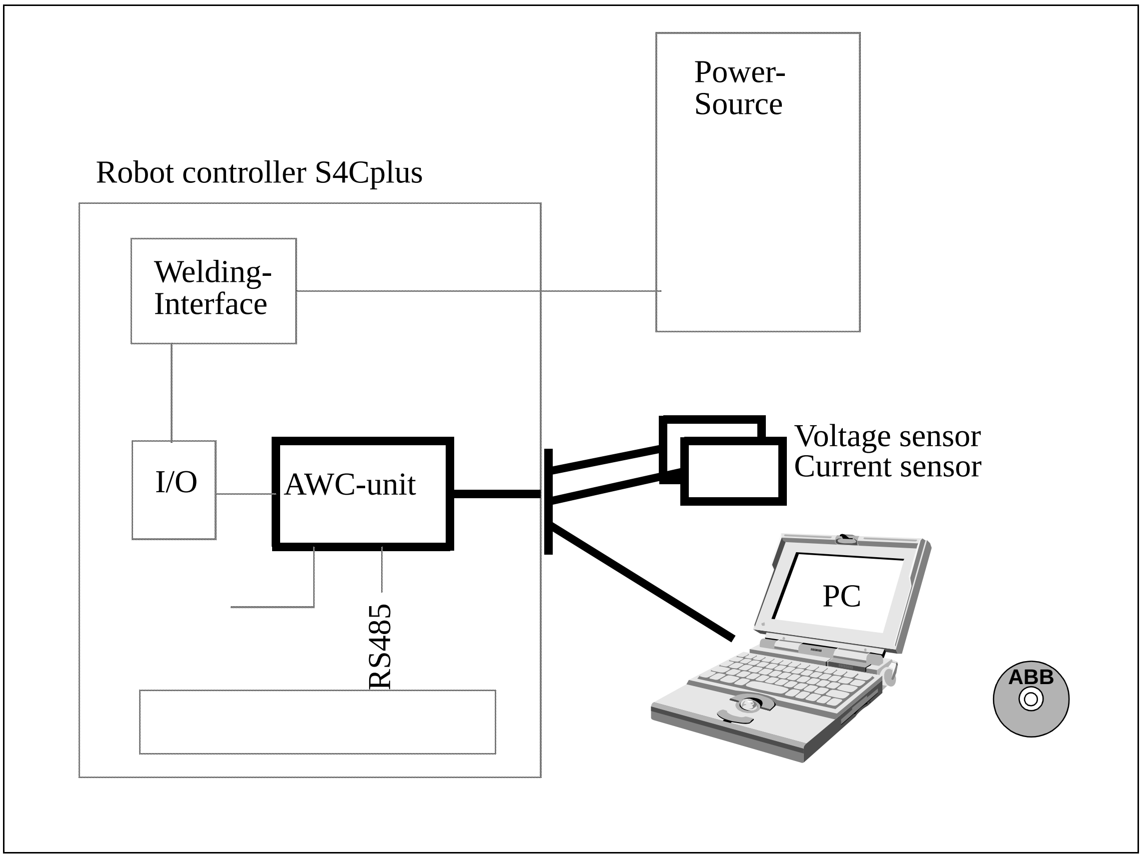
User's Guide

Advanced Weld Controller
AWC

505 498-102
2002-09

Seam Trackning System
Tracking and Multi Pass Functionality



ABB

Artikelnummer: 505 498-102

Datum: 2002-09

ABB Automation Technology Products AB

Arc Welding & Application Equipment

S-695 82 Laxå

Sverige

TABLE OF CONTENTS

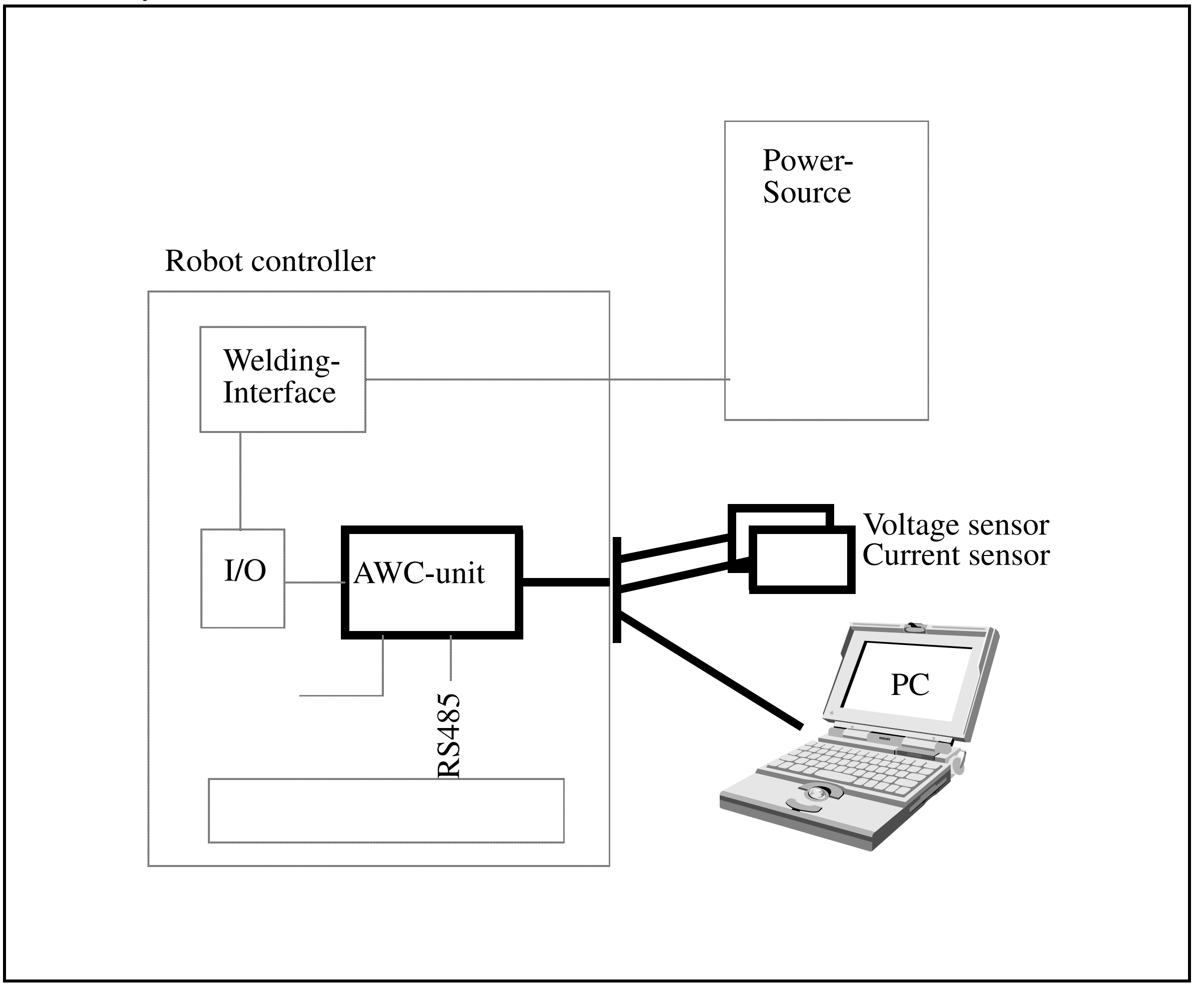
1.0 Functional description

The Advanced Weld Control (AWC) is a microprocessor based weld sequence controller. The controller is seamlessly integrated to the robot controller via a serial link. The system gives an tracking functionality of the path. Adjusting the robot to the actual path location.

The AWC measures current and voltage of the arc and sends pathcorrections to the robot.

Measurements are made at the edge of the weave pattern.

System



2.0 Mechanical Installation

Figure 1 and Figure 2 show the dimension and mounting pattern for the AWC. Locate the AWC in a suitable enclosure. Allow access to the rear terminal block panel and operational visibility of the front panel. Locate the connector panel on the side of the enclosure so as to provide reasonable access to the connectors. Route the control cables from the connector panel to the AWC controller. Keep the AWC control cables away from any power control lines or cables. It is recommended that the cables be placed in a wiring duct independent of other high voltage control solenoid or motor wiring. Refer to Figure 3 and connect the control cables to the indicated terminal block connections. The cables should be trimmed to the correct length before installation. Do not coil any excess cable. Connect a suitable source of 115 Vac power to the AWC to TB2. Connect a ground wire from TB2-3 to chassis ground.

