosch Timer desktop.

**WARNING**

Changing parameters in this view requires very good knowledge about the welding equipment and the welding parameters, otherwise it is easy to damage or destroy the welding equipment and/or compromise the welding quality.

**Note**

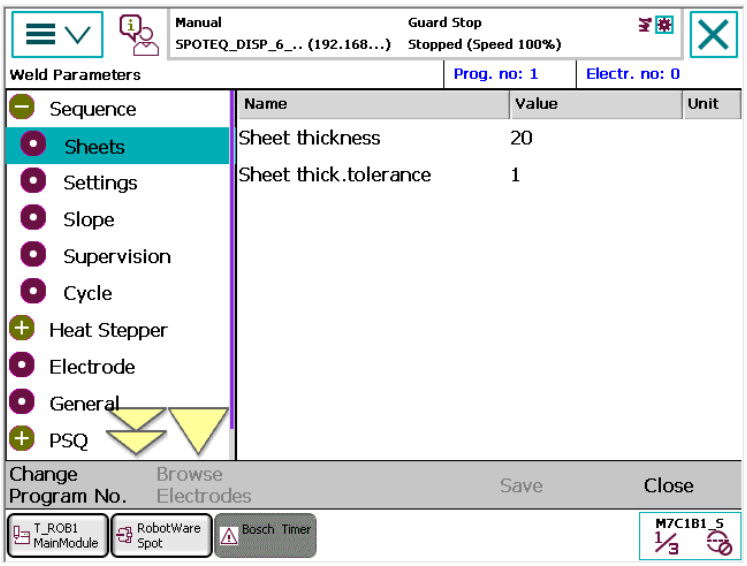
In order to edit RAPID data, the right access is needed, *Modify current value*. This gives access to modify the value of any RAPID variable. For more information about grants see *Operating manual - IRC5 with FlexPendant*.

*Continues on next page*

### Sheets node

The sheets node contains parameters related to the sheet thickness and sheet tolerance for each welding program.

See *Bosch weld timer manual volume 2* for more information about the parameters.




xx1600000035

1. Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2. Tap **Save** to save the changes done to the parameters and stay in current window.
3. Tap **Close** to close the window and return to the previous window.

**Note**

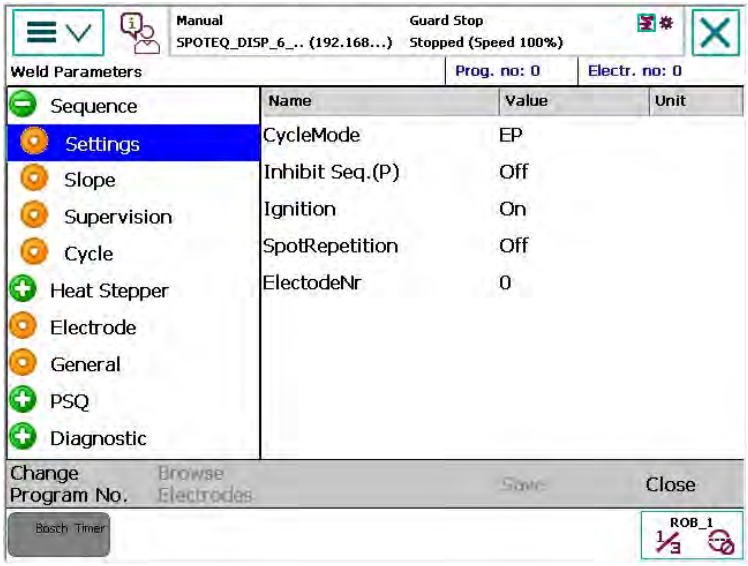
The values can be specified as a I/O group value. The values can be recalculated into millimeters if configured, see *How to use spot data programmed in the weld timer on page 258*.

