ROBOTICS

Application manual

SpotWare

Trace back information:
Workspace 23B version a17 (not checked in)
Published 2023-06-28 at 10:23:21
Skribenta version 5.5.019

Application manual

SpotWare

RobotWare 7.10

Document ID: 3HAC078759-001

Revision: A

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

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

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

This manual and parts thereof must not be reproduced or copied without ABB's written permission.

Keep for future reference.

Additional copies of this manual may be obtained from ABB.

Original instructions.

Table of contents

Overview of this manual 9

1 Introduction to SpotWare 11

1.1 Spot option and features 11
1.2 Principles of the Spot option 14
1.3 Programming principles 15

2 Installation 17

2.1 Prerequisites 17

3 Configuration 19

3.1 Spot process configuration 19
3.1.1 The Spot System instance 21
3.1.2 The Spot Error Handling instance 25
3.1.3 The Spot Equipment instance 27
3.1.4 The Spot Weld Equipment instance 28
3.1.5 The Spot Gun Equipment instance 34
3.1.6 The Spot Media Equipment instance 39
3.1.7 The Spot SoftWare Equalizing instance 42
3.1.8 The Spot GUI instance 43
3.2 Spot I/O configuration 44
3.2.1 Spot template I/O configuration for single gun system 45
3.2.2 Spot template I/O configuration dual guns systems 49
3.2.3 Spot template I/O configuration four guns system 53

4 Programming 57

4.1 Quick start for servo gun 58
4.2 Spot weld instructions and data 59
4.3 Testing spot weld instructions in simulated mode 66
4.4 Gun control 67
4.5 Manual actions 68
4.6 Process sequence and error handling 70
4.7 Weld process timing 82
4.8 Process data access 85
4.9 Miscellaneous information 86
4.9.1 Jogging the robot after unintentional servo gun disconnection 86
4.9.2 Tip dressing for servo guns 87
4.9.3 Pneumatic spot welding gun and gripper 88
4.10 Supervision task SW_SUP 89

5 RAPID references 91

5.1 Instructions 91
5.1.1 SpotL/SpotJ - The basic spot welding instructions 91
5.1.2 SpotML/SpotMJ - Spot welding with multiple guns 96
5.1.3 SetForce - Close and Open a gun with desired force and time 100
5.1.4 CalibL/CalibJ - Calibrate a servo gun during robot movement 105
5.1.5 Calibrate - Calibrate a servo gun 111
5.1.6 MeasureWearL - Measure current electrode wear and recalculate the TCP 115
5.1.7 ReCalcTCP - Calculate current electrode wear and recalculate the TCP 123
5.1.8 IndGunMove - Activates independent mode for a servo gun 127
5.1.9 IndGunMoveReset - Resets servo gun from independent mode 129
5.1.10 OpenHighLift/CloseHighLift - Control the position of a pneumatic gun 130
5.1.11 GunArmSearch - Used to search for a surface 132
5.2 Data types 134
5.2.1 gundata - Equipment specific weld data 134

5.2.2 spotdata - Spot weld data ... 138
5.2.3 forcedata - Spot gun force data ... 142
5.2.4 simdata - Simulation data ... 145
5.2.5 searchdata - Search data ... 148
5.3 Global instructions and functions ... 150
5.3.1 SwGetCurrTargetName - Get the current robtarget name ... 150
5.3.2 SwGetCurrSpotName - Get the current spotdata name ... 151
5.3.3 SwSetIntSpotData - Set the internal spotdata ... 152
5.3.4 SwSetIntForceData - Set the internal forcedata ... 153
5.3.5 SwSetIntGunData - Set the internal gundata ... 154
5.3.6 SwSetIntSimData - Set the internal simdata ... 155
5.3.7 SwGetCalibData - Get the latest total tip wear and position adjustment ... 156
5.3.8 SwGetFixTipData - Get the latest fixed tip wear and position adjustment ... 157
5.3.9 SwGetCurrThickness - Get the latest measured thickness for a spot instruction ...158
5.3.10 SwGetCurrProcInfo - Get the latest process data for a spot instruction ... 159
5.3.11 SwDebugState - Activate and deactivate debug state ... 160
5.3.12 SGClose - Used to close a servo gun ... 161
5.3.13 SGOpen - Used to open a servo gun ... 164
5.4 System modules ... 166
5.4.1 SWUSER ... 166
5.4.2 SWUSRM ... 173

6 Software Equalizing 177

6.1 Introduction to Software Equalizing ... 177
6.2 Some basic definitions ... 179
6.3 Weld position Touch Up ... 182
6.4 Releasing of the fixed gun arm and gun opening ... 184
6.5 Gun arm deflection compensation ... 186
6.6 Tip wear compensation ... 189
6.7 Movable gun arm search mode ... 195

7 Servo gun motion control 199

7.1 Servo gun introduction ... 199
7.2 Installation and service ... 200
7.3 General motion control for servo guns ... 208
7.4 Asynchronous movements with force control ... 210
7.5 Tip management ... 212
7.6 Stationary gun ... 214
7.7 Servo tool change ... 215

8 Customizing RobotWare-Spot 219

8.1 Introduction ... 219
8.2 Files to be changed during customizing ... 220
8.3 Customizing guides ... 222
8.3.1 How to remove not used signals from the process sequence ... 222
8.3.2 How to remove not used process hooks ... 223
8.3.3 How to change the number of guns equipment to be used ... 224
8.3.4 How to define max/min values for data components ... 225
8.3.5 How to change the Spot data types ... 226
8.3.6 How to add functionality in the process sequence ... 227
8.3.7 How to use spot data programmed in the weld timer ... 229
8.3.8 How to set the number of automatic rewelds after weld error ... 231
8.3.9 How to change the user modules names and file path ... 232
8.3.10 How to package and install the result from the customizing ... 233

9 FlexPendant Interface 235

9.1 Application Overview ... 235
9.2 Main View ... 236

9.3 Manual Actions ... 240
9.4 Process Equipment ... 241
9.5 Simulation ... 242
9.6 Configuration ... 243
9.7 Customizing the UI ... 245

Index 247

This page is intentionally left blank

Overview of this manual

About this manual

This manual describes the *SpotWare* add-in product and its different option variants *Spot Standard*, *Spot Premium* and Spot *Premium Plus*.

Usage

This manual should be used during installation and configuration of the *SpotWare* product.

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:

- OmniCore 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 configuration files(*.cfg).

References

References	Document ID
Operating manual - OmniCore	*3HAC065036-001*
Operating manual - RobotStudio	*3HAC032104-001*
Product manual - OmniCore V250XT Type A	*3HAC084692-001*
Product specification - OmniCore V line	*3HAC074671-001*
Technical reference manual - System parameters	*3HAC065041-001*
Technical reference manual - RAPID Instructions, Functions and Data types	*3HAC065038-001*
Technical reference manual - RAPID Overview	*3HAC065040-001*
Application manual - Additional axes for OmniCore	*3HAC082287-001*
Application manual - Servo Gun Setup	*3HAC065014-001*
Application manual - RobotWare add-ins	*3HAC070207-001*

Continues on next page

References	Document ID
Application manual - DeviceNet Master/Slave	*3HAC066562-001*
Application manual - PROFINET Controller/Device	*3HAC066558-001*

Revisions

Revision	Description
A	Released with RobotWare 7.10.

1 Introduction to SpotWare

1.1 Spot option and features

The Spot options

Spot 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** options supporting spot welding.

- The *Spot Welding Standard* option provides support for sequential welding with **one servo gun**, robot held or stationary. Can also be used for servo guns without mechanical equalizing systems.
- The *Spot Welding Premium* option provides support for sequential welding with **one servo gun**. It is also possible to install multi process support for **up to four servo guns** at the same time. Can also be used for servo guns without mechanical equalizing systems using different software features included.
- The *Spot Welding Premium Plus* option has the same functionality as **Spot Welding Premium**, but also with the possibility to search with the movable gun arm to compensate for varying geometry and access to process related data via a log file or RAPID interface.

All Spot options 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 standard I/O interface.

The Spot options are general and can be extensively customized. They 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 from RobotStudio.

Distribution

The Spot Welding software is distributed as a separate RobotWare add-in product, SpotWare for OmniCore. This allows for independent release cycles not dependent on the RobotWare releases.

Any version of the product can be downloaded from the RobotStudio Add-Ins Gallery.

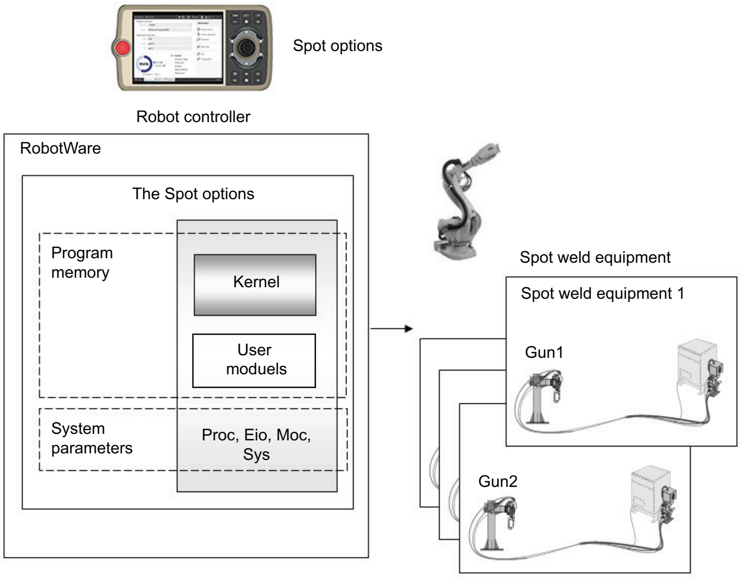
Spot Welding requires a license for one of the following options:

- [3417-1] Spot Welding Standard
- [3417-2] Spot Welding Premium
- [3417-3] Spot Welding Premium Plus

Continues on next page

Overview of the Spot options

The Spot software can be customized to handle different types of process equipment, the picture below shows schematic examples of different spot welding system variants.



xx2300000125

Features

The Spot software packages 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.
- Software equalizing functions.
- Support for mechanical gun equalizing systems.
- Support functions for tip wear management.
- Constant or changeable gun force during welding for servo guns.
- Gravity compensation of gun force during welding.
- Calibration functions for servo guns.
- Detection of missing or improper plates for servo guns.
- Reverse execution with gun control.
- Manual actions for welding and gun control.
- Support for fast tool changing between up to 8 different servo guns. Note that this requires the option *3110-1 Servo Tool Change*.
- Support for simultaneous welding with up to **four** guns at the same time. Note that this requires the option *3417-2 Spot Welding Premium* or *3417-3 Spot Welding Premium Plus*.
- Several simulation possibilities for test purposes.

Continues on next page

- Weld error recovery with automatic rewelding.
- Wide customizing possibilities, process data types, spotdata, gundata, such as weld counters and tip wear data, for each used gun.
- Built in error handling and possibility for customizable user-defined supervision and error recovery.
- A dedicated Spot operator interface on the FlexPendant.
- Possibility to run some Spot functionality without a robot selected, i.e. a stand alone controller system.
- Possibility for weld process data access. Note that option *3417-3 Spot Welding Premium Plus* is required.
- Possibility to search with the movable gun arm to compensate for varying geometry. Note that option *3417-3 Spot Welding Premium Plus* is required.

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 available for that purpose. See *RAPID references on page 91*.

- Each spot welding process is specified by:
- `spotdata`: spot weld process data. See *spotdata - Spot weld data on page 138*.
- `gundata`: spot weld equipment data. See *gundata - Equipment specific weld data on page 134*.
- The system module SWUSRM: Process data and RAPID routines for data transfer between user code and kernel code. See *SWUSRM on page 173*.
- The system module SWUSER: RAPID routines for customization of the **process behavior**, for example, checking additional external equipment etc. See *SWUSER on page 166*.
- System parameters: the I/O signal configuration and the manipulator configuration. See *Configuration on page 19* and *Installation and service on page 200*.
- See *Operating manual - OmniCore* 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 in option *3417-2 Spot Welding Premium* or *3417-3 Spot Welding Premium* Plus.
`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 in option *3417-2 Spot Welding Premium* or *3417-3 Spot Welding Premium* Plus.
`IndGunMove`	Set the servo gun in independent mode and thereafter move the gun to a specific independent position.

Continues on next page

Instruction	Used to
`IndGunMoveReset`	Reset the independent mode for servo gun.
`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.
`ReCalcTCP`	Measure the tip wear and recalculates the TCP.

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. Normally used for tip dressing.
`simdata`	Simulation modes, controller simulation, weld equipment simulation etc.

2 Installation

2.1 Prerequisites

RobotWare options

Spot Welding requires a license for one of the following options:

- *3417-1 Spot Welding Standard*
- *3417-2 Spot Welding Premium*
- *3417-3 Spot Welding Premium Plus*

The Spot product

The Spot Welding software is distributed as a separate RobotWare-Addin product, SpotWare for Omnicore. This allows for independent release cycles not dependent on the RobotWare releases.

Any version of the product can be downloaded from the RobotStudio AddIns Gallery.

During installation of the system with Installation Manager, add the downloaded add-in as a separate product in the Distributions next to RobotWare.

This page is intentionally left blank

3 Configuration

3.1 Spot process configuration

Introduction

This chapter describes the process configuration for the Spot options.

The parameters used for the Spot options are configured in the system parameters, in the **Process** domain.

Some parts of the configuration for the Spot options can be accessed from the Spot user interface, see *Configuration on page 243*

From the Home menu on the FlexPendant:

1. Tap **SpotWare.**
2. Tap **Configuration.**

The complete application and system configuration can be accessed from RobotStudio.

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

Not used equipment's can also be removed if not needed, or shared in between several equipment instances in the Spot Equipment.

Note

Configuration setup will depend on the selected default equipment template variant.

Spot process configuration

The system parameters for SpotWare are divided in the following instances.

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

Configuration instances

Configuration instances	Definitions
Spot System	Configuration of global Spot system specific parameters.
Spot Error Handling	Configuration of global Spot error handling parameters.

Continues on next page

Configuration instances	Definitions
Spot Equipment	Defines the number of used Spot welding equipment's, up to 10 different equipment's are possible.
Spot Weld Equipment	Configuration of the Spot welding equipment's, parameters, signals needed in the process.
Spot Gun Equipment	Configuration of the Spot gun equipment's, parameters, signals needed in the process.
Spot Media Equipment	Configuration of the Spot media equipment's, parameters, signals needed in the process.
Spot Equalizing	Configuration of global Software Equalizing specific parameters.
Spot GUI	Configuration of the Spot user interface.

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

3.1.1 The Spot System instance

Description

The *Spot System* contains parameters for global system settings.

Parameters

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

> **Note**
>
> Settings are dependent on actual spot configuration.

Parameter	Default value	Data type	Note
Name	spot_system	string	The name of the system.
Gun force unit type	N	string	The used gun force unit type. (Newtons [N], Decanewtons [daN] or Pounds Force [lbf]). **Note** Default value is Newton, the configured value is used as a string to present the actual gun force values shown on the FlexPendant. No automatic recalculations of any force values will be done if this parameter is changed, it is up to the user to specify correct force unit type and corresponding force values. Force calibration is required if values are changed. See *Servo gun force calibration on page 201*.
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)
No. of force calibration measurements	2	num	The number of force calibration measurements in the *ManualForceCalib* service routine. (2-10)
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)

Continues on next page

Parameter	Default value	Data type	Note
Set simulation mode [GI]		signalgi	Used to configure a group input signal that can be used to control simulation modes from an external source, for example from a PLC. Possible values are: 0 - 3. 0 = Simulation is deactivated. 1 = Weld simulation in the robot controller. 2 = Weld simulation in the weld controller. 3 = Touch-up mode. For more information about simulation modes see *Simulation modes on page 74*. If no signal name is specified here, there will be no function. **Note** The DefineSimData routine in `SWUSRM` module will be run with optional parameters if this method is used. Note that the content and values in curr_simdata structure will still be used during program execution. See DefineSimData in *Data definition routines on page 174*. **Note** An information message will be logged if any simulation mode is activated during startup, start and restart.
Simulation mode active [GO]		signalgo	Used to configure a group output signal that can be used to reflect what simulation mode that is currently selected and active. If no signal name is specified here, there will be no function.
Set inhibit gun close simulation [DI]		signaldi	Used to configure an input signal that can be used to control inhibit gun closure simulation mode. If no signal name is specified here, there will be no function. For more information about simulation modes see *Simulation modes on page 74*.
Set no plates simulation [DI]		signaldi	Used to configure an input signal that can be used to control no plates simulation mode. If no signal name is specified here, there will be no function. For more information about simulation modes see *Simulation modes on page 74*.

Continues on next page

Parameter	Default value	Data type	Note
Process data log activated	Yes	bool	At system startup a log file will be created. The log file will be updated in each spot instruction and also if any error during the process occur. It contains various information related to the process, e.g target id, spot id, gun force, process ok/not ok etc. Location: HOME:/Spot/Logs/SwProcLog.csv. If this parameter is set to No the logging to local file will be deactivated. **Note** Process data log file feature requires option *3417-3 Spot Welding Premium Plus*. **Note** The content of this log file may be changed and/or expanded in later software releases.
Process data log file size	1 MB	num	Size of the process data log in HOME:/Spot/Logs/SwProcLog.csv. Possible values are: 0.5MB: 1MB: 2MB: 4MB **Note** When the log file size exceeds the specified value a new log file will be created and the original log file will be renamed to SwProcLog_old.csv. When the **new** log file exceeds the limit the SwProcLog_old.csv will be permanently deleted and a new backup file will be created.
Use Spot Equipment1 - 10	spotequipment1	string	The name of the used spot equipment(s) used in the system. Max number of spot equipment's are 10.
Motion task user module name	SWUSRM.SYSX	string	The name of the user module running in the motion task only. **Note** If the user module names are to be changed from the default name this data needs to changed accordingly.
All task user module name	SWUSER.SYSX	string	The name of the user module running in all tasks. **Note** If the user module names are to be changed from the default name this data needs to changed accordingly.

Continues on next page

Parameter	Default value	Data type	Note
Spot user modules file path	HOME:/Spot	string	The location and file path of the spot user modules. Used when saving user data from the calibration and measurement routines, e.g CalibL, MeasureWearLetc. Default path HOME:/Spot.

3.1.2 The Spot Error Handling instance

Description

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

Parameters

The following parameters and signals are used to define the error handling in 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 before the weld error handling is activated. 0-3 are possible to configure.
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.
Show 'skip' button in auto mode	No	bool	Show 'Skip' button in automatic mode for weld error recovery, Yes/No. **Note** This parameter has no function if User defined error handling is set to Yes.
Show 'skip' button in manual mode	Yes	bool	Show 'Skip' button in manual mode for weld error recovery, Yes/No. **Note** This parameter has no function if User defined error handling is set to Yes.
Show 'ignore' button in auto mode	No	bool	Show 'Ignore' button in automatic mode for tip position error recovery mode, Yes/No. **Note** This parameter has no function if User defined error handling is set to Yes.

Continues on next page

Parameter	Default value	Data type	Note
Show 'ignore' button in manual mode	Yes	bool	Show 'Ignore' button in manual mode for tip position error recovery, Yes/No. **Note** This parameter has no function if User defined error handling is set to Yes.
Break error dialogue with 'Skip'	skip_proc	signaldi	Signal to acknowledge operator error dialogue with 'Skip' from an external source, e.g a PLC. **Note** This parameter has no function if User defined error handling is set to Yes.
Break error dialogue with 'Retry'	reweld_proc	signaldi	Signal to acknowledge operator error dialogue with 'Reweld/Retry' from an external source, e.g a PLC. **Note** This parameter has no function if User defined error handling is set to Yes.
Break error dialogue with 'Skip' or 'Retry'	ext_override	signaldi	Summary signal to break operator error dialogue, 'Skip' or 'Reweld/Retry'. **Note** This parameter has no function if User defined error handling is set to Yes.

3.1.3 The Spot Equipment instance

Description

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

Parameters

The following parameters and signals are used to define the equipment's in Spot.

Note

Settings will depend on the 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 equipment.
Use Gun Equipment	servogun1	string	Pointer to the used gun equipment.
Use Media Equipment	mediapanel1	string	Pointer to the used media equipment.
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 any process fault occurs during process. **Note** This signal will be reset if the PP is moved and current instruction is aborted.

3.1.4 The Spot Weld Equipment instance

Description

The *Spot Weld Equiment* contains parameters for the connected weld equipment's. This instance can be multiplied or shared between several equipment's.

Parameters

The following parameters and signals are used to define the weld equipment's in Spot.

Parameter	Default value	Data type	Note
Name	weldtimer1	string	The name of the weld equipment instance.
Weld process start [DO]	doStartWeld	signaldo	Used to configure the weld start signal for the weld timer. This signal will be set when gun is closed with force and the robot is in position. This signal is mandatory. **Note** The setting of this signal can be delayed via a system parameter if the servo gun needs some extra time for the force to be stabilized, Force Ready Delay, see System Parameters, Topic Motion and Type SG Process *Technical reference manual - System parameters*.
Weld process complete [DI]	diWeldComplete	signaldi	Used to configure the weld complete signal for the weld timer. Signal will be set by the weld timer when the current weld sequence is ready. This signal is mandatory.
Weld timeout	2 s	num	The max time waiting for the weld process complete signal, after this time error handling is activated. (0-10 s) **Note** If the auto reweld function is activated, the weld will be restarted automatically the specified times before the error handling is activated, see *The Spot Error Handling instance on page 25*

Continues on next page

Parameter	Default value	Data type	Note
Enable current [DO]	doEnableCurrent	signaldo	Used to configure the weld current enable signal. This signal is normally set during the process unless simulation in the weld timer or simulation in robot controller is activated. For more information about simulation modes see *Simulation modes on page 74*. If no signal name is specified here, there will be no function. ℹ **Note** Normally this signal is used in the weld timer to decide if the weld sequence should be run with or without current ("dry weld").
Timer ready to weld [DI]	diTimerReady	signaldi	Used to configure the weld timer ready signal. This signal will be set by the weld timer when it is ready to weld and no error is present. 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 error handling is activated. (0-5 s)
Stop weld process [DI]	diWeldFault	signaldi	Used to configure the stop weld process signal. This signal can be set by the weld timer if an error is detected in the weld equipment during the weld sequence. If set the current weld will be stopped and error handling will be activated. 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 used by the weld timer. This signal will be set at the beginning of a spot instruction. Allowed values: 0 - to the configured size of the I/O group. Absolute max is the size of dnum, 4294967295 - 32bit. If no signal name is specified here, there will be no function.

Continues on next page

Parameter	Default value	Data type	Note
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. If no signal name is specified here, there will be no function. **Note** Some weld timers requires a handshaking sequence regarding the weld program selection. This signal can be used to let the weld timer know that a "new" program has been selected at the weld program output group. Normally the weld timer responds with a "valid" program signal if the selected weld program is valid. If not error handling is activated.
New program selection delay	0 s	num	Used to configure a delay before the new weld program selection signal is set after the weld program group is set. (0-0.5 s) Default value 0 s. **Note** A delay can be added here if timing issues are experienced with the connected weld timer regarding the handshaking sequence, e.g. if the new program selection signal is set too close to the weld program group. **Note** Adding a delay here will affect the cycle time negatively, check the device configuration settings first, production inhibit time, poll rate etc. See System Parameters, Topic I/O System and Type Device *Technical reference manual - System parameters*.

Continues on next page

Parameter	Default value	Data type	Note
Weld program valid [DI]	diProgSelectValid	signaldi	Used to configure the weld program valid signal. This signal will be set by the weld timer when a valid weld program is selected. It will be checked during the process before the weld is started. If no signal name is specified here, no check will be done. **Note** Some weld timers requires a handshaking sequence regarding the weld program selection. This signal can be used to let the robot know that the weld timer has validated the selected weld program output group, and that it is ready to weld. **Note** If automatic rewelding is activated the program valid signal will only be checked at the first try, not on the consequent retries. If the number of retries has been executed, a new check will be done when the operator selects "Reweld" and the reweld sequence is restarted from the beginning.
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 weld timer fault signal. This signal can be used to reset weld timer faults before a reweld is done. 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 weld 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.

Continues on next page

Parameter	Default value	Data type	Note
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. If no signal name is specified here, there will be no function.
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. If no signal name is specified here, there will be no function. **Note** To activate the use of the weld 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 the weld timer. (0-100) Example with 8 bit group input, 255 * 39.2 ~ 10000 N Max in weld 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. If no signal name is specified here, there will be no function. **Note** To activate the use of the weld 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 the weld timer. (0-100) Example with 8 bit group input, 255 * 0.1 = 25.5mm max thickness.