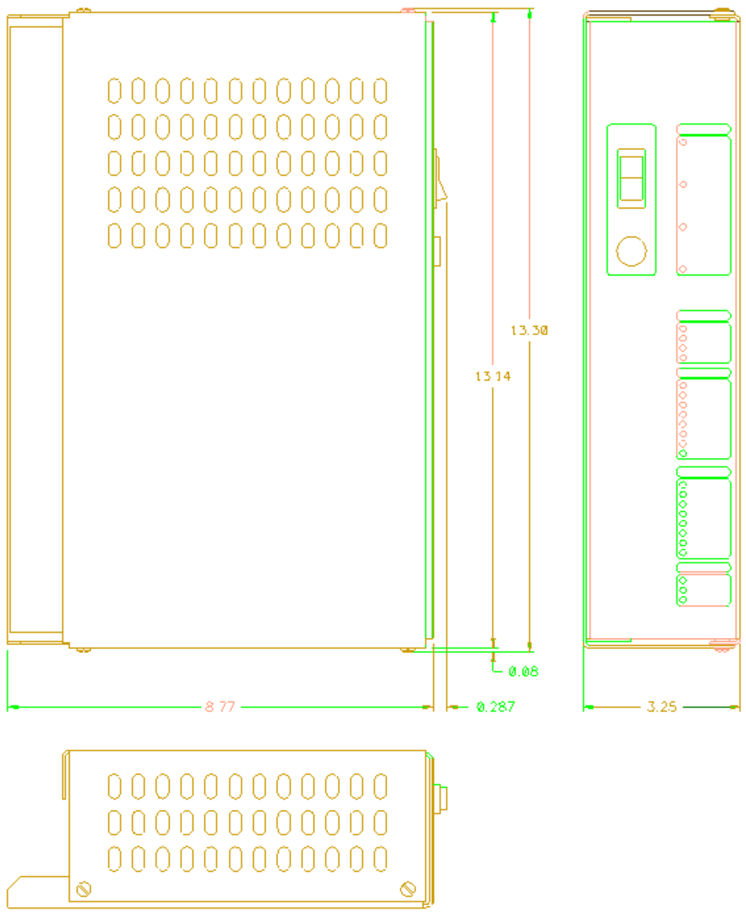


FIGURE 1. AWC dimensional outline

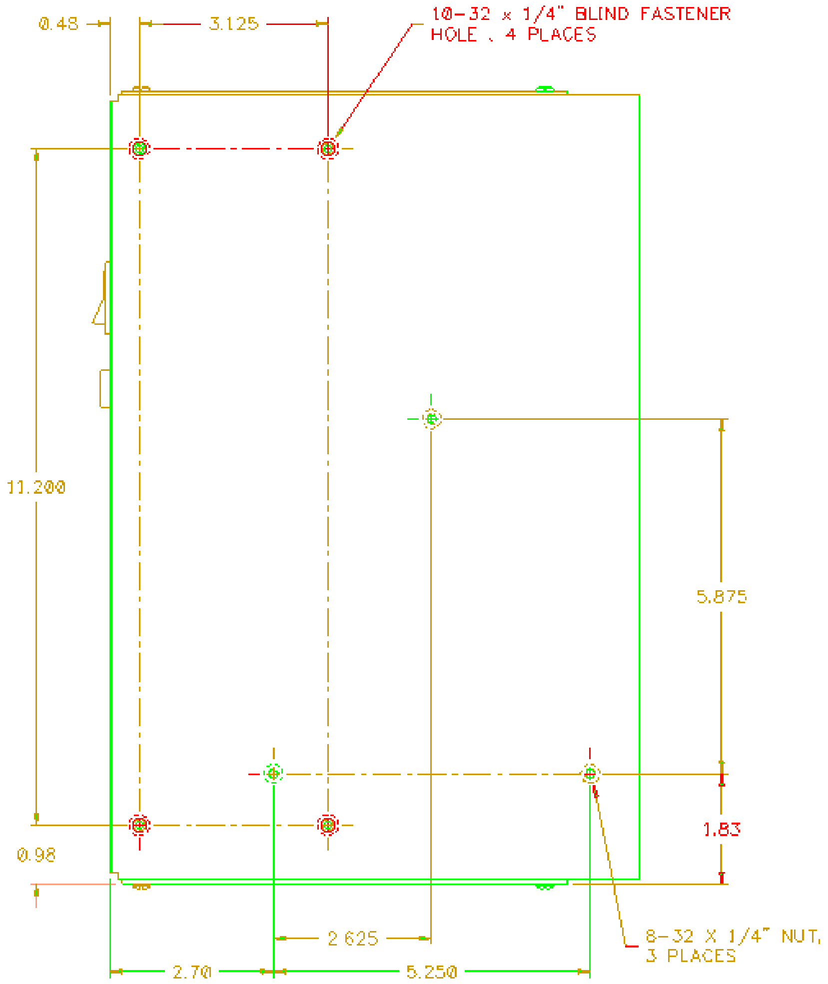


FIGURE 2. Mounting Dimensions for AWC enclosure

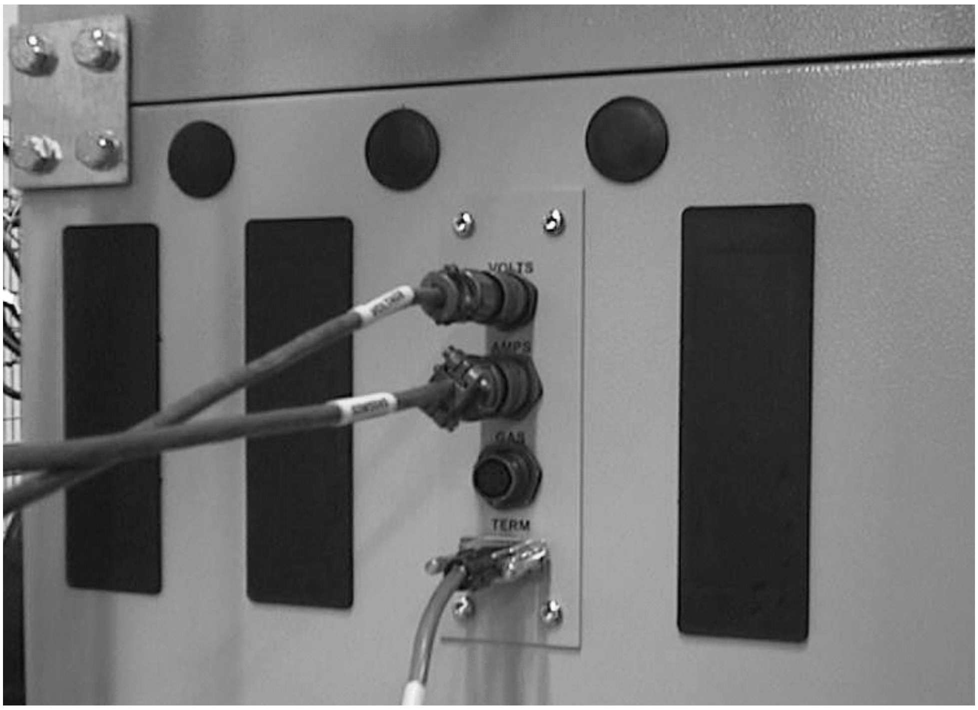
2.1 Location

Normal location of the AWC unit and cable connector for:

- PC.
- Voltage sensor.
- Current sensor.

See pictures on next page.


ABB
Advanced
Weld
Control


VOLTS
AMPS
GAS
TERM

3.0 Electrical Installation

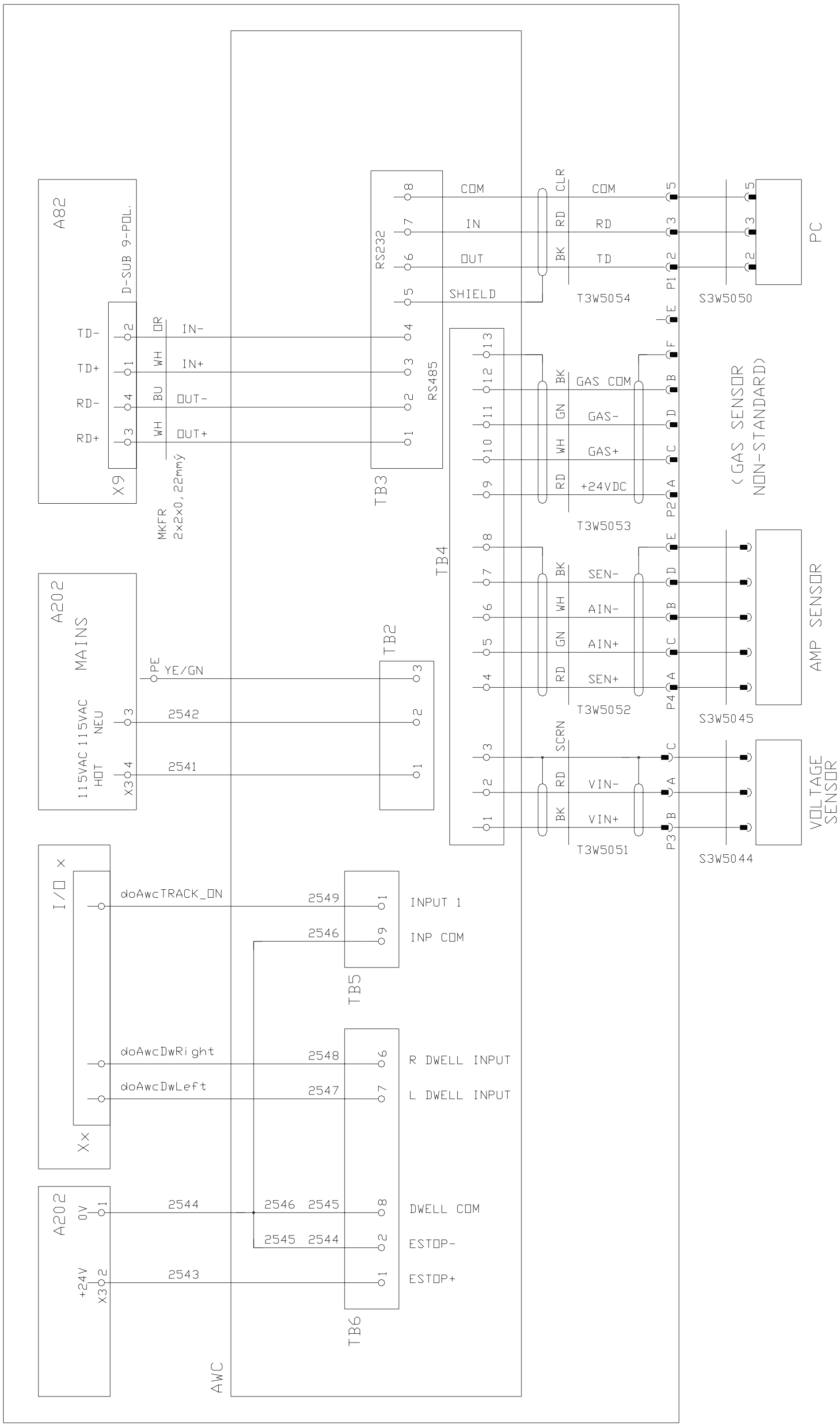


FIGURE 3. Circuit diagram of external wiring to AWC

3.1 Arc Current Sensor

Install the arc current sensor (Model CT-4080) around the welding ground cable. The sensor is a split shell design and the top of the sensor is removed by releasing two side mounted latches. It is important to orient the two red dots on the sensor so that they point toward the welding power supply on the negative side. Pass the welding cable through the opening and reinstall the top of the sensor. Be certain that the top and bottom half are properly aligned and the latches are positively engaged. The current sensor can be mounted using two M6 bolt if necessary. Connect the current sensor cable to the receptacle marked "AMP", or P4.

3.2 Voltage Sensor

Install the voltage sensor (Part number 442890-880) with the black lead attached to the negative terminal of the welding power supply, and the red lead attached to welding positive terminal. Connect the voltage sensor cable to the receptacle marked "VOLT", or P3.