*Continues on next page*

### Settings node

The settings node contains parameters related to the selected welding program.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000595

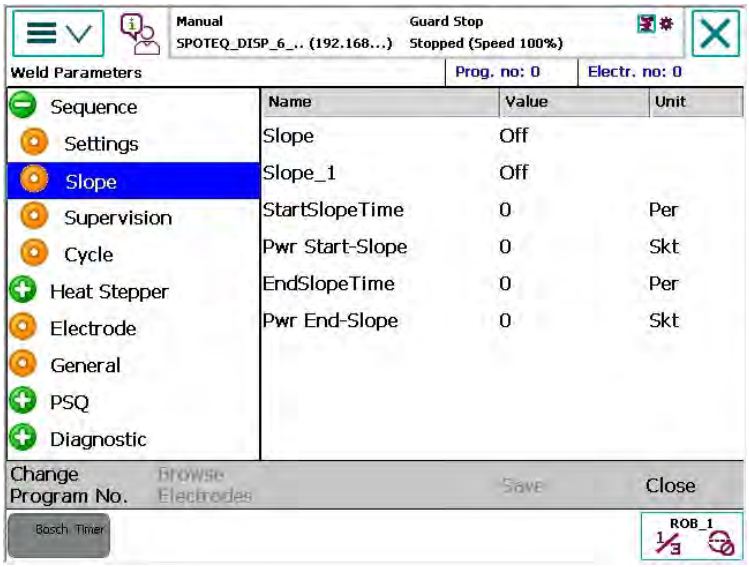
1 Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2 Tap **Save** to save the changes done to the parameters and stay in current window.
3 Tap **Close** to close the window and return to the previous window.

*Continues on next page*

### Slope node

The slope node contains parameters related to the welding sequence, if up and down slope of the current is required changes are done here.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000594

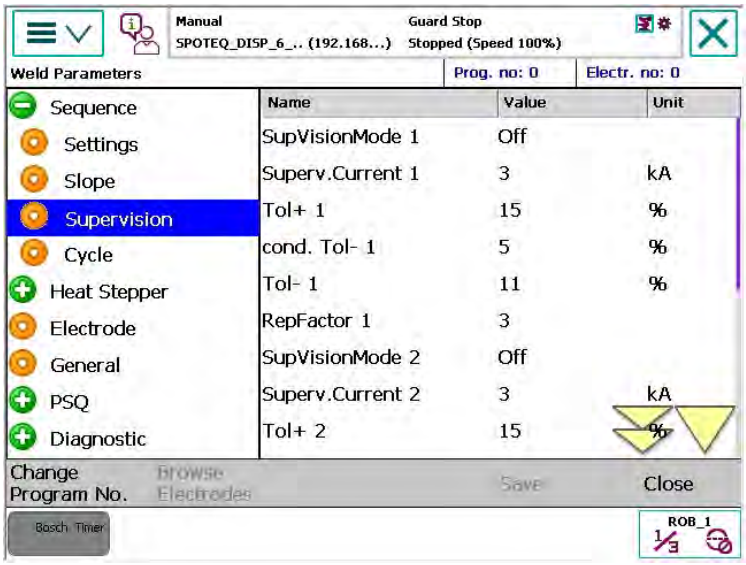
1. Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2. Tap **Save** to save the changes done to the parameters and stay in current window.
3. Tap **Close** to close the window and return to the previous window.

*Continues on next page*

### Supervision node

The supervision node contains parameters related to the supervision of the current. Each weld time (1, 2, 3) can be supervised separately.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000601

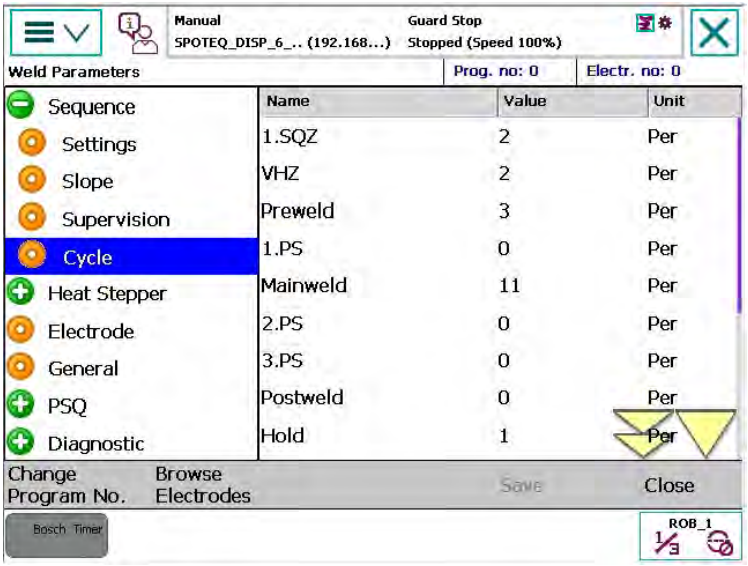
1 Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2 Tap **Save** to save the changes done to the parameters and stay in current window.
3 Tap **Close** to close the window and return to the previous window.