Continues on next page

Parameter	Default value	Data type	Note
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. If no signal name is specified here, there will be no function. **Note** To activate the use of the weld 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 the weld timer. (0-100) Example with 8 bit group input, 255 * 0.1 = 25.5 max tolerance.
Timer status [GI]	giTimerStatus	signalgi	Used to configure the weld timer status group signal. This signal is used to check the timer status when a weld fault has occurred. If no signal name is specified here, there will be no function. The timer status will be available as an optional argument if this signal is configured in the SwPreWeld and SwWeldFault user hooks, see example code in these routines *Process hooks on page 168*.
Reset fault with reweld [DO]		signaldo	Used to configure the reset timer fault with reweld functionality. This signal can be used to enable a reweld in KSR mode in a Bosch weld timer when performing a reweld 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. If no signal name is specified here, there will be no function.

3.1.5 The Spot Gun Equipment instance

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 equipment's in Spot.

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. If no signal name is specified here, no check will be done.
Use SoftWare equalizing	Yes	bool	Software equalizing specific data. 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 **Note** Setting this parameter to Yes will also deactivate the use of the mechanical equalizing signal in all spot instructions.
Tip change supervision value	3 mm	num	Software equalizing specific data. Tip change supervision value. Max allowed digression [mm] in positive and negative direction from stored reference values. Default value 3 mm. 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 40 mm)

Continues on next page

Parameter	Default value	Data type	Note
Tip wear supervision value	0.2 mm	num	Software equalizing specific data. Tip wear supervision value. Max allowed digression [mm] in positive and negative direction since last tip wear compensation. Default value 0.2 mm. This data is used to supervise the tip wear and is used in the `MeasureWearL`, `ReCalcTCP` and `Calibrate/CalibL/`Jinstructions. (Max 5 mm)
Tip wear ratio, fixed vs total wear	Deactivated	num	Software equalizing specific data. 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%. 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. ℹ **Note** If the measuring method is used, (`MeasureWearL`) this value has to be set to Deactivated.
Opposite z-direction	No	bool	Software equalizing specific data. Defined z-direction for the TCP, gives move direction for search and compensations movements. **Yes**/**No**. **No** = positive z-direction out from the fixed tip (Normal setting). **Yes** = positive z-direction into the fixed tip (Setting for stationary tools to achieve the same jogging behavior as with a robot held tool). This parameter also influences the direction of the gun arm deflection compensation. For more information, see *How to define the TCP on page 179*.
MeasureWearL search I/O		signaldi	Software equalizing specific data. Used to configure an input signal that can be used instead of the reference plate for the search sequence in the `MeasureWearL` instruction. If this signal is specified the search will be done against a sensor signal instead of a fixed reference surface.

Continues on next page

Parameter	Default value	Data type	Note
Tip change fault		signaldo	Software equalizing specific data. 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. If no signal name is specified here, there will be no function.
Tip wear fault		signaldo	Software equalizing specific data. 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. If no signal name is specified here, there will be no function.
Gun equalizing	doEqualize	signaldo	Used to configure the gun equalizing signal for the gun. This signal will be set a predefined time(Gun pre equalizing time) before the weld position, and it will be reset when opening the gun after weld. If no signal name is specified here, there will be no function.
Gun pre equalizing time	0 s	num	Time before gun is 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 opened	diGunOpen	signaldi	Pneumatic gun specific data. Used to configure the gun opened (work stroke position) sensor signal for a **pneumatic** gun. Signal will be checked during process. If no signal name is specified here, no check will be done during process.
Gun open timeout	2 s	num	Pneumatic gun specific data. 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	Pneumatic gun specific data. Used to configure the high lift open (max opening position) sensor signal for a **pneumatic** gun. Signal will be checked during the spot process. If no signal name is specified here, no check will be done during process.

Continues on next page

Parameter	Default value	Data type	Note
Gun close	doCloseGun	signaldo	Pneumatic gun specific data. Used to configure the gun close signal for a **pneumatic** gun. This signal will be set a predefined time(Gun pre closing time) before the weld position, and reset when the weld is completed. If no signal name is specified here, there will be no function.
Gun pre closing time	0.1 s	num	Pneumatic gun specific data. Time before gun is in weld position, when the asynchronous gun closure is started. For a **pneumatic** gun when the gun close signal is set. (0-0.5 s) 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	Pneumatic gun specific data. 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	Pneumatic gun specific data. Used to configure the open high lift (max opening position) signal for a **pneumatic** gun. Signal will be set during the spot process if the optional argument \OpenHighLift is selected in the spot instruction. If no signal name is specified here, there will be no function.
Gun close highlift	doCloseHighLift	signaldo	Pneumatic gun specific data. Used to configure the close high lift (work stroke position) signal for a **pneumatic** gun. Signal will be set during the spot process if the optional argument \CloseHighLift is selected in the spot instruction. If no signal name is specified here, there will be no function.
Gun pressure group		signalgo	Pneumatic gun specific data. 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. If no signal name is specified here, there will be no function. **Note** Normally the gun pressure is controlled from the weld timer, and in that case this signal is not used.

Continues on next page

Parameter	Default value	Data type	Note
Gun pressure OK	diPressureOk	signaldi	Pneumatic gun specific data. Used to configure the gun pressure ok signal for a **pneumatic** gun. If no signal name is specified here, no check will be done during process.
Gun pressure timeout	2 s	num	Pneumatic gun specific data. The max time waiting for gun pressure ok signal for a **pneumatic** gun, after this time the error handling is activated. (0-10 s)
Force complete	force_complete	signaldi	Pneumatic gun specific data. Used to configure a gun force complete signal that can be used in the `SetForce` instruction. If no signal name is specified here, there will be no function.

3.1.6 The Spot Media Equipment instance

Description

The *Spot Media Equipment* contains parameters for the connected media equipment's. This instance can be multiplied or shared between several equipment's or removed completely if not needed.

Parameters

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

Parameter	Default value	Data type	Note
Name	mediapanel1	string	The name of the media panel instance.
Air flow sensor	diAirOk	signaldi	Used to configure an air pressure ok signal for a media equipment. Signal will be checked during the spot process. If no signal name is specified here, no check will be done.
Water flow start	doStartWater	signaldo	Used to configure a water start signal for a media equipment. 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. If no signal name is specified here, there will be no function.
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 flow sensor1	diWaterFlow1Ok	signaldi	Used to configure a water flow ok sensor signal for water return 1 in a media equipment. Signal will be checked during the spot process and from the supervision task if configured. If no signal name is specified here, no check will be done. **Note** The flow rate is set in the media equipment.

Continues on next page

Parameter	Default value	Data type	Note
Water flow sensor2	diWaterFlow2Ok	signaldi	Used to configure a water flow ok sensor signal for water return 2 in a media equipment. Signal will be checked during the spot process and from the supervision task if configured. If no signal name is specified here, no check will be done. **Note** The flow rate is set in the media equipment.
Water sensor flow timeout	2 s	num	Used to configure a time out for the flow sensors. 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)
Media equipment OK	diWaterOk	signaldi	Used to configure a media equipment ok summary signal. This signal will be checked during the process before the weld start, and continuously from the supervision task if configured. The spot process will wait for this signal the specified time in water sensor flow timeout. If no signal name is specified here, the water flow sensor signals will be checked independently. **Note** Make sure that the corresponding cross connection is changed if only one of the water flow sensor signals are used, see *Cross-connected signals on page 48*. **Note** Continuous water supervision can not be used if this signal is undefined.
Continuous water supervision	No = FALSE	bool	Used to configure the continuous water supervision in the `SW_SUP` task. 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. **Note** Continuous water supervision will only be active if the water ok summary signal is defined.

Continues on next page

Parameter	Default value	Data type	Note
Water saving in cycle off state	No	bool	Used to configure a water saving 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.

3.1.7 The Spot SoftWare Equalizing instance

Description

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

Parameters

The following parameters are used to define global software equalizing settings in 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).
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)
	3 mm	num	

3.1.8 The Spot GUI instance

Description

The *Spot GUI* contains parameters for the Spot user interface.

Parameters

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

Parameter	Default value	Data type	Note
Name	spot_gui	string	The name of the spot gui instance.
Show simulation settings view	Yes	bool	Set to FALSE to hide simulation view in Spot UI.
Show Spot service routines	Yes	bool	Set to 'No' to hide specific service routines in Spot UI.
User defined service routines 1-4	Empty string	string	Enter the names of any required customer defined service routines.

3.2 Spot I/O configuration

Introduction

This chapter describes the different predefined template I/O configurations that are available for the Spot variants, 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 or to no device.

The signals used are configured in the system parameters.

The I/O configuration can 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 19*.

Continues on next page

3.2.1 Spot template 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 can be connected to any device type if needed, for example Profinet, EtherNetIP, and so on.

Note

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

Predefined I/O devices

There are two predefined simulated I/O devices:

- One EtherNetIP device, named *SW_TIMER*, with specific signals for a weld timer.
- One EtherNetIP device, named *SW_DEVICE*, with signals for media panel and gun.

The devices are configured on the EtherNetIP 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 66*.
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 138*. 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 138*

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. **Note** 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. **Note** Some timers has the possibility to check if a valid weld program selection has been done. Valid program = 1, Not valid = 0 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. (0 = deactivated)
doEqualize	output	Gun equalizing signal if mechanical equalizing system are used.
doCloseGun	output	Gun close signal for a pneumatic gun.

Continues on next page

Name	Type	Information
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. This signal is set depending on status of several cross-connected signals. See cross-connections section below.
doWeldPowerContact	output	Signal used to pull the weld contactor. This signal is set depending on status of several cross-connected signals See cross-connections section below.

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. **Note** The default configuration of the safe level for this signal is to keep the last value used.

Other signals for external control

Name	Type	Information
force_complete	input	Can be used to interrupt the `SetForce` instruction before the programmed force time is elapsed for a pneumatic 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.
giSimulationMode	input group	Signal group to set simulation mode, 0,1, 2 or 3. For more information about simulation modes see *Simulation modes on page 74*.
diInhibCloseSim	input	Signal to set inhibit gun close simulation.
diNoPlatesSim	input	Signal to set no plates simulation.
goSimulationMode	output group	Signal to reflect simulation mode, 0, 1, 2 or 3. For more information about simulation modes see *Simulation modes on page 74*.

Continues on next page

Cross-connected signals

Resultant	Activator(s)	Information
doStartWater	motor_on AND doEnableCurrent AND doProcessFault (Inverted)	Water start signal, if any of the activators are not set the water start signal will be reset.
doWeldPowerContact	motor_on AND doEnableCurrent AND doProcessFault (Inverted)	Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset.
diWaterOk	diWaterFlow1Ok AND diWaterFlow2Ok	Water status signal, if any of the activators are not set the water ok signal will be reset. **Note** This cross connection needs to be modified if one of the actor signals are removed from the process configuration, in order to get correct functionality. See *The Spot Media Equipment instance on page 39*.
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. See *Other signals for external control on page 47*.

3.2.2 Spot template I/O configuration dual guns systems

Introduction

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

Note

This configuration is also suitable if the system will use servo tool change.

Default configuration

The I/O configuration is prepared for one or two weld equipment's depending on multiple gun process support configuration. This configuration can be used for servo tool change or simultaneous welding with several guns at the same time, that is, stationary guns.A set of prepared user modules are also installed if this configuration is selected.

The signals can be connected to any device type if needed, for example Profinet, DeviceNet, and so on.

Note

For simultaneous welding with several guns at the same time the option **multiple guns process support** must be selected when building the system.

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 three predefined simulated I/O devices if a dual gun equipment configuration is selected:

- One EtherNetIP device, named *SW_TIMER1*, with specific signals for weld timer 1.
- One EtherNetIP device, named *SW_TIMER2*, with specific signals for weld timer 2.
- One EtherNetIP device, named *SW_DEVICE*, with signals for media panel and gun.

The devices are configured on the EtherNetIP network by default.

Continues on next page

Weld timer signals for equipment 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 66*.
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 138*. 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 138*
doNewProgram	output	This signal can be used as handshaking to let the timer know a new program has been selected in the *goWeldProgram* group. **Note** 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. **Note** Some timers has the possibility to check if a valid weld program selection has been done. Valid program = 1, Not valid = 0 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.

Continues on next page

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.
diAirOk	input	Signal indicating low air pressure in the equalize cylinder.
diWeldContact	input	Signal indicating the state of the weld contactor. (0 = deactivated)
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. This signal is set depending on status of several cross-connected signals. See cross-connections section below.
doWeldPowerContact	output	Signal used to pull the weld contactor. This signal is set depending on status of several cross-connected signals See cross-connections section below.

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. **Note** This signal will be reset if the PP is moved and current instruction is aborted.

Other signals for external control

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.

Continues on next page

Name	Type	Information
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.
giSimulationMode	input group	Signal group to set simulation mode, 0,1, 2 or 3. For more information about simulation modes see *Simulation modes on page 74*.
diInhibCloseSim	input	Signal to set gun inhib close simulation.
diNoPlatesSim	input	Signal to set no plates simulation.
goSimulationMode	output group	Signal to reflect simulation mode, 0, 1, 2 or 3. For more information about simulation modes see *Simulation modes on page 74*.

Cross-connected signals

Resultant	Activator(s)	Information
doStartWater	motor_on AND doEnableCurrent AND doProcessFault (Inverted)	Water start signal, if any of the activators are not set the water start signal will be reset.
doWeldPowerContact	motor_on AND doEnableCurrent AND doProcessFault (Inverted)	Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset.
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. See *Other signals for external control on page 47*.

3.2.3 Spot template I/O configuration four guns system

Introduction

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

Note

This configuration is also suitable if the system will use servo tool change.

Default configuration

The I/O configuration is prepared for four weld equipment's depending on multiple gun process support configuration. This configuration can be used for servo tool change or simultaneous welding with several guns at the same time, that is, stationary guns. A set of customized user modules are also installed if this configuration is selected.

The signals can be connected to any device type if needed, for example Profinet, DeviceNet, and so on.

Note

For simultaneous welding with several guns at the same time the option **multiple guns process support** must be selected when building the system.

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 simulated I/O devices if a 4 gun equipment configuration is selected:

- One EtherNetIP device, named *SW_TIMER1*, with specific signals for weld timer 1.
- One EtherNetIP device, named *SW_TIMER2*, with specific signals for weld timer 2.
- One EtherNetIP device, named *SW_TIMER3*, with specific signals for weld timer 3.
- One EtherNetIP device, named *SW_TIMER4*, with specific signals for weld timer 4.

Continues on next page

- One EtherNetIP device, named *SW_DEVICE*, with signals for media panel and gun.

The devices are configured on the EtherNetIP network by default.

Weld timer signals for equipment 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 66*.
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 138*. 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 138*
doNewProgram	output	This signal can be used as handshaking to let the timer know a new program has been selected in the *goWeldProgram* group. **Note** 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. **Note** Some timers has the possibility to check if a valid weld program selection has been done. Valid program = 1, Not valid = 0 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.

Continues on next page

Name	Type	Information
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.
diAirOk	input	Signal indicating low air pressure in the equalize cylinder.
diWeldContact	input	Signal indicating the state of the weld contactor. (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. This signal is set depending on status of several cross-connected signals. See cross-connections section below.
doWeldPowerContact	output	Signal used to pull the weld contactor. This signal is set depending on status of several cross-connected signals See cross-connections section below.

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.

Continues on next page

Name	Type	Information
doProcessFault	output	Is set when an error situation occurs and the process is interrupted. **Note** This signal will be reset if the PP is moved and current instruction is aborted.

Other signals for external control

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.
giSimulationMode	input group	Signal group to set simulation mode, 0,1, 2 or 3. For more information about simulation modes see *Simulation modes on page 74*.
diInhibCloseSim	input	Signal to set gun inhib close simulation.
diNoPlatesSim	input	Signal to set no plates simulation.
goSimulationMode	output group	Signal to reflect simulation mode, 0, 1, 2 or 3. For more information about simulation modes see *Simulation modes on page 74*.

Cross-connected signals

Resultant	Activator(s)	Information
doStartWater	motor_on AND doEnableCurrent AND doProcessFault (Inverted)	Water start signal, if any of the activators are not set the water start signal will be reset.
doWeldPowerContact	motor_on AND doEnableCurrent AND doProcessFault (Inverted)	Weld contactor signal, if any of the activators are not set the weld contactor signal will be reset.
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. See section Other signals above.

4 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 199*, and *Application manual - Servo Gun Setup*.

Continues on next page

4.1 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 200*.

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 200*.

Servo gun force calibration

To protect the gun from too high forces there is a RAPID service routine to calibrate the motor torque versus max tip force of the gun. See *Servo gun force calibration on page 201*.

Servo gun initialization

Before running any spot instructions, the gun 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 to find the contact position or **zero** position of the gun. See *Servo gun initialization calibration on page 204*.

4.2 Spot weld instructions and data

Defining spot welding data

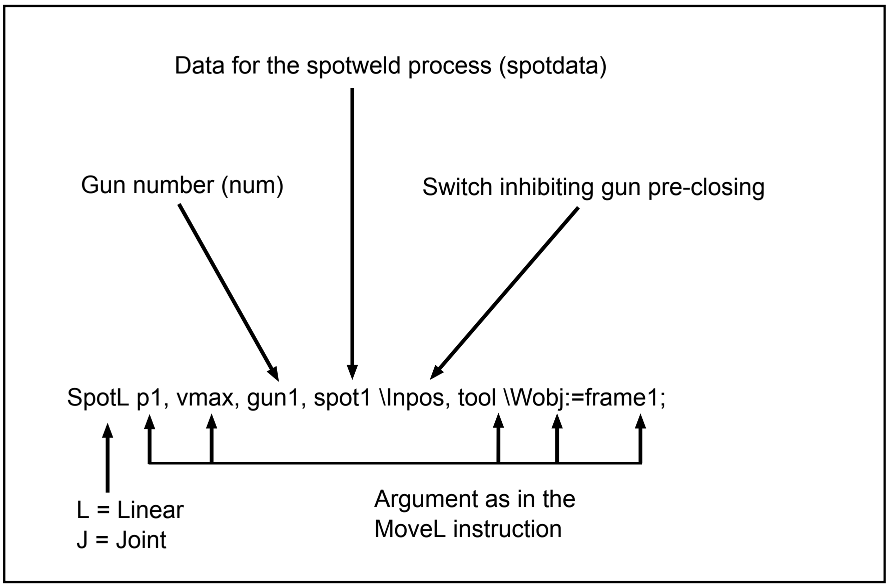
Before starting to program the instructions, define the spot welding data to be used. This data is divided into two types:

- `spotdata`; describes the spot welding process specific data for a specific spot (target). See *spotdata - Spot weld data on page 138*.
- `gundata`; describes spot welding gun specific data, used mechanical unit, weld counters, tip wear data etc. The used **spot equipment** is specified by a gun index number (gun1 or G1). This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

Spot weld instructions for sequential welding

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

For further details, see *SpotL/SpotJ - The basic spot welding instructions on page 91*.


Data for the spotweld process (spotdata)
Gun number (num)
Switch inhibiting gun pre-closing
SpotL p1, vmax, gun1, spot1 \Inpos, tool \Wobj:=frame1;
L = Linear
J = Joint
Argument as in the
MoveL instruction

en1200000250

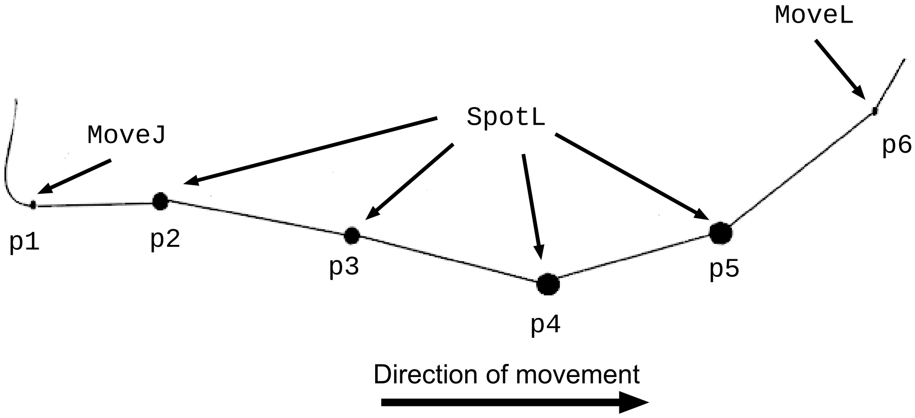
Programming example for one servo gun

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.

Continues on next page

The gun index number used in the instruction (gun1) will use the servo gun *SGUN_1* specified in the corresponding `gundata` array index in *`curr_gundata`* located in the SWUSER module, and after weld update the weld counter in the same instance.

The targets p2 and p3 will be welded with weld program number 10 and the tip force 2000N. Thickness is set to 2mm and the supervision tolerance is set to 0.5mm. If the tolerance is exceeded the execution will be stopped with an operator dialog. The next two targets 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                                 curr_gundata{1}
  prog_num = 10                          gun_name = "SGUN_1"
  tip_force = 2500                       weld_counter = 112
  plate_thickness = 2                    max_nof_welds = 1000
  plate_tolerance = 0.5                  curr_tip_wear = 5.2
  release_dist = 5                       max_tip_wear = 8
                                         curr_wear_fix = 3
                                         curr_wear_mov = 2.2

spot20
  prog_num = 20
  tip_force = 3000
  plate_thickness = 3
  plate_tolerance = 0
  release_dist = 5
```

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;
```

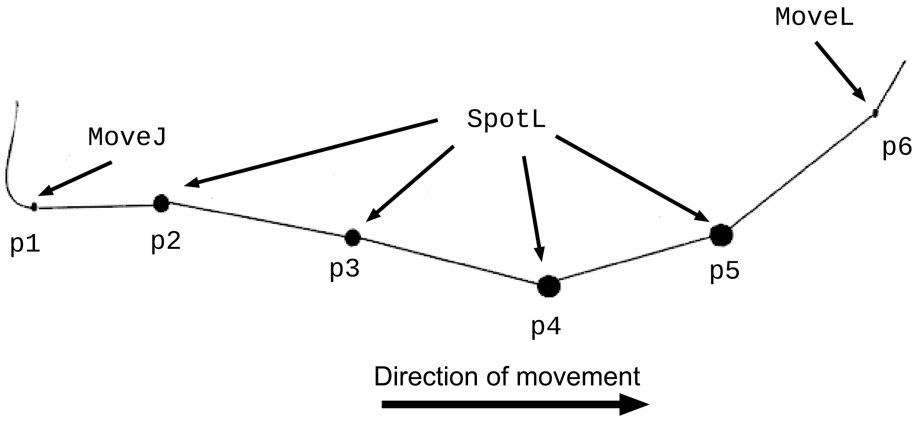
Programming example for one pneumatic gun

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

Continues on next page

The targets p2 and p3 will be welded with weld program number 10 and the tip force output group is set to 2. The next two targets 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 226*.



xx1200000245

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

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

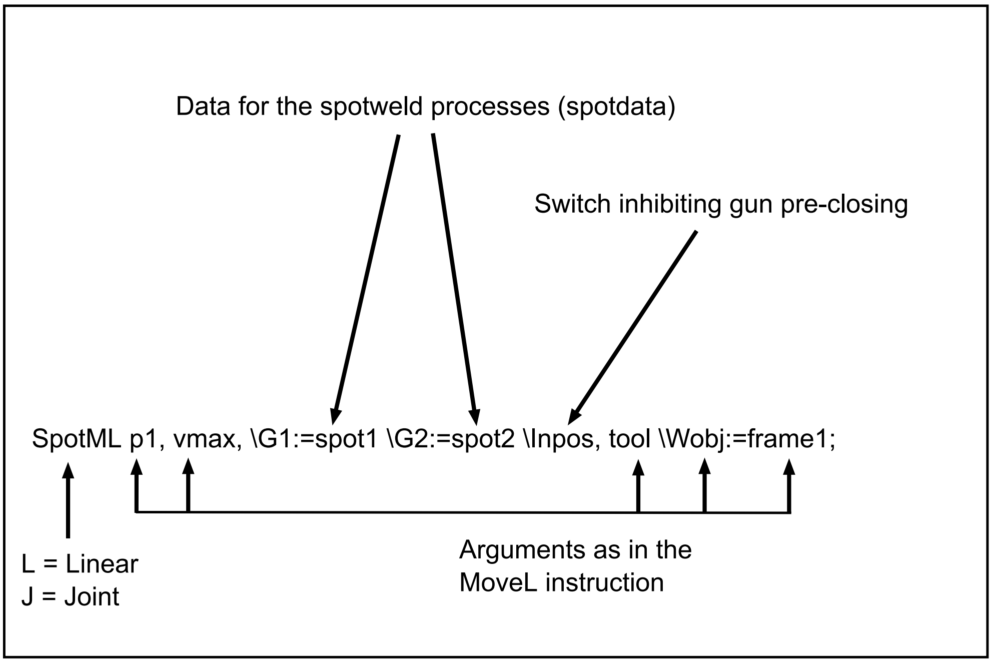
Spot weld instructions for simultaneous welding with multiple guns

`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, and also arguments that serves as data for the different spot welding processes. See *SpotML/SpotMJ - Spot welding with multiple guns on page 96*.