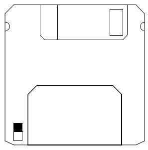
3.3 Communication cable AWC - PC cable

Install the communicationcable to the serial interface RS-232 of Your PC and to the receptacle marked "P1 TERM".

4.0 Software installation

4.1 Robot controller

Installed during the bootsequence of the robot (option disk).



4.1.1 Tracking functionality only

Task	File	Storage	Shared
main	ram1disk:Awc_Ext#.sys	LOADED	No

4.1.2 Tracking with Multi Pass functionality (optional)

Task	File	Storage	Shared
main	ram1disk:Awc_Ext4.sys	LOADED	No
main	ram1disk:AwcBas_0.sys	BUILDIN	Yes
tAdpt	ram1disk:AwcAdapt_2.mod	LOADED	No
main	ram1disk:AwcMulti.sys	LOADED	No

4.2 Robot system requirements

4.2.1 Tracking functionality

Base Ware OS 4.0/30 and ArcWare Plus.

4.2.2 Tracking with Multi Pass functionality (optional)

Base Ware OS 4.0/30, ArcWare Plus, Advanced Functions, Developer's functions and Multitasking.

4.3 PC software

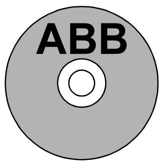
4.3.1 AWC-companion

Use the CD to install the AWC companion software.

- Open AWC companion folder.
- Start setup.exe.
- Follow instructions on the screen.

4.4 Software downloaded to AWC

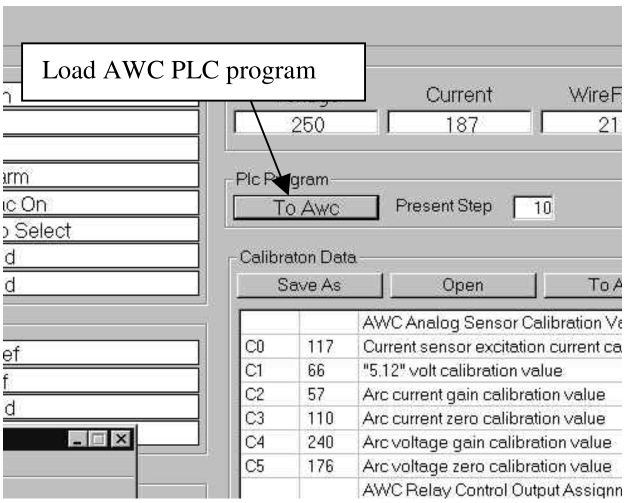
The AWC system need to be loaded with appropriate data before it can be used. The AWC companion PC program can be used for this task.



4.4.1 Load the AWC PLC program

A predefined PLC program needs to be loaded in to the AWC. This is done with the To Awc button on the companion test screen. A file menu will come up were the appropriate file can be selected.

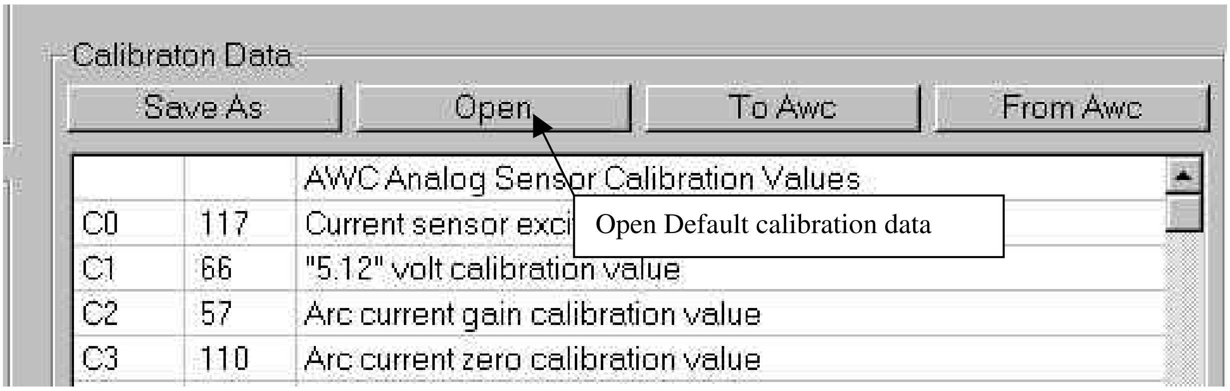
- DownLoad the file Awc_SeArc_V1.plc supplied on the CD.



4.4.2 Calibration data to the AWC

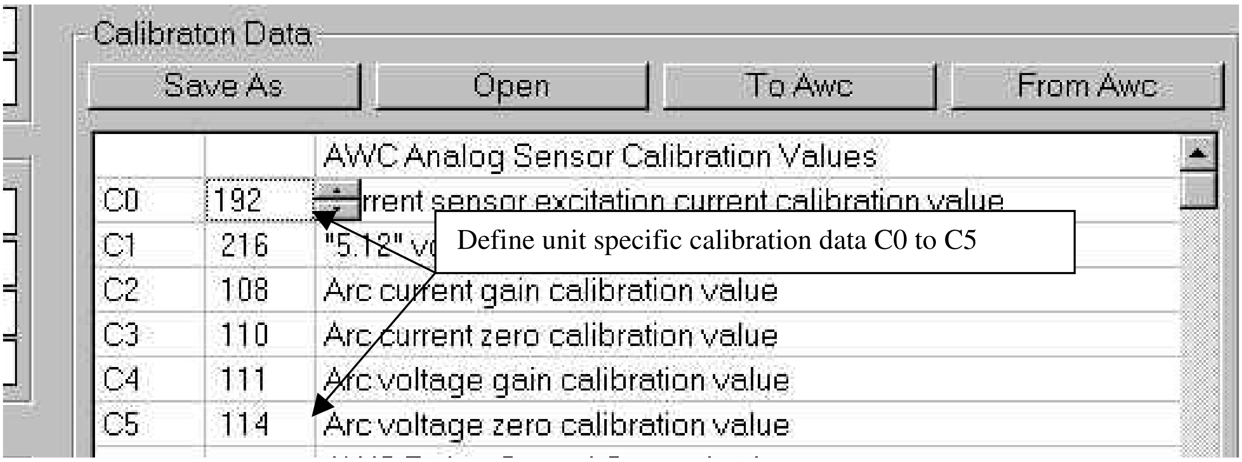
The calibration data defines the AWC output/input assignment and analog scaling. The calibration data also contains unit specific calibration data.

- Open the file Default.cfg. Default.cfg located on the CD and in the default AWC companion directory. This will load default data in to the companion.



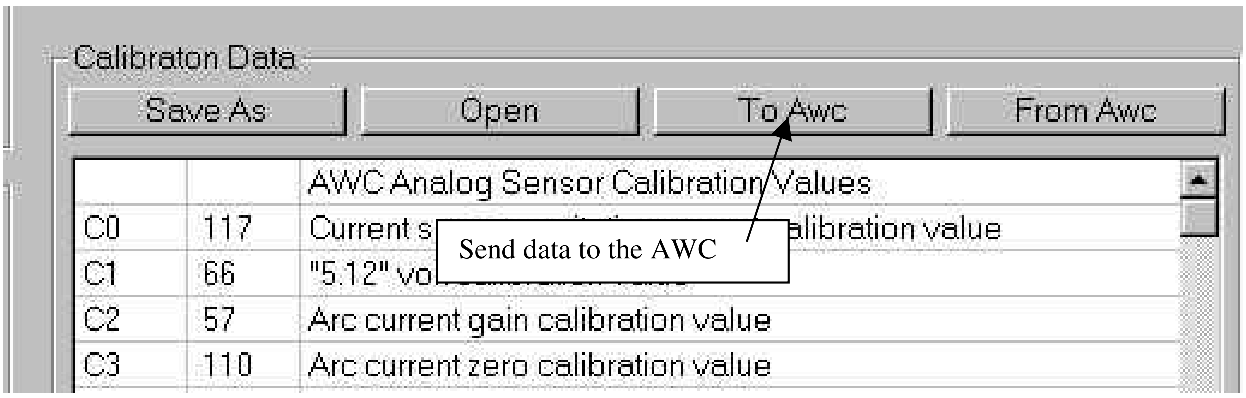
Read in calibration data to the Companion

- Adjust specific parameters C0-C5. The numbers can be found on the front of the AWC cpu board (behind screwterminals).



Define unit specific data

- Send the data to the AWC with the button ToAWC.



Send data to the AWC

4.5 Resetting the AWC

This operation will erase all calibration data and load default data in to the AWC. After reset download .plc and .cfg as described earlier.

Turn the AWC OFF

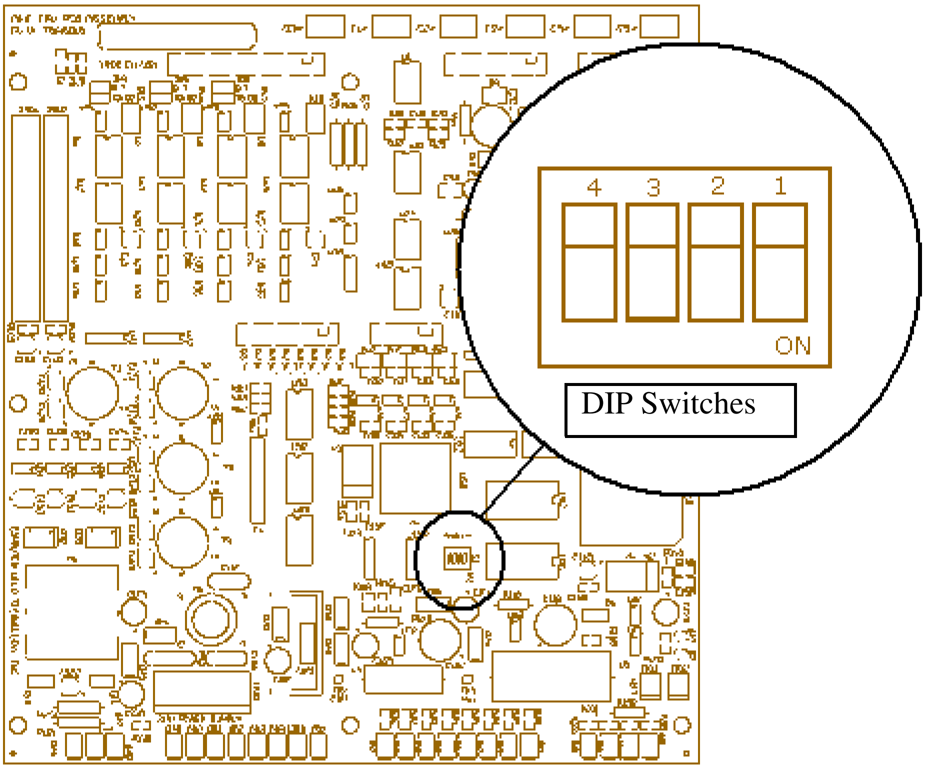
Open the cover of the AWC

Switch dip switch No 4 to the ON position

Turn the AWC On and wait 2-3 sec.