*Continues on next page*

### Cycle node

The cycle node contains parameters related to the welding sequence.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000593

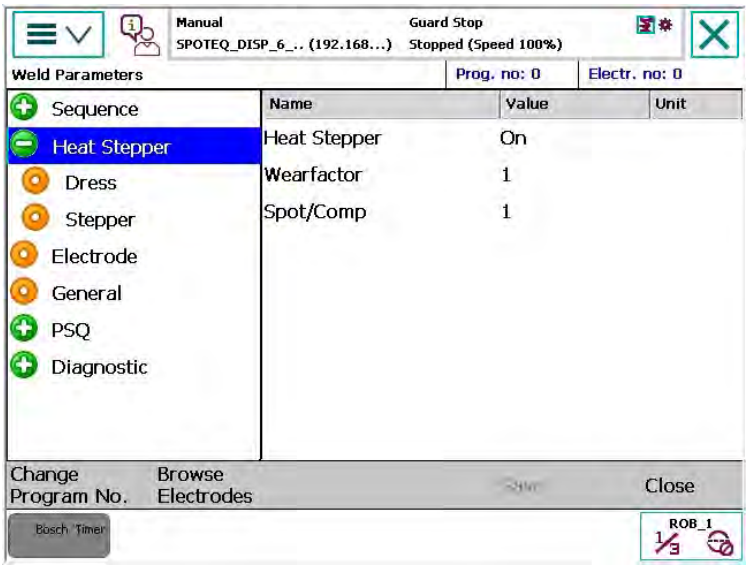
1. Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2. Tap **Save** to save the changes done to the parameters and stay in current window.
3. Tap **Close** to close the window and return to the previous window.

*Continues on next page*

### HeatStepper node

The heatstepper node contains parameters related to the wear of the electrodes, if the customer use this parameters changes are done continuously.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000596

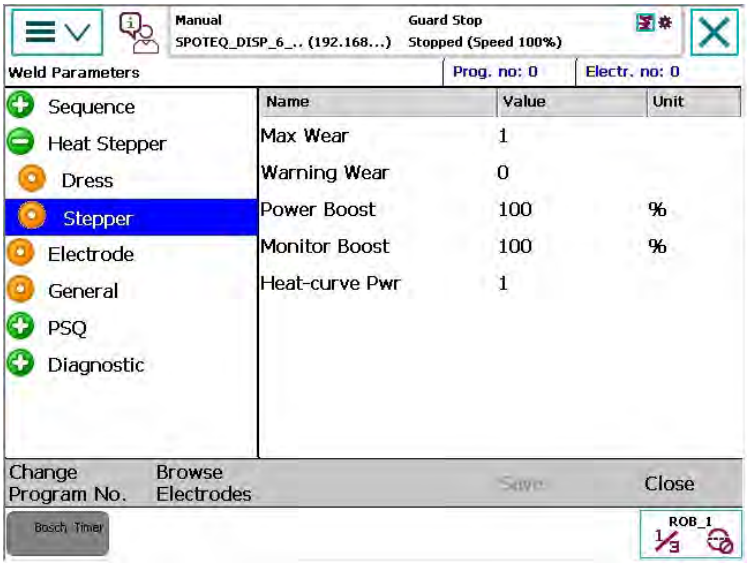
1 Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2 Tap **Browse Electrodes** to change electrode number.
3 Tap **Save** to save the changes done to the parameters and stay in current window.
4 Tap **Close** to close the window and return to the previous window.