Note

It is not possible to use Software Equalizing mode for the `SpotML/SpotMJ` instructions.

For more information, see *Software Equalizing on page 177*.


Data for the spotweld processes (spotdata)
Switch inhibiting gun pre-closing
SpotML p1, vmax, \G1:=spot1 \G2:=spot2 \Inpos, tool \Wobj:=frame1;
L = Linear
J = Joint
Arguments as in the
MoveL instruction

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`.

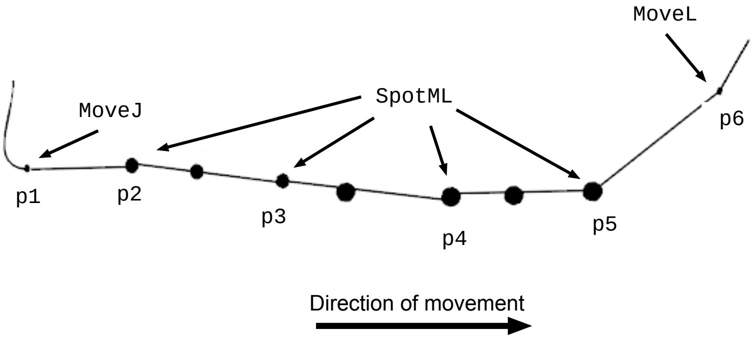
The gun index numbers used in the instruction (G1) and (G2) will use the servo guns *SGUN_1* and *SGUN_2* specified in the corresponding `gundata` array indexes in `curr_gundata` located in the SWUSER module, and after weld update the weld counters in the same instances.

The target p2 will be welded with G1(*SGUN_1*) with weld program number 10 and the tip force 2000N. Thickness is set to 2mm and the supervision tolerance is set to 0.5mm. If the tolerance is exceeded the execution will be stopped with an operator dialog.

Continues on next page

The next targets p3 and p4 will also be welded with G1(*SGUN_1*), and also at the same time with G2(*SGUN_2*) 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**.

The target p5 will be welded with just G1(*SGUN_1*) 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 = 2                 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.

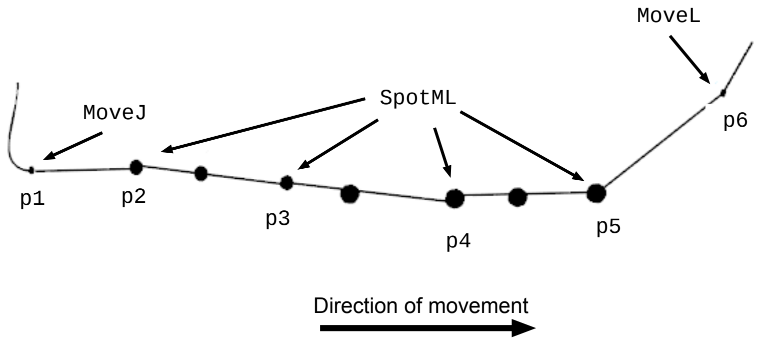
The target p2 will be welded with G1(*PNEU_G1*) with weld program number 10 and the tip force output group is set to 2.

The next targets p3 and p4 will also be welded with G1(*PNEU_G1*) but also at the same time with G2(*PNEU_G2*) with weld program number 20 and the selected tip force output group is set to 3.

The target p5 will be welded with only G2(*PNEU_G2*) also with weld program 20 and the tip force output group is set to 3.

Continues on next page

It is more common to control the the tip force from the weld timer, and in those cases the tip force parameter can be ignore or removed. See *How to change the Spot data types on page 226*.



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:

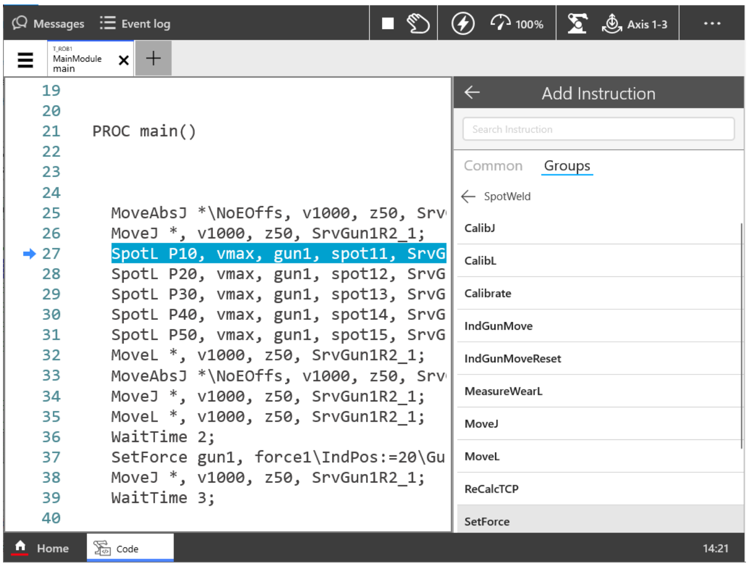
```
MoveJ p1, v600, z50, toolGrip1\Wobj:= frame1;
SpotML p2, vmax\G1:=spot10,toolGrip1\Wobj:= frame1;
SpotML p3, vmax\G1:=spot10\G2:=spot20, toolGrip1\Wobj:= frame1;
SpotML p4, vmax\G1:=spot20\G2:=spot20, toolGrip1\Wobj:= frame1;
SpotML p5, vmax\G2:=spot20, toolGrip1\Wobj:= frame1;
MoveL p6, v600, z50, 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 Editor, tap **Add instruction**, and then select the **SpotWeld** Group.
3. Select the instruction **SpotL** or **SpotJ** and **Add.**

   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



```


PROC main()



    MoveAbsJ *\NoEOffs, v1000, z50, Srv
    MoveJ *, v1000, z50, SrvGun1R2_1;
    SpotL P10, vmax, gun1, spot11, SrvG
    SpotL P20, vmax, gun1, spot12, SrvG
    SpotL P30, vmax, gun1, spot13, SrvG
    SpotL P40, vmax, gun1, spot14, SrvG
    SpotL P50, vmax, gun1, spot15, SrvG
    MoveL *, v1000, z50, SrvGun1R2_1;
    MoveAbsJ *\NoEOffs, v1000, z50, Srv
    MoveJ *, v1000, z50, SrvGun1R2_1;
    MoveL *, v1000, z50, SrvGun1R2_1;
    WaitTime 2;
    SetForce gun1, force1\IndPos:=20\Gu
    MoveJ *, v1000, z50, SrvGun1R2_1;
    WaitTime 3;

```

xx2300000083

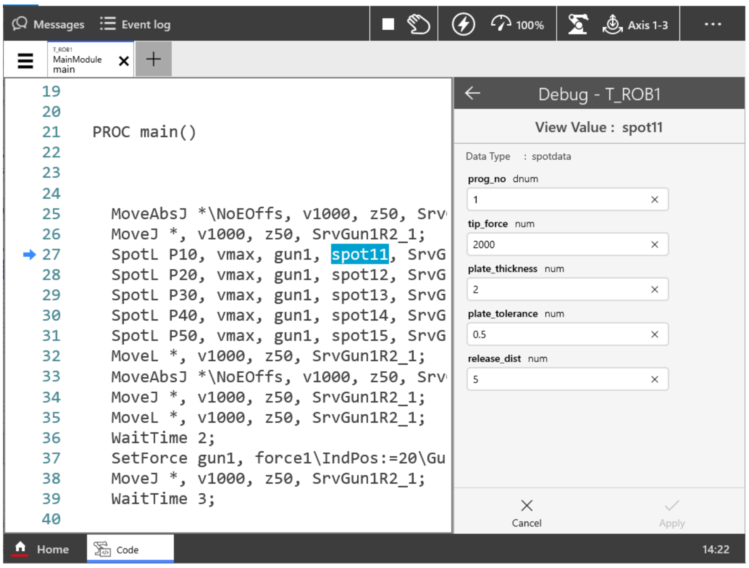
Edit current used spotdata

1 Select current `spotdata` in the instruction.

2 Tap **Debug**, and then tap **View Value**.

3 Change the value on the desired parameter.

4 Tap **Apply**.



```


PROC main()



    MoveAbsJ *\NoEOffs, v1000, z50, Srv
    MoveJ *, v1000, z50, SrvGun1R2_1;
    SpotL P10, vmax, gun1, spot11, SrvG
    SpotL P20, vmax, gun1, spot12, SrvG
    SpotL P30, vmax, gun1, spot13, SrvG
    SpotL P40, vmax, gun1, spot14, SrvG
    SpotL P50, vmax, gun1, spot15, SrvG
    MoveL *, v1000, z50, SrvGun1R2_1;
    MoveAbsJ *\NoEOffs, v1000, z50, Srv
    MoveJ *, v1000, z50, SrvGun1R2_1;
    MoveL *, v1000, z50, SrvGun1R2_1;
    WaitTime 2;
    SetForce gun1, force1\IndPos:=20\Gu
    MoveJ *, v1000, z50, SrvGun1R2_1;
    WaitTime 3;

```

View Value : spot11

Data Type : spotdata

prog_no dnum
1

tip_force num
2000

plate_thickness num
2

plate_tolerance num
0.5

release_dist num
5

xx2300000084

Changing to another spotdata

1 Select the desired `spotdata` in the instruction.

2 Tap **Modify Instruction** or double click on the **`spotdata`**.

3 Select a **`spotdata`** from the **Spot** list of available spotdata.

4 Tap **Apply**.

4.3 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 74*.

Tip

The fastest and easiest way to change the simulation mode is to use the Spot FlexPendant interface, see *Simulation on page 242*.

4.4 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 71*.

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 70*.

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.

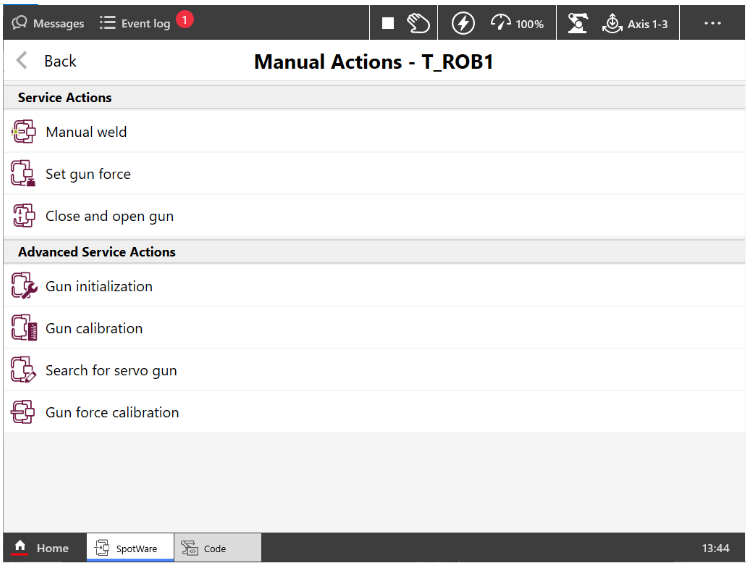
For more information, see *Software Equalizing on page 177*.

4.5 Manual actions

Service routines

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

- From the Spot UI application, select **SpotWare** and **Manual Actions**.
- From the Code Editor, tap **Debug**, and then tap **Call Service Routine**.



Manual Actions - T_ROB1

Service Actions

- Manual weld
- Set gun force
- Close and open gun

Advanced Service Actions

- Gun initialization
- Gun calibration
- Search for servo gun
- Gun force calibration

xx2300000086

Note

Some of the following routines are also possible to run without a robot selected, i.e. in a stand alone controller system.

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.
ManualGunPosition	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. • Option 1. Tool Change calibration, used after changing tool if using more than one gun. • Option 2: Tip Change calibration, used after worn tips has been replaced with new tips. • Option 3: Tip Wear calibration, used after the tip has been dressed. See *Tip management on page 212*.
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 201*. From this routine is also possible to setup the gravitational impact on the gun force. This function can be used if a servo gun loses force when the movable gun arm moves against gravity. For more information, see *Servo gun force gravity compensation on page 203* and *The Spot Gun Equipment instance on page 34*.
ManualServiceCalib	This routine will perform a gun init calibration of the servo gun, find the zero position. • Option 1: Will synchronize the servo gun without jogging after the revolution counter has been updated. The servo gun will close slowly until it reaches the contact position. • Option 2: Will synchronize the servo gun without jogging after the gun has been fine calibrated. The servo gun will move fast to a predefined pre-position. and then continue to close slowly until it reaches the contact position. See *Servo gun initialization calibration on page 204*.
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. **Note** 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, e.g unsuitable robot axis configuration. 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.

4.6 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, DefineGunData)
2. The weld program number is set (goWeldProgram).
3. The new program selection signal is set (doNewProgram).
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 (diProgSelectValid).
6. If valid program selection (diProgSelectValid), the process will read the weld controller groups if configured, (eg. giGunForce).
7. The new program selection signal is reset, if configured (doNewProgram).
8. User routine UpdateSpotData is executed.
9. User routine SwInitUserIO is executed.
10. User routine SwPrepare is executed.
11. User routine SwCloseGun is executed (for pneumatic guns).
12. The gun will start to close before the position is reached (unless argument \InPos is used), according to the predefined gun pre closing time.
13. The equalizing signal is set according to the predefined pre equalizing time. (eg. doEqualize).
14. User routine SwPreWeld is executed when the weld position is reached (Preweld supervision).
15. The plate thickness is checked. (Servo guns only). The requested gun force is established if OK.
16. The start signal to the weld controller is set (eg. doStartWeld).
17. The weld controller performs the weld, and can change the gun force during the weld sequence if configured (eg. new value on giGunForce).
18. 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.
19. User routine SwOpenGun is executed (for pneumatic guns).
20. User routine SwPostWeld is executed.
21. The instruction is ready.

Gun equalizing

For pneumatic and servo guns without *Software Equalizing* activated the signal for the mechanical gun equalizing is activated at a defined time before the weld position. 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 Spot Gun Equipment instance on page 34*.

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 177*.

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.

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 199*.

For pneumatic and servo guns without *Software Equalizing* activated 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 Spot Gun Equipment instance on page 34*.

Note

The data `Gun pre closing time` is not used if *Software Equalizing* is active, see *Software Equalizing on page 177*.

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 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.

Continues on next page

Welding

Before weld the plate thickness is checked (Servo guns only). The weld start signal is set as soon as the internal supervisions and 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 229*.

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 time out time elapsed.

Gun opening

The gun starts to open to the programmed position after the weld process is finished. When the gun is opened enough and 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 the release distance 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 used release distance.

For more information about the SoftWare Equalizing functionality, see *Software Equalizing on page 177*.

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.

Continues on next page

On restart, the robot continues with next instruction.

Halt with motors off and restart

Halt with motors off 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.

Halt with motors off 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.
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.

Continues on next page

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

The simulation modes can be set from the Spot FlexPendant interface, they can also be set from RAPID and are located in `curr_simdata` in SWUSER.

For more information, see *simdata - Simulation data on page 145*.

Note

It is also possible to control all simulation modes via an I/O interface, for RobotStudio see *The Spot System instance on page 21*, or from the Spot FlexPendant interface see *Configuration on page 243*

Note

Control simulation modes via the Spot FlexPendant interface. See *Simulation on page 242*

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 pre-weld supervision is performed, water air etc. 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, welding without current.

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 pre-weld supervision is performed, water air etc. Program valid check and reading of timer input groups will still be done in this mode, tip force, plate thickness etc

Testing without closing the guns

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

Continues on next page

This inhibits the gun closing and opening.

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

Set `sim_type = 3` to activate the weld position TouchUp function. See *Software Equalizing on page 177*

Activated by setting `sim_type = 3` in `curr_simdata` or via the simulation view in Spot FlexPendant interface.

Note

It is possible to weld the new position after it has been modified.

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
- Equipment status supervision in the beginning of the movement
- Supervision of valid program selection
- Gun closure supervision (Pneumatic guns)
- Detection of missing or improper plates (Servo guns only)
- Weld equipment supervision before weld start
- Weld error
- Supervision after welding
- Gun opening supervision (Pneumatic guns)

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.

Continues on next page

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.

Note

If automatic rewelding is configured and used the program valid will only be checked at the first try, not on the consequent retries.

If the number of retries has been executed, a new check will be done when the operator selects "**Reweld**", see *Weld error on page 78*

Supervision in the beginning of the movement

The internal default supervision checks are executed if configured, and the `SwPrepare` routine is run. See *Process hooks on page 168*.

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 Spot Media Equipment instance on page 39*

Gun closure supervision

For a pneumatic gun the internal default gun closing sequence are executed if configured, and the `SwCloseGun` routine is run. See *Process hooks on page 168*.

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 Spot Gun Equipment instance on page 34*.

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:

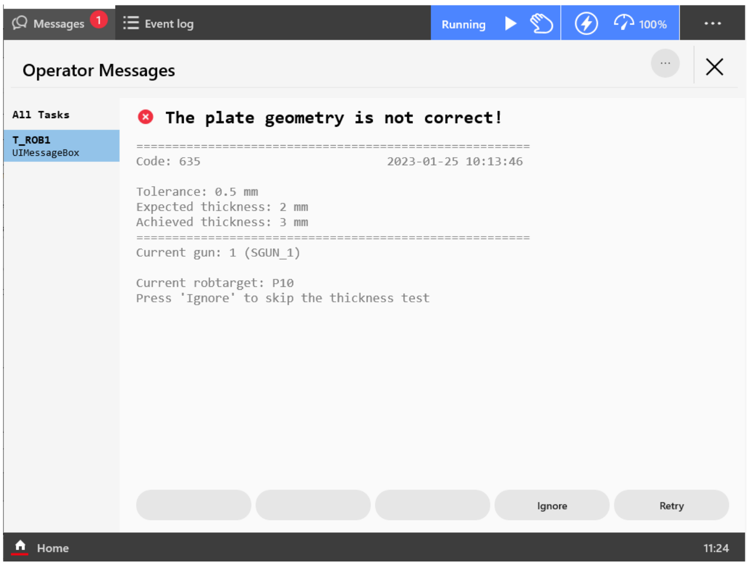
- Negative gun position, one of the tips are missing on the gun, or a tip_wear calibration is needed.
- Missing plates, the plate thickness is smaller than the thickness defined in spotdata.
- Improper geometry, the plate thickness exceeds the tolerance defined in spotdata.

The gun opens.

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

Continues on next page

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



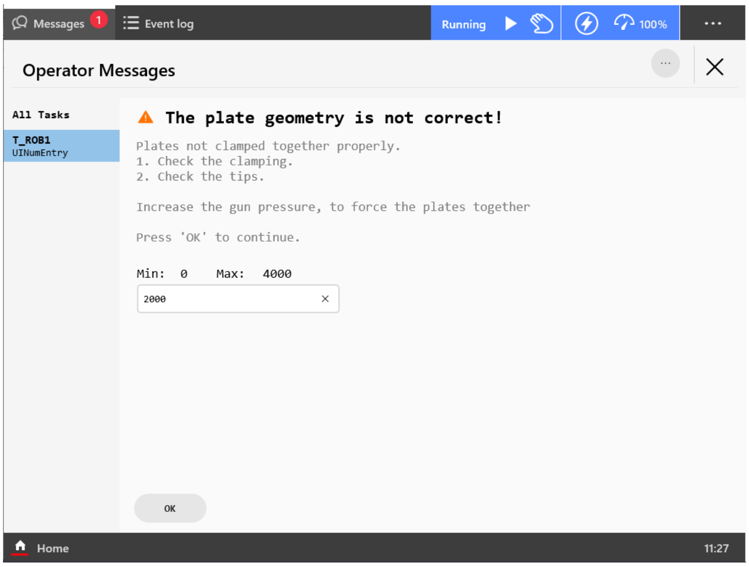
The plate geometry is not correct!

```
=======================================================
Code: 635                          2023-01-25 10:13:46

Tolerance: 0.5 mm
Expected thickness: 2 mm
Achieved thickness: 3 mm
=======================================================
Current gun: 1 (SGUN_1)

Current robtarget: P10
Press 'Ignore' to skip the thickness test
```

xx2300000072

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.



The plate geometry is not correct!

```
Plates not clamped together properly.
1. Check the clamping.
2. Check the tips.

Increase the gun pressure, to force the plates together

Press 'OK' to continue.

Min:  0    Max:  4000
2000
```

xx2300000074

Note

The accuracy of the thickness supervision is highly dependent of good gun tuning and correct mechanical data, e.g. *Transmission Gear Ratio*. It is also recommended to use a small value in *Close Position Adjust*.

Continues on next page

> **Note**
>
> If the system has been reset (**Reset system**) the calibrated zero position is not known anymore and a new gun initialization calibration needs to performed in order to find a new zero position. If this step is ignored the value **-1000** will be returned instead of the measured thickness and an error will be raised.

Supervision before the weld is started

The internal default preweld supervisions are executed if configured, and the `SwPreWeld` routine is run. See *Process hooks on page 168*.

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 Spot Weld Equipment instance on page 28*.

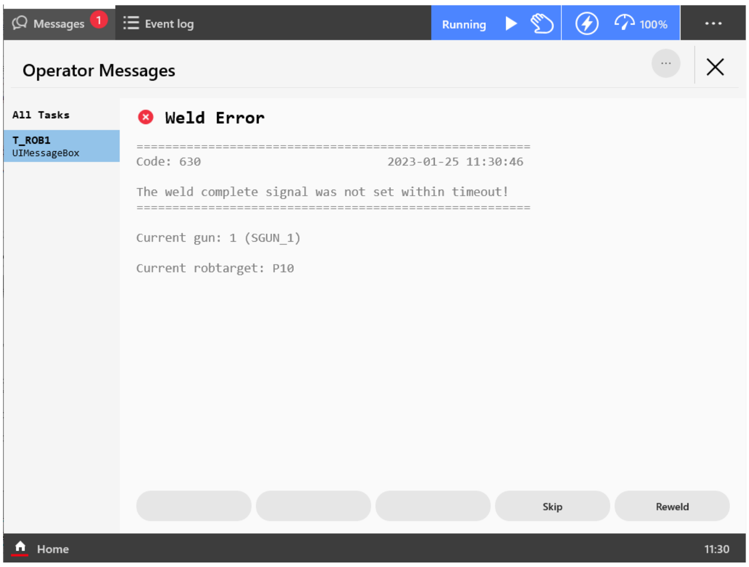
Weld error

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 Spot Weld Equipment instance on page 28*.

`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 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 and the current robtarget name is logged.

Continues on next page



```
Weld Error

========================================================
Code: 630                        2023-01-25 11:30:46

The weld complete signal was not set within timeout!
========================================================

Current gun: 1 (SGUN_1)