Turn the AWC Off.

Switch dip switch No 4 back to the OFF position



5.0 Troubleshooting

5.1 Voltage & Current sensor

Define weld data and then make a simple weld program using regular ArcL instructions and set doAwcTRACK_ON.

Make a weld when having the AWC companion in the test window. Check to see that current and voltage values show up on the screen.

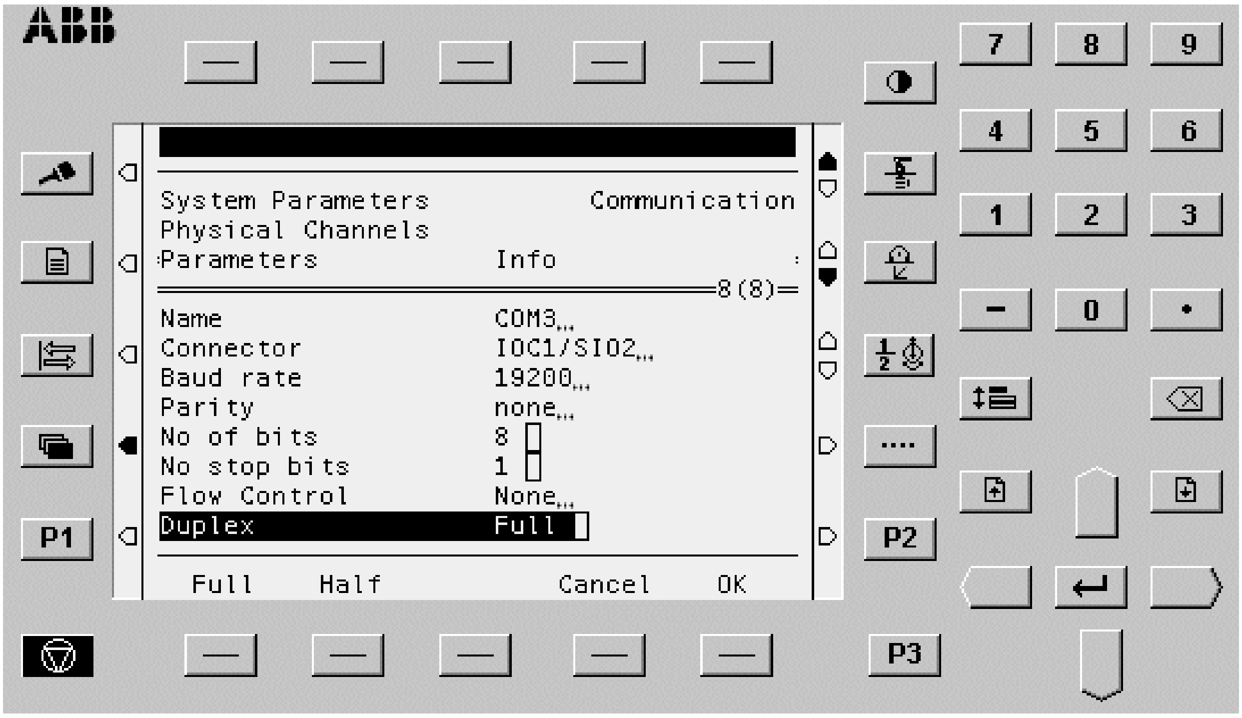
5.2 Communication between robot controler and AWC

AWC needs to be on at robot control startup. The robot controler fails to to open the RS-485 serial link if the AWC unit mainswitch is off during startup.

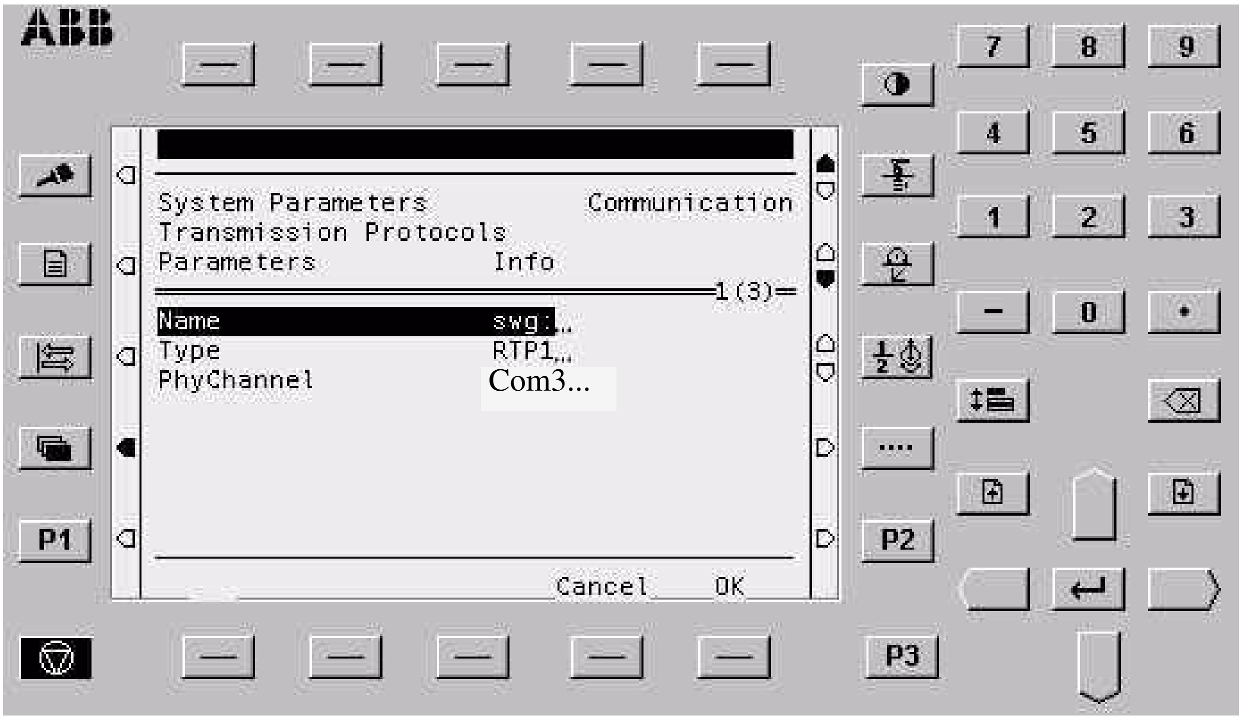
6.0 Reference

6.1 Communication Setup

Physical Channel for the AWC



Sensor protocol definition for the AWC



Note: AWC needs to be on at robot control startup.

6.2 I/O Configuration

The following I/O configuration is required for an AWC system. These I/O points have to be added to the target systems I/O configuration

User signals on physical I/O board:

Signal	I/O pointl
doAwcTRACK_ON	Phsig Output xx
doAwcDwLeft	Phsig Output xx
doAwcDwRight	Phsig Output xx

7.0 General Description

The AWC (Advanced Weld Control) is a weld seam controller integrated into the robot controller. It is designed to track difficult welding joint variations due to cast components or other pre-process problems. It monitors and controls through-the-arc seam tracking.

Features include:

- Centerline, Torch-to-Work and Single side through-the-arc tracking capability.
- Multi-pass capability with variable replay of paths.

7.1 Tracking methods

A through-the-arc tracking system uses the arc as a sensor to adjust the robot path to the actual location of the part. Measuring the arc voltage and welding current, synchronized with the robot weave pattern, the stick-out length is calculated on both sides and in the middle of the weld. The stick-out length in the middle and the difference between the sides are converted in to robot vertical and horizontal corrections.

7.1.1 Torch-to-Work tracking (Height, Z direction)

In Torch to Work mode. The same contact tip to work length is maintained. The contact tip to work distance is specified as voltage and current settings in the weld data. Weaving with almost zero width is required because the correction calculations are synchronized with the weave pattern. See Figure 1-1.

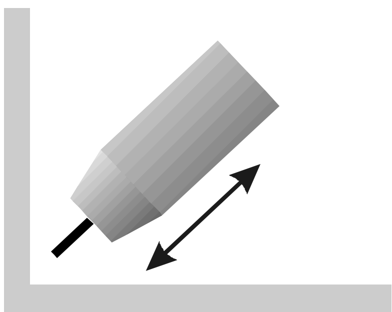


Figure 1-1. Torch to Work Mode

7.1.2 Centerline tracking (Center, Y and Z direction)

Centerline tracking is the most commonly used tracking method. Torch to work tracking is based on measurements made in the middle of the weave pattern. The cross seam tracking is based on measurement made on the sides of the weave pattern. Corrections are calculated based on the difference in stick-out between the sides. The position of the weld can be adjusted side to side using the Bias parameter. See Figure 1-2.

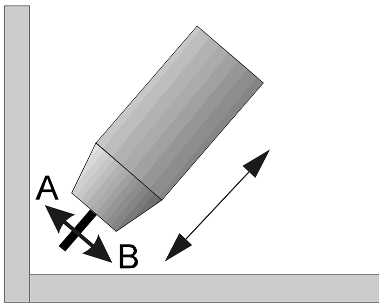


Figure 1-2. Centerline tracking

7.1.3 Single side tracking (Right and Left)

The difference between the centerline tracking method and the single side method is the way cross seam corrections are calculated. When using the single side method, data from one side of the weave is used. The length of the stick-out in the center of the weave is stored as a reference. The side of the grove is then detected as a difference in stick-out at one of the sides compared with the center. The difference in stick-out required for detecting the side is defined as a Penetration level. A higher penetration level makes the weld move further into the selected side.