*Continues on next page*

### Stepper node

The stepper node contains parameters related to the wear of the electrodes, if the customer use this parameters changes are done continuously.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000597

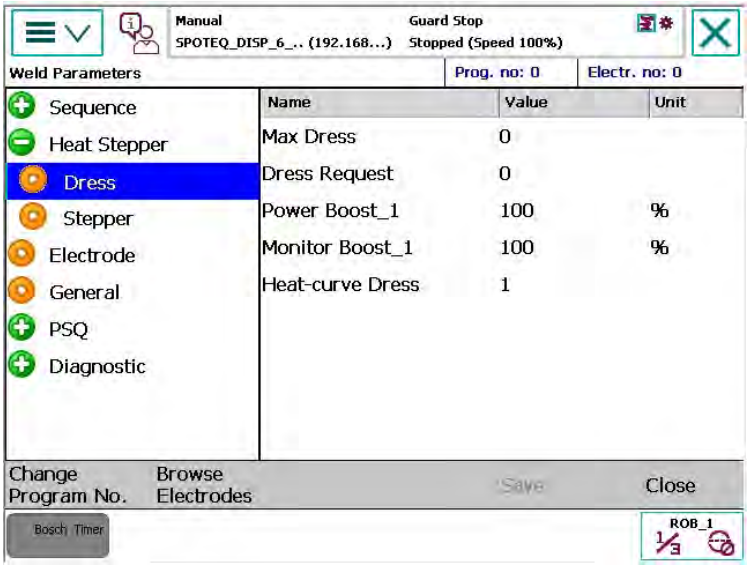
1. Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2. Tap **Browse Electrodes** to change electrode number.
3. Tap **Save** to save the changes done to the parameters and stay in current window.
4. Tap **Close** to close the window and return to the previous window.

*Continues on next page*

### Dress node

The dress node contains parameters related to the wear of the electrodes, if the customer use this parameters changes are done continuously.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000598

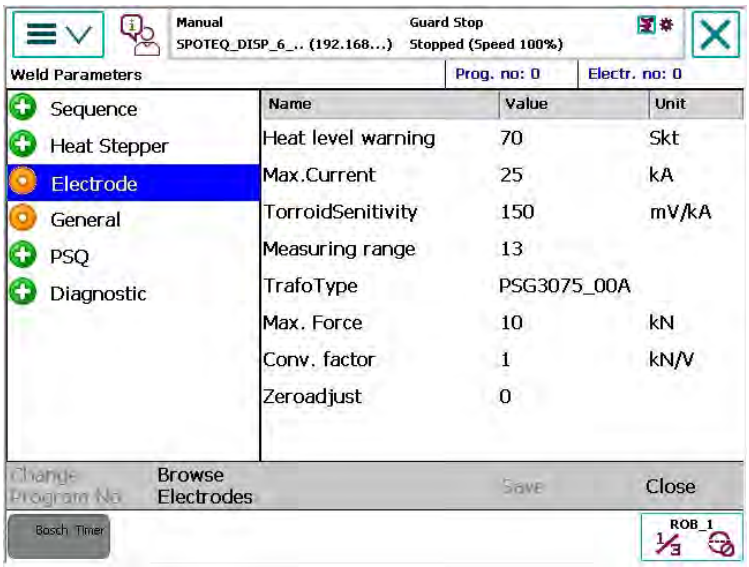
1. Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2. Tap **Browse Electrodes** to change electrode number.
3. Tap **Save** to save the changes done to the parameters and stay in current window.
4. Tap **Close** to close the window and return to the previous window.

*Continues on next page*

### Electrode node

The electrode node contains parameters related to the electrode (electrode number), changes to this parameters are done in the start phase or when a new gun is initiated.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000599