Current robtarget: P10
```

xx2300000073

Skip (only available in manual mode)	• The corresponding `reset fault` sgnal is pulsed. • The corresponding `process fault` signal is reset. • The current robtarget name will be stored in the log. The program execution is resumed but omitting the faulty weld.
Reweld	• The corresponding `reset fault` sgnal is pulsed. • The corresponding `process fault` signal is reset. • The gun closes. • The start signal is set after a short time delay and the program execution is resumed. **Note** 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 reweld in adaptive weld mode. See *The Spot Weld Equipment instance on page 28*.

Skip and **Reweld** error recovery can also be activated by using the digital inputs `skip_proc` and `reweld_proc`, see *Spot template I/O configuration for single gun system on page 45*.

Tip

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

Supervision after welding

For a pneumatic gun the internal default gun opening sequence are executed if configured, and the `SwOpenGun` routine is run. See *Process hooks on page 168*.

Continues on next page

An error occurs if the `gun open` signal is not set within a certain time.

- 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 Spot Gun Equipment instance on page 34*.

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 Spot Error Handling instance on page 25*.

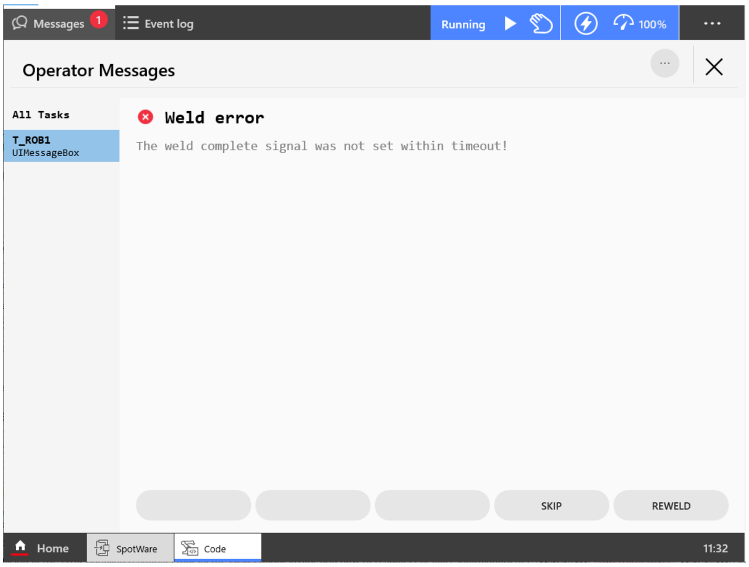
If the "user defined" error handling is activated, a dedicated routine `SwErrorRecover` in `SWUSER` will be executed if any of the error cases described in section *Error handling on page 75* occur, except for parameter errors. `SwErrorRecover` is always executed from the robot task.

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 166*.

Example

Default example if weld error occurs.



Weld error

The weld complete signal was not set within timeout!

xx2300000075

Continues on next page

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 pre close 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 177*.

Multiple gun forces during welding

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

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

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 229* and *Servo gun force calibration on page 201*.

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 201*.

Note

A fast I/O response time is critical for this functionality.

Customizing

The Spot package gives the user plenty of scope for customizing the Spot functionality, see *Customizing RobotWare-Spot on page 219*.

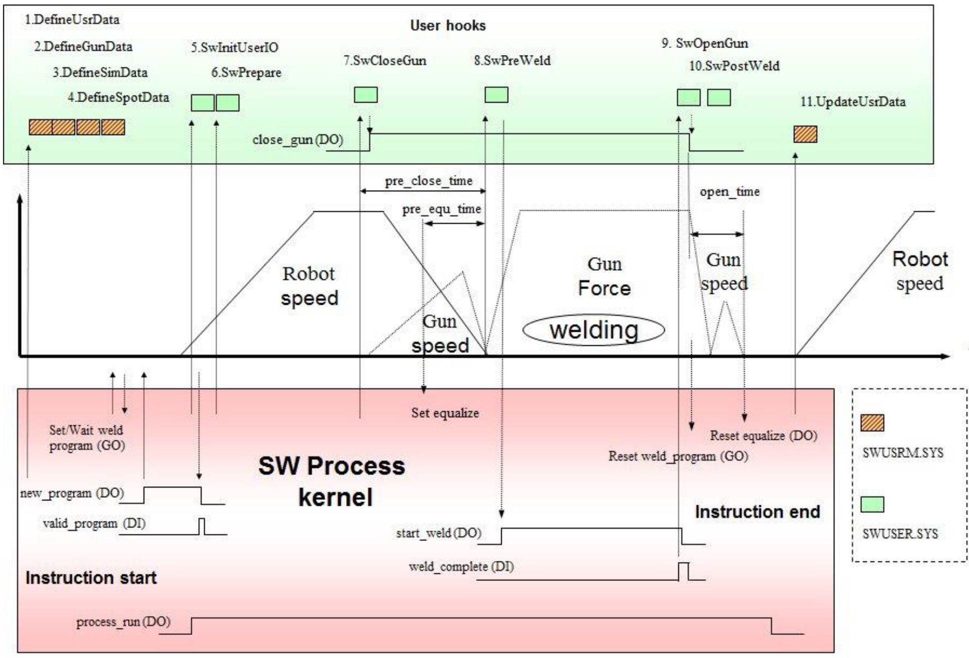
However the main subject of the `SpotL/J` and `SpotML/MJ` instructions description is the default setup.

4.7 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.

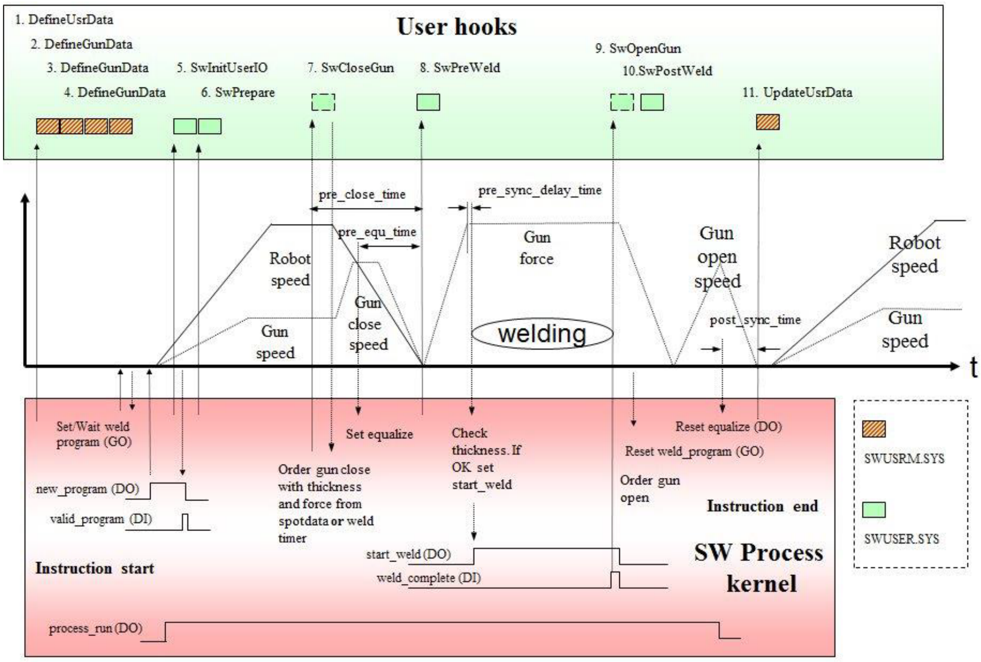
If welding is done with several guns at the same time then each process is handled in separate tasks independent of each other.

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. See System Parameters, Topic Motion and Type SG Process *Technical reference manual - System parameters*.

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 for OmniCore*.

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. See System Parameters, Topic Motion and Type SG Process *Technical reference manual - System parameters*

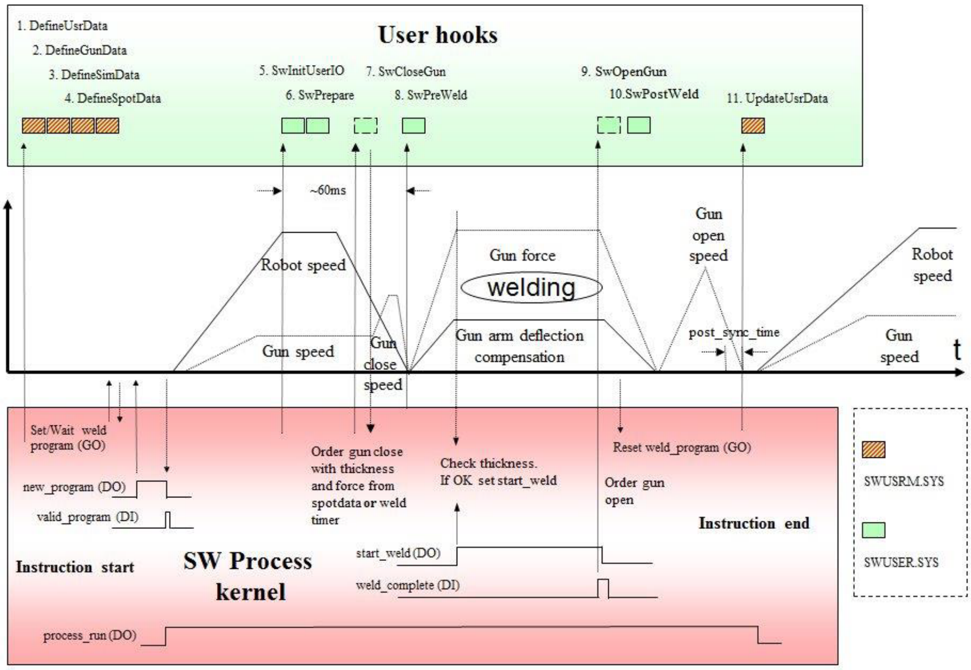
Note

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 177*.

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 for OmniCore*.

Continues on next page



en1200000237

4.8 Process data access

General

If the additional Spot option Process data access is included when creating a spot system, some data from the weld process sequence and the status of the current ongoing weld will be stored in an internal data record and a log file when running spot welding instructions. The log file contains various information related to the process, e.g. target id, spot id, gun force, process ok/not ok etc.

The log file and data record will always be updated regardless of situation, weld completed, an error situation or if the spot instruction is aborted and skipped for some reason.

Prerequisites

Process data access is only available if the RobotWare option *3417-3 Spot Welding Premium Plus* is installed.

Function overview

After executing a spot instruction it is also possible to retrieve the last weld information by using the instruction *SwGetLastProcInfo*, see *SwGetCurrProcInfo - Get the latest process data for a spot instruction on page 159*.

The size of the log file can be configured if needed, see *The Spot System instance on page 21*.

Note

The content of the log file and data record may be changed and/or expanded in later software releases.

4.9 Miscellaneous information

4.9.1 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 207*.

4.9.2 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 134* and *Tip management on page 212*.

4.9.3 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.

4.10 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 Spot Media Equipment instance on page 39*.

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 172*.

This page is intentionally left blank

5 RAPID references

5.1 Instructions

5.1.1 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.

- `SpotL` moves the TCP linearly to the weld position.
- `SpotJ` moves the TCP non-linearly to the weld position.

These instructions 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 instruction can be used 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 force.
- The parameter `gun1` is an index number corresponding to the used gun equipment. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

[i] May differ depending on configuration.

Arguments

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

Continues on next page

SpotJ ToPoint Speed GunNo [\GunD] Spot [\InPos] [\OpenHLift] [\CloseHLift] [\QuickRelease] [\Search] 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).

`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 index number. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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 138*

`[\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

Continues on next page

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.

`[|Search]`

Data type: searchdata (Search data)

If the optional data \Search is used the external axis for the gun will be used to search for the plates, and the robot TCP will be **adjusted** in the tool z-direction based on the search hit distance.

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 *Movable gun arm search mode on page 195* and *searchdata - Search data on page 148*.

Note

The search functionality is only available for spot option *3417-3 Spot Welding Premium Plus*.

Note

The search functionality has to be tuned for the used gun, see *Application manual - Servo Gun Setup*.

`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`.

Communication

`SpotL/J` instructions communicates with the surrounding weld equipment using a standard I/O interface with digital signals.

For a complete description of the I/O configuration, see *Spot I/O configuration on page 44*.

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 70*.

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 127*.

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.

Note

`SpotL/SpotJ` cannot be executed in an UNDO handler or RAPID routine connected to any of the following special system events: PowerOn, Stop, QStop, Restart, Reset or Step.

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 ]
```

Continues on next page

```
[ '\' OpenHLift ]
[ '\' CloseHLift ]
[ '\' QuickRelease ]
[ '|' Search ':='] < persistent(PERS) of searchdata > ] ','
[ 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 138*
Definition of gun data, `gundata`	*gundata - Equipment specific weld data on page 134*
SpotML/MJ	*SpotML/SpotMJ - Spot welding with multiple guns on page 96*
Overview Spot options	*Spot option and features on page 11*
Customizing possibilities	*Customizing RobotWare-Spot on page 219*
I/O configuration	*Spot I/O configuration on page 44*
Servo gun introduction	*Servo gun motion control on page 199*
Servo gun motion parameters	*Application manual - Additional axes for OmniCore*
Motion in general	*Technical reference manual - RAPID Overview*
Software Equalizing	*Software Equalizing on page 177*
Movable gun arm search	*Movable gun arm search mode on page 195*

5.1.2 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. The instructions are used to control the complete welding sequences, that is, the motion, gun closure/opening, and the welding process.

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.
- `SpotMJ` moves the TCP non-linearly to the weld position.

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 instruction can be used to implement a complete welding operation with two gun equipment's.

- 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 use gun equipment 1 and gun equipment 2. The parameter `spot10` is a data of type *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.

The parameters `G1` and `G2` serves also as index numbers corresponding to the used gun equipment's. The index numbers points at the corresponding `gundata` array indexes in *`curr_gundata`* in SWUSER and the equipment instances in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

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

Continues on next page

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

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 138* and *gundata - Equipment specific weld data on page 134*. The parameter serves also as index numbers corresponding to the used gun equipment's. The index numbers points at the corresponding `gundata` array indexes in *`curr_gundata`* in SWUSER and the equipment instances in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

`[\Gun1] - [\Gun4]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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

Continues on next page

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

`SpotML/MJ` instructions communicates with the surrounding weld equipment using a standard I/O interface with digital signals.

For a complete description of the I/O configuration, see *Spot I/O configuration on page 44*.

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 70*.

Limitations

`SpotML/SpotMJ` cannot be executed in an UNDO handler or RAPID routine connected to any of the following special system events: PowerOn, Stop, QStop, Restart, Reset or Step.

Note

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

For more information, see *Software Equalizing on page 177*.

Note

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

Continues on next page

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 > ]
  [ '\' 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 138*
Definition of gun data, gundata	*gundata - Equipment specific weld data on page 134*
SpotL/J	*SpotL/SpotJ - The basic spot welding instructions on page 91*
Overview Spot options	*Spot option and features on page 11*
Customizing possibilities	*Customizing RobotWare-Spot on page 219*
I/O configuration	*Spot I/O configuration on page 44*
Servo gun introduction	*Servo gun motion control on page 199*
Servo gun motion parameters	*Application manual - Additional axes for OmniCore*
Motion in general	*Technical reference manual - RAPID Overview*

5.1.3 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 an index number corresponding to the used gun equipment. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

Arguments

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

`GunNo`

Data type: num

Used gun number. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

`[\GunD]`

Data type: gundata

Optional parameter. Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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 142*.

`[\RetThickness]`

(returned thickness)

Data type: num

Continues on next page

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

Note

If the system has been reset (**Reset system**) the calibration position is not know anymore and a new gun init calibration needs to performed in order to find a new zero position.

`[\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).

`[\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).

`[\NoEqualize]`

Data type: switch

If the optional argument \NoEqualize is used the mechanical equalizing signal will not be set when executing the SetForce instruction.

Note

The switch only has a function if the software equalizing functionality is deactivated. See *The Spot Gun Equipment instance on page 34*.

Continues on next page

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 the used `forcedata` is established, see *forcedata - Spot gun force data on page 142*.
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 44*.

Error handing

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)

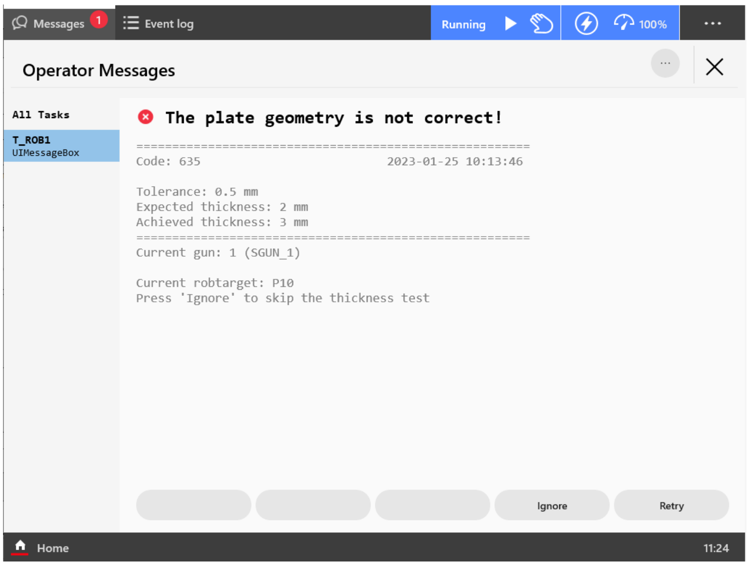
An error will be detected by the process kernel if the plate thickness differ more than the allowed limit defined by the tolerance from the programmed thickness.

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 fault` signal for the current 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.

Continues on next page



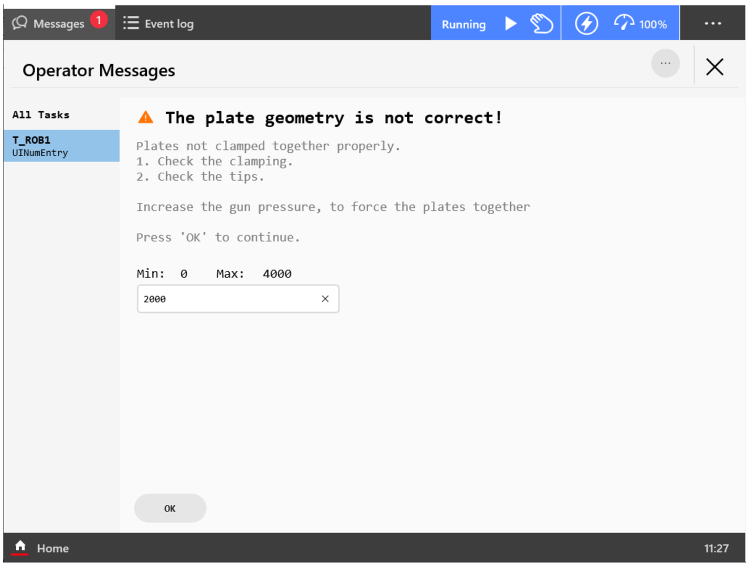
The plate geometry is not correct!

```
=======================================================
Code: 635                          2023-01-25 10:13:46

Tolerance: 0.5 mm
Expected thickness: 2 mm
Achieved thickness: 3 mm
=======================================================
Current gun: 1 (SGUN_1)

Current robtarget: P10
Press 'Ignore' to skip the thickness test
```

xx2300000072

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 instruction, that is. when the plates are not properly fixed together.



The plate geometry is not correct!

```
Plates not clamped together properly.
1. Check the clamping.
2. Check the tips.

Increase the gun pressure, to force the plates together

Press 'OK' to continue.

Min:  0     Max:  4000
2000
```

xx2300000074

Note

The accuracy of the thickness supervision is highly dependent of good gun tuning and correct mechanical data, e.g. *Transmission Gear Ratio*. It is also recommended to use a small value in *Close Position Adjust*.

Continues on next page

Note

If the system has been reset (**Reset system**) the calibrated zero position of the gun is not known anymore and a new gun initialization calibration has to performed in order to find a new zero position. The value **-1000** will be returned instead of the measured thickness and an error will be raised.

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 127* and *IndGunMoveReset - Resets servo gun from independent mode on page 129*.

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 ]
  [ '\' NoEqualize ]';'
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 ]
  [ '\' NoEqualize ]';'
```

5.1.4 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.

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

Example

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

- The gun `gun1` is calibrated for tip wear during the linear movement to `p400`.
- The parameter `gun1` is an index number corresponding to the used gun equipment. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.
- The data `curr_tip_wear` in `curr_gundata` will be automatically updated.

For more information about tip management, see *Tip management on page 212*.

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. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance

Continues on next page

in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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 Software equalizing is used 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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*, *Tip wear compensation on page 189*, *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 115* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 123*.

`[\ToolChg]`

(tool change calibration)

Data type: switch

Calibration type. This calibration type is used after tool change, see *Servo tool change on page 215*.

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 Software equalizing is used 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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*, *Tip wear compensation on page 189*, *MeasureWearL - Measure current electrode*

Continues on next page

wear and recalculate the TCP on page 115 or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 123*.

`[\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 Software equalizing is used 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. If Software equalizing is used 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

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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*.

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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*.

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 78*.

It is also possible to handle a weld error in the user defined error handling if needed, see *User defined error handling on page 80*.

Limitations

Note

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

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 > ]
```

Continues on next page

```
[ '\' 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	*Spot option and features on page 11*
Servo gun introduction	*Servo gun motion control on page 199*
Calibration without movement	*Servo gun motion control on page 199*
Software Equalizing	*Software Equalizing on page 177*
Setup data for Software Equalizing	*The Spot SoftWare Equalizing instance on page 42*

5.1.5 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.

Example

```
Calibrate gun1\ TipChange;
```

- The gun `gun1` is calibrated after tip change.
- The parameter `gun1` is an index number corresponding to the used gun equipment. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.
- 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 212*.

Arguments

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

`GunNo`

Data type: num

Used gun equipment number. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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 Software equalizing is used 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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*, *Tip wear compensation on page 189*, *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 115* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 123*.

`[\ToolChg]`

Data type: switch

Calibration type. This calibration type is used after tool change, see *Servo tool change on page 215*.

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 Software equalizing is used 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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*, *Tip wear compensation on page 189*, *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 115* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 123*.

`[\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. Different tip speeds depending on selected calibration type.

Continues on next page

- 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 Software equalizing is used 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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*.

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 Spot Gun Equipment instance on page 34*.

For more information about tip management, see *Tip management on page 212*.

Syntax

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

Continues on next page

```
| [\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*	*Spot option and features on page 11*
Servo gun introduction	*Servo gun motion control on page 199*
Calibration with movement	*Servo gun motion control on page 199*
Software Equalizing	*Software Equalizing on page 177*
Setup data for Software Equalizing	*The Spot SoftWare Equalizing instance on page 42*

5.1.6 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`.

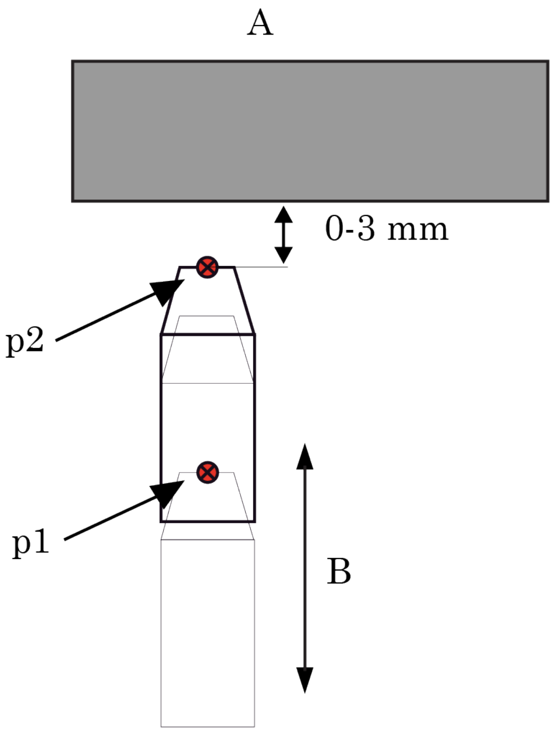
When the gun is held by the robot the gun performs a search movement during the measurement. The gun is moving in the z-direction, in the tool coordinate system, until the fixed electrode touches a fixed reference plate or a sensor of some sort, e.g. *BullsEye.*

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 can only be used in the main task T_ROB1 or, if in a MultiMove system, in Motion tasks.

Example

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



xx1200000206

A	Fix plate
B	Search direction

Continues on next page

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.

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 68*.

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.

Continues on next page

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;
```

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 166*

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.

Continues on next page

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 166*.

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 68*.

Arguments

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

`ToPoint`

Data type: robtarget

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. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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

Continues on next page

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 about tip management, see *Tip wear compensation on page 189*.

`[\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 212* and *Tip wear compensation on page 189*.

`[\TipChange]`

(tip change measurement)

Data type: switch

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 212* and *Tip wear compensation on page 189*.

`[\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 Spot Gun Equipment instance on page 34*.

`Tool`

Data type: tooldata

Continues on next page

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.

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 Spot SoftWare Equalizing instance on page 42*.
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.

Note

If `MeasureWearL` is executed in touchup mode and with the `\Reference` or `\RefChange` switch active the current position will be automatically checked if it's suitable for tip wear measurement or not.

Continues on next page

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 Spot Gun Equipment instance on page 34*.
- 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 a 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 **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 touch up position is not possible to reach for some reason (poor position, axis configuration). 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.