This method can be used when tracking a lap joint, were the arc might consume one of the side of the grove.

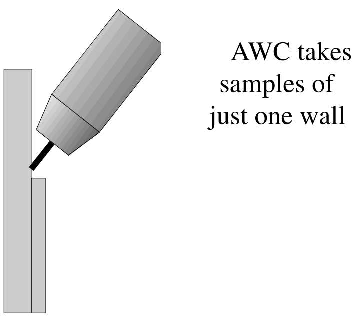


Figure 1-4. Single-Side Tracking

7.2 Multi-Pass (OPTION)

Sometimes multiple weld passes are required due to the required weld size and thickness of the material being joined. AWC makes this easy by tracking the first pass and storing the actual tracked path so it can offset for subsequent passes.

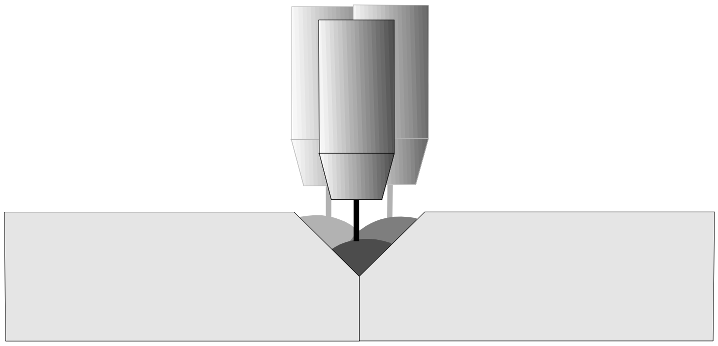
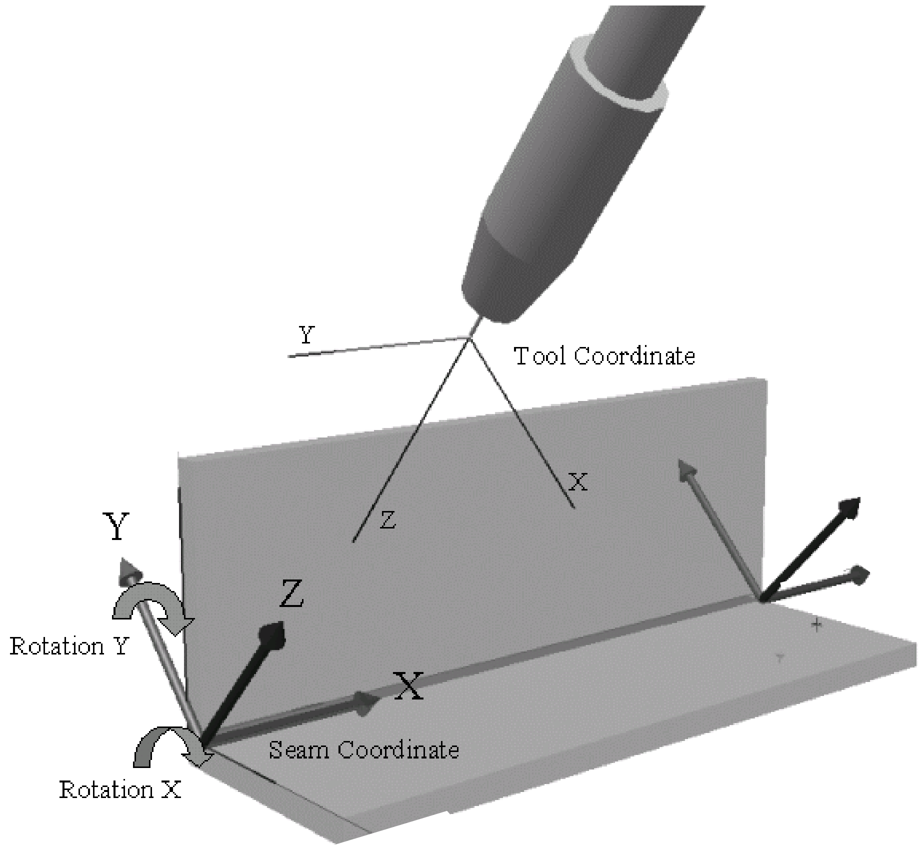


Figure 1-6. Multi-Pass welding

The first weld pass is recorded by using ArcLSto instructions (see chapter Instructions). The interval between stored path points is dictated by the weave length times the spacing used in the ArcLSto instruction. For example, if the weave length is 6 millimeters and the spacing is set to 4, then the distance between stored path points will be 24 millimeters. If the optional argument "Spacing" is not used, the spacing = 4. Multi-pass welding can be used in conjunction with seam tracking.

7.2.1 Multi-Pass (OPTION)

The replayed path can be offset in either the plus or minus "Y" and "Z" seam coordinates and rotated plus or minus "X" and "Y" in seam coordinates. Replayed paths can also be executed in the forward or reverse direction. The start and end path points can be lengthened or shortened by a specified distance in millimeters. If the path is lengthened, the new end point is projected outward by using the last two points that were stored in the path. Lengthening and shortening the path allows for the weld to be tied into previous welds or the parent material itself.



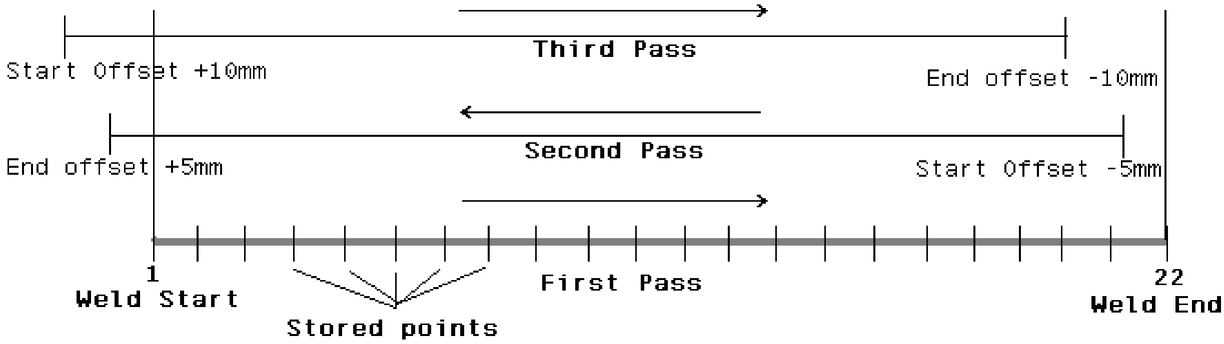


Figure 1-7. Multi-pass path definition

7.2.2 Typical Weld Program w/Tracking

Line #	Code
1	PROC Line( )
2	MoveL *,v100,z1,tWeldGun;
3	AwcSetUp\Center,27,255\Gain Y=20\Gain Z=20
4	ArcLSto\On,*,v300,sm1,wd1,wv1,fine,tWeldGun \Track:=tr1\Spacing:=4;
5	ArcLSto\Off,*, v300,sm1,wd1,wv1,fine,tWeldGun \Track:=tr1;
6	ArcLRep\On\Off,Ly2,v300,sm1, wd2, wv2,z5,tWeldGun \Track:=tr1;
7	ArcLRep\On\Off,Ly3,v300,sm1, wd3, wv3,z5,tWeldGun \Track:=tr1;
8	MoveL *,v100,z1,tWeldGun;
9	ENDPROC

Line #	Explanation
1	Name of procedure.
2	Linear move.
3	Tracking data. Sends tracking data to the AWC.
4	Activates point storage along weld path. "Spacing" determines how often points are stored along the path (i.e. every 4th weave point, etc.) Also, the weld is started, with data similar to a standard ArcL command. Note that the root pass uses tracking, here. "Track:=tr1" must be on every line to enable tracking process.
5	Ends point storage for the weld and finishes the weld as an ArcL\Off would. An "ArcCSto\Off" command does not exist.
6	Replays the stored pass specified by the information contained in the multi data shown as Ly2. Ly2 is the second weld pass with reversed direction, a new torch angle and position offset with a start offset of -5mm and an end offset of +5mm.
7	Replays the stored pass specified by the information contained in the multi data shown as Ly3. Ly3 is the third weld pass with a new torch angle and position offset with a start offset of +10mm and an end offset of -10mm.
8	Linear Move.
9	End of procedure.

8.0 Programming a seam

8.1 Tracking procedures

The performance of a seam tracking system depends on the input from its sensors. The sensor for a Thru-Arc tracking system is the arc. A stable and clean process will enhance the systems tracking performance. Below is a simple guide on how to use an seam tracking welding system.

1 Start simple

Start by using regular ArcL instructions and a simple T-joint configuration weld piece. Program the path with proper stick-out and block the tracking.

To enable data to be sent to AWC companion without tracking active set digital output doAwcTRACK_ON.

```
AwcSetUp\Center,27,255\Gain Y=20\Gain Z=20
MoveL p10,v1000,fine,tWeldGun;
ArcL\On,p20,v1000,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
ArcL\Off,p30,v1000,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
MoveL p10,v1000,fine,tWeldGun;
```

Don't forget to set the arguments in weavedata (wd1): weave_sync_left and _right to 90 define also TrackData (tr1, max_corr=n). "n = maximum mm from programmed path"

2 Develop weld data

Develop the weld data that gives you the weld size that you need, voltage, wire feed speed and robot travel speed. Add some weave to see how wide you can weave and still get a good weld. The wider the robot weaves the better the tracking. The rule of thumb is that the weave width should be minimum two times the wire diameter.