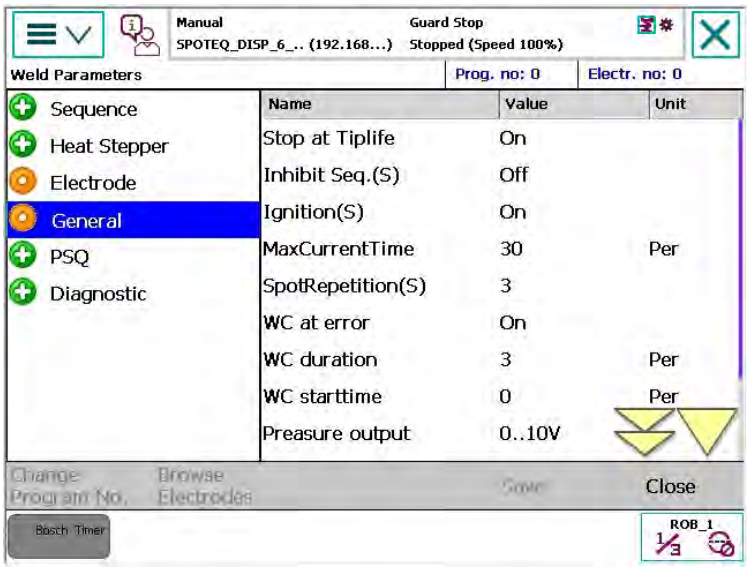
1. Tap **Change Program No.** to view weld parameters for different programs in the weld timer.
2. Tap **Browse Electrodes** to change electrode number.
3. Tap **Save** to save the changes done to the parameters and stay in current window.
4. Tap **Close** to close the window and return to the previous window.

*Continues on next page*

## General node

The general node contains parameters related to the complete weld timer. They are normally set up in the start up phase.

See *Bosch weld timer manual volume 2* for more information about the parameters.




en1200000600

1 Tap **Save** to save the changes done to the parameters and stay in current window.
2 Tap **Close** to close the window and return to the previous window.

## 8.6 Communication and configuration

### Configuration

The communication is done via an RS232 cable connected from the controller to the weld timer.

The necessary configuration is loaded when installing the system, and it is located in the SIO configuration, ABB/Configuration/Topics/Communication.

Example:

```
SIO:CFG_1.0:5:0::
#
COM_PHY_CHANNEL:
-Name "COM1" -Connector "COM1" -Baudrate 19200 -Parity "even"
#
COM_TRP:
-Name "trpbosv24_1:" -Type "BOSV24" -PhyChannel "COM1"
```

Currently it is not possible to connect more than one weld timer to the robot controller.

# 9 Customizing RobotWare-Spot

## 9.1 Introduction

**Customizing possibilities**

The Spot Options are general and can be extensively customized to fit to different spotweld equipments. The have a default "ready to use" functionality after installation, but can easily be customized by changing configuration data, RAPID data, and RAPID routines from RobotStudio for example.

One purpose of the customizing process can be to reduce the amount of data and number of variables presented to the operator.

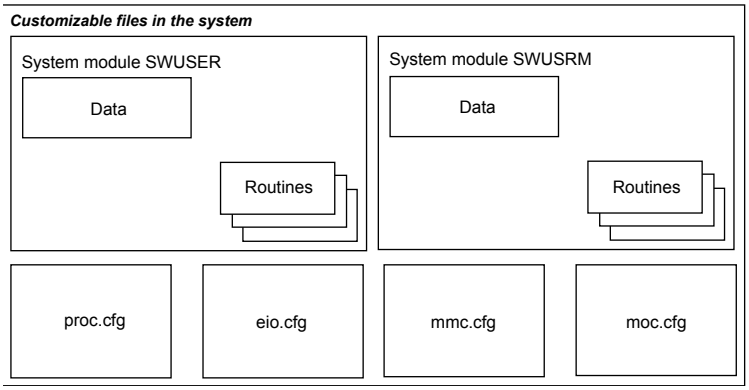
The following customizing is described in this manual:

- *How to remove not used signals from the process sequence on page 250*
- *How to remove not used process hooks on page 251*
- 
- *How to change the number of guns equipments to be used on page 252*
- *How to define max/min values for data components on page 254*
- *How to change the Spot data types on page 255*
- *How to add functionality in the process sequence on page 256*
- 
- *How to use spot data programmed in the weld timer on page 258*
- 
- *How to set the number of automatic rewelds after weld error on page 260*
- 
- 
- *How to package and install the result from the customizing on page 261*

## 9.2 Files to be changed during customizing

### Description

Customizing can be done by changing a number of predefined data and routines, preferably using a standard PC with RobotStudio. The following RAPID modules and configuration files can be changed during the customizing process:




xx1200000602

### SWUSRM

This module can be modified if the the default data types are changed. Normally there is no need to edit this module, but the possibility exists.

SWUSRM is running in **all motion tasks** and contains routines for data transfer between the user code and the kernel code, for example `DefineSpotData` and `DefineGunData`. This module can be changed from RobotStudio if needed.