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 > ] ';'
```

Continues on next page

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	*Spot option and features on page 11*
System module SWUSER	*SWUSER on page 166*
Definition of gundata	*gundata - Equipment specific weld data on page 134*
Software Equalizing	*The Spot SoftWare Equalizing instance on page 42*

5.1.7 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.

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 Spot Gun Equipment instance on page 34*.

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 167*.

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 curr_tip_wear and in gundata and the Tip wear ratio, fixed vs total

Continues on next page

`wear` in the process configuration is used for the calculations. See *The Spot Gun Equipment instance on page 34*.

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. This index number points at the corresponding `gundata` array index in *`curr_gundata`* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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, e.g different tip sizes.

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

Fore more information about tip management, see *Tip wear compensation on page 189*.

`[\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.

For more information about tip management, see *Tip management on page 212* and *Tip wear compensation on page 189*.

Continues on next page

`[\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.

For more information about tip management, see *Tip management on page 212* and *Tip wear compensation on page 189*.

`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 Spot Gun Equipment instance on page 34*.

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 134*
Overview Spot options	*Spot option and features on page 11*

Continues on next page

	Described in:
System module SWUSER	*SWUSER on page 166*
Software Equalizing	*The Spot SoftWare Equalizing instance on page 42*

5.1.8 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.

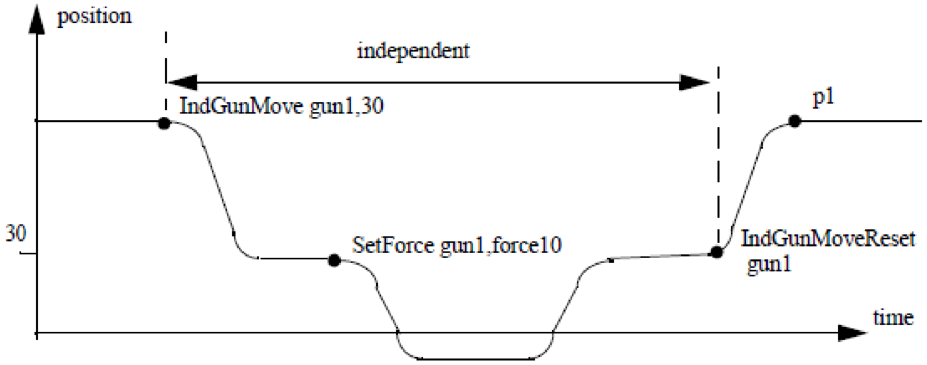
During independent mode, the control of the servo gun is separated from the robot. The gun can be closed, opened, calibrated or moved to a new independent position, but it will not follow coordinated robot movements.

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 166*

`[\GunD]`

Data type: gundata

Used gun equipment data for the process, see *gundata - Equipment specific weld data on page 134*.

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 Software 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 100*
`STIndGun`	*Technical reference manual - RAPID Instructions, Functions and Data types*

5.1.9 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 134*.

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 the reset will be delayed. 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.

Limitations

Note that the reset movement of the gun only will be finished if the coordinated speed of the tool between two points are zero or if the consecutive point is a stop point.

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 134*

5.1.10 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 134*.

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

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*	*Spot option and features on page 11*

5.1.11 GunArmSearch - Used to search for a surface

Description

GunArmSearch (Gun arm search) is used to search with the movable gun arm for a surface or an object.

Example

```
PROC my_search_routine()
VAR num ret_pos;
......
! The gun will start to move from 50mm to zero position and return
      the hit position in the variable ret_pos.
! After the search hit it will open to 50mm.
GunArmSearch gun1, ret_pos \GunOpenPos:=50;
......
TPWrite "Object is "+ValToStr(ret_pos)+"mm";
      ......
ENDPROC
```

Arguments

GunArmSearch GunNo, RetPosition [\GunOpenPos]

GunNo

Data type: num

Used gun equipment number. This index number points at the corresponding gundata array index in *curr_gundata* in SWUSER and the equipment instance in the process configuration, see *gundata - Equipment specific weld data on page 134* and *The Spot Equipment instance on page 27*.

RetPosition

Data type: num

The achieved position after search [mm].

[\GunOpenPos]

Data type: num

If the optional argument \GunOpenPos is used the gun arm will move to the specified position before and after the instruction. If the argument is omitted only the search movement will performed.

Program execution

The movable gun arm will move to the position specified in the argument \GunOpenPos if used, and then start the search movement to zero position. If an object is present between the tips the movement is stopped and the achieved position is returned in the argument RetPosition. After the search hit the movable gun arm is moved back to the position specified in the argument \GunOpenPos if used, if not, the movable gun arm is moved back a small distance specified in the system parameter 'Search reverse distance' in 'FORCE_MASTER' for the corresponding external axis.

Continues on next page

The movable gun arm of the gun is moved with the speed defined in the system parameter 'Search speed' in 'FORCE_MASTER_CONTROL' for the corresponding external axis.

For more details, see System Parameters, Topic Motion, *Technical reference manual - System parameters*.

Limitations

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

Error handling

Instruction parameter supervision

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

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

Syntax

```
GunArmSearch
  [ GunNo ':='] < expression (IN) of num > ','
  [ RetPosition ':=' < variable or persistent(INOUT) of num >
  [ '\' GunOpenPos ':=' < expression (IN) of num > ] ';'
```

Related information

	Described in:
Movable gun arm search	*Movable gun arm search mode on page 195*
Servo Gun Setup	*Application manual - Servo Gun Setup*
System Parameters, Topic Motion	*Technical reference manual - System parameters.*

5.2 Data types

5.2.1 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.
- Current tip changes and a max value.
- Specific parameters for the Software Equalizing functions.

`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 corresponding **mechanical unit** name defined in the motion 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, see *Manual actions on page 68*.

`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

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

`tip_changes`

(tip changes)

Data type: num

Number of tip changes. This data is automatically updated after each tip change calibration. Use of this data is optional. (Servo guns only).

`max_tip_changes`

(max tip changes)

Data type: num

Max number of tip changes. Use of this data is optional and can be used to supervise number of changed tips.

`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).

`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.

Default structure

```
For servo guns:
<dataobject of gundata>
  <gun_name of num>
  <weld_counter of num>
  <max_nof_welds of num>
  <tip_changes of num>
  <max_tip_changes of num>
  <curr_tip_wear of num>
  <max_tip_wear of num>
```

Continues on next page

```
      <curr_wear_fix of num>
      <wear_moveable of num>
```

For pneumatic guns:

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

Predefined data

```
PERS gundata curr_gundata{4} :=
  [["SGUN_1",0,1000,0,200,0,10,0,0],
  ["SGUN_2",0,1000,0,200,0,10,0,0],
  ["SGUN_3",0,1000,0,200,0,10,0,0],
  ["SGUN_4",0,1000,0,200,0,10,0,0]];
```

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 91*.

Note

The size of the gundata array may depend on the selected spot option, for a single gun configuration the size of the array will be only one instance as well as the process and I/O configuration.

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 219*.

Related information

	Described in:
`SpotL/SpotJ`	*SpotL/SpotJ - The basic spot welding instructions on page 91*
Overview Spot options	*Spot option and features on page 11*

Continues on next page

	Described in:
Customizing possibilities	*Customizing RobotWare-Spot on page 219*
Definition of spot data, spotdata	*Technical reference manual - RAPID Instructions, Functions and Data types*
System module SWUSER	*SWUSER on page 166*

5.2.2 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 *Customizing RobotWare-Spot on page 219*

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.

`part_id`

(part id)

Data type: string

Defines the part identity that can be used for tracing the actual part that is welded, e.g a specific door etc.

It is not used internally, but the information will be part of the process log file if the data is used.

This data is optional and not part of the default structure, so it needs to be added in the template modules by customizing the spot data type.

Max length is limited to 32 characters. Longer string will be truncated.

See *SwSetIntSpotData - Set the internal spotdata on page 152*

Continues on next page

`tip_force`

(gun tip force)

Data type: num

Defines the desired gun tip force. [Default - Newton], unit type can be changed via configuration, see *The Spot System instance on page 21*.

Permitted values: -1, to the defined max gun force, see

System Parameters, Topic Motion and Type SG Process *Technical reference manual - System parameters*.

-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 229*.

If pneumatic gun is used, this value controls a group output, see *The Spot Gun Equipment instance on page 34*.

> **Note**
>
> Normally the max force parameter is supplied by the gun manufacturer and is by default configured via system parameters, see System Parameters, Topic Motion and Type SG Process *Technical reference manual - System parameters*.

`plate_thickness`

(plate thickness)

Data type: num

Defines the expected total plate thickness. [mm].

Permitted values: -1, to the defined max plate thickness, see *The Spot System instance on page 21*.

-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 229*.

> **Note**
>
> This parameter is 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.

Permitted values: -1, to the defined max plate tolerance, see *The Spot System instance on page 21*.

-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 229*.

> **Note**
>
> This parameter is only valid for servo guns.

Continues on next page

`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.

Predefined data

Servo guns:

```
PERS spotdata spot1 := [1, 1000, 0, 0, 5];
```

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],[0,0,0,0,0]];
```

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 68*).

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 219*.

Default structure

For servo guns:

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

Continues on next page

```
        <plate_tolerance of num>
        <release_dist 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 91*
Overview Spot options	*Spot option and features on page 11*
Customizing possibilities	*Customizing RobotWare-Spot on page 219*
Definition of gun data, gundata	*gundata - Equipment specific weld data on page 134*
System module SWUSER	*SWUSER on page 166*

5.2.3 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 [Default - Newton], unit type can be changed via configuration, see *The Spot System instance on page 21*.

If pneumatic gun is used, this value controls a group output, see *The Spot Gun Equipment instance on page 34*.

`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].

Note

This parameter is 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.

Note

This parameter is only valid for servo guns.

Continues on next page

`\tip_force2`

(optional second gun tip force)

Data type: num

Defines the second gun tip force [Default - 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

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 68*.

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.

Continues on next page

However, the main subject of this description is the default setup. See *Customizing RobotWare-Spot on page 219*.

Default structure

For servo guns:

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

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 100*
Overview Spot options	*Spot option and features on page 11*
Customizing possibilities	*Customizing RobotWare-Spot on page 219*
Definition of spot data, spotdata	*spotdata - Spot weld data on page 138*
System module SWUSER	*SWUSER on page 166*

5.2.4 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.

`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 will be deactivated. (Only valid for servo guns).

Predefined data

Servo guns:

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

Continues on next page

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 219*.

Note

It is possible to control all simulation modes via the Spot FlexPendant interface, see *Simulation on page 242*.

Note

It is also possible to control all simulation modes via an I/O interface, for RobotStudio see *The Spot System instance on page 21*, or from the Spot FlexPendant interface see *Configuration on page 243*

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 91*
`SpotML`	*SpotML/SpotMJ - Spot welding with multiple guns on page 96*

Continues on next page

	Described in:
Overview *Spot*	*Spot option and features on page 11*
Customizing possibilities	*Customizing RobotWare-Spot on page 219*
System module SWUSER	*SWUSER on page 166*
System module SWUSRM	*SWUSRM on page 173*

5.2.5 searchdata - Search data

Description

Searchdata is used to activate the movable gun arm search functionality that can be used when welding a certain spot.

This optional data can be used by the SpotL/J instructions if movable gun arm search is required.

searchdata has the following structure:

- Desired search_pos_adjust plate sheet stiffness compensation for the specific position.

Components

search_pos_adjust

(search position adjust)

Data type: num

Defines the distance in mm, that can be used to adjust the searched position depending on the stiffness of the sheet metal for example if it is weaker in a specific position. The value can be positive or negative.

A positive value will move the gun and TCP position back from the search hit position.

Default value is 0 mm.

Example

Program example with and without movable gun arm search functionality: P20 and P50 is executed in standard software equalizing mode.

P30 is executed with in gun arm search mode because of tolerance variations in that specific position. Since the sheet is relatively stiff in this position there is no need to compensate for softer sheet metal.

P40 is also executed with gun arm search mode for the same reason. But because of the softer sheet metal in this specific target, and to not deform the sheet the gun will move back 2 mm to reduce the impact.

Since the search functionality only is needed in these two specific positions, it is only used there because of the additional cycle time impact.

```
PERS searchdata search1:=[0];
PERS searchdata search2:=[2];
PROC main()
  MoveJ P10, v1000, z50, tool1;
  SpotL P20, vmax, gun1, spot11, tool1;
  SpotL P30, vmax, gun1, spot12 \Search:=search1, tool1;
  SpotL P40, vmax, gun1, spot13 \Search:=search2, tool1;
  SpotL P50, vmax, gun1, spot14, tool1;
ENDPROC
```

Continues on next page

Default structure

```
<dataobject of searchdata>
  <search_pos_adjust of num>
```

Related information

	Described in:
`SpotL/J`	*SpotL/SpotJ - The basic spot welding instructions on page 91*
Movable gun arm search	*Movable gun arm search mode on page 195*
Servo Gun Setup	*Application manual - Servo Gun Setup*

5.3 Global instructions and functions

5.3.1 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.

5.3.2 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.

5.3.3 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 > ]
[ '\' ReleaseDist ':=' ] < expression (IN) of num > ]
'[ '\' PartID ':=' ] < expression (IN) of string > ]';'
```

5.3.4 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 > ]';'
```

5.3.5 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 > ] ';'
```

5.3.6 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 > ]';'
```

5.3.7 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 > ]';'
```

5.3.8 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 > ]';'
```

5.3.9 SwGetCurrThickness - Get the latest measured thickness for a spot instruction

Description

SwGetCurrThickness can be used to retrieve the current plate thickness for the specified gun after a SpotX instruction has been run..

Example

A basic example of the routine SwGetCurrThickness is illustrated below.

```
PROC MyProc()
    VAR num curr_thickness;
    SpotL p10, vmax, gun1, spot1, tool1;
    curr_thickness := SwGetCurrThickness(gun1);
ENDPROC
```

- The robot executes a SpotL instruction.
- The instruction SwGetCurrThickness is called.
- The curr_thickness variable are assigned with the measured thickness from the SpotL instruction.

Syntax

```
SwGetCurrThickness '('[ GunNum ':='] < expression (VAR) of num >
     ')'
A function with a return value of data type num.
```

5.3.10 SwGetCurrProcInfo - Get the latest process data for a spot instruction

Description

`SwGetCurrProcInfo` can be used to retrieve latest process information for the specified gun after a `SpotX` instruction has been run.

Example

A basic example of the routine `SwGetCurrProcInfo` is illustrated below.

```
PROC MyProc()
    VAR swprocinfo curr_proc_info;
    SpotL p10, vmax, gun1, spot1, tool1;
    SwGetCurrProcInfo gun1, curr_proc_info;
ENDPROC
```

- The robot executes a `SpotL` instruction.
- The instruction SwGetCurrProcInfo is called.
- The `curr_proc_info` string array variable is assigned with the latest process information from the `SpotL` instruction.

Note

The content of this data may be changed and/or expanded in later software releases.

Syntax

```
[ GunNum ':=' ] < expression (IN) of num > ]
[ ProcessInfo ':='] < expression (INOUT) of swprocinfo > ]';'
```

5.3.11 SwDebugState - Activate and deactivate debug state

Description

The SwDebugState routine can be used to activate or deactivate debug mode for the spot instructions. Log results will be stored in a log file located in the Home/Spot/Logs directory, SwDebug.log.

Note

Do not use the debug mode unless there is a specific need for it. If activated cycle time will be increased.

Example

An example of the instruction SwDebugState is illustrated below.

```
PROC main()
  SwDebugState \On;
  SpotL p100, vmax, gun1, spot10, tool1;
  SwDebugState \Off;
ENDPROC
```

1. The robot executes the SwDebugState instruction with switch \On to activate the logging.
2. The robot executes a SpotL instruction.
3. The robot executes the SwDebugState instruction with switch \Off to deactivate the logging.

Syntax

```
[ '\' On ]
[ '\' Off ] ';'
```

5.3.12 SGClose - Used to close a servo gun

Description

SGClose (Servo gun close) can be used to close the servo gun with a specific force. The gun will stay in force control mode until the instruction SGOpen is executed.

Example

```
PROC tipdress()
VAR string gun1:="SGUN_1";
......
! Note that the gun will move to current robtarget position, if
     already in independent mode.
STIndGunReset gun1;
......
! The gun will move to a position just above the cutters.
STIndgun gun1, 30;
......
SetDO doStartDresser, 1;
! The gun will close with reduced speed, 10% not to slam into the
     cutters.
SGClose gun1, 1500, 20 \GunSpeed:=10;
WaitTime 1;
! The gun will open with reduced speed, 10% to remove eventual
     chips on the electrodes.
SGOpen gun1 \GunSpeed:=10;
SetDO doStartDresser, 0;
......
STIndGunReset gun1;
ENDPROC
```

Independent mode is activated and the gun is moved to a position just above the cutters (30 mm). During independent mode the dresser is started and the instruction SGClose is executed with reduced gun speed, without interfering with robot motion. After the dress time of 1 second the instruction SGopen is executed with reduced speed and the dresser is stopped. The instruction STIndGunReset will take the gun out of independent mode and move the gun to current robtarget position.

Arguments

SGClose GunName, GunForce, Thickness [\RetThickness] [\GunPos]

GunName

Data type: string

The name of the servo gun.

GunForce

Data type: num

The desired gun force [N].

Continues on next page

`Thickness`

Data type: num

The expected contact position for the servo gun [mm].

`\RetThickness`

Data type: num

The achieved thickness [mm].

`\GunSpeed`

Data type: num

The closing speed of the servo gun in percent, 0 - 100%.

Program execution

If the servo gun exists then it is ordered to close to the expected thickness and force. The closing will start to move the gun arm to the expected contact position (thickness). The movement is stopped in this position, and a switch from position control mode to force control mode is done.

The movable gun arm of the gun is moved with max speed and acceleration as it is defined in the system parameters for corresponding external axis. If the optional argument \GunSpeed is used the speed of the gun arm can be reduced. As for other axes movements, the speed is also reduced in manual mode. When the desired tip force is achieved the instruction is ready and the achieved thickness is returned if the optional argument \RetThickness is specified.

It is possible to close the servo gun during a programmed robot movement as long as the robot movement **does not** include a movement of the movable gun arm. For more details see Servo tool motion control.

Error handling

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

Name	Cause of error
ERR_NO_SGUN	The specified servo tool name is not a configured servo gun (mecunit).
ERR_SGUN_ESTOP	Emergency stop during servo gun movement. **Note** Note that if the instruction is invoked from the main task then the program pointer will be stopped at the instruction, and the instruction will be restarted from the beginning at program restart.
ERR_SGUN_MOTOFF	The instruction is invoked from a background task and the system is in motors off state.
ERR_SGUN_NOTACT	The servo gun mechanical unit is not activated. Use instruction ActUnit to activate the servo gun.

Continues on next page

Name	Cause of error
ERR_SGUN_NOTOPEN	The servo gun is not open when SGClose is invoked.
ERR_SGUN_NOTINIT	The servo gun position is not initialized. The servo gun position must be initialized the first time the gun is installed or after a fine calibration is made. Use the service routine ManualServiceCalib. The current tip wear data will be reset in this case.
ERR_SGUN_NOTSYNC	The servo gun position are not synchronized. The servo gun tool position must be synchronized if the revolution counter has been lost and/or updated. Use the service routine ManualServiceCalib. Tip wear data will not be reset in this case.

Syntax

```
SGClose
  ['GunName ':=' ] < expression (IN) of string > ','
  ['Gunforce ':=' ] < expression (IN) of num > ','
  ['Thickness ':='] < expression (IN) of num >
  ['\' RetThickness ':=' < variable or persistent (INOUT) of num
       ] >
  ['\' GunSpeed] < expression (IN) of num > ] ';'
```

Related information

	Described in:
SGOpen	*SGOpen - Used to open a servo gun on page 164*
STIndGun	*Technical reference manual - RAPID Instructions, Functions and Data types*
STIndGunReset	*Technical reference manual - RAPID Instructions, Functions and Data types*

5.3.13 SGOpen - Used to open a servo gun

Description

`SGOpen` (Servo gun open) can be used to open a servo gun.

Example

Example 1:

```
SGOpen "SGUN_1";
Open the servo gun SGUN_1.
```

Example 2:

```
SGOpen "SGUN_1" \WaitZeroSpeed;
Stop the servo gun SGUN_1, wait until any coordinated movement has
     finished, and then open the servo gun SGUN_1.
```

Arguments

STOpen GunName [\WaitZeroSpeed] [\GunSpeed]

`GunName`

Data type: string

The name of the servo gun.

`[\WaitZeroSpeed]`

Data type: switch

Stop the servo gun, wait until any coordinated movement has finished, and then open the servo gun.

`GunSpeed`

Data type: num

The opening speed of the servo gun in percent, 0 - 100%.

Program execution

If the servo gun exists then the servo gun is ordered to open. The tip force is reduced to zero and the movable gun arm is moved back to the pre_close position.

The movable gun arm is moved with max speed and acceleration as it is defined in the system parameters for the corresponding external axis. f the optional argument \GunSpeed is used the speed of the gun arm can be reduced. As for other axes movements, the speed is also reduced in manual mode.

It is possible to open the gun during a programmed robot movement as long as the robot movement **does not** include a movement of the movable gun arm. If the gun is opened during such movement then an error *50251 Tool opening failed* will be displayed. The switch \WaitZeroSpeed can be used to reduce the risk for this error.

For more details, see Servo tool motion control.

Continues on next page

Error handling

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

Name	Cause of error
ERR_NO_SGUN	The specified servo gun name is not a configured servo gun (mecunit).
ERR_SGUN_NOTACT	The servo gun mechanical unit is not activated. Use instruction ActUnit to activate the servo gun.
ERR_SGUN_NOTINIT	The servo gun position is not initialized. The servo gun position must be initialized the first time the gun is installed or after a fine calibration is made. Use the service routine ManualServiceCalib. The current tip wear data will be reset in this case.
ERR_SGUN_NOTSYNC	The servo gun position are not synchronized. The servo gun tool position must be synchronized if the revolution counter has been lost and/or updated. Use the service routine ManualServiceCalib. Tip wear data will not be reset in this case.

Syntax

```
SGOpen
  ['GunName ':=' ] < expression (IN) of string >
  ['\' WaitZeroSpeed ]
  ['\' GunSpeed ':=' < expression (IN) of num > ] ';'
```

Related information

	Described in:
SGClose	*SGClose - Used to close a servo gun on page 161*
STIndGun	*Technical reference manual - RAPID Instructions, Functions and Data types*
STIndGunReset	*Technical reference manual - RAPID Instructions, Functions and Data types*

5.4 System modules

5.4.1 SWUSER

Description

The `SWUSER` user module is configured to run in **all tasks** in the system, and contains the default 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 the content of this module. The default functionality should be good enough in most cases. But if the default data types needs to be modified for any reason, and/or additional logic has to be added in the process sequence this module needs to be changed. See *How to change the Spot data types on page 226*.

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;
		num force_time;
		num plate_thickness;
		num plate_tolerance;

simdata	Definition of simulation data	num sim_type;
		num sim_time;
		bool inhib_close;
		bool no_plates;

Continues on next page

Record data	Description	Default value

spotdata	Definition of process spot data	dnum prog_no;
		num tip_force;
		num plate_thickness;
		num plate_tolerance;
		num release_dist;;

gundata	Definition of process gun data	string gun_name;
		num weld_counter;
		num max_nof_welds;
		num curr_tip_wear;
		num max_tip_wear;
		num curr_wear_fix;
		num curr_wear_mov;

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 190*. (This only concerns Spot servo equalizing)
tw_ref_tool{4}	PERS tooldata	Reference tooldata for each gun, used in software equalizing. See *Tip measurement sequence on page 190*. (This only concerns Spot servo equalizing)

Continues on next page

Name	Declaration	Description
reference_done{4}	PERS bool	Boolean used in routine ReCalcTCP to check if a reference measurement has been done. (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.

Note

The process hooks below is run from the internal process task(s) in the application when running spot instructions. However the SwErrorRecover hook is **only** run from the robot task, so any errors raised from the process hooks will be handled in the robot task.

Note

Any errors discovered in the process hooks should be raised and then handled in the SwErrorRecover hook.

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 Spot Media Equipment instance on page 39*.

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 Spot Gun Equipment instance on page 34*

No default functionality.

Note

The presence of this routine depends on the selected spot configuration.

Continues on next page

PROC SwPreWeld(num GunNum, INOUT string ErrText \num TimerStatus)

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 Spot Weld Equipment instance on page 28*.

Here the weld equipment status can be check before weld, template example code included.

```
PROC SwPreWeld(num GunNum, INOUT string ErrText \num TimerStatus)
  !Example code: Check timer status before weld
  IF Present(TimerStatus) AND TimerStatus <> 0 THEN
  SwWeldFault GunNum, ErrText \TimerStatus:=TimerStatus;
  ENDIF
ENDPROC
```

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 Spot Gun Equipment instance on page 34*.

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 \num TimerStatus)

This routine is called when the configured 'weld timeout' time has elapsed without receiving the weld complete signal from 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 Spot Weld Equipment instance on page 28*.

Here the weld equipment status can be check before weld, template example code included.