3 Check the real time current value

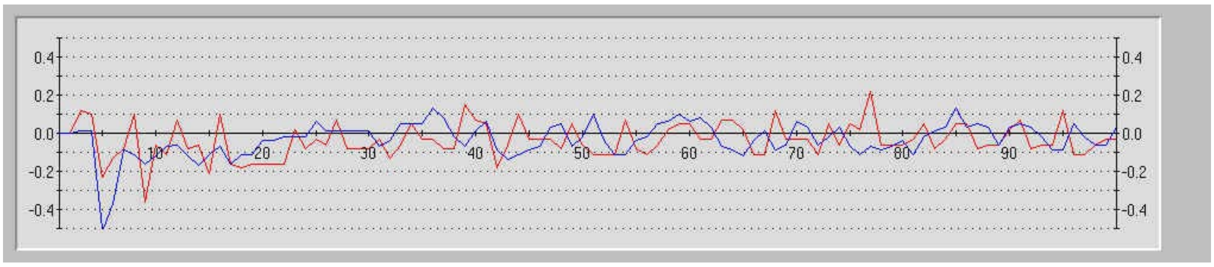
Check the weldingcurrent and arcvoltage via the PC-program AWC-companion for example actual values are V=25 A=201. Update the weld current and arcvoltage in the Awc_Ext instruction. Before doing this operation, ensure that proper path and stick-out was used.

```
AwcSetUp\Center,25,201\Gain Y=20\Gain Z=20
```

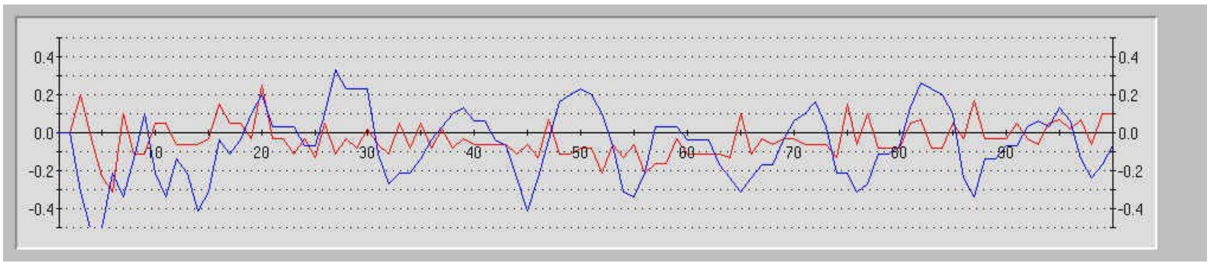
4 Centerline tracking, gain adjustment

Take away blocking of the optional tracking argument to the ArcL instruction. Make a weld program were both the start and endpoint are outside the joint. Make a weld and look for the robot response. Using the AWC companion could be a good help for adjusting the gain levels. The robot should move in to the weld and then follow the weld without any overshoot.

```
AwcSetUp\Center,27,255\Gain Y=20\Gain Z=20
MoveL p10,v1000,fine,tWeldGun;
ArcL\On,p20,v1000,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
ArcL\Off,p30,v1000,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
MoveL p10,v1000,fine,tWeldGun;
```



Proper gain level



Gain to high

9.0 Instructions

9.1 AwcSetUp, instruction for Awc Setup

9.1.1 Description

Used to define trackingtype, welddata and gain for AWC (tracking)

9.1.2 Argument

[\switch Center] Data type: switch
Centerline tracking. (Both height & sidetracking).

[\switch Right] Data type: switch
Both height & sidetracking. Priority right.

[\switch Left] Data type: switch
Both height & sidetracking. Priority left.

[\switch Height] Data type: switch
Only height control.

Voltage Data type: num
Arcvoltage.

Current Data type: num
Weldingcurrent.

GainY Data type: num
Side gain. Normal 10 - 40

GainZ Data type: num
Height gain. Normal 10 - 40

Penetration Data type: num
Penetration. Normal = 2 - 7

Bias Data type: num
Bias. Normal = 10 -20

TorchClim Data type: num
Maximum correction in Z (in mm) default is 0.5mm

CrossClim Data type: num
Maximum correction in Y (in mm) default is 0.5mm

9.2 ArcLSto

Initialization or termination of path storage

The "ArcLSto" instruction is used when it is necessary to store or *memorize* a weld path for the robot to follow on a subsequent pass. The ArcLSto instruction has the following functions:

- Initialization or termination of path storage
- Initialization or termination of the arc welding process (ArcL\On\Off)

9.2.1 Arguments

ArcLSto [\On] | [\Off], ToPoint, Speed, Seam, Weld, Weave, Zone, Tool,[\WObj] [\Track] [\SeamName] [\Spacing]

[\On] Data type: *switch*

For description see the ArcL instruction

[\Off] Data type: *switch*

For description see the ArcL instruction

ToPoint Data type: robtarget

The destination position of the robot and external axes. This is either defined as a named position or stored directly in the instruction (indicated by an * in the instruction).

Speed Data type: speeddata

The speed of the TCP is controlled by the argument Speed in the following cases:

- \- When the argument \On is used (weld start preparations at a flying start).
- \- When the program is run instruction-by-instruction (forward step).

The speed of the TCP during welding is the same as for the arguments Seam and Weld. Speed data also describes the speed of the tool's reorientation and the speed of any uncoordinated external axes.

Seam Data type: seamdata

Seam data describes the start and end phases of a welding process. The argument *Seam* is included in all arc welding instructions so that, regardless of the position of the robot when the process is interrupted, a proper weld end and restart is achieved. Normally the same seam data is used in all instructions of a seam.

Weld Data type: welddata

Weld data describes the weld phase of the welding process. Weld data is often changed from one instruction to the next along a seam.

Weave Data type: weavedata

Weave data describes the weaving that is to take place during the heat and weld phases. Welding without weaving is obtained by specifying, for example, the weave data noweave. (No weaving if the weave_shape component value is zero.)

Zone Data type: zonedata

Zone data defines how close the axes must be to the programmed position before they can start moving towards the next position.In the case of a fly-by point, a corner path is generated past that position. In the case of a stop point (fine), the movement is interrupted until all axes have reached the programmed point. A stop point is always generated automatically at the start position of a weld (even in the case of a flying start) and at a controlled weld end position. Fly-by points, such as z10, should be used for all other weld positions. Weld data changes over to the next arc welding instruction at the centre point of the corner path (if not delayed by the delay_distance component in the Weld argument).

Tool Data type: tooldata

The tool used in the movement. The TCP of the tool is the point moved to the specified destination position. The z-axis of the tool should be parallel with the torch.

[\WObj] (Work Object) Data type: wobjdata

The work object (coordinate system) to which the instruction's robot position isreferenced. When this argument is omitted, the robot position is referenced to the world coordinate system. It must, however, be specified if a stationary TCP or coordinated external axes are used.

[\Track] Data type: trackdata

Track data describes the parameters used for tracking. In the case of AWC tracking the only valid parameter in the track data is the track max correction parameter.

[\SeamName] (Seam Name) Data type: string

The seam name is a string, which will be added to error logs if an error occurs during the welding sequence. *\SeamName* is only applicable in the first instruction of a sequence of weld instructions i.e. together with the *\On argument*.

[\Spacing] Data type: num

Defines the distance between stored points during path storage. An integer number between 1 and 10 can be entered. Set to one a point will be stored every weave. (Default = 4)

9.2.2 Program execution

Controlling process equipment

The process equipment is controlled by the robot in such a way that the entire process and each of its phases are coordinated with the robot's movements.

Performed functions

- Robot is moved to the ToPoint
- Path storage is initialized or terminated
- The arc is initialized or terminated

The destination position is referenced to:

- the specified object coordinate system if the argument \WObj is used; the world coordinate system if the argument \WObj is not used.

Execution in forward step mode

Robot is moved to the ToPoint location

Backward execution

Robot is moved to the ToPoint

9.2.3 Fault management

The process is supervised by a number of signal inputs. If anything abnormal is detected, program execution will stop. The AWC specific error handler is supplied (in AwcMulti.sys and/or ErrHand.sys) and can be used for faultmanagement.

```
TEST ErrComHandler(\StopKey)
CASE 1:
  RETRY;
CASE 3:
  RETURN;
CASE 4:
  !
  !  Press Forward step
  Stop;
  RETURN;
CASE 5:
  RAISE;
ENDTEST
```

Syntax

ArcLSto(

['\'On','] | ['\'Off',']
[ToPoint ':='] < expression (**IN**) of *robtarget* > ','
[Speed ':='] < expression (**IN**) of *speeddata* >','
[Seam ':='] < persistent (**PERS**) of *seamdata* > ','
[Weld ':='] < persistent (**PERS**) of *welddata* > ','
[Weave ':='] < persistent (**PERS**) of *weavedata* > ','
[Zone ':='] < expression (**IN**) of *zonedata* >','
[Tool ':='] < persistent (**PERS**) of *tooldata* >','
['\' WObj ':=' < persistent (**PERS**) of *wobjdata* > ';'
['\' Track':='] < persistent (**PERS**) of trackdata >','
['\' SeamName ':=' < expression (**IN**) of *string* >]
['\Spacing:='] < expression (**IN**) of *num* >';'

9.2.4 Description

ArcLSto

The "ArcLSto" instruction replaces the traditional **"ArcL\On"** and **"ArcL\Off"** instructions when storing a path. This instruction does not replace the "ArcL" or "ArcC" instructions therefore they are used as normal.

The distance between the path points or *resolution* of the stored path is based on the weave cycle (length) in mm times the spacing specified in the argument **"\spacing"**. This argument has a range of 1 to 10 with a default value of 2. A minimum of 10 points must be stored in a path before it can be replayed with the "ArcLRep" instruction.

As with a traditional ArcL, the "ArcLSto" can be used in conjunction with Through Arc Seam Tracking by inserting the **"\track"** argument.

9.2.5 Program Example

```
MoveL,*, vmax,fine,tWeldGun;
ArcLSto\On,*,vmax,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1\spacing:=4;
ArcL,*, vmax,sm1,wd1,wv1,z5,tWeldGun\Track:=tr1;
ArcL,*, vmax,sm1,wd1,wv1,z5,tWeldGun\Track:=tr1;
ArcLSto\Off,*, vmax,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
MoveL,*, vmax,fine,tWeldGun;
```

9.2.6 Related Information

The stored path is replayed with the **"ArcLRep"** instruction.
The stored path can be saved for a later replay with the instruction **"PathSaveToFile"**.
The stored path can be retrieved for replay with the instruction **"PathReadFromFile".**

9.3 ArcLRep

Replay of a stored path

The ArcLRep instruction is used for re-playing a stored path. The instruction can be used for re-play of one complete layer or a section of a layer. The path is stored using the ArcLSto or ArcLSto1 instruction.

9.3.1 Arguments

ArcLRep [\On] [\Off] MultiData, [\StartInd] [\EndInd] Speed, Seam, Weld, Weave, Zone, Tool, [\Wobj] [\Track] [\SeamName]

[\On] Data type: switch

The \On argument is used at the start of a re-play sequence.Regardless of what is specified in the *Zone* argument, the destination position will be a stop point.