See *SWUSRM on page 164*.

### SWUSER

This module can be modified if there is a need to customize the process sequence, ie add additional logic or conditions during the process, or change the content of the default types. The data and routines in this module are possible to modify from RobotStudio.

SWUSER is running in **all task** in the system and contains all the data definitions for the Spot data types and current values for the different defined Spot related data types. It also contains a number of process hook that can be modified if needed. See *SWUSER on page 158*.

### Process configuration

The process configuration is used to to setup the spot system. See *Spot process configuration on page 21*.

**Note**

Depending on the spot configuration, different default process configuration will be installed.

*Continues on next page*

**I/O configuration**

Depending on the spot configuration, a different default setup of spot weld signals will be installed, and all signals are connected to virtual I/O units. See *Spot I/O configuration on page 41* or *Spot Weld timer configuration options on page 54*.

**Tip**

If a predefined Weld Timer Configuration option is installed, only signals for one gun equipment will be defined.

**MMC configuration**

This configuration file contains for example information about which instructions are included in the different instruction pick lists, and which routines are added to the **Debug**/**Call routine** menu in the program editor, to be used as manual actions. See *Manual actions on page 74*.

**SYS configuration**

This configuration file contains for example information about which tasks that the user modules are loaded in. See *System modules on page 158*.

**MOC configuration**

This configuration file contains for example parameters for servo guns. See *Servo gun motion control on page 195*

## 9.3 Customizing guides

### 9.3.1 How to remove not used signals from the process sequence

**Description**

Use RobotStudio or the `FlexPendant` to edit the process configuration.

Example on `FlexPendant`:

Remove the `diWaterFlow2Ok` signal from the `Water flow sensor2` instance, this will disable the function of the signal. See *The type Spot Media Equipment on page 36*

1. Press ABB/Control Panel/Configuration/Process/Spot Media Equipment.
2. Replace the signal name with the predefined `NO_SIGNAL` string in the `Water flow sensor2` instance. See *The type Spot Media Equipment on page 36*
3. Save the configuration and restart the system.

The same procedure can be used on other not used signals if needed.

## 9.3.2 How to remove not used process hooks

**Description**

Use RobotStudio or the `FlexPendant` to edit the `SWUSER` module.

By default the `SWUSER` module are setup with a number of process hooks (routines), where custom code can be added if there is a need to add additional logic that is not part of the default process.

These routines can be removed if not needed to get a cleaner code.

Remove the `SwInitUserIO` user routine from the `SWUSER` module. See *Process hooks on page 160*

1. Save the module and/or apply the changes.
2. Restart the system, with **Reset RAPID**.

The same procedure can be used on other not used process hooks if needed.

**Note**

The user hook `SwPostWeld` is used to update the weld counter in the used gundata. If this hook is removed the counter will not update after weld.

**Note**

Code changes in this module requires a restart using the mode **Reset RAPID**.

## 9.3.3 How to change the number of guns equipments to be used

### Description

Use RobotStudio to edit the SWUSRM and SWUSER modules.

By default the user modules are setup for one gun equipment, or four gun equipments depending on the selected configuration (Multiple Guns etc). But it is possible to use and configure up to ten (10) gun equipments in the system if the default configuration in not sufficient.

1. Add or remove the number of instances in following spot data releated arrays in SWUSER.
   - curr_gundata
   - curr_spotdata
   - curr_forcedata

   Example:

   ```
   PERS spotdata curr_spotdata{2} := [[0,0,0,0],[0,0,0,0]];
   ```

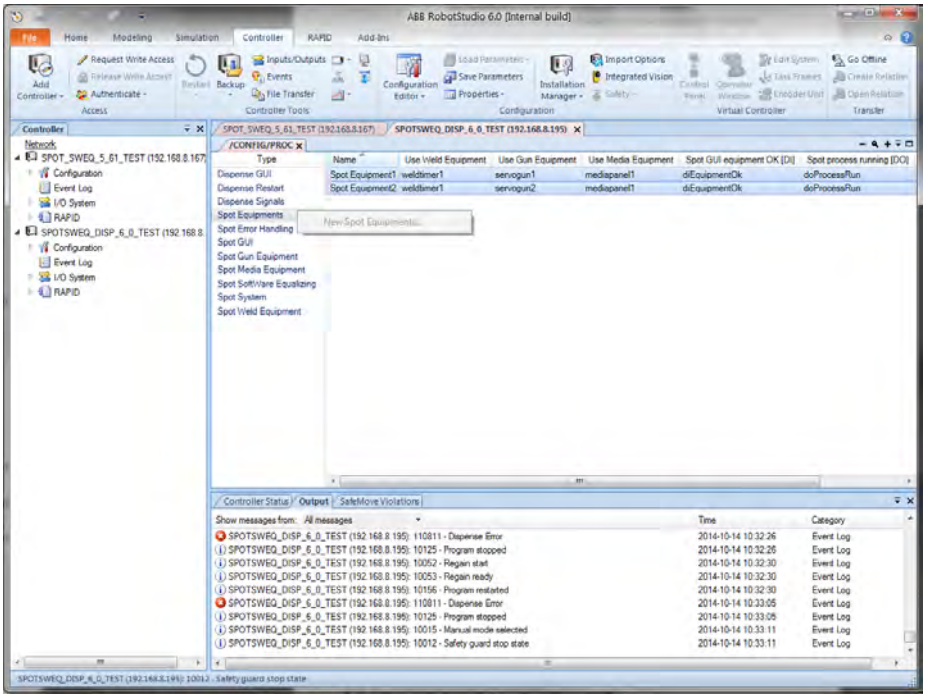
2. Add or remove the predefined gunnum gun index data in SWUSRM accordingly, ie. gun1, gun2.

   Example:

   ```
   PERS gunnum gun1 := 1;
   PERS gunnum gun2 := 2;
   ```

3. Add or remove spot equipments and the signals needed for that equipments in the process configuration. See *Spot process configuration on page 21*.

   Example:




xx1400002301

*Continues on next page*

4 Add or remove signals and I/O units in the I/O configuration for the equipments to be used if required. See *Spot I/O configuration on page 41* or *Spot Weld timer configuration options on page 54*.

**Tip**

If an additional spot weld timer option is selected when creating a new system, that is, a Bosch weld timer option (options 782-X), only one process task and configurations for one gun equipment will be installed. The installed I/O configuration can then be easily modified in RobotStudio to fit the required equipment.

**Note**

Code changes in this module requires a restart using the mode **Reset RAPID**.

## 9.3.4 How to define max/min values for data components

**Description**

Use RobotStudio to access the process configuration.

It is possible to change the max and min values for a number of data components. The limits will be tested at runtime. See *The type Spot System on page 23*.

## 9.3.5 How to change the Spot data types

### Description

Use RobotStudio to edit the SWUSER and the SWUSRM modules.

| To... | Note |
|---|---|
| Change the definition of the Spot data types in SWUSER to desired. For more information see *SWUSER on page 158*.<br>Example: add new components for second gun force in forcedata.<br>`RECORD forcedata`<br>`num tip_force;`<br>`num force_time;`<br>`num plate_thickness;`<br>`num plate_tolerance;`<br>`num tip_force2;`<br>`num force_time2;`<br>`ENDPROC` | It is possible to:<br>• Add or delete data components<br>• Move data components from for example `gundata` to `spotdata`<br>• Change the names of the data components |
| Change the structure and the default values of following arrays in SWUSER (if corresponding data type is changed).<br>Example new components in forcedata:<br>`PERS forcedata force1 :=`<br>`[1000,2,0,0,500,2];` | • curr_gundata<br>• curr_spotdata<br>• curr_simdata<br>• curr_forcedata |
| Change corresponding instructions in the data definition routines in SWUSRM if needed. These routines are used to connect the user defined data components to internal data. For more information see *SWUSRM on page 164*. | • DefineSpotData<br>• DefineGunData<br>• DefineForceData<br>• DefineSimData |

**Note**

Code changes in this module requires a restart using the mode **Reset RAPID**.

## 9.3.6 How to add functionality in the process sequence

**Description**

Use RobotStudio to edit the SWUSER module.

If the supervision during the weld process needs to be changed, add code to the process hooks.