```
PROC SwWeldFault(num GunNum, INOUT string ErrText \num TimerStatus)
  !Example code: Check timer status
  ! Check timer error status
  IF Present(TimerStatus) THEN
  TEST TimerStatus
  CASE 1:
  ! Transformer temperature too high
  ErrText := "Transformer temperature too high";
  ! ...
  DEFAULT:
  ! Unknown fault
  ErrText := "Unknown fault";
  ENDTEST
```

Continues on next page

```
    ENDIF
  ENDPROC
```

PROC SwErrorRecover(num GunNum, string ErrType, string ErrText, 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 default built-in error recovery is used. See *The Spot Error Handling instance on page 25*.

Parameter	Description
GunNum	Current gun number

Continues on next page

Parameter	Description
ErrType	Type of error that occurred. Possible cases are: • SW_PREPARE_ERR: prepare error. Error reported in the prepare sequence or by the `SwPrepare` routine. • SW_CLOSE_GUN_ERR: close gun error. Error reported in the close gun sequence or by the `SwCloseGun` routine. • SW_PRE_WELD_ERR: preweld supervision error. Error reported in the preweld sequence or by the `SwPreWeld` routine. • SW_WELD_ERR: weld error timeout. • SW_WELD_TEST_ERR: weld test error timeout. Error if test weld error in the `CalibL` or `CalibJ` instructions. • SW_OPEN_GUN_ERR: open gun error. Error reported in the open gun sequence or by the `SwOpenGun` routine. • SW_POST_WELD_ERR: postweld supervision error. Error reported in the post weld sequence or by the `SwPostWeld` routine. • SW_TIP_POS_ERR: tip position error. Only for servo guns. • SW_PROG_VALID_ERR: program valid timeout. • 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 Spot Media Equipment instance on page 39*. • SW_MEAS_TIP_CHANGE_ERR: Tip change supervision error in the `MeasureWearL` instruction. • SW_MEAS_TIP_WEAR_ERR: Tip wear supervision error in the `MeasureWearL` instruction. • SW_RECAL_TIP_WEAR_ERR: Tip wear supervision error in the `ReCalcTCP` instruction. • SW_CALIB_TIP_CHANGE_ERR Tip change supervision error in the `CalibX` instructions. • SW_CALIB_TIP_WEAR_ERR Tip wear supervision error in the `CalibX` instruction.

Continues on next page

Parameter	Description
	Note 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.

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: • `SwPrepare` • `SwCloseGun` • `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 219*

5.4.2 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. **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 229*.	prog_no - 1 tip_force - 1000 N plate_thickness 0 mm plate_tolerance - 0 mm release_dist - 5 mm

Continues on next page

Name	Description	Default value
force1	Default forcedata when programming the SetForce instruction on the FlexPendant	tip_force - 1000 N force_time 1 s plate_thickness 0 mm 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 91*.

```
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 91*

```
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

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(\num IOCtrlSimType \bool IOCtrlInhibClose \bool IOCtrlNoPlates)

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*.

The optional parameters \IOCtrlSimType, \IOCtrlInhibClose and \IOCtrlNoPlates are used when the simulation modes are controlled via optional I/O signals defined in the configuration, see *The Spot System instance on page 21*. This way the simulation modes can be controlled from a PLC etc.

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 halted with motors off (E-stop).

There is no default functionality.

6 Software Equalizing

6.1 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.

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.
- Movable gun arm search.

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 134*.

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.

Limitations

- The functions Gun Arm Release and Deflection Compensation are available for the `SpotL`/`SpotJ` instructions are used.
- 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.

Continues on next page

- 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.

Note

Software Equalizing functions does not work for the `SpotML` and `SpotMJ` instructions.

6.2 Some basic definitions

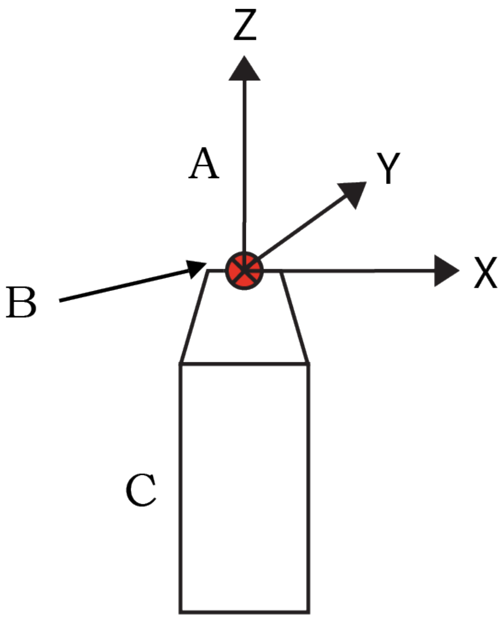
How to define the TCP

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 Spot Gun Equipment instance on page 34*.

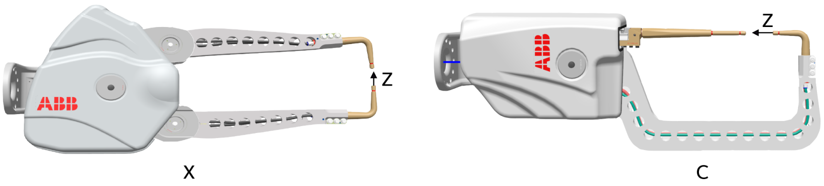


en1200000236

A	Directions in the tool coordinate system
B	TCP
C	Fixed electrode

Continues on next page

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.

How to set up the tool TCP

1 Define the TCP for the used gun with a new tip mounted. Use the 5 point method.

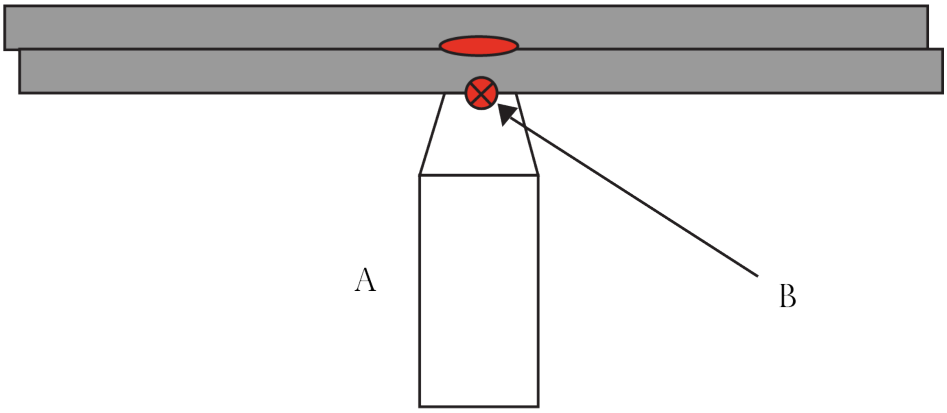
See *Operating manual - OmniCore*.

2 Store the result in a `tooldata` for example `ref_tool1`.

3 Save the tip (tool) as a reference tip (the physical tip).

Continues on next page

How to program the weld positions



en1200000234

A	Fixed tip
B	Programmed weld position

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

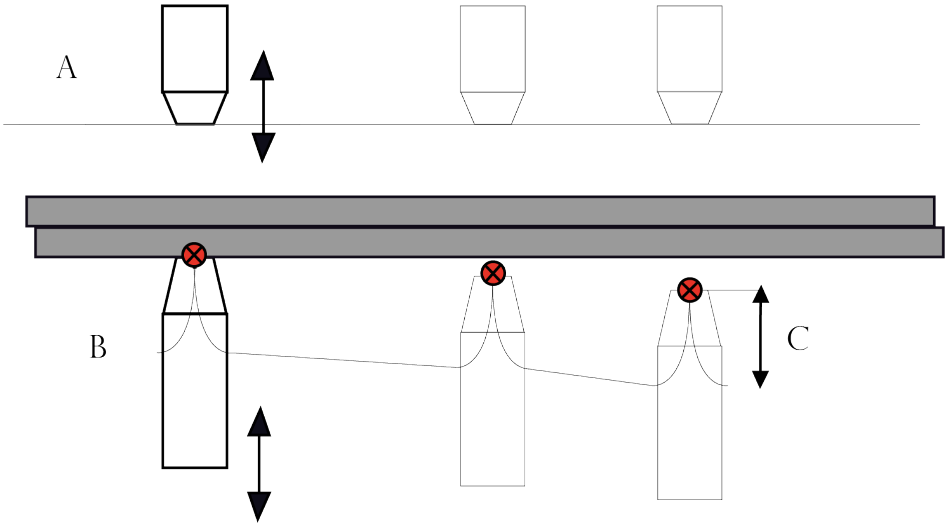
Before touching up the weld positions a `MeasureWearL` with the `Reference` switch has to be done. See *Tip measurement sequence on page 190*.

6.3 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

A	Movable tip
B	Fixed tip
C	Release distance

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' in the simulation view in Spot UI, or via the I/O signal interface. See *simdata - Simulation data on page 145*.
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 or weld 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.

Continues on next page

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 spot, `release_dist`. See *spotdata - Spot weld data on page 138*.

   This release movement can be skipped to save cycle time if the optional switch `\QuickRelease` is selected in the `SpotL` or `SpotJ` instruction.

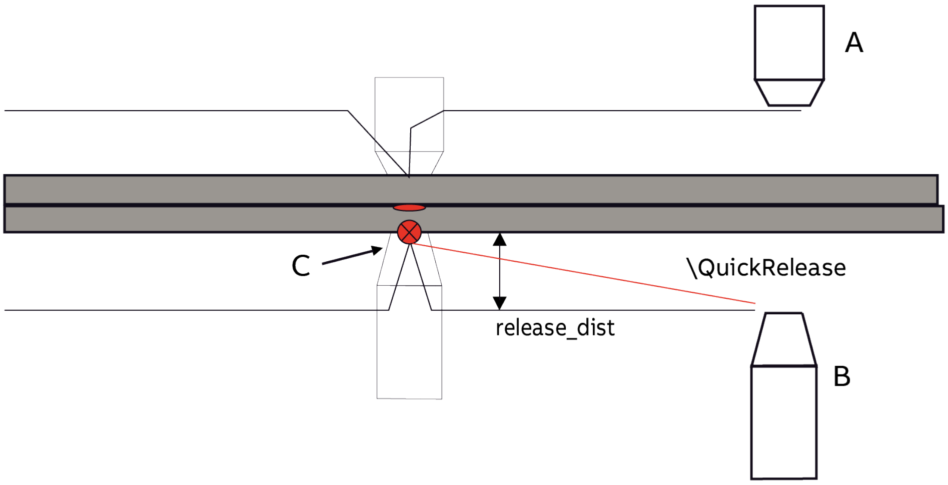
   See *SpotL/SpotJ - The basic spot welding instructions on page 91*.
8. When all weld positions are checked, reset the simulation mode to disable the 'touch up mode' mode.

   See *simdata - Simulation data on page 145*.

6.4 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

A	Movable electrode
B	Fixed electrode
C	The weld position

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 *spotdata - Spot weld data on page 138*.

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 5 mm, the moving tip will open to 1.5 x release distance + used plate thickness and used close position adjust 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.

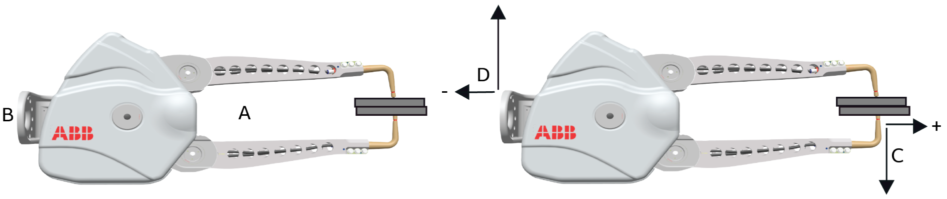
For more information about the release_dist parameter, see *spotdata - Spot weld data on page 138*.

6.5 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.

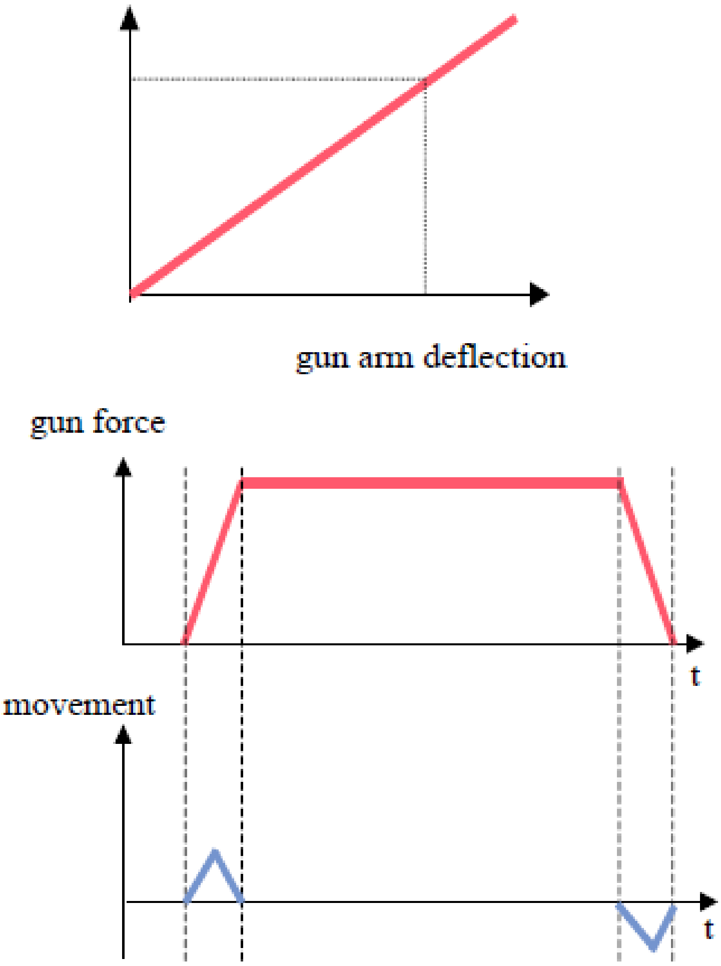


xx1200000230

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 in z direction`, `Deflection in x direction` , `Force matching deflection values` and `Ramp time matching deflection values` . These data is normally found in the data sheet for the used gun. See *Technical reference manual - System parameters*, Topic Motion, Type SG_Process.

Note

Note that the deflection values in the configuration are specified in meters (m).

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 * Deflection in z direction /
     Force matching deflection values;
```

Continues on next page

```
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.

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 or using a dial indicator.

2. Enter the measured values in the SG_Process configuration. This can be done from the Servo gun motion settings in the Spot UI, `Deflection in z direction` and `Force matching deflection values`, for example 5mm (0.005m) at 4000 N. See *Technical reference manual - System parameters*, Topic Motion, Type SG_Process.
3. Save the motion configuration.

6.6 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

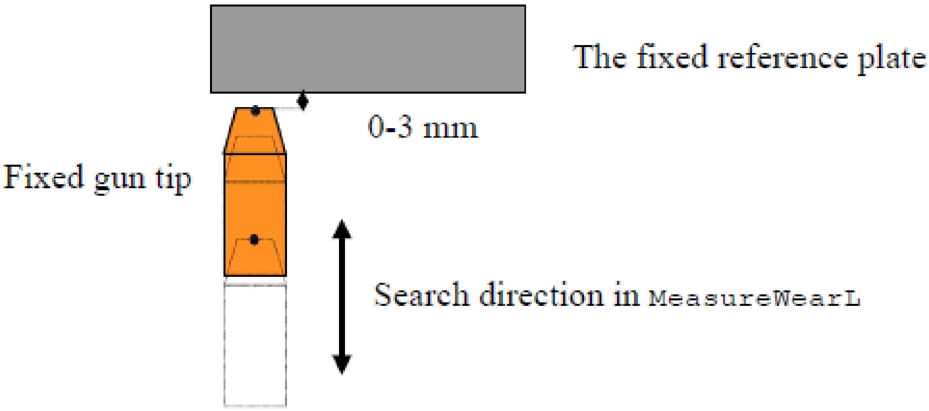
`MeasureWearL`, the tip wear for the fixed tip is measured and the tip wear is then compensated in current used `tooldata`.

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 116*.

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 68*.
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 115*.

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 117*.

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 Spot SoftWare Equalizing instance on page 42*.

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. 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 115*.

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.

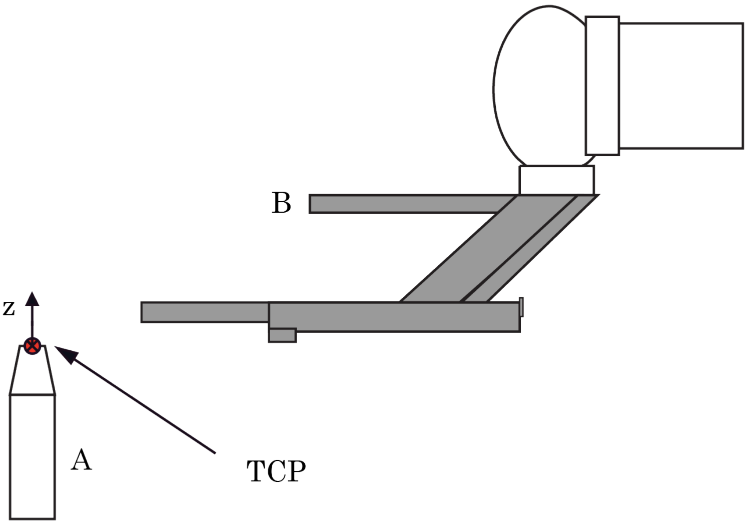
Tip wear measurement and compensation for stationary guns

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 179*

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 - OmniCore*.



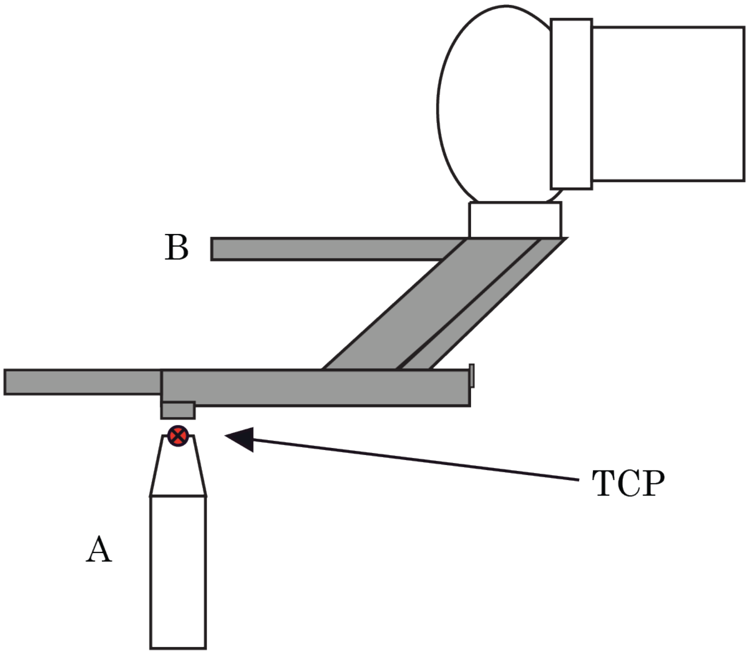
en1200000227

A	Fixed electrode
B	Gripper

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

Continues on next page

the same principles for the measurement and the `MeasureWearL` instruction, it has to be done as described in following items:



en1200000226

A	Fixed electrode
B	Gripper

- 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 111* and *CalibL/CalibJ - Calibrate a servo gun during robot movement on page 105*.
- 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 115*.

Continues on next page

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 123*.

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.

6.7 Movable gun arm search mode

General

The movable gun arm search method can be used as a complement to the standard software equalizing method if there is a need to compensate for programming errors, or tolerance variations, that is, the plates are not located exactly in the nominal position.

When a spot instruction with search mode activated is run, the robot will move to the programmed and nominal position, and then use the external axis for the gun to search for the plates and adjust the TCP in the tool z-direction based on search hit distance.

Prerequisites

Movable gun arm search mode is available for the `SpotL` and `SpotJ` instructions. Available in the RobotWare option *3417-3 Spot Welding Premium* Plus.

Recommendations

When using movable gun arm search mode it is important that the gun search performance is properly tuned.

Some general tips when using movable gun arm search mode:

- Verify that the search performance is tuned properly, bad tuning may have a negative or harder impact on the plates or the search movement may stop before the plates are actually reached.
- The clamping distances cannot be too far apart, as there must be some resistance for the gun arm to react on.

For information about how to tune the search performance, see *Application manual - Servo Gun Setup*.

Limitations

The following limitations apply when using movable gun arm search mode:

- The search performance is gun dependent, and it may be difficult to tune some guns because of various reasons, motor type, construction etc.
- This function should in general have a **low** force impact on the plates during the welding sequence, however the search performance is dependent on the gun tuning.
- Cannot be used to compensate for too large variations, only up to a few mm.
- Cycle times will be longer when using search mode because of the the slower gun arm speed during the search sequence.

Note

Movable gun arm search mode is only implemented for the `SpotL` and `SpotJ` instructions and does not work for the `SpotML` and `SpotMJ` instructions.

Continues on next page

Note

This function allows the tolerances of the parts to vary a bit more compared to using only standard software equalizing. Note that the total weld process cycle time will be longer than without, up to 500 ms per spot depending on the current settings and the actual plate position.

Function overview

Movable gun arm search mode is activated by selecting an optional switch in the `SpotL` and `SpotJ` instructions, see *SpotL/SpotJ - The basic spot welding instructions on page 91*.

With the `\Search` data activated in the spot instruction the external axis for the gun will be used to search for the plates, and the robot TCP will be adjusted in the tool z-direction based on the search hit distance.

Note

If the normal software equalizing functionality is deactivated then the movable gun arm search mode 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 movable gun arm search functionality (`\Search`). The parameter is an on/off switch for the search 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.

The optional parameter `\Search` of type `searchdata` has one data component for adjusting the position after search, specific for each position.

The `searchdata` has the following structure:

- `search_pos_adjust` (search position adjust) for the specific position.

The parameter `search_pos_adjust` can be used to compensate for soft sheet metal stiffness of the plates for example if the plates are weaker in a specific position. The value can be positive or negative, see *searchdata - Search data on page 148*.

Example

Program example with and without movable gun arm search functionality: P20 and P50 is executed in standard software equalizing mode.

P30 is executed with in gun arm search mode because of tolerance variations in that specific position. Since the sheet is relatively stiff in this position there is no need to compensate for softer sheet metal.

P40 is also executed with gun arm search mode for the same reason. But because of the softer sheet metal in this specific target, and to not deform the sheet the gun will move back 2 mm to reduce the impact.

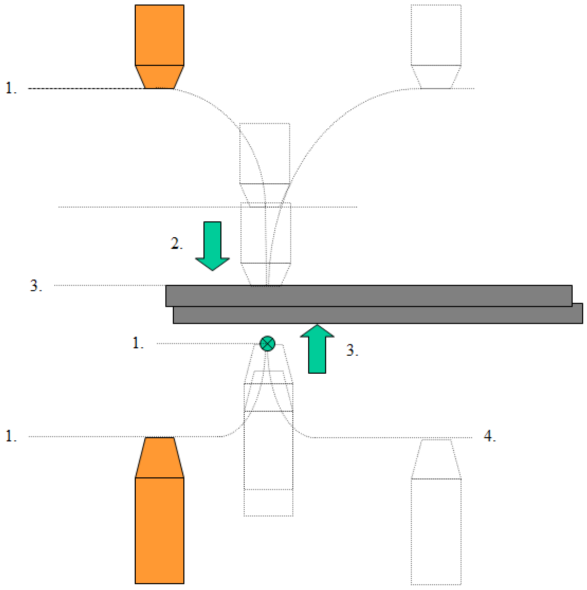
Continues on next page

Since the search functionality only is needed in these two specific positions, it is only used there because of the additional cycle time impact.