[\Off] Data type: switch

If the argument *\Off* is used, welding ends when the robot reaches the destination position (end of the stored path). Regardless of what is specified in the *Zone* argument, the destination position will be a stop point.
Note: For the ArcLRep instruction both the On and Off switch can be activated.

MultiData Data type: Multi

The Multi data contains the offset information and adaptive information for the path.

[\StartInd] Data type: num

The optional argument StartInd is used if the path should be re-played from a specific index not from the beginning of the stored path.
Note: First index in a path is always 1. Last index is contained in the variable *nPthEnd*.

[\EndInd] Data type: num

The optional argument EndInd is used if the path should be re-played to a specific index not the end of the stored path.
Note: First index in a path is always 1. Last index is contained in the variable *nPthEnd*.

Speed Data type: speeddata

The speed of the TCP is controlled by the argument Speed during the movement towards the start of the re-play sequence.The speed of the TCP during welding is the same as for the arguments Seam and Weld. Speed data also describes the speed of the tool's reorientation and the speed of any uncoordinated external axes.

Seam Data type: seamdata

Seam data describes the start and end phases of a welding process. The argument *Seam* is included in all arc welding instructions so that, regardless of the position of the robot when the process is interrupted, a proper weld end and restart is achieved.

Weld Data type: welddata

Weld data describes the weld phase of the welding process.

Weave Data type: weavedata

Weave data describes the weaving that is to take place during the heat and weld phases. Welding without weaving is obtained by specifying, for example, the weave data noweave. (No weaving if the weave_shape component value is zero.)

Zone Data type: zonedata

Zone data defines how close the axes must be to the programmed position before they can start moving towards the next position. In the case of a fly-by point, a corner path is generated past that position. In the case of a stop point (fine), the movement is interrupted until all axes have reached the programmed point. A stop point is always generated automatically at the start position of a weld (even in the case of a flying start) and at a controlled weld end position. Fly-by points, such as z10, should be used for all other weld positions. It is recommended to use a z5 data for the re-play instruction.

Tool Data type: tooldata

The tool used in the movement. The TCP of the tool is the point moved to the specified destination position. The z-axis of the tool should be parallel with the torch.

[\WObj] (Work Object) Data type: wobjdata

The work object (coordinate system) to which the instruction's robot position is referenced. When this argument is omitted, the robot position is referenced to the world coordinate system. It must, however, be specified if a stationary TCP or coordi-nated external axes are used.

[\Track] Data type: trackdata

Track data describes the parameters used for tracking. In the case of AWC tracking the only valid parameter in the track data is the track max correction parameter.

[\SeamName] (Seam Name) Data type: string

The seam name is a string which will be added to error logs if an error occurs during the welding sequence. *\SeamName* is only applicable in the first instruction of a sequence of weld instructions i.e together with the *\On argument*.

9.3.2 Program execution

Controlling process equipment

The process equipment is controlled by the robot in such a way that the entire process and each of its phases are coordinated with the robot's movements.

Fault management

The process is supervised by a number of signal inputs. If anything abnormal is detected, program execution will stop. The AWC specific error handler is supplied (in AwcMulti.sys and/or ErrHand.sys) and can be used for faultmanagement.

```
TEST ErrComHandler(\StopKey)
CASE 1:
  RETRY;
CASE 3:
  RETURN;
CASE 4:
  !
  ! Press Forward step
  Stop;
  RETURN;
CASE 5:
  RAISE;
ENDTEST
```

Syntax

```
ArcLRep(
[ '\'On',' ] | [ '\'Off',' ]
[ MultiData ':=' ] < expression (IN) of multi >','
[ Speed ':=' ] < expression (IN) of speeddata >','
[ Seam ':=' ] < persistent (PERS) of seamdata > ','
[ Weld ':=' ] < persistent (PERS) of welddata > ','
[ Weave ':=' ] < persistent (PERS) of weavedata > ','
[Zone ':=' ] < expression (IN) of zonedata >','
[ Tool ':=' ] < persistent (PERS) of tooldata >','
[ '\' WObj ':=' < persistent (PERS) of wobjdata > ';'
['\' Track':=' ] < persistent (PERS) of trackdata >','
[ '\' SeamName ':=' < expression (IN) of string > ]
```

9.3.3 Description

ArcLRep

The "ArcLRep" instruction is used in multipass welding to replay a stored weld path without teaching each subsequent pass. The replayed path direction, start and end offset, Y and Z path offset and Y and X torch rotation information are setup in the multi data. See "**Multi data**" under Data types. In this program example the multi data has been named "Layer2".In the program example you will notice that we are using \On and \Off in the same instruction therefore the entire weld process will be initiated and terminated in this single instruction. Transition welding can be accomplished by using separate ArcLRep\On and ArcLRep\Off instructions with unique multi, seam, weld and weave data.

Note: It is recommended to use a zone z5 in this instruction. If a fine point is used in the ArcLRep instruction the weave will stop and restart at every path point recorded by the ArcLSto instruction.

9.3.4 Example

MoveL ...
ArcLSto\On,*,v100,sm1,wdFL104m,wvAdapt,fine,tWeldGun\Track:=tr1;
ArcLSto\Off,*,v100,sm1,wdFL104m,wvAdapt,fine,tWeldGun\Track:=tr1;
MoveL ...
ArcLRep\On\Off,Layer2,v100,sm1,wdFL10m,wv2,z5,tWeldGun;
MoveL ...

9.3.5 Related Information

The path is stored with the **"ArcLSto"** instruction.
The stored path can be saved for a later replay with the instruction **"PathSaveToFile"**.
The stored path can be reloaded for replay with the instruction **"PathReadFromFile".**

9.4 PathSaveToFile

Store path to a file

The PthSaveToFile instruction has the following function:

- Store a recorded weld path to a file device

9.4.1 Arguments

PthSaveToFile, PathNo, [\FilePath]

PathNo Data type: num

When the path is stored it is associated with a number in the event that multiple paths are stored. Therefore when reading a path in, the path number must be specified.

[\FilePath] Data type: string

This optional argument is used in the event that the path is stored to a file path other than default. The default file path is Ramdisk.

9.4.2 Program execution

Execution in program

A weld path is executed and recorded. The instruction **PthSaveToFile** is executed and the recorded weld path is saved with a specific path number. The saved path is loaded with PathReadFromFile and replayed using the replay instructions.

Execution in forward step mode

The specified path file is saved.

Backward execution

No operation.

9.4.3 Description

It is only necessary to save the path to a file if another path is to be stored before replaying this path with the replay instructions.

9.4.4 Fault management

If anything abnormal is detected, program execution will stop.

ERROR
  RAISE;
ENDTEST

Syntax

PthSaveToFile
[PathNo:='] < expression (**IN**) of *num* >','
[\FilePath:='] < expression (**IN**) of *string* >','

9.4.5 Example

MoveL ...
ArcLSto\On ,*,v100,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
ArcLSto\Off ,*,v100,sm1, wd1,wv1,fine,tWeldGun\Track:=tr1;
MoveL ...
PathSaveToFile,1;
MoveL ...
ArcLSto\On ,*,v100,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
ArcLSto\Off ,*,v100,sm1, wd1,wv1,fine,tWeldGun\Track:=tr1;
MoveL ...
PathSaveToFile,2;
MoveL ...
PathReadFromFile,1;
ArcLRep\On\Off,Layer2,v100,sm1,wd1,wv1,z5,tWeldGun;
MoveL ...
PathReadFromFile,2;
ArcLRep\On\Off,Layer2,v100,sm1,wd1,wv1,z5,tWeldGun;
MoveL ...

9.5 PthReadFromFile

Stored path file retrieval

The PthReadFromFile instruction has the following function:
Upload of Stored path for replay

9.5.1 Arguments

PthReadFromFile, PathNo, [\FilePath]

PathNo Data type: num

When the path is stored it is associated with a number in the event that multiple paths are stored. Therefore when reading a path in, the path number must be specified.

[\FilePath] Data type: string

This optional argument is used in the event that the path was stored to a file path other than default. The default file path is Ramdisk.

9.5.2 Program execution

Execution in program

A weld path is executed and recorded. The instruction PthSaveToFile is executed and the recorded weld path is saved with a specific path number. The saved path is loaded with PathReadFromFile and replayed using the replay instructions.

Execution in forward step mode

The specified path file is loaded.

Backward execution

No operation.

9.5.3 Description

It is only necessary to save the path to a file if another path is to be stored before replaying this path with the replay instructions.

9.5.4 Fault management

If anything abnormal is detected, program execution will stop.

```
ERROR
  RAISE;
ENDTEST
```

Syntax

PthReadFromFile
[PathNo:=’] < expression (**IN**) of *num* >’,’
[\FilePath:=’] < expression (**IN**) of *string* >’,’

9.5.5 Example

```
MoveL ...
ArcLSto\On ,*,v100,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
ArcLSto\Off ,*,v100,sm1, wd1,wv1,fine,tWeldGun\Track:=tr1;
MoveL ...
PathSaveToFile,1;
MoveL ...
ArcLSto\On ,*,v100,sm1,wd1,wv1,fine,tWeldGun\Track:=tr1;
ArcLSto\Off ,*,v100,sm1, wd1,wv1,fine,tWeldGun\Track:=tr1;
MoveL ...
PathSaveToFile,2;
MoveL ...
PathReadFromFile,1;
ArcLRep\On\Off,Layer2,v100,sm1,wd1,wv1,z5,tWeldGun;
MoveL ...
PathReadFromFile,2;
ArcLRep\On\Off,Layer2,v100,sm1,wd1,wv1,z5,tWeldGun;
MoveL ...
```

9.6 Multi

Multi-pass offset data

The multi data is used to define the path offset for a re-played path (instruction ArcL-Rep)

9.6.1 Description

The data type contains the information on how the robot should position the tool relative to a stored path.

9.6.2 Components

Direction Data type: num

Direction of travel for the replayed path. Can be set to 1 or – 1. A setting of 1 means that the path will be re-played in the same direction as it was stored. A –1 will re-play the path in the opposite direction as it was stored.

StartOffset Data type: num

Offset in mm for the start of the path relative to the first or last stored point depending on *direction*. A negative number shortens the weld path.

EndOffset Data type: num

Offset in mm for the end of the path relative to the first or last stored point depending on *direction*. A negative number shortens the weld path.