For example:

Add an ErrWrite instruction in the error handling sequence and set a custom signal doMyAlarmSignal.

```
PROC SwPreWeld(num GunNum, INOUT string ErrText)
  VAR bool timeout;
  ! Wait for my equipment ok signal, max 2 seconds.
  WaitDI diMyEquipmentOK, 1 \MaxTime := 2 \TimeFlag := timeout;
  IF timeout THEN
    ErrText := "My equipment is not ok";
    SetDO doMyAlarmSignal, 1;
    RETURN;
  ENDIF
ENDPROC
```

1. Add or change the code in the process hooks in SWUSER. See description of the process hooks in *Process hooks on page 160*.
2. Apply changes and perform a Restart Rapid restart.

If the default autonomous supervision has to be changed, the supervision task routine in SWUSER has to be changed.

The normal way to add supervisions is to connect the supervised signal to a trap routine, e.g. (MySupTrap) and create a new supervision routine which is called from the trap routine.

For example:

```
PROC SupervisionInit()
  IDelete my_sup_init;
  CONNECT my_sup_init WITH MySupTrap;
  ISignalDO doMySupSignal, 1, my_sup_init;
ENDPROC
TRAP MySupTrap()
  TEST INTNO
  CASE my_sup_init:
    MySupervisionProc;
  ENDTEST
ENDTRAP
PROC MySupervisionProc()
  TPWrite "Executing MySupervisionProc";
  SetDO doMyAlarmSignal, 1;
ENDPROC
```

*Continues on next page*

See the predefined supervisions in *Supervision task hook on page 163*.

**Note**

Code changes in this module requires a restart using the mode **Reset RAPID**.

## 9.3.7 How to use spot data programmed in the weld timer

### Description

Some weld timers are prepared for storing data like `tip_force` and `plate_thickness` for each weld program in the timer. When the robot controller sends a new program number the timer responds with this data (for example on separate input groups). Then it is possible to use this data instead of corresponding data from the current `spotdata`.

1 Make sure that the process configuration is setup to use weld timer data instead of the default spotdata parameters, see *The type Spot Weld Equipment on page 27*.

If the optional group signals `Gun force from timer [GI]`, `Plate thickness from timer [GI]` and `Plate tolerance from timer [GI]` are used, the default data in spotdata will be disabled if the corresponding data are set to -1.

Using the optional `Gun force from timer [GI` signal group will also enable the possibility to use multiple forces during the weld cycle, that is after the weld start signal has been set and before the weld complete signal is set. The kernel will supervise this signal during the weld and change to a higher or lower value when the group value changes. See *Multiple gun forces during welding on page 86*.

**Note**

This functionality may already be prepared depending on the spot configuration.

**Note**

To activate the use of the timer input signals the corresponding parameter in spotdata must be set to -1, e.g. my_spot.tip_force := -1;

See *spotdata - Spot weld data on page 140*.

2 If needed, not used data components can be removed from `spotdata` in SWUSER. Do not forget to modify the default data declarations int the modules. See *How to change the Spot data types on page 255*.

For example, remove all parameters except the weld program parameter.

Definition of default process spot data:

```
RECORD spotdata
  num prog_no;
  num tip_force;
  num plate_thickness;
  num plate_tolerance;
ENDRECORD
```

*Continues on next page*

**Definition of customized process spot data:**

```
RECORD spotdata
  num prog_no;
ENDRECORD
```

## 9.3.8 How to set the number of automatic rewelds after weld error

**Description**

By default the automatic reweld function is deactivated.

Use RobotStudio to change the number of automatic rewelds, set the data `Number of automatic rewelds` in the system configuration to the desired value. See *The type Spot Error Handling on page 25*.

- 
- 
- 

## 9.3.9 How to package and install the result from the customizing

**Description**

After customizing the default template user modules and configuration files, it is appropriate to create a new directory with the changed files for each customized variant and load it as an RobotWare Add-In. This add-in can then be included via Installation Manager in RobotStudio.

When the Add-In is loaded into the system, the default spot user modules located in the home directory and configuration files will be replaced with the customized modules and configuration files included in the Add-In.

Fore more information about RobotWare Add-Ins, see *Application manual - RobotWare Add-Ins*.

This page is intentionally left blank

# Index

# Contact us

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

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

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

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

www.abb.com/robotics

3HAC050979-001, Rev E, en