```
PERS searchdata search1:=[0];
PERS searchdata search2:=[2];
PROC main()
  MoveJ P10, v1000, z50, tool1;
  SpotL P20, vmax, gun1, spot11, tool1;
  SpotL P30, vmax, gun1, spot12 \Search:=search1, tool1;
  SpotL P40, vmax, gun1, spot13 \Search:=search2, tool1;
  SpotL P50, vmax, gun1, spot14, tool1;
ENDPROC
```

Execution

When a spot instruction with `\Search` data active is run, the robot will move to the programmed and nominal position, and then use the movable gun arm to search for the plates. When the plates are found the position will be **recalculated and adjusted** in the tool z-direction based on the search hit distance, and the weld will be performed in the new position, see figure below.



SearchSequen

1. Movement to the nominal position, via release distance.
2. The gun will start the search movement from the open position.
3. When the gun hits the plate the position will be recalculated and adjusted in the z-direction based on the search hit distance and the weld will be done from that new position.
4. After weld the robot will move out to the release distance at the same time as the gun opens.

Related information

	Described in
Manual actions	*Manual actions on page 68*

Continues on next page

	Described in
Process configuration	*Spot process configuration on page 19*
MeasureWearL	*MeasureWearL - Measure current electrode wear and recalculate the TCP on page 115*
ReCalcTcp	*ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 123*
Customizing	*Customizing RobotWare-Spot on page 219*
SpotL/SpotJ	*SpotL/SpotJ - The basic spot welding instructions on page 91*
Search data	*searchdata - Search data on page 148*
Servo Gun Setup	*Application manual - Servo Gun Setup*

7 Servo gun motion control

7.1 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 (ADU's) 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. With the Spot 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.

Note

Normally the gun specific parameters are supplied by the gun manufacturer.

References

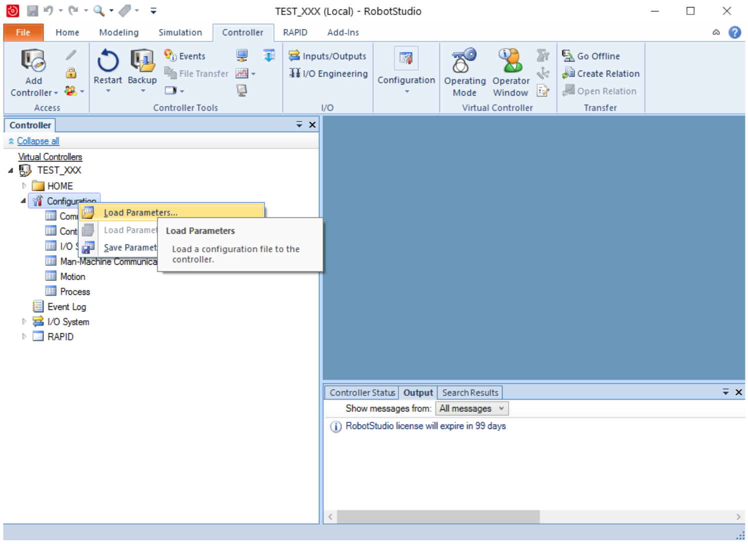
Type of information	See
`CalibL`, `CalibJ`, `Calibrate`, `SpotL`, `SpotJ`, `SetForce`, `STTune`, `gundata`, `spotdata`, `forcedata`	*RAPID references on page 91*
General motion control and programming	*Operating manual - OmniCore*
`ActUnit`, `DeactUnit`, `MoveL`, `MoveJ`, `robtarget`, `tooldata`	*Technical reference manual - RAPID Instructions, Functions and Data types*
How to tune a servo gun	*Application manual - Servo Gun Setup*
Hardware: motors, resolvers, drives, servo gun parameters, tuning a servo gun	*Application manual - Additional axes for OmniCore*

7.2 Installation and service

Install servo gun parameters

If the system is cold started, the servo gun parameters are most likely not loaded. If no backup of the system is available then follow these steps, otherwise restore the backup, then the complete system will be ready for production after restart

1. Load the servo gun parameters from RobotStudio via the configuration editor, tap **Configuration**, **Load 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 already activated via the configuration, see *Servo tool change on page 215*.



Load a configuration file to the controller.

RobotStudio license will expire in 99 days

xx2300000124

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	• From the Spot FlexPendant interface, select **SpotWare** and **Manual Actions** and tap **Search for servo gun**. • From the Code Editor, tap **Debug**, and then tap **Call Service Routine** and tap `ManualGunSearch`. See *gundata - Equipment specific weld data on page 134*.	Follow the instructions in the routine.
2	Ready.	

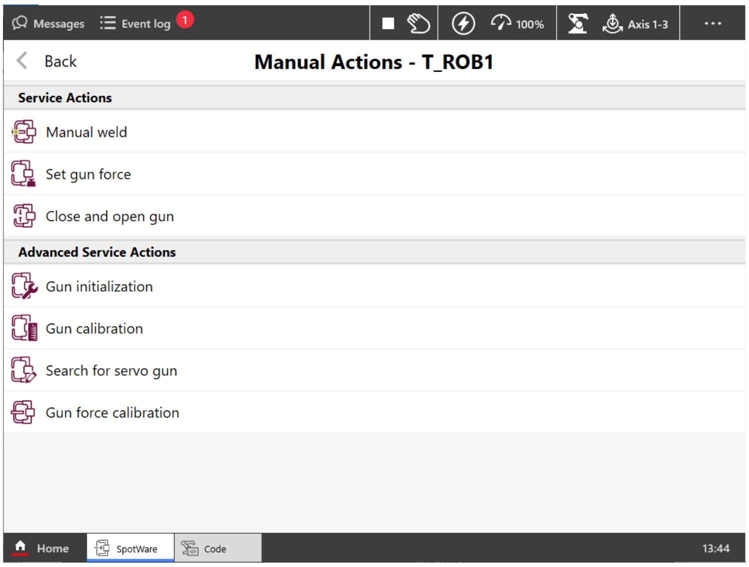
Continues on next page

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.

From the Spot FlexPendant interface, tap **Manual actions** then tap **Gun force calibration.** Or from the Code Editor, Tap **Debug** and then tap **Call Service Routine**.



xx2300000086

	Action	Note
1	Run the service routine `ManualForceCalib`, tap **Setup**.	Enter the max allowed force for the gun, the lower force limit, the number of force measurements and the sensor thickness to be used during the calibration sequence.
2	Then tap **Run**. This will perform the calibrations.	Follow the instructions in the routine.
3	Ready.	

Number of measurements

Enter number of measurements, for example 2.

Most times 2 measurements are good enough, but sometimes more samples may be needed, e.g. for non linear guns.

Max gun force

Enter the max force for the gun, for example 4000 N.

The lower force limit will then be recalculated based on the specified max force and number of measurements.

Continues on next page

Lower force limit

The lower force limit is always set to the calculated step value at start of the routine, for example 4000 / 2 = 2000 N or if the number of measurements and/or max gun force is changed.

Lower limit can be tuned to a different value, and that value will be valid as long as the force calibration routine is run. It will be reset to the calculated step value if the routine is restarted.

Sensor thickness

Enter the force sensor thickness used to measure the forces, for example 15 mm.

Squeeze time

Enter the duration of the force measurements, for example 2 s.

Note

The lower force limit will be recalculated if the maximum gun force and/or number of measurements are changed.

Note

The lower force value will be limited to `1100 [N]`, or `110 [daN]` or `250 [lbf]` depending on configuration setting, but it can be changed to a lower value if needed. It is mandatory to validate the selected force if a value lower than the calculated step value is entered.

Note

It is important to specify the correct sensor thickness to achieve the correct gun force accuracy.

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 81*.

Continues on next page

Note

If a gun service / repair has been made, the force calibration should be done again to ensure the forces are 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.

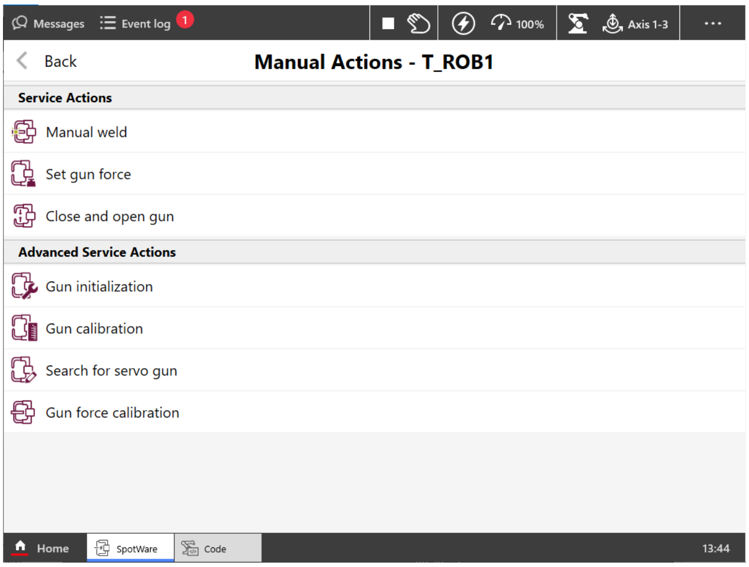
This function can be used if a servo gun loses force when the movable gun arm moves against gravity when closing. Normally there should be no need to compensate for loss of gun force, but for certain types of guns there may be a risk that gravity can influence the gun force negatively depending on on 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.

From the Spot FlexPendant interface, tap **Manual actions** then tap **Gun force calibration**. Or from the Code Editor, Tap **Debug** and then tap **Call Service Routine**.



xx2300000086

	Action	Note
1	Run the service routine **Gun force calibration**, 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.	

Continues on next page

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

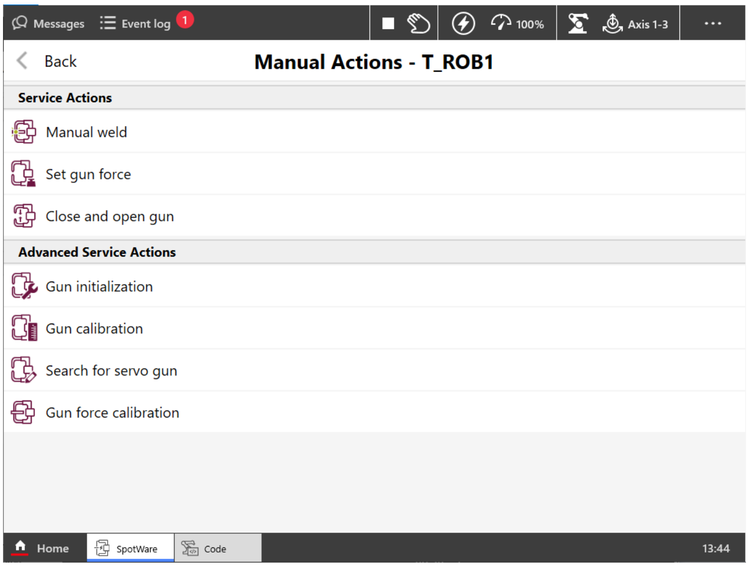
The actual force will not be compensated with 100% accuracy, but the deviation from ordered gun force will be less than without compensation.

Servo gun initialization 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 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 68*.

From the Spot FlexPendant interface, tap **Manual actions** then tap **Gun initialization**. Or from the Code Editor, Tap **Debug**, **Call Service Routine** and then tap `ManualServiceCalib`.



xx2300000086

Continues on next page

Servo gun initialization 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 **Calibrate** view, select the desired gun and then tap **Calibration Methods**, **Calibration Parameters** and **Fine calibration**. There is no need to jog the axis to any particular position.	**Note** Make sure that new electrode tips are used since all tip wear data will be cleared.
2	Run the service routine **Gun initialization** with *Initialize.* 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. **Note** If the gun is not force calibrated and properly tuned, follow the tuning procedure described in *Application manual - Servo Gun Setup*.
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.

Servo gun initialization 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 **Calibration Methods** and **Revolution Counters.** There is no need to jog the axis to any particular position.	

Continues on next page

	Action	Note
2	Run the service routine **Gun initialization** with *Synchronize.* 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. **Note** If the gun is not force calibrated and properly tuned, follow the tuning procedure described in *Application manual - Servo Gun Setup*.
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 initialization' 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 Setup*.

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.

Continues on next page

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 215*.

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, select the mechanical unit and tap the **Activate** button 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 synchronization, see *Servo gun initialization calibration on page 204*.

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. Start up the system.
3. Perform a fine calibration of the gun.
4. Perform a gun position initialization, see *Servo gun initialization calibration on page 204*.
5. To make sure the gun force is correct a force calibration should be performed if it is not already done for that gun, see *Servo gun force calibration on page 201*.

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`.

7.3 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 - OmniCore*). 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 used.

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 127*. This mode can be reset by executing the instruction `IndGunMoveReset`. See *IndGunMoveReset - Resets servo gun from independent mode on page 129*.

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.

7.4 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 81*.

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 138*.

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 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 91* and *SpotML/SpotMJ - Spot welding with multiple guns on page 96*.

Squeezing without welding, tip dressing

A gun closing is also typically done after tip dressing and after changing tips. The force will be held constant for a certain time, and then the gun is opened up again.

If using the `SetForce` instruction it 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 100*.

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 for OmniCore*) or the Servo Gun Setup wizard (see *Application manual - Servo Gun Setup*).

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.

7.5 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 134*). 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 111*, or during a robot movement, see *CalibL/CalibJ - Calibrate a servo gun during robot movement on page 105*.

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 177*.

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 134*.

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 189* and *MeasureWearL - Measure current electrode wear and recalculate the TCP on page 115* or the *ReCalcTCP - Calculate current electrode wear and recalculate the TCP on page 123*.

Supervision during tip calibration

The same supervision will be active during calibration as during asynchronous movements with force control.

7.6 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.

Note

In case software equalizing functions are used for the stationary gun, there may be a need to configure the deflection and release movement direction.

See *The Spot Gun Equipment instance on page 34* on how to configure movement direction.

7.7 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

Servo tool change can be used up to 8 different tools but is limited by 14 axes in total for the drive module. E.g if robot is on a track motion or if another additional axis is connected to a drive module it reduces that number of allowed tools that can be used with servo disconnect.

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 SGUN_1.
4. Define an *Input Signal* in type *Relay* with a signal that is also defined in the *I/O* configuration.

   For example *diToolConnected*, and this input signal should be connected to a sensor on the tool changer that indicates a physically connected gun.
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 a servo gun is activated and have a safe way to tool change, it is strongly recommended that the connection relay functionality is used when tool changing.

Basic example, tool change procedure

The procedure to switch between gun SGUN_1 and gun SGUN_2 must includes these minimal actions (excluded here is the needed communication with the tool changer, the tool stand and necessary robot movements):

Specify a connection relay signal for the gun to prevent accidental activation of not not connected gun as described above.

1. Deactivate gun SGUN_1 with the instruction DeactUnit.

   The position of gun SGUN_1 is stored.
2. Disconnect gun SGUN_1.

   Disconnect the servo gun motor cables.
3. Connect gun SGUN_2.

   Connect the motor cables to motor SGUN_2.
4. Activate gun SGUN_2 with the instruction ActUnit.

   The latest position of gun SGUN_2 is restored.
5. Run a tool change calibration of gun SGUN_2 with the instruction STCalib \ToolChg or the corresponding spot instruction Calibrate \ToolChg.

   Verify that 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 (Tool ID). That information may be used to make sure correct mechanical unit is activated.

It is recommended to block activation of not connected mechanical units by specifying a digital input (DI) in the connection relay motion system parameter (type *Relay* in topic *Motion*) for each servo gun. This digital input, which also is setup in the I/O configuration, is read when the mechanical unit is activated from the tool change program sequence. 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

8 Customizing RobotWare-Spot

8.1 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.

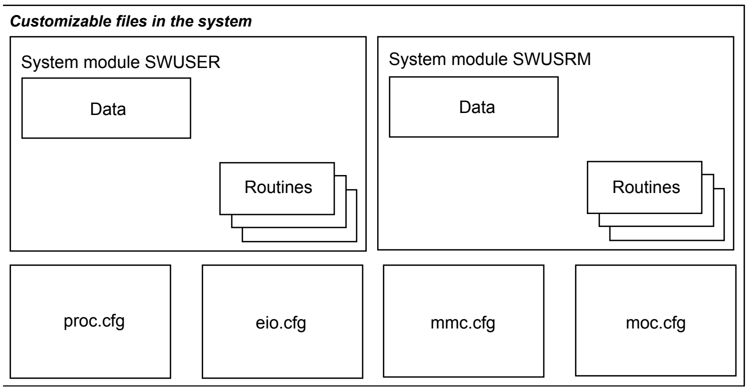
The following customizing is described in this manual:

- *How to remove not used signals from the process sequence on page 222*
- *How to remove not used process hooks on page 223*
- *How to change the number of guns equipment to be used on page 224*
- *How to define max/min values for data components on page 225*
- *How to change the Spot data types on page 226*
- *How to add functionality in the process sequence on page 227*
- *How to use spot data programmed in the weld timer on page 229*
- *How to set the number of automatic rewelds after weld error on page 231*
- *How to change the user modules names and file path on page 232*
- *How to package and install the result from the customizing on page 233*

8.2 Files to be changed during customizing

Description

Customizing can be done by changing a number of predefined data and routines, preferably using a ordinary 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 173*.

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 166* .

Process configuration

The process configuration is used to to setup the spot system. See *Spot process configuration on page 19*.

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 44*.

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 68*.

SYS configuration

This configuration file contains for example information about which tasks that the user modules are loaded in. See *System modules on page 166*.

MOC configuration

This configuration file contains for example parameters for servo guns. See *Servo gun motion control on page 199*

8.3 Customizing guides

8.3.1 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 Spot Media Equipment instance on page 39*.

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 Spot Media Equipment instance on page 39*.
3. Save the configuration and restart the system.

The same procedure can be used on other not used signals if needed.

8.3.2 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 168*.

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**.

8.3.3 How to change the number of guns equipment 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 equipment depending on the selected configuration (Multiple Guns etc). But it is possible to use and configure up to ten (10) gun equipment in the system if the default configuration in not sufficient.

1. Add or remove the number of instances in following spot data related arrays in SWUSER.
   - `curr_gundata`
   - `curr_spotdata`
   - `curr_forcedata`

   Example:

   ```
   PERS spotdata curr_spotdata{2} := [[0,0,0,0],[0,0,0,0]];
   ```

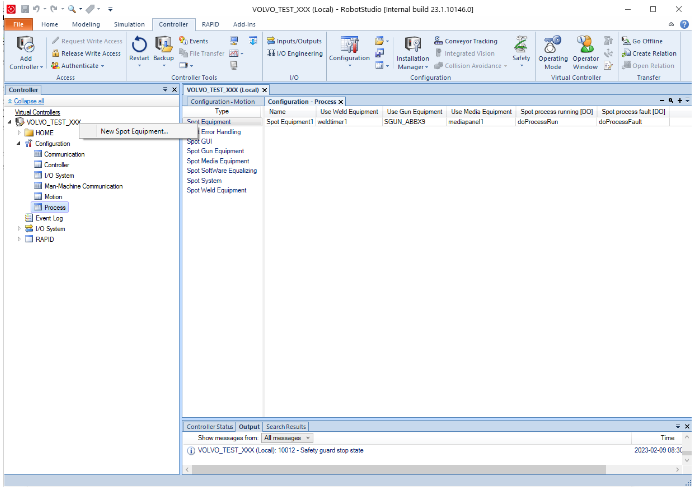
2. Add or remove the predefined `gunnum` gun index data in SWUSRM accordingly, ie. gun1, gun2.

   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 19*.

   Example:



Type	Name	Use Weld Equipment	Use Gun Equipment	Use Media Equipment	Spot process running [DO]	Spot process fault [DO]
Spot Equipment	Spot Equipment1	weldtimer1	SGUN_ABBX9	mediapanel1	doProcessRun	doProcessFault

Show messages from: All messages

	Time
VOLVO_TEST_XXX (Local): 10012 - Safety guard stop state	2023-02-09 08:30

   xx2300000135

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 44*.

Note

Code changes in this module requires a restart using the mode **Reset RAPID**.

8.3.4 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 Spot System instance on page 21*, *The Spot SoftWare Equalizing instance on page 42* and *The Spot Gun Equipment instance on page 34*.

8.3.5 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 166*.
Example: add new components for second gun force in forcedata.
`RECORD forcedata`
`  num tip_force;`
`  num force_time;`
`  num plate_thickness;`
`  num plate_tolerance;`
`  num tip_force2;`
`  num force_time2;`
`ENDPROC`
	It is possible to:
	• Add or delete data components
	• Move data components from for example `gundata` to `spotdata`
	• 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).
Example new components in forcedata:
`PERS forcedata force1 :=`
`    [1000,2,0,0,500,2];`
	• curr_gundata
	• curr_spotdata
	• curr_simdata
	• 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 173*.	• DefineSpotData • DefineGunData • DefineForceData • DefineSimData

Note

Code changes in this module requires a restart using the mode **Reset RAPID**.

8.3.6 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 for some reason, and that is not handled by the built in supervision, 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 168*.
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 *Supervision task hook on page 172*.

Note

Code changes in this module requires a restart using the mode **Reset RAPID**.

8.3.7 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 Spot Weld Equipment instance on page 28*.

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 81*.

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 138*.

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 226*.

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;
  num release_dist;
ENDRECORD
```

Continues on next page

3. Definition of customized process spot data:

   ```
   RECORD spotdata
     num prog_no;
     num release_dist;
   ENDRECORD
   ```

8.3.8 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 Spot Error Handling instance on page 25*.

8.3.9 How to change the user modules names and file path

Description

Since the user routines are called and executed with the user module name in the path this needs to be changed if "custom" user modules are to be used instead.

Perform the following steps to modify or change the user modules names and path location.

1. Rename the and 'swusm.sys' 'swuser.sysx' and 'swusm.sysx'modules to something more suitable for the specific application, or create new modules with the same content.
2. Change spot system process configuration and specify the new module names in "Process task user module name" and "Motion task user module name".
3. Change the system process configuration and specify the new file path in 'Spot user modules file path', e.g. HOME:/MyMods.
4. Change the controller configuration, 'Automatic Loading of Modules' to reflect the new names and path, e.g. HOME:/MyMods/MyUsrMod.sys.
5. Reboot the system with **Reset RAPID** to reload all the modules.

For more information, see *The Spot System instance on page 21*.

8.3.10 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 additional RobotWare Add-In product, that is, a function package. This add-in product can then be included via Installation Manager in RobotStudio.

When the Add-In product 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 product.

Fore more information about RobotWare Add-Ins and how to create one, see *Application manual - RobotWare Add-Ins*.

This page is intentionally left blank

9 FlexPendant Interface

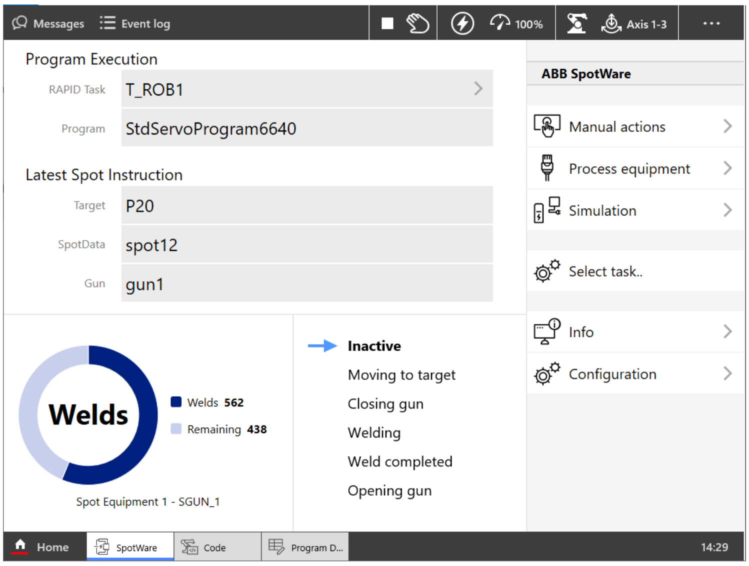
9.1 Application Overview

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 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 the Spot UI application, go to the Home menu and then tap **SpotWare**. The main view will be started. From here all spot related functions can be accessed. For further information on using the FlexPendant, see *Operating manual - OmniCore*.



Program Execution

RAPID Task: T_ROB1

Program: StdServoProgram6640

Latest Spot Instruction

Target: P20

SpotData: spot12

Gun: gun1

Welds 562

Remaining 438

Spot Equipment 1 - SGUN_1

Inactive
Moving to target
Closing gun
Welding
Weld completed
Opening gun

xx2300000088

Content

The Spot UI has the following content:

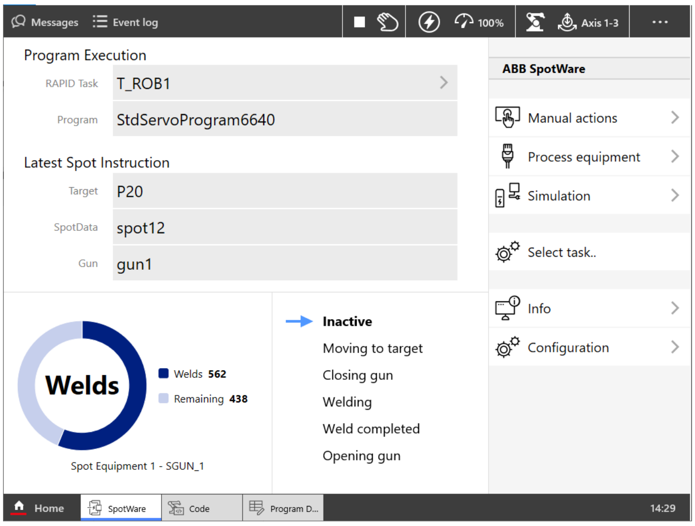
- **Main View**
- **Manual Actions**
- **Process Equipment**
- **Simulation**
- **Information**
- **Configuration**

The Spot main view provides basic status information about the current executing spot program with possibilities to reach other views and sub views. The welding process can be followed for each spot instruction.

9.2 Main View

Basic functionality in the main view

1 Tap the **Home menu** and then select **SpotWare**. A window will appear containing status information to follow the welding progress, latest/current spot instruction, name of the executing weld program, access manual actions, set simulation modes and so on.



Program Execution

RAPID Task: T_ROB1

Program: StdServoProgram6640

Latest Spot Instruction

Target: P20

SpotData: spot12

Gun: gun1

Welds 562

Remaining 438

Spot Equipment 1 - SGUN_1

Inactive

Moving to target

Closing gun

Welding

Weld completed

Opening gun

xx2300000088

2 Tap **Simulation** control to access and edit simulation data.

3 Tap **Process equipment** to view the connected equipment status.

4 Tap **Manual Actions** to access an run the spot service routines.

5 Tap **Configuration** to view and edit **some** of the spot related configuration data.

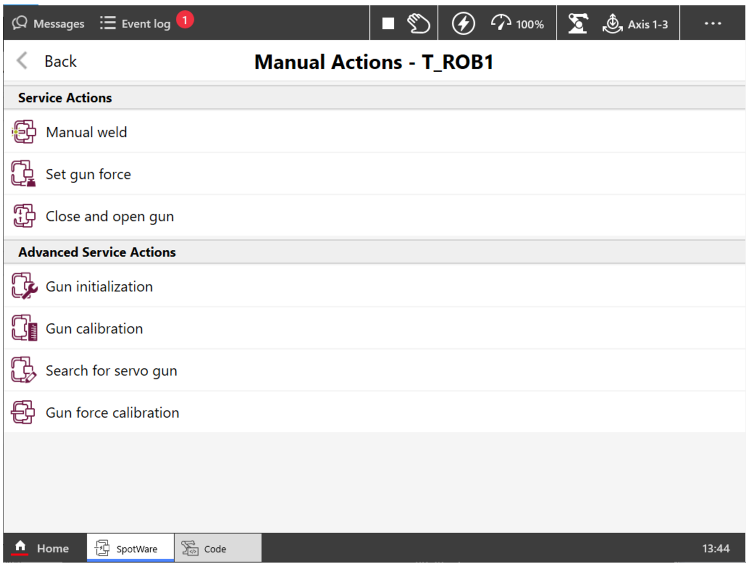
6 Tap the **Info** to view version info etc.

7 Tap **Select task** button to view another spot welding robot.

Continues on next page

Manual actions

Contains all available application related service routines in the system.



Manual Actions - T_ROB1

Service Actions

- Manual weld
- Set gun force
- Close and open gun

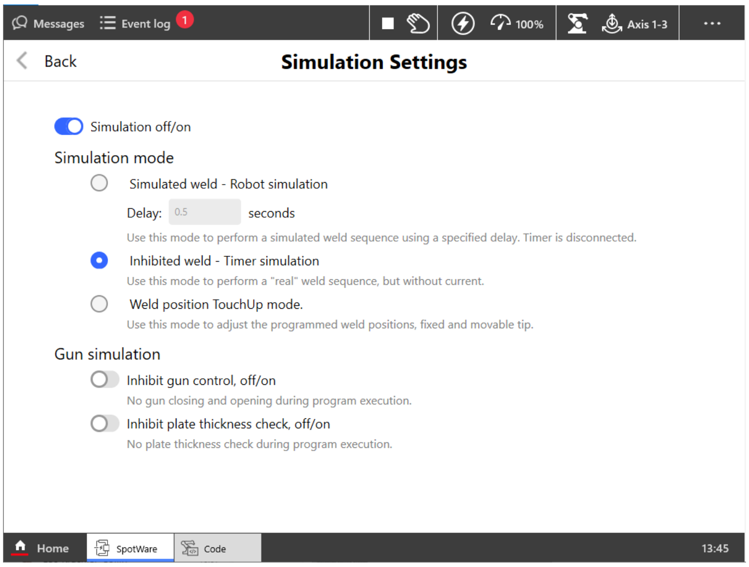
Advanced Service Actions

- Gun initialization
- Gun calibration
- Search for servo gun
- Gun force calibration

xx2300000086

Simulation

If the simulation mode is changed the **Simulation** button will flash with yellow tone as long as the simulation is active. For more information about simulation see *Simulation modes on page 66*.



Simulation Settings

● Simulation off/on

Simulation mode

○ Simulated weld - Robot simulation

Delay: 0.5 seconds

Use this mode to perform a simulated weld sequence using a specified delay. Timer is disconnected.

● Inhibited weld - Timer simulation

Use this mode to perform a "real" weld sequence, but without current.

○ Weld position TouchUp mode.

Use this mode to adjust the programmed weld positions, fixed and movable tip.

Gun simulation

○ Inhibit gun control, off/on

No gun closing and opening during program execution.

○ Inhibit plate thickness check, off/on

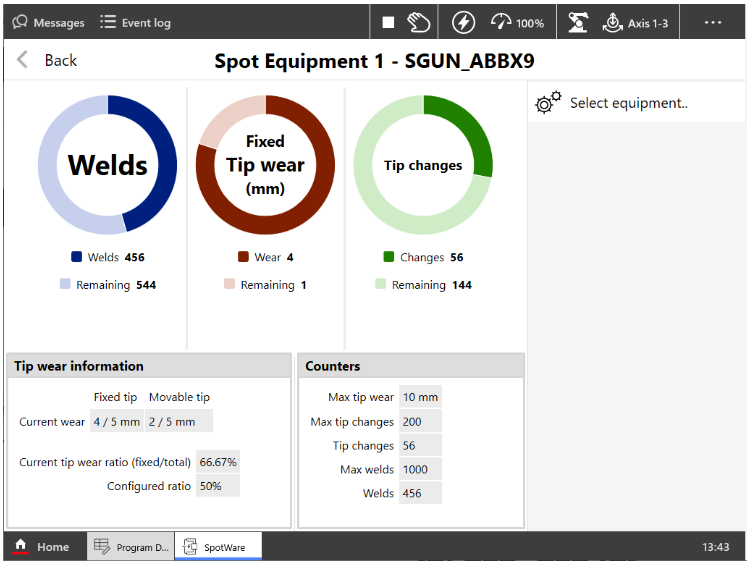
No plate thickness check during program execution.

xx2300000087

Continues on next page

Process equipment

Information in this area shows the current equipment counters, for example tip wear, number of welds etc. The information in this will be updated when Spot instructions are executed.



Spot Equipment 1 - SGUN_ABBX9

- Welds 456
- Remaining 544
- Wear 4
- Remaining 1
- Changes 56
- Remaining 144

Tip wear information

	Fixed tip	Movable tip
Current wear	4 / 5 mm	2 / 5 mm

Current tip wear ratio (fixed/total)	66.67%
Configured ratio	50%

Counters

Max tip wear	10 mm
Max tip changes	200
Tip changes	56
Max welds	1000
Welds	456

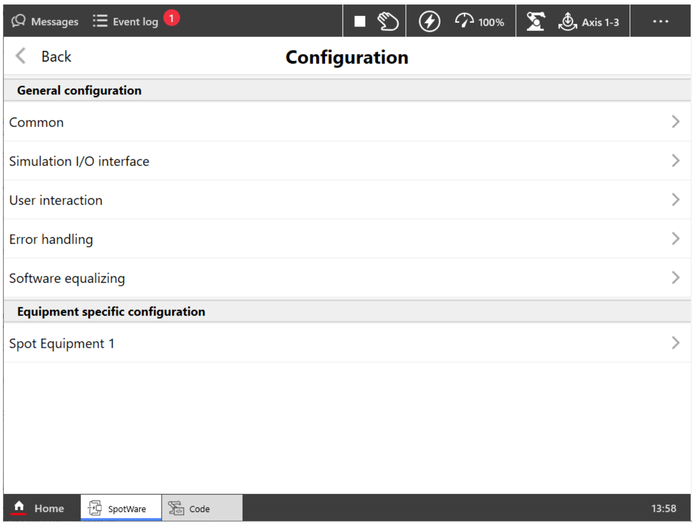
xx2300000105

Information

Information in this view shows application version and misc. system info.

Configuration

1 Tap tap **Configuration** in the SpotWare main window



Configuration

General configuration

- Common
- Simulation I/O interface
- User interaction
- Error handling
- Software equalizing

Equipment specific configuration

- Spot Equipment 1

xx2300000090

2 Select the configuration instance to be changed.

3 View or edit the value.

4 New value will be written to the controller as soon as a value has changed.

Tap **Back** to return to the previous window without any changes.

Continues on next page

Note

Access to the configuration may be limited by the user grants for the specific user. **Modify system parameters** grant is required.

Note

Not **all** application and system configuration are accessible from here, only a subset of SpotWare related configuration data can be viewed and/or edited.

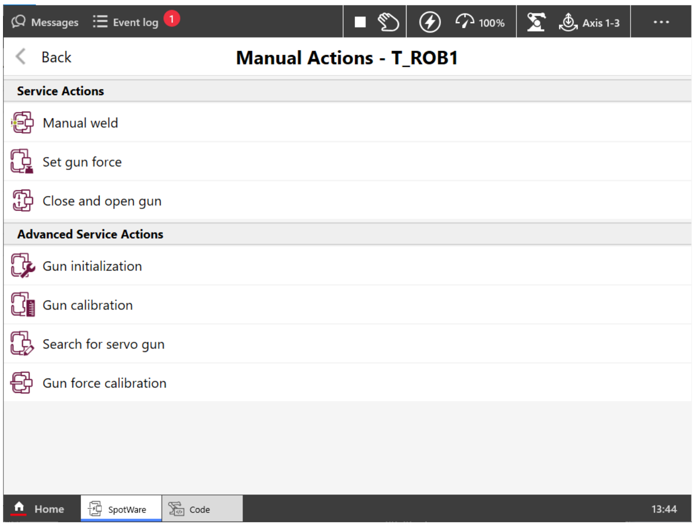
9.3 Manual Actions

Manual Actions

Manual actions contains all available application 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 main view. A window appears containing all available spot related service routines in the system.



xx2300000086

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 68*.

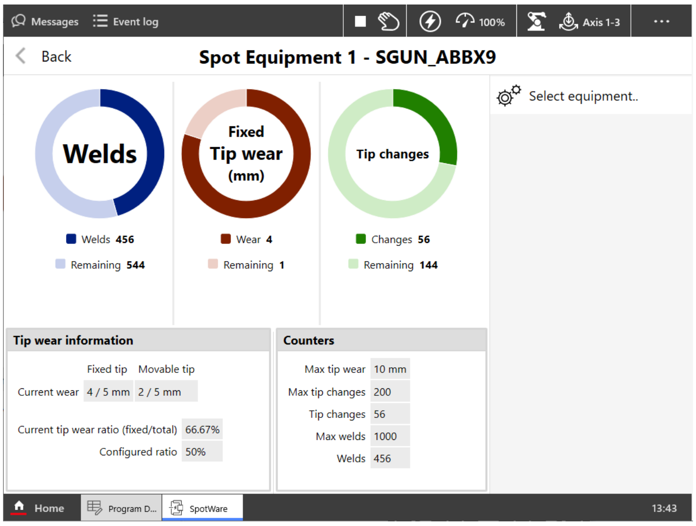
Manual versus Automatic mode

If the system is in manual mode you can run service routines, view or edit data, set simulation mode. When you switch to automatic mode all views are locked, but it's possible to see the current status.

9.4 Process Equipment

Basic functionality in the Process equipment view

1 Tap the **Process equipment** button in the main view. A window appears containing status information connected to different parts of the equipment, tip wear etc.



Spot Equipment 1 - SGUN_ABBX9

Legend	Value
Welds	456
Remaining	544
Wear	4
Remaining	1
Changes	56
Remaining	144

Tip wear information

	Fixed tip	Movable tip
Current wear	4 / 5 mm	2 / 5 mm

Current tip wear ratio (fixed/total) 66.67%

Configured ratio 50%

Counters

Max tip wear	10 mm
Max tip changes	200
Tip changes	56
Max welds	1000
Welds	456

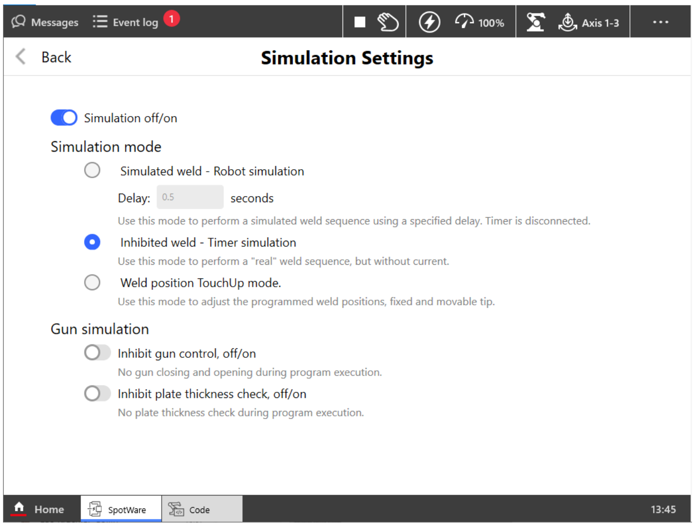
xx2300000105

2 Tap **Select equipment** to switch between all available gun equipment's in the system.

9.5 Simulation

Basic functionality in the Simulation view

1 Tap the **Simulation** button in the main view. A window appears containing quick settings for the different simulation modes.



Simulation Settings

● Simulation off/on

Simulation mode

○ Simulated weld - Robot simulation

Delay: 0.5 seconds

Use this mode to perform a simulated weld sequence using a specified delay. Timer is disconnected.

● Inhibited weld - Timer simulation

Use this mode to perform a "real" weld sequence, but without current.

○ Weld position TouchUp mode.

Use this mode to adjust the programmed weld positions, fixed and movable tip.

Gun simulation

○ Inhibit gun control, off/on

No gun closing and opening during program execution.

○ Inhibit plate thickness check, off/on

No plate thickness check during program execution.

xx2300000087

2 Tap **Simulation Off/On** to activate or deactivate simulation mode.

3 Select desired **Simulation mode**.

4 **Gun simulation**, tap **Inhibit gun control** to prevent gun closing/opening or tap **Inhibit plate thickness check** during program execution.

Note

In order to be able to change simulation modes, the proper access is needed, *Modify current value*. This gives access to modify RAPID data. For more information about grants see *Operating manual - OmniCore*.

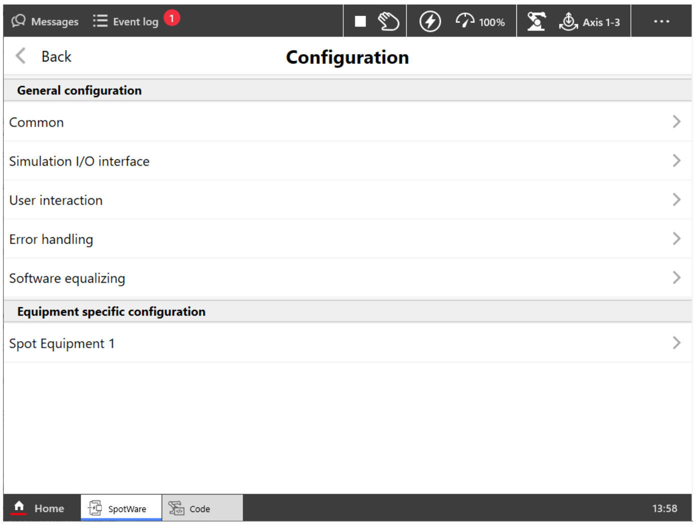
9.6 Configuration

Process Configuration

Process Configuration presents application specific configuration data, thus offering a quick and easy way to edit or view some parts of the application related configuration.

Basic functionality in the Configuration window

1 Tap **Configuration** in the SpotWare main window.



xx2300000090

2 Select the configuration instance to be changed.

3 View or edit the value.

4 New value will be written to the controller as soon as a value has changed.

Tap **Back** to return to the previous window without any changes.

Note

Not **all** application and system configuration are accessible from here, only a subset of SpotWare related configuration data can be viewed and/or edited. To access all system configuration RobotStudio is required.

Note

In order to edit configuration data, the right access is needed, *Modify system parameters*. This gives access to modify configuration parameters. For more information about grants see *Operating manual - OmniCore*.

Note

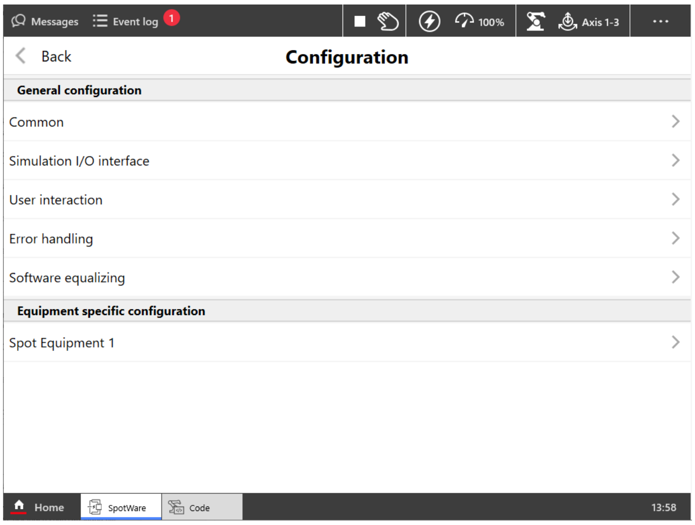
Some configuration changes may need a system restart.

Continues on next page

Change configuration

1 Tap **Configuration** in the SpotWare main window.

2 Select the configuration instance to be changed, for example **Error handling**.



Configuration

General configuration
- Common
- Simulation I/O interface
- User interaction
- Error handling
- Software equalizing

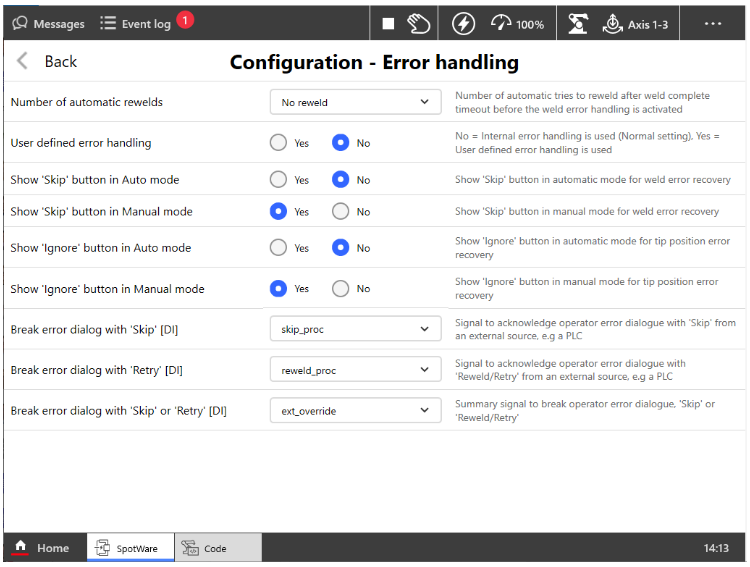
Equipment specific configuration
- Spot Equipment 1

xx2300000090

3 Select the parameter you want to change. New value will be written to the controller as soon as a value has changed.

Tap **Back** to return to the previous window without any changes.

4 Depending on the selected parameter, numeric pad, drop down list, alpha pad or radio buttons with **Yes**/**No** selection.



Configuration - Error handling

Number of automatic rewelds	No reweld	Number of automatic tries to reweld after weld complete timeout before the weld error handling is activated
User defined error handling	○ Yes ● No	No = Internal error handling is used (Normal setting), Yes = User defined error handling is used
Show 'Skip' button in Auto mode	○ Yes ● No	Show 'Skip' button in automatic mode for weld error recovery
Show 'Skip' button in Manual mode	● Yes ○ No	Show 'Skip' button in manual mode for weld error recovery
Show 'Ignore' button in Auto mode	○ Yes ● No	Show 'Ignore' button in automatic mode for tip position error recovery
Show 'Ignore' button in Manual mode	● Yes ○ No	Show 'Ignore' button in manual mode for tip position error recovery
Break error dialog with 'Skip' [DI]	skip_proc	Signal to acknowledge operator error dialogue with 'Skip' from an external source, e.g a PLC
Break error dialog with 'Retry' [DI]	reweld_proc	Signal to acknowledge operator error dialogue with 'Reweld/Retry' from an external source, e.g a PLC
Break error dialog with 'Skip' or 'Retry' [DI]	ext_override	Summary signal to break operator error dialogue, 'Skip' or 'Reweld/Retry'

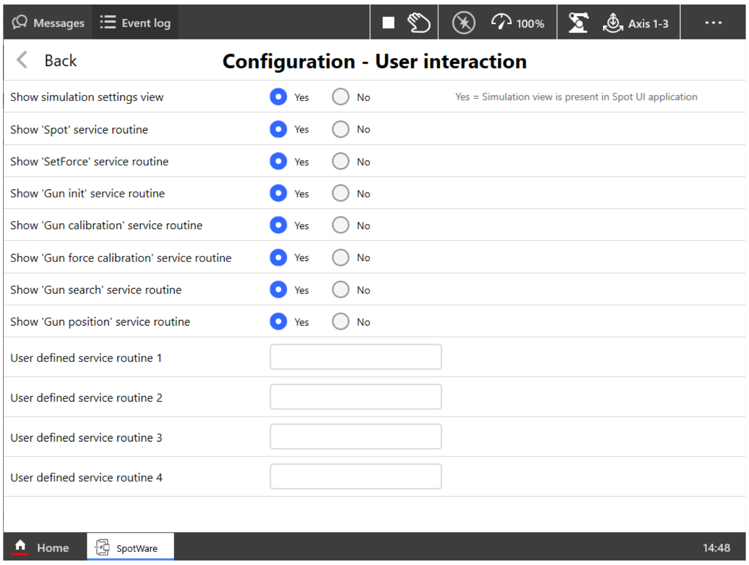
xx2300000094

More information about the application process configuration can be found in *Introduction on page 19*

9.7 Customizing the UI

Configuration of the Manual Actions view

The Manual Actions view can be modified to fit the current configuration. It is possible to add/remove service routines and also add custom service routines if needed.



Configuration - User interaction

Show simulation settings view	● Yes ○ No	Yes = Simulation view is present in Spot UI application
Show 'Spot' service routine	● Yes ○ No	
Show 'SetForce' service routine	● Yes ○ No	
Show 'Gun init' service routine	● Yes ○ No	
Show 'Gun calibration' service routine	● Yes ○ No	
Show 'Gun force calibration' service routine	● Yes ○ No	
Show 'Gun search' service routine	● Yes ○ No	
Show 'Gun position' service routine	● Yes ○ No	
User defined service routine 1		
User defined service routine 2		
User defined service routine 3		
User defined service routine 4		

xx2300000123

This page is intentionally left blank

Index

A
Additional axes, 199

C
CalibJ, 105
CalibL, 105
Calibrate, 111
Changing Motion parameters, 215
CloseHighLift, 130
Configuration, 19
Customizing Spot, 219

D
Deflection Compensation, 177

E
Editing instructions, 65
Error handling, 75
Error handling, user defined, 80
Example tool change procedure, 216

F
Fine calibration, 58
FlexPendant Interface, 235
Forcedata, 142

G
Gripper and pneumatic gun, 88
Gun Arm Release, 177
Gun closure, 71
Gundata, 134
Gun equalizing, 67, 70
Gun force calibration, 58
Gun init after fine calibration, 205
Gun init after updated revolution counters, 205
Gun opening, 72
Gun preclosing, 67

H
Halt with motors off, behavior, 73

I
I/O configuration, 44
Independent gun movement, 209
IndGunMove, 127, 132, 161, 164
IndGunMoveReset, 129
Install servo gun parameters, 200

M
Manual actions, 68, 240
ManualCalib, 69
ManualCheckMeasPos, 69
ManualForceCalib, 69
ManualGunControl, 68
ManualGunPosition, 68
ManualGunSearch, 68
ManualServiceCalibceCalib, 69
ManualSetForce, 68
ManualSpot, 68
MeasureWearL, 115
Multiple gun forces during weld, 81

O
OpenHighLift , 130
options
  Servo Tool Change, 215
  Simulated Weld Timer I/O, 45, 49, 53
  Spot, 11
  Spot Premium, 11
  Spot Premium Plus, 11

P
Power failure behaviour, 73
Process configuration, 19, 243
Process sequence, 70
Programming example pneumatic gun, 60
Programming example servo gun, 59
Programming principles, 15
Programming spot welding instructions, 64
Program stop behaviour, 72

Q
Quick start for servo gun, 58

R
ReCalcTCP, 123
Replace servo gun, 207
Reweld process, 79

S
Searchdata, 148
Service routines, 68
Servo gun activation, 208
Servo gun deactivation, 208
Servo gun disconnection, 86
Servo gun force calibration, 201
Servo gun force gravity compensation, 203
Servo gun tuning, 57
Servo Tool Change, 215
SetForce, 100
Simdata, 145
Simulated welding, 74
Simulated Weld Timer I/O, 45, 49, 53
Simulation, 74
Simulation mode, 66
Skip process, 79
Software Equalizing, 177
Spotdata, 138
SpotJ, 59, 91
SpotL, 59, 91
SpotMJ, 62, 96
SpotML, 62, 96
Spot options, 11
Spot welding data, 59
Stationary servo gun, 214
Step backward behaviour, 74
Step forward behaviour, 73
Supervision, 75–76, 79–80
SwGetCurrSpotName, 151–155, 160
SwGetCurrTargetName, 150
SwGetCurrThickness, 157–159
SWUSER, 166
SWUSRM, 173
Synchronous gun movement, 208

T
TCP, 179
Tip change, 212
Tip dressing, 87
Tip management, 212
Tip wear, 212
Tool change, 212

V
Virtual I/O unit, 49, 53

W
Weld error, 78
Welding, 72
Weld process timing, 82–83

ABB AB
Robotics & Discrete Automation
S-721 68 VÄSTERÅS, Sweden
Telephone +46 10-732 50 00

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

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

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

abb.com/robotics