SeamOffs_y Data type: num

Fixed path offset in millimeters for the seam Y direction.

SeamOffs_z Data type: num

Fixed path offset in millimeters for the seam Z direction.

SeamRot_x Data type: num

Torch rotation in degrees around seam x-axis. Rotation is relative to the stored path.

SeamRot_y Data type: num

Torch rotation in degrees around seam y-axis. Rotation is relative to the stored path.

9.6.3 Structure

<data object of multi>
< Direction of num>
< StartOffset of num>
< EndOffset of num>
< SeamOffs_y of num>
< SeamOffs_z of num>
< SeamRot_x of num>
< SeamRot_y of num>

9.7 TrackData

Through Arc Seam Tracking data

TrackData is used to define tracking parameters (instruction ArcL)

9.7.1 Description

This data type contains the information on how much the robot should react to AWC path trajectory corrections.

9.7.2 Components

joint_no Data type: num

Used for camera tracking only. Should be set to zero.

max_blind Data type: num

Used for camera tracking only. Should be set to zero.

filter Data type: num

Used for camera tracking only. Should be set to zero.

max_corr Data type: num

Maximum allowable deveation in millimeters from the programmed path.

SeamOffs_y Data type: num

Used for camera tracking only. Should be set to zero.

SeamRot_z Data type: num

Used for camera tracking only. Should be set to zero.

9.7.3 Structure

<data object of trackdata>
< joint_no>
< max_blind>
< filter>
< max_corr>
< seamoffs_y>
< seamoffs_z>

10.0 Spare parts

10.1 AWC with Tracking and Multi Pass (optional) functionality

Complete AWC installation 505 500-880

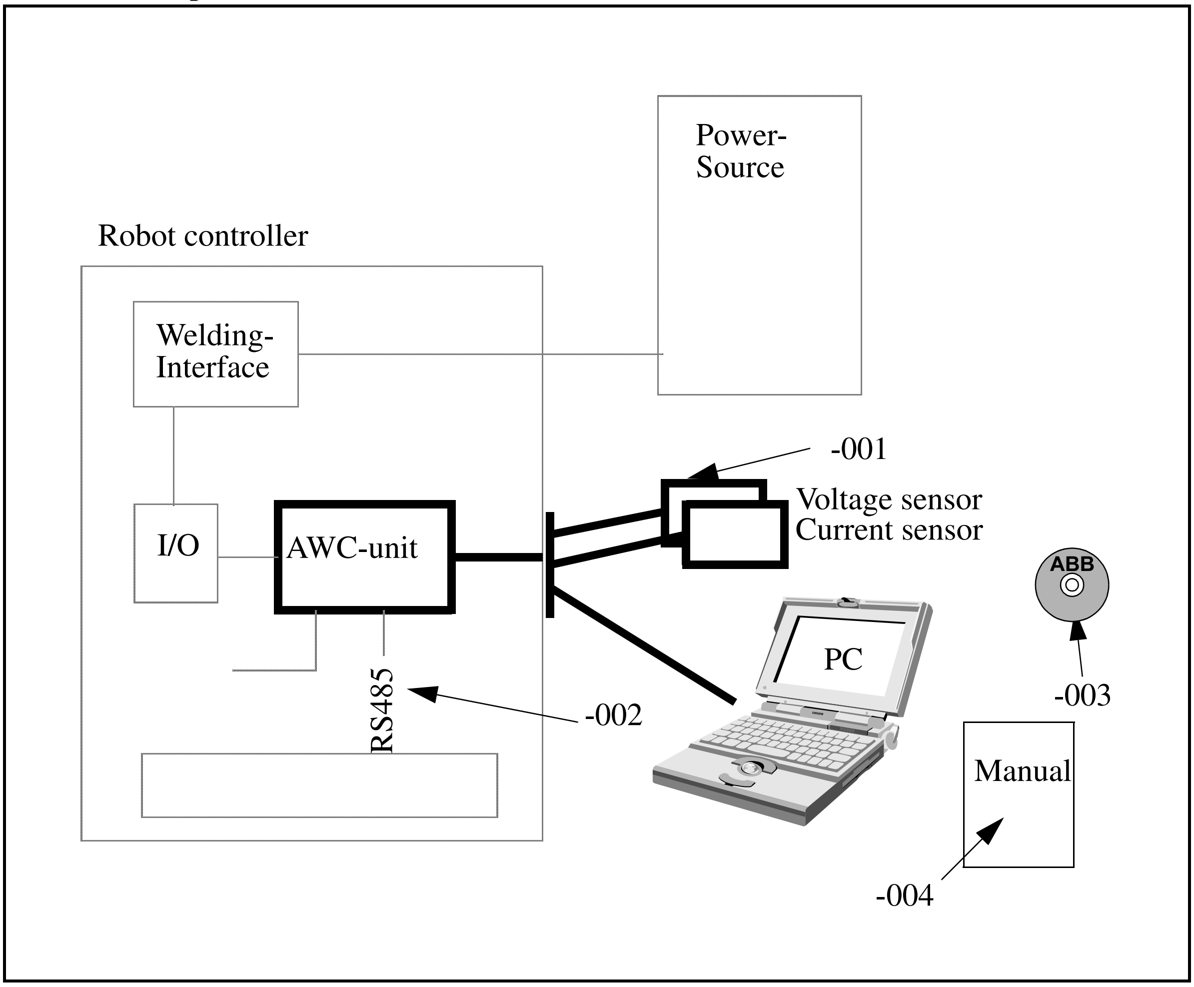


Table 1: Complete AWC installation, group 3HEA 505 500-880

Pos.	Articlenumber	Description	Model
001	3HEA 554934-880	AWC, see table 2	
002	3HEA 558120-880	Communication cable AWC-S4Cplus	
003	3HEA 505 499-001	Software CD	AWC-companion AWC-files .plc .cfg & Manual
004	3HEA 505 498-102	Manual	

10.2 AWC-unit with accessories

AWC, group 3HEA 554 934-880

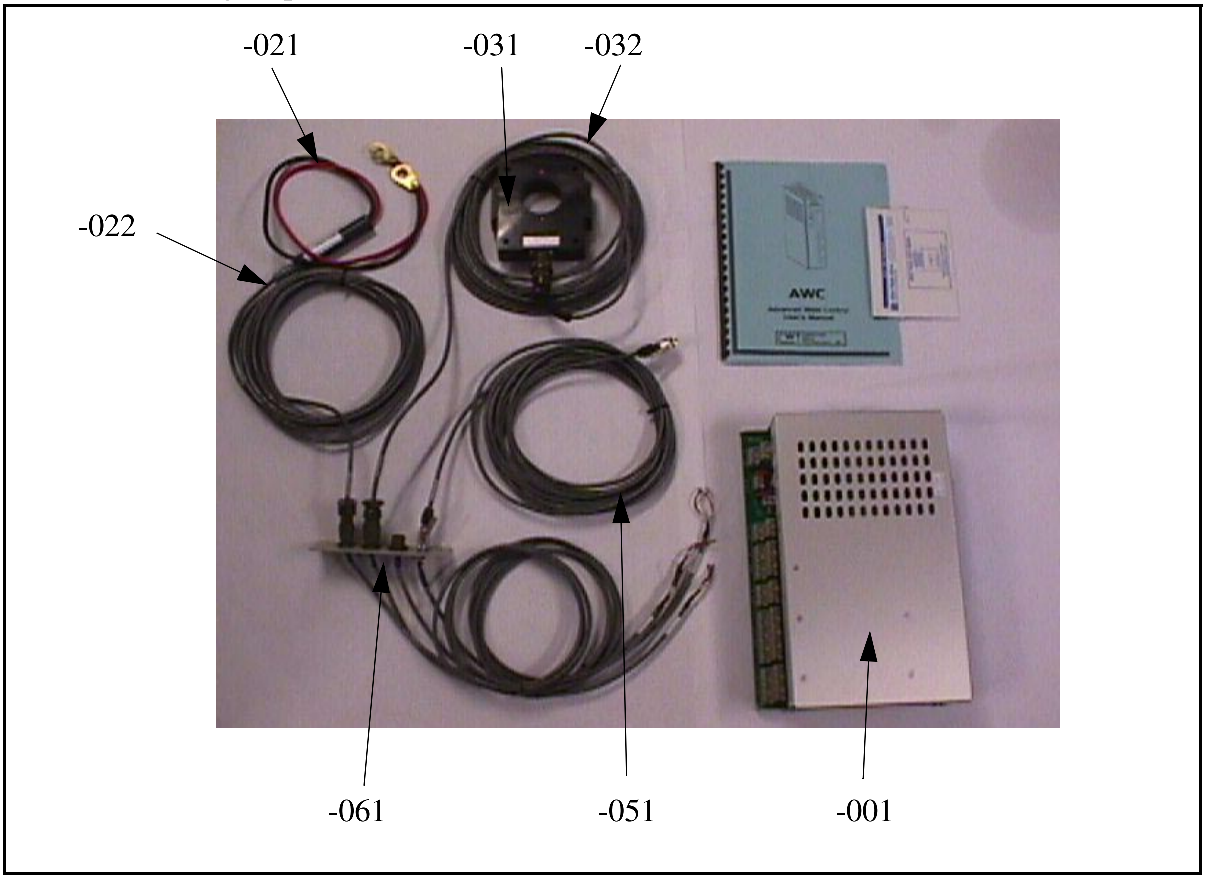


Table 2: AWC-unit with sensors, group 3HEA 554 934-880

Articlenumber	Description	P/N.	Model
3HEA 554934-001	AWC without Drive Control Board (with Thruarc control EPROM)	T0A5019	
3HEA 554934-021	Voltage Probe Sensor	T3A5034	442890-880
3HEA 554934-022	Voltage Probe Cable L=7,5m	S3W5044	
3HEA 554934-031	Current Transducer	X3Q5010	CT-4080
3HEA 554934-032	Current sensor cable L=7,5m	S3W5045	
3HEA 554934-051	PC - AWC terminal Cable L=7,5m	S3W5050	
3HEA 554934-061	Internal cabling. Flange with plugs & cables for connection of external sensors and PC	T3A5040 includes fol-lowing T3E5026 + T3W5051 + T3W5052 + T3W5053 + T3W5